



OpenShift Container Platform 4.20

Network APIs

Reference guide for network APIs

OpenShift Container Platform 4.20 Network APIs

Reference guide for network 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 network API objects and their detailed specifications.

Table of Contents

CHAPTER 1. NETWORK APIS	18
1.1. CLUSTERUSERDEFINEDNETWORK [K8S.OVN.ORG/V1]	18
1.2. ADMINNETWORKPOLICY [POLICY.NETWORKING.K8S.IO/V1ALPHA1]	18
1.3. ADMINPOLICYBASEDEXTERNALROUTE [K8S.OVN.ORG/V1]	18
1.4. BASELINEADMINNETWORKPOLICY [POLICY.NETWORKING.K8S.IO/V1ALPHA1]	18
1.5. CLOUDPRIVATEIPCONFIG [CLOUD.NETWORK.OPENSIFT.IO/V1]	18
1.6. EGRESSFIREWALL [K8S.OVN.ORG/V1]	19
1.7. EGRESSIP [K8S.OVN.ORG/V1]	19
1.8. EGRESSQOS [K8S.OVN.ORG/V1]	19
1.9. EGRESSSERVICE [K8S.OVN.ORG/V1]	19
1.10. ENDPOINTS [V1]	19
1.11. ENDPOINTS_SLICE [DISCOVERY.K8S.IO/V1]	20
1.12. EGRESSROUTER [NETWORK.OPERATOR.OPENSIFT.IO/V1]	20
1.13. GRPCROUTE [GATEWAY.NETWORKING.K8S.IO/V1]	21
1.14. GATEWAY [GATEWAY.NETWORKING.K8S.IO/V1]	21
1.15. GATEWAYCLASS [GATEWAY.NETWORKING.K8S.IO/V1]	21
1.16. HTTPROUTE [GATEWAY.NETWORKING.K8S.IO/V1]	22
1.17. INGRESS [NETWORKING.K8S.IO/V1]	22
1.18. INGRESSCLASS [NETWORKING.K8S.IO/V1]	22
1.19. IPADDRESS [NETWORKING.K8S.IO/V1]	22
1.20. IPAMCLAIM [K8S.CNI.CNCF.IO/V1ALPHA1]	22
1.21. IPPOOL [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]	23
1.22. MULTINETWORKPOLICY [K8S.CNI.CNCF.IO/V1BETA1]	23
1.23. NETWORKATTACHMENTDEFINITION [K8S.CNI.CNCF.IO/V1]	23
1.24. NETWORKPOLICY [NETWORKING.K8S.IO/V1]	23
1.25. NODESLICEPOOL [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]	23
1.26. OVERLAPPINGRANGEIPRESERVATION [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]	23
1.27. PODNETWORKCONNECTIVITYCHECK [CONTROLPLANE.OPERATOR.OPENSIFT.IO/V1ALPHA1]	24
1.28. REFERENCEGRANT [GATEWAY.NETWORKING.K8S.IO/V1BETA1]	24
1.29. ROUTE [ROUTE.OPENSIFT.IO/V1]	24
1.30. SERVICE [V1]	25
1.31. SERVICECIDR [NETWORKING.K8S.IO/V1]	25
1.32. IPADDRESSCLAIM [IPAM.CLUSTER.X-K8S.IO/V1BETA1]	25
1.33. USERDEFINEDNETWORK [K8S.OVN.ORG/V1]	25
CHAPTER 2. CLUSTERUSERDEFINEDNETWORK [K8S.OVN.ORG/V1]	26
2.1. SPECIFICATION	26
2.1.1. .spec	27
2.1.2. .spec.namespaceSelector	27
2.1.3. .spec.namespaceSelector.matchExpressions	28
2.1.4. .spec.namespaceSelector.matchExpressions[]	28
2.1.5. .spec.network	29
2.1.6. .spec.network.layer2	30
2.1.7. .spec.network.layer2.ipam	31
2.1.8. .spec.network.layer3	32
2.1.9. .spec.network.layer3.subnets	34
2.1.10. .spec.network.layer3.subnets[]	34
2.1.11. .spec.network.localnet	34
2.1.12. .spec.network.localnet.ipam	38
2.1.13. .spec.network.localnet.vlan	40
2.1.14. .spec.network.localnet.vlan.access	40

2.1.15. .status	40
2.1.16. .status.conditions	41
2.1.17. .status.conditions[]	41
2.2. API ENDPOINTS	42
2.2.1. /apis/k8s.ovn.org/v1/clusteruserdefinednetworks	43
2.2.2. /apis/k8s.ovn.org/v1/clusteruserdefinednetworks/{name}	45
2.2.3. /apis/k8s.ovn.org/v1/clusteruserdefinednetworks/{name}/status	47
CHAPTER 3. ADMINNETWORKPOLICY [POLICY.NETWORKING.K8S.IO/V1ALPHA1]	51
3.1. SPECIFICATION	51
3.1.1. .spec	52
3.1.2. .spec.egress	53
3.1.3. .spec.egress[]	54
3.1.4. .spec.egress[].ports	55
3.1.5. .spec.egress[].ports[]	56
3.1.6. .spec.egress[].ports[].portNumber	56
3.1.7. .spec.egress[].ports[].portRange	57
3.1.8. .spec.egress[].to	58
3.1.9. .spec.egress[].to[]	58
3.1.10. .spec.egress[].to[].namespaces	59
3.1.11. .spec.egress[].to[].namespaces.matchExpressions	60
3.1.12. .spec.egress[].to[].namespaces.matchExpressions[]	60
3.1.13. .spec.egress[].to[].nodes	61
3.1.14. .spec.egress[].to[].nodes.matchExpressions	62
3.1.15. .spec.egress[].to[].nodes.matchExpressions[]	62
3.1.16. .spec.egress[].to[].pods	63
3.1.17. .spec.egress[].to[].pods.namespaceSelector	63
3.1.18. .spec.egress[].to[].pods.namespaceSelector.matchExpressions	64
3.1.19. .spec.egress[].to[].pods.namespaceSelector.matchExpressions[]	64
3.1.20. .spec.egress[].to[].pods.podSelector	65
3.1.21. .spec.egress[].to[].pods.podSelector.matchExpressions	65
3.1.22. .spec.egress[].to[].pods.podSelector.matchExpressions[]	66
3.1.23. .spec.ingress	66
3.1.24. .spec.ingress[]	66
3.1.25. .spec.ingress[].from	68
3.1.26. .spec.ingress[].from[]	68
3.1.27. .spec.ingress[].from[].namespaces	69
3.1.28. .spec.ingress[].from[].namespaces.matchExpressions	70
3.1.29. .spec.ingress[].from[].namespaces.matchExpressions[]	70
3.1.30. .spec.ingress[].from[].pods	71
3.1.31. .spec.ingress[].from[].pods.namespaceSelector	71
3.1.32. .spec.ingress[].from[].pods.namespaceSelector.matchExpressions	72
3.1.33. .spec.ingress[].from[].pods.namespaceSelector.matchExpressions[]	72
3.1.34. .spec.ingress[].from[].pods.podSelector	73
3.1.35. .spec.ingress[].from[].pods.podSelector.matchExpressions	73
3.1.36. .spec.ingress[].from[].pods.podSelector.matchExpressions[]	74
3.1.37. .spec.ingress[].ports	74
3.1.38. .spec.ingress[].ports[]	74
3.1.39. .spec.ingress[].ports[].portNumber	75
3.1.40. .spec.ingress[].ports[].portRange	76
3.1.41. .spec.subject	76
3.1.42. .spec.subject.namespaces	77
3.1.43. .spec.subject.namespaces.matchExpressions	77

3.1.44. .spec.subject.namespaces.matchExpressions[]	78
3.1.45. .spec.subject.pods	78
3.1.46. .spec.subject.pods.namespaceSelector	79
3.1.47. .spec.subject.pods.namespaceSelector.matchExpressions	79
3.1.48. .spec.subject.pods.namespaceSelector.matchExpressions[]	79
3.1.49. .spec.subject.pods.podSelector	80
3.1.50. .spec.subject.pods.podSelector.matchExpressions	81
3.1.51. .spec.subject.pods.podSelector.matchExpressions[]	81
3.1.52. .status	82
3.1.53. .status.conditions	83
3.1.54. .status.conditions[]	83
3.2. API ENDPOINTS	85
3.2.1. /apis/policy.networking.k8s.io/v1alpha1/adminnetworkpolicies	86
3.2.2. /apis/policy.networking.k8s.io/v1alpha1/adminnetworkpolicies/{name}	87
3.2.3. /apis/policy.networking.k8s.io/v1alpha1/adminnetworkpolicies/{name}/status	90
CHAPTER 4. ADMINPOLICYBASEDEXTERNALROUTE [K8S.OVN.ORG/V1]	94
4.1. SPECIFICATION	94
4.1.1. .spec	95
4.1.2. .spec.from	95
4.1.3. .spec.from.namespaceSelector	96
4.1.4. .spec.from.namespaceSelector.matchExpressions	96
4.1.5. .spec.from.namespaceSelector.matchExpressions[]	96
4.1.6. .spec.nextHops	97
4.1.7. .spec.nextHops.dynamic	98
4.1.8. .spec.nextHops.dynamic[]	98
4.1.9. .spec.nextHops.dynamic[].namespaceSelector	99
4.1.10. .spec.nextHops.dynamic[].namespaceSelector.matchExpressions	100
4.1.11. .spec.nextHops.dynamic[].namespaceSelector.matchExpressions[]	100
4.1.12. .spec.nextHops.dynamic[].podSelector	101
4.1.13. .spec.nextHops.dynamic[].podSelector.matchExpressions	101
4.1.14. .spec.nextHops.dynamic[].podSelector.matchExpressions[]	102
4.1.15. .spec.nextHops.static	102
4.1.16. .spec.nextHops.static[]	102
4.1.17. .status	103
4.2. API ENDPOINTS	103
4.2.1. /apis/k8s.ovn.org/v1/adminpolicybasedexternalroutes	104
4.2.2. /apis/k8s.ovn.org/v1/adminpolicybasedexternalroutes/{name}	105
4.2.3. /apis/k8s.ovn.org/v1/adminpolicybasedexternalroutes/{name}/status	108
CHAPTER 5. BASELINEADMINNETWORKPOLICY [POLICY.NETWORKING.K8S.IO/V1ALPHA1]	112
5.1. SPECIFICATION	112
5.1.1. .spec	113
5.1.2. .spec.egress	114
5.1.3. .spec.egress[]	114
5.1.4. .spec.egress[].ports	116
5.1.5. .spec.egress[].ports[]	116
5.1.6. .spec.egress[].ports[].portNumber	117
5.1.7. .spec.egress[].ports[].portRange	117
5.1.8. .spec.egress[].to	118
5.1.9. .spec.egress[].to[]	118
5.1.10. .spec.egress[].to[].namespaces	120
5.1.11. .spec.egress[].to[].namespaces.matchExpressions	120

5.1.12. .spec.egress[].to[].namespaces.matchExpressions[]	121
5.1.13. .spec.egress[].to[].nodes	121
5.1.14. .spec.egress[].to[].nodes.matchExpressions	122
5.1.15. .spec.egress[].to[].nodes.matchExpressions[]	122
5.1.16. .spec.egress[].to[].pods	123
5.1.17. .spec.egress[].to[].pods.namespaceSelector	123
5.1.18. .spec.egress[].to[].pods.namespaceSelector.matchExpressions	124
5.1.19. .spec.egress[].to[].pods.namespaceSelector.matchExpressions[]	124
5.1.20. .spec.egress[].to[].pods.podSelector	125
5.1.21. .spec.egress[].to[].pods.podSelector.matchExpressions	125
5.1.22. .spec.egress[].to[].pods.podSelector.matchExpressions[]	126
5.1.23. .spec.ingress	126
5.1.24. .spec.ingress[]	126
5.1.25. .spec.ingress[].from	128
5.1.26. .spec.ingress[].from[]	128
5.1.27. .spec.ingress[].from[].namespaces	129
5.1.28. .spec.ingress[].from[].namespaces.matchExpressions	129
5.1.29. .spec.ingress[].from[].namespaces.matchExpressions[]	130
5.1.30. .spec.ingress[].from[].pods	130
5.1.31. .spec.ingress[].from[].pods.namespaceSelector	131
5.1.32. .spec.ingress[].from[].pods.namespaceSelector.matchExpressions	131
5.1.33. .spec.ingress[].from[].pods.namespaceSelector.matchExpressions[]	132
5.1.34. .spec.ingress[].from[].pods.podSelector	132
5.1.35. .spec.ingress[].from[].pods.podSelector.matchExpressions	133
5.1.36. .spec.ingress[].from[].pods.podSelector.matchExpressions[]	133
5.1.37. .spec.ingress[].ports	134
5.1.38. .spec.ingress[].ports[]	134
5.1.39. .spec.ingress[].ports[].portNumber	135
5.1.40. .spec.ingress[].ports[].portRange	135
5.1.41. .spec.subject	136
5.1.42. .spec.subject.namespaces	136
5.1.43. .spec.subject.namespaces.matchExpressions	137
5.1.44. .spec.subject.namespaces.matchExpressions[]	137
5.1.45. .spec.subject.pods	138
5.1.46. .spec.subject.pods.namespaceSelector	138
5.1.47. .spec.subject.pods.namespaceSelector.matchExpressions	139
5.1.48. .spec.subject.pods.namespaceSelector.matchExpressions[]	139
5.1.49. .spec.subject.pods.podSelector	140
5.1.50. .spec.subject.pods.podSelector.matchExpressions	140
5.1.51. .spec.subject.pods.podSelector.matchExpressions[]	141
5.1.52. .status	141
5.1.53. .status.conditions	142
5.1.54. .status.conditions[]	142
5.2. API ENDPOINTS	144
5.2.1. /apis/policy.networking.k8s.io/v1alpha1/baselineadminnetworkpolicies	144
5.2.2. /apis/policy.networking.k8s.io/v1alpha1/baselineadminnetworkpolicies/{name}	146
5.2.3. /apis/policy.networking.k8s.io/v1alpha1/baselineadminnetworkpolicies/{name}/status	149
CHAPTER 6. CLOUDPRIVATEIPCONFIG [CLOUD.NETWORK.OPENSIFT.IO/V1]	153
6.1. SPECIFICATION	153
6.1.1. .spec	154
6.1.2. .status	154
6.1.3. .status.conditions	155

6.1.4. .status.conditions[]	155
6.2. API ENDPOINTS	156
6.2.1. /apis/cloud.network.openshift.io/v1/cloudprivateipconfigs	157
6.2.2. /apis/cloud.network.openshift.io/v1/cloudprivateipconfigs/{name}	158
6.2.3. /apis/cloud.network.openshift.io/v1/cloudprivateipconfigs/{name}/status	161
CHAPTER 7. EGRESSFIREWALL [K8S.OVN.ORG/V1]	165
7.1. SPECIFICATION	165
7.1.1. .spec	166
7.1.2. .spec.egress	166
7.1.3. .spec.egress[]	166
7.1.4. .spec.egress[].ports	167
7.1.5. .spec.egress[].ports[]	167
7.1.6. .spec.egress[].to	167
7.1.7. .spec.egress[].to.nodeSelector	168
7.1.8. .spec.egress[].to.nodeSelector.matchExpressions	169
7.1.9. .spec.egress[].to.nodeSelector.matchExpressions[]	169
7.1.10. .status	170
7.2. API ENDPOINTS	170
7.2.1. /apis/k8s.ovn.org/v1/egressfirewalls	171
7.2.2. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressfirewalls	171
7.2.3. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressfirewalls/{name}	173
7.2.4. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressfirewalls/{name}/status	176
CHAPTER 8. EGRESSIP [K8S.OVN.ORG/V1]	179
8.1. SPECIFICATION	179
8.1.1. .spec	180
8.1.2. .spec.namespaceSelector	180
8.1.3. .spec.namespaceSelector.matchExpressions	181
8.1.4. .spec.namespaceSelector.matchExpressions[]	181
8.1.5. .spec.podSelector	182
8.1.6. .spec.podSelector.matchExpressions	182
8.1.7. .spec.podSelector.matchExpressions[]	183
8.1.8. .status	183
8.1.9. .status.items	184
8.1.10. .status.items[]	184
8.2. API ENDPOINTS	184
8.2.1. /apis/k8s.ovn.org/v1/egressips	185
8.2.2. /apis/k8s.ovn.org/v1/egressips/{name}	186
CHAPTER 9. EGRESSQOS [K8S.OVN.ORG/V1]	190
9.1. SPECIFICATION	190
9.1.1. .spec	191
9.1.2. .spec.egress	191
9.1.3. .spec.egress[]	191
9.1.4. .spec.egress[].podSelector	192
9.1.5. .spec.egress[].podSelector.matchExpressions	192
9.1.6. .spec.egress[].podSelector.matchExpressions[]	193
9.1.7. .status	193
9.1.8. .status.conditions	194
9.1.9. .status.conditions[]	194
9.2. API ENDPOINTS	196
9.2.1. /apis/k8s.ovn.org/v1/egressqoses	197
9.2.2. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressqoses	197

9.2.3. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressqoses/{name}	199
9.2.4. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressqoses/{name}/status	202
CHAPTER 10. EGRESSSERVICE [K8S.OVN.ORG/V1]	205
10.1. SPECIFICATION	205
10.1.1. .spec	206
10.1.2. .spec.nodeSelector	206
10.1.3. .spec.nodeSelector.matchExpressions	207
10.1.4. .spec.nodeSelector.matchExpressions[]	207
10.1.5. .status	208
10.2. API ENDPOINTS	208
10.2.1. /apis/k8s.ovn.org/v1/egressservices	209
10.2.2. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressservices	209
10.2.3. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressservices/{name}	211
10.2.4. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressservices/{name}/status	214
CHAPTER 11. ENDPOINTS [V1]	218
11.1. SPECIFICATION	218
11.1.1. .subsets	219
11.1.2. .subsets[]	220
11.1.3. .subsets[].addresses	221
11.1.4. .subsets[].addresses[]	221
11.1.5. .subsets[].addresses[].targetRef	222
11.1.6. .subsets[].notReadyAddresses	223
11.1.7. .subsets[].notReadyAddresses[]	223
11.1.8. .subsets[].notReadyAddresses[].targetRef	224
11.1.9. .subsets[].ports	225
11.1.10. .subsets[].ports[]	225
11.2. API ENDPOINTS	227
11.2.1. /api/v1/endpoints	227
11.2.2. /api/v1/watch/endpoints	228
11.2.3. /api/v1/namespaces/{namespace}/endpoints	228
11.2.4. /api/v1/watch/namespaces/{namespace}/endpoints	230
11.2.5. /api/v1/namespaces/{namespace}/endpoints/{name}	231
11.2.6. /api/v1/watch/namespaces/{namespace}/endpoints/{name}	234
CHAPTER 12. ENDPOINTSLSICE [DISCOVERY.K8S.IO/V1]	235
12.1. SPECIFICATION	235
12.1.1. .endpoints	237
12.1.2. .endpoints[]	237
12.1.3. .endpoints[].conditions	238
12.1.4. .endpoints[].hints	239
12.1.5. .endpoints[].hints.forNodes	240
12.1.6. .endpoints[].hints.forNodes[]	240
12.1.7. .endpoints[].hints.forZones	240
12.1.8. .endpoints[].hints.forZones[]	241
12.1.9. .ports	241
12.1.10. .ports[]	241
12.2. API ENDPOINTS	243
12.2.1. /apis/discovery.k8s.io/v1/endpointslices	244
12.2.2. /apis/discovery.k8s.io/v1/watch/endpointslices	244
12.2.3. /apis/discovery.k8s.io/v1/namespaces/{namespace}/endpointslices	244
12.2.4. /apis/discovery.k8s.io/v1/watch/namespaces/{namespace}/endpointslices	246
12.2.5. /apis/discovery.k8s.io/v1/namespaces/{namespace}/endpointslices/{name}	247

12.2.6. /apis/discovery.k8s.io/v1/watch/namespaces/{namespace}/endpointslices/{name}	250
CHAPTER 13. EGRESSROUTER [NETWORK.OPERATOR.OPENSIFT.IO/V1]	251
13.1. SPECIFICATION	251
13.1.1. .spec	252
13.1.2. .spec.addresses	253
13.1.3. .spec.addresses[]	253
13.1.4. .spec.networkInterface	253
13.1.5. .spec.networkInterface.macvlan	254
13.1.6. .spec.redirect	254
13.1.7. .spec.redirect.redirectRules	255
13.1.8. .spec.redirect.redirectRules[]	255
13.1.9. .status	256
13.1.10. .status.conditions	256
13.1.11. .status.conditions[]	256
13.2. API ENDPOINTS	257
13.2.1. /apis/network.operator.openshift.io/v1/egressrouters	258
13.2.2. /apis/network.operator.openshift.io/v1/namespaces/{namespace}/egressrouters	258
13.2.3. /apis/network.operator.openshift.io/v1/namespaces/{namespace}/egressrouters/{name}	260
13.2.4. /apis/network.operator.openshift.io/v1/namespaces/{namespace}/egressrouters/{name}/status	263
CHAPTER 14. GRPCROUTE [GATEWAY.NETWORKING.K8S.IO/V1]	267
14.1. SPECIFICATION	267
14.1.1. .spec	268
14.1.2. .spec.parentRefs	272
14.1.3. .spec.parentRefs[]	273
14.1.4. .spec.rules	276
14.1.5. .spec.rules[]	277
14.1.6. .spec.rules[].backendRefs	282
14.1.7. .spec.rules[].backendRefs[]	283
14.1.8. .spec.rules[].backendRefs[].filters	286
14.1.9. .spec.rules[].backendRefs[].filters[]	286
14.1.10. .spec.rules[].backendRefs[].filters[].extensionRef	288
14.1.11. .spec.rules[].backendRefs[].filters[].requestHeaderModifier	289
14.1.12. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.add	291
14.1.13. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.add[]	291
14.1.14. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.set	292
14.1.15. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.set[]	292
14.1.16. .spec.rules[].backendRefs[].filters[].requestMirror	293
14.1.17. .spec.rules[].backendRefs[].filters[].requestMirror.backendRef	295
14.1.18. .spec.rules[].backendRefs[].filters[].requestMirror.fraction	297
14.1.19. .spec.rules[].backendRefs[].filters[].responseHeaderModifier	297
14.1.20. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.add	299
14.1.21. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.add[]	299
14.1.22. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.set	300
14.1.23. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.set[]	300
14.1.24. .spec.rules[].filters	301
14.1.25. .spec.rules[].filters[]	301
14.1.26. .spec.rules[].filters[].extensionRef	304
14.1.27. .spec.rules[].filters[].requestHeaderModifier	304
14.1.28. .spec.rules[].filters[].requestHeaderModifier.add	306
14.1.29. .spec.rules[].filters[].requestHeaderModifier.add[]	306
14.1.30. .spec.rules[].filters[].requestHeaderModifier.set	307

14.1.31. .spec.rules[].filters[].requestHeaderModifier.set[]	307
14.1.32. .spec.rules[].filters[].requestMirror	308
14.1.33. .spec.rules[].filters[].requestMirror.backendRef	310
14.1.34. .spec.rules[].filters[].requestMirror.fraction	312
14.1.35. .spec.rules[].filters[].responseHeaderModifier	312
14.1.36. .spec.rules[].filters[].responseHeaderModifier.add	314
14.1.37. .spec.rules[].filters[].responseHeaderModifier.add[]	314
14.1.38. .spec.rules[].filters[].responseHeaderModifier.set	315
14.1.39. .spec.rules[].filters[].responseHeaderModifier.set[]	315
14.1.40. .spec.rules[].matches	316
14.1.41. .spec.rules[].matches[]	317
14.1.42. .spec.rules[].matches[].headers	317
14.1.43. .spec.rules[].matches[].headers[]	318
14.1.44. .spec.rules[].matches[].method	318
14.1.45. .status	319
14.1.46. .status.parents	320
14.1.47. .status.parents[]	320
14.1.48. .status.parents[].conditions	322
14.1.49. .status.parents[].conditions[]	323
14.1.50. .status.parents[].parentRef	324
14.2. API ENDPOINTS	327
14.2.1. /apis/gateway.networking.k8s.io/v1/grpcroutes	328
14.2.2. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/grpcroutes	328
14.2.3. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/grpcroutes/{name}	330
14.2.4. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/grpcroutes/{name}/status	333
CHAPTER 15. GATEWAY [GATEWAY.NETWORKING.K8S.IO/V1]	337
15.1. SPECIFICATION	337
15.1.1. .spec	338
15.1.2. .spec.addresses	344
15.1.3. .spec.addresses[]	344
15.1.4. .spec.infrastructure	345
15.1.5. .spec.infrastructure.parametersRef	347
15.1.6. .spec.listeners	348
15.1.7. .spec.listeners[]	351
15.1.8. .spec.listeners[].allowedRoutes	355
15.1.9. .spec.listeners[].allowedRoutes.kinds	356
15.1.10. .spec.listeners[].allowedRoutes.kinds[]	357
15.1.11. .spec.listeners[].allowedRoutes.namespaces	357
15.1.12. .spec.listeners[].allowedRoutes.namespaces.selector	358
15.1.13. .spec.listeners[].allowedRoutes.namespaces.selector.matchExpressions	358
15.1.14. .spec.listeners[].allowedRoutes.namespaces.selector.matchExpressions[]	359
15.1.15. .spec.listeners[].tls	359
15.1.16. .spec.listeners[].tls.certificateRefs	362
15.1.17. .spec.listeners[].tls.certificateRefs[]	362
15.1.18. .status	363
15.1.19. .status.addresses	364
15.1.20. .status.addresses[]	365
15.1.21. .status.conditions	365
15.1.22. .status.conditions[]	366
15.1.23. .status.listeners	367
15.1.24. .status.listeners[]	367
15.1.25. .status.listeners[].conditions	369

15.1.26. .status.listeners[].conditions[]	369
15.1.27. .status.listeners[].supportedKinds	370
15.1.28. .status.listeners[].supportedKinds[]	371
15.2. API ENDPOINTS	371
15.2.1. /apis/gateway.networking.k8s.io/v1/gateways	372
15.2.2. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/gateways	372
15.2.3. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/gateways/{name}	374
15.2.4. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/gateways/{name}/status	377
CHAPTER 16. GATEWAYCLASS [GATEWAY.NETWORKING.K8S.IO/V1]	380
16.1. SPECIFICATION	380
16.1.1. .spec	381
16.1.2. .spec.parametersRef	382
16.1.3. .status	383
16.1.4. .status.conditions	384
16.1.5. .status.conditions[]	384
16.2. API ENDPOINTS	385
16.2.1. /apis/gateway.networking.k8s.io/v1/gatewayclasses	386
16.2.2. /apis/gateway.networking.k8s.io/v1/gatewayclasses/{name}	388
16.2.3. /apis/gateway.networking.k8s.io/v1/gatewayclasses/{name}/status	391
CHAPTER 17. HTTPROUTE [GATEWAY.NETWORKING.K8S.IO/V1]	394
17.1. SPECIFICATION	394
17.1.1. .spec	395
17.1.2. .spec.parentRefs	399
17.1.3. .spec.parentRefs[]	400
17.1.4. .spec.rules	403
17.1.5. .spec.rules[]	404
17.1.6. .spec.rules[].backendRefs	408
17.1.7. .spec.rules[].backendRefs[]	409
17.1.8. .spec.rules[].backendRefs[].filters	412
17.1.9. .spec.rules[].backendRefs[].filters[]	412
17.1.10. .spec.rules[].backendRefs[].filters[].extensionRef	415
17.1.11. .spec.rules[].backendRefs[].filters[].requestHeaderModifier	415
17.1.12. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.add	417
17.1.13. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.add[]	417
17.1.14. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.set	418
17.1.15. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.set[]	418
17.1.16. .spec.rules[].backendRefs[].filters[].requestMirror	419
17.1.17. .spec.rules[].backendRefs[].filters[].requestMirror.backendRef	421
17.1.18. .spec.rules[].backendRefs[].filters[].requestMirror.fraction	423
17.1.19. .spec.rules[].backendRefs[].filters[].requestRedirect	423
17.1.20. .spec.rules[].backendRefs[].filters[].requestRedirect.path	426
17.1.21. .spec.rules[].backendRefs[].filters[].responseHeaderModifier	428
17.1.22. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.add	429
17.1.23. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.add[]	430
17.1.24. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.set	430
17.1.25. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.set[]	430
17.1.26. .spec.rules[].backendRefs[].filters[].urlRewrite	431
17.1.27. .spec.rules[].backendRefs[].filters[].urlRewrite.path	432
17.1.28. .spec.rules[].filters	433
17.1.29. .spec.rules[].filters[]	434
17.1.30. .spec.rules[].filters[].extensionRef	437

17.1.31. .spec.rules[].filters[].requestHeaderModifier	437
17.1.32. .spec.rules[].filters[].requestHeaderModifier.add	439
17.1.33. .spec.rules[].filters[].requestHeaderModifier.add[]	439
17.1.34. .spec.rules[].filters[].requestHeaderModifier.set	440
17.1.35. .spec.rules[].filters[].requestHeaderModifier.set[]	440
17.1.36. .spec.rules[].filters[].requestMirror	441
17.1.37. .spec.rules[].filters[].requestMirror.backendRef	443
17.1.38. .spec.rules[].filters[].requestMirror.fraction	445
17.1.39. .spec.rules[].filters[].requestRedirect	445
17.1.40. .spec.rules[].filters[].requestRedirect.path	448
17.1.41. .spec.rules[].filters[].responseHeaderModifier	450
17.1.42. .spec.rules[].filters[].responseHeaderModifier.add	451
17.1.43. .spec.rules[].filters[].responseHeaderModifier.add[]	452
17.1.44. .spec.rules[].filters[].responseHeaderModifier.set	452
17.1.45. .spec.rules[].filters[].responseHeaderModifier.set[]	452
17.1.46. .spec.rules[].filters[].urlRewrite	453
17.1.47. .spec.rules[].filters[].urlRewrite.path	454
17.1.48. .spec.rules[].matches	455
17.1.49. .spec.rules[].matches[]	456
17.1.50. .spec.rules[].matches[].headers	458
17.1.51. .spec.rules[].matches[].headers[]	458
17.1.52. .spec.rules[].matches[].path	459
17.1.53. .spec.rules[].matches[].queryParams	459
17.1.54. .spec.rules[].matches[].queryParams[]	460
17.1.55. .spec.rules[].timeouts	462
17.1.56. .status	464
17.1.57. .status.parents	465
17.1.58. .status.parents[]	466
17.1.59. .status.parents[].conditions	467
17.1.60. .status.parents[].conditions[]	468
17.1.61. .status.parents[].parentRef	469
17.2. API ENDPOINTS	472
17.2.1. /apis/gateway.networking.k8s.io/v1/httproutes	473
17.2.2. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/httproutes	473
17.2.3. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/httproutes/{name}	475
17.2.4. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/httproutes/{name}/status	478
CHAPTER 18. INGRESS [NETWORKING.K8S.IO/V1]	482
18.1. SPECIFICATION	482
18.1.1. .spec	482
18.1.2. .spec.defaultBackend	484
18.1.3. .spec.defaultBackend.service	484
18.1.4. .spec.defaultBackend.service.port	485
18.1.5. .spec.rules	485
18.1.6. .spec.rules[]	485
18.1.7. .spec.rules[].http	487
18.1.8. .spec.rules[].http.paths	487
18.1.9. .spec.rules[].http.paths[]	487
18.1.10. .spec.rules[].http.paths[].backend	489
18.1.11. .spec.rules[].http.paths[].backend.service	490
18.1.12. .spec.rules[].http.paths[].backend.service.port	490
18.1.13. .spec.tls	491
18.1.14. .spec.tls[]	491

18.1.15. .status	491
18.1.16. .status.loadBalancer	492
18.1.17. .status.loadBalancer.ingress	492
18.1.18. .status.loadBalancer.ingress[]	492
18.1.19. .status.loadBalancer.ingress[].ports	493
18.1.20. .status.loadBalancer.ingress[].ports[]	493
18.2. API ENDPOINTS	494
18.2.1. /apis/networking.k8s.io/v1/ingresses	495
18.2.2. /apis/networking.k8s.io/v1/watch/ingresses	495
18.2.3. /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses	495
18.2.4. /apis/networking.k8s.io/v1/watch/namespaces/{namespace}/ingresses	497
18.2.5. /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}	498
18.2.6. /apis/networking.k8s.io/v1/watch/namespaces/{namespace}/ingresses/{name}	501
18.2.7. /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}/status	501
CHAPTER 19. INGRESSCLASS [NETWORKING.K8S.IO/V1]	505
19.1. SPECIFICATION	505
19.1.1. .spec	505
19.1.2. .spec.parameters	506
19.2. API ENDPOINTS	507
19.2.1. /apis/networking.k8s.io/v1/ingressclasses	507
19.2.2. /apis/networking.k8s.io/v1/watch/ingressclasses	509
19.2.3. /apis/networking.k8s.io/v1/ingressclasses/{name}	510
19.2.4. /apis/networking.k8s.io/v1/watch/ingressclasses/{name}	513
CHAPTER 20. IPADDRESS [NETWORKING.K8S.IO/V1]	514
20.1. SPECIFICATION	514
20.1.1. .spec	514
20.1.2. .spec.parentRef	515
20.2. API ENDPOINTS	515
20.2.1. /apis/networking.k8s.io/v1/ipaddresses	516
20.2.2. /apis/networking.k8s.io/v1/watch/ipaddresses	518
20.2.3. /apis/networking.k8s.io/v1/ipaddresses/{name}	518
20.2.4. /apis/networking.k8s.io/v1/watch/ipaddresses/{name}	521
CHAPTER 21. IPAMCLAIM [K8S.CNI.CNCF.IO/V1ALPHA1]	523
21.1. SPECIFICATION	523
21.1.1. .spec	523
21.1.2. .status	524
21.2. API ENDPOINTS	524
21.2.1. /apis/k8s.cni.cncf.io/v1alpha1/ipamclaims	525
21.2.2. /apis/k8s.cni.cncf.io/v1alpha1/namespaces/{namespace}/ipamclaims	525
21.2.3. /apis/k8s.cni.cncf.io/v1alpha1/namespaces/{namespace}/ipamclaims/{name}	527
21.2.4. /apis/k8s.cni.cncf.io/v1alpha1/namespaces/{namespace}/ipamclaims/{name}/status	530
CHAPTER 22. IPPOL [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]	534
22.1. SPECIFICATION	534
22.1.1. .spec	534
22.1.2. .spec.allocations	535
22.1.3. .spec.allocations{}	535
22.2. API ENDPOINTS	536
22.2.1. /apis/whereabouts.cni.cncf.io/v1alpha1/ippools	536
22.2.2. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/ippools	536
22.2.3. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/ippools/{name}	538

CHAPTER 23. MULTINETWORKPOLICY [K8S.CNI.CNCF.IO/V1BETA1]	542
23.1. SPECIFICATION	542
23.1.1. .spec	542
23.1.2. .spec.egress	545
23.1.3. .spec.egress[]	545
23.1.4. .spec.egress[].ports	546
23.1.5. .spec.egress[].ports[]	546
23.1.6. .spec.egress[].to	547
23.1.7. .spec.egress[].to[]	547
23.1.8. .spec.egress[].to[].ipBlock	548
23.1.9. .spec.egress[].to[].namespaceSelector	549
23.1.10. .spec.egress[].to[].namespaceSelector.matchExpressions	549
23.1.11. .spec.egress[].to[].namespaceSelector.matchExpressions[]	550
23.1.12. .spec.egress[].to[].podSelector	550
23.1.13. .spec.egress[].to[].podSelector.matchExpressions	551
23.1.14. .spec.egress[].to[].podSelector.matchExpressions[]	551
23.1.15. .spec.ingress	552
23.1.16. .spec.ingress[]	552
23.1.17. .spec.ingress[].from	553
23.1.18. .spec.ingress[].from[]	553
23.1.19. .spec.ingress[].from[].ipBlock	554
23.1.20. .spec.ingress[].from[].namespaceSelector	555
23.1.21. .spec.ingress[].from[].namespaceSelector.matchExpressions	555
23.1.22. .spec.ingress[].from[].namespaceSelector.matchExpressions[]	556
23.1.23. .spec.ingress[].from[].podSelector	556
23.1.24. .spec.ingress[].from[].podSelector.matchExpressions	557
23.1.25. .spec.ingress[].from[].podSelector.matchExpressions[]	557
23.1.26. .spec.ingress[].ports	558
23.1.27. .spec.ingress[].ports[]	558
23.1.28. .spec.podSelector	559
23.1.29. .spec.podSelector.matchExpressions	559
23.1.30. .spec.podSelector.matchExpressions[]	559
23.2. API ENDPOINTS	560
23.2.1. /apis/k8s.cni.cncf.io/v1beta1/multi-networkpolicies	561
23.2.2. /apis/k8s.cni.cncf.io/v1beta1/namespaces/{namespace}/multi-networkpolicies	561
23.2.3. /apis/k8s.cni.cncf.io/v1beta1/namespaces/{namespace}/multi-networkpolicies/{name}	563
CHAPTER 24. NETWORKATTACHMENTDEFINITION [K8S.CNI.CNCF.IO/V1]	567
24.1. SPECIFICATION	567
24.1.1. .spec	567
24.2. API ENDPOINTS	568
24.2.1. /apis/k8s.cni.cncf.io/v1/network-attachment-definitions	568
24.2.2. /apis/k8s.cni.cncf.io/v1/namespaces/{namespace}/network-attachment-definitions	569
24.2.3. /apis/k8s.cni.cncf.io/v1/namespaces/{namespace}/network-attachment-definitions/{name}	570
CHAPTER 25. NETWORKPOLICY [NETWORKING.K8S.IO/V1]	574
25.1. SPECIFICATION	574
25.1.1. .spec	574
25.1.2. .spec.egress	576
25.1.3. .spec.egress[]	577
25.1.4. .spec.egress[].ports	578
25.1.5. .spec.egress[].ports[]	578
25.1.6. .spec.egress[].to	578

25.1.7. .spec.egress[].to[]	579
25.1.8. .spec.egress[].to[].ipBlock	580
25.1.9. .spec.ingress	580
25.1.10. .spec.ingress[]	581
25.1.11. .spec.ingress[].from	582
25.1.12. .spec.ingress[].from[]	582
25.1.13. .spec.ingress[].from[].ipBlock	583
25.1.14. .spec.ingress[].ports	583
25.1.15. .spec.ingress[].ports[]	584
25.2. API ENDPOINTS	584
25.2.1. /apis/networking.k8s.io/v1/networkpolicies	585
25.2.2. /apis/networking.k8s.io/v1/watch/networkpolicies	585
25.2.3. /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies	586
25.2.4. /apis/networking.k8s.io/v1/watch/namespaces/{namespace}/networkpolicies	588
25.2.5. /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies/{name}	588
25.2.6. /apis/networking.k8s.io/v1/watch/namespaces/{namespace}/networkpolicies/{name}	591
CHAPTER 26. NODESLICEPOOL [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]	593
26.1. SPECIFICATION	593
26.1.1. .spec	593
26.1.2. .status	594
26.1.3. .status.allocations	594
26.1.4. .status.allocations[]	595
26.2. API ENDPOINTS	595
26.2.1. /apis/whereabouts.cni.cncf.io/v1alpha1/nodeslice pools	595
26.2.2. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/nodeslice pools	596
26.2.3. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/nodeslice pools/{name}	597
CHAPTER 27. OVERLAPPINGRANGEIPRESERVATION [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]	601
27.1. SPECIFICATION	601
27.1.1. .spec	602
27.2. API ENDPOINTS	602
27.2.1. /apis/whereabouts.cni.cncf.io/v1alpha1/overlappingrangeipreservations	602
27.2.2. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/overlappingrangeipreservations	603
27.2.3. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/overlappingrangeipreservations/{name}	604
CHAPTER 28. PODNETWORKCONNECTIVITYCHECK [CONTROLPLANE.OPERATOR.OPENSIFT.IO/V1ALPHA1]	608
28.1. SPECIFICATION	608
28.1.1. .spec	609
28.1.2. .spec.tlsClientCert	609
28.1.3. .status	610
28.1.4. .status.conditions	611
28.1.5. .status.conditions[]	611
28.1.6. .status.failures	611
28.1.7. .status.failures[]	611
28.1.8. .status.outages	612
28.1.9. .status.outages[]	612
28.1.10. .status.outages[].endLogs	613
28.1.11. .status.outages[].endLogs[]	613
28.1.12. .status.outages[].startLogs	614
28.1.13. .status.outages[].startLogs[]	614

28.1.14. .status.successes	615
28.1.15. .status.successes[]	615
28.2. API ENDPOINTS	615
28.2.1. /apis/controlplane.operator.openshift.io/v1alpha1/podnetworkconnectivitychecks	616
28.2.2. /apis/controlplane.operator.openshift.io/v1alpha1/namespaces/{namespace}/podnetworkconnectivitycheck	616
28.2.3. /apis/controlplane.operator.openshift.io/v1alpha1/namespaces/{namespace}/podnetworkconnectivitycheck	618
28.2.4. /apis/controlplane.operator.openshift.io/v1alpha1/namespaces/{namespace}/podnetworkconnectivitycheck	621
CHAPTER 29. REFERENCEGRANT [GATEWAY.NETWORKING.K8S.IO/V1BETA1]	625
29.1. SPECIFICATION	625
29.1.1. .spec	626
29.1.2. .spec.from	627
29.1.3. .spec.from[]	627
29.1.4. .spec.to	628
29.1.5. .spec.to[]	628
29.2. API ENDPOINTS	629
29.2.1. /apis/gateway.networking.k8s.io/v1beta1/referencegrants	630
29.2.2. /apis/gateway.networking.k8s.io/v1beta1/namespaces/{namespace}/referencegrants	630
29.2.3. /apis/gateway.networking.k8s.io/v1beta1/namespaces/{namespace}/referencegrants/{name}	632
CHAPTER 30. ROUTE [ROUTE.OPENSIFT.IO/V1]	636
30.1. SPECIFICATION	636
30.1.1. .spec	638
30.1.2. .spec.alternateBackends	640
30.1.3. .spec.alternateBackends[]	641
30.1.4. .spec.httpHeaders	641
30.1.5. .spec.httpHeaders.actions	642
30.1.6. .spec.httpHeaders.actions.request	644
30.1.7. .spec.httpHeaders.actions.request[]	645
30.1.8. .spec.httpHeaders.actions.request[].action	645
30.1.9. .spec.httpHeaders.actions.request[].action.set	646
30.1.10. .spec.httpHeaders.actions.response	647
30.1.11. .spec.httpHeaders.actions.response[]	647
30.1.12. .spec.httpHeaders.actions.response[].action	648
30.1.13. .spec.httpHeaders.actions.response[].action.set	648
30.1.14. .spec.port	649
30.1.15. .spec.tls	649
30.1.16. .spec.tls.externalCertificate	651
30.1.17. .spec.to	652
30.1.18. .status	652
30.1.19. .status.ingress	653
30.1.20. .status.ingress[]	653
30.1.21. .status.ingress[].conditions	654
30.1.22. .status.ingress[].conditions[]	654
30.2. API ENDPOINTS	655
30.2.1. /apis/route.openshift.io/v1/routes	656
30.2.2. /apis/route.openshift.io/v1/watch/routes	656
30.2.3. /apis/route.openshift.io/v1/namespaces/{namespace}/routes	656
30.2.4. /apis/route.openshift.io/v1/watch/namespaces/{namespace}/routes	658
30.2.5. /apis/route.openshift.io/v1/namespaces/{namespace}/routes/{name}	659
30.2.6. /apis/route.openshift.io/v1/watch/namespaces/{namespace}/routes/{name}	662

30.2.7. /apis/route.openshift.io/v1/namespaces/{namespace}/routes/{name}/status	662
CHAPTER 31. SERVICE [V1]	666
31.1. SPECIFICATION	666
31.1.1. .spec	667
31.1.2. .spec.ports	678
31.1.3. .spec.ports[]	679
31.1.4. .spec.sessionAffinityConfig	682
31.1.5. .spec.sessionAffinityConfig.clientIP	682
31.1.6. .status	682
31.1.7. .status.loadBalancer	682
31.1.8. .status.loadBalancer.ingress	683
31.1.9. .status.loadBalancer.ingress[]	683
31.1.10. .status.loadBalancer.ingress[].ports	684
31.1.11. .status.loadBalancer.ingress[].ports[]	684
31.2. API ENDPOINTS	685
31.2.1. /api/v1/services	686
31.2.2. /api/v1/watch/services	686
31.2.3. /api/v1/namespaces/{namespace}/services	687
31.2.4. /api/v1/watch/namespaces/{namespace}/services	689
31.2.5. /api/v1/namespaces/{namespace}/services/{name}	689
31.2.6. /api/v1/watch/namespaces/{namespace}/services/{name}	692
31.2.7. /api/v1/namespaces/{namespace}/services/{name}/status	693
CHAPTER 32. SERVICECIDR [NETWORKING.K8S.IO/V1]	696
32.1. SPECIFICATION	696
32.1.1. .spec	696
32.1.2. .status	697
32.2. API ENDPOINTS	697
32.2.1. /apis/networking.k8s.io/v1/servicecidrs	698
32.2.2. /apis/networking.k8s.io/v1/watch/servicecidrs	700
32.2.3. /apis/networking.k8s.io/v1/servicecidrs/{name}	700
32.2.4. /apis/networking.k8s.io/v1/watch/servicecidrs/{name}	703
32.2.5. /apis/networking.k8s.io/v1/servicecidrs/{name}/status	704
CHAPTER 33. IPADDRESSCLAIM [IPAM.CLUSTER.X-K8S.IO/V1BETA1]	707
33.1. SPECIFICATION	707
33.1.1. .spec	707
33.1.2. .spec.poolRef	708
33.1.3. .status	708
33.1.4. .status.addressRef	709
33.1.5. .status.conditions	709
33.1.6. .status.conditions[]	710
33.1.7. .status.v1beta2	711
33.1.8. .status.v1beta2.conditions	711
33.1.9. .status.v1beta2.conditions[]	711
33.2. API ENDPOINTS	712
33.2.1. /apis/ipam.cluster.x-k8s.io/v1beta1/ipaddressclaims	713
33.2.2. /apis/ipam.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/ipaddressclaims	713
33.2.3. /apis/ipam.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/ipaddressclaims/{name}	715
33.2.4. /apis/ipam.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/ipaddressclaims/{name}/status	718
CHAPTER 34. USERDEFINEDNETWORK [K8S.OVN.ORG/V1]	722
34.1. SPECIFICATION	722

34.1.1. .spec	723
34.1.2. .spec.layer2	723
34.1.3. .spec.layer2.ipam	724
34.1.4. .spec.layer3	725
34.1.5. .spec.layer3.subnets	727
34.1.6. .spec.layer3.subnets[]	727
34.1.7. .status	727
34.1.8. .status.conditions	728
34.1.9. .status.conditions[]	728
34.2. API ENDPOINTS	729
34.2.1. /apis/k8s.ovn.org/v1/userdefinednetworks	730
34.2.2. /apis/k8s.ovn.org/v1/namespaces/{namespace}/userdefinednetworks	730
34.2.3. /apis/k8s.ovn.org/v1/namespaces/{namespace}/userdefinednetworks/{name}	732
34.2.4. /apis/k8s.ovn.org/v1/namespaces/{namespace}/userdefinednetworks/{name}/status	735

CHAPTER 1. NETWORK APIS

1.1. CLUSTERUSERDEFINEDNETWORK [K8S.OVN.ORG/V1]

Description

ClusterUserDefinedNetwork describe network request for a shared network across namespaces.

Type

object

1.2. ADMINNETWORKPOLICY [POLICY.NETWORKING.K8S.IO/V1ALPHA1]

Description

AdminNetworkPolicy is a cluster level resource that is part of the AdminNetworkPolicy API.

Type

object

1.3. ADMINPOLICYBASEDEXTERNALROUTE [K8S.OVN.ORG/V1]

Description

AdminPolicyBasedExternalRoute is a CRD allowing the cluster administrators to configure policies for external gateway IPs to be applied to all the pods contained in selected namespaces. Egress traffic from the pods that belong to the selected namespaces to outside the cluster is routed through these external gateway IPs.

Type

object

1.4. BASELINEADMINNETWORKPOLICY [POLICY.NETWORKING.K8S.IO/V1ALPHA1]

Description

BaselineAdminNetworkPolicy is a cluster level resource that is part of the AdminNetworkPolicy API.

Type

object

1.5. CLOUDPRIVATEIPCONFIG [CLOUD.NETWORK.OPENSIFT.IO/V1]

Description

CloudPrivateIPConfig performs an assignment of a private IP address to the primary NIC associated with cloud VMs. This is done by specifying the IP and Kubernetes node which the IP should be assigned to. This CRD is intended to be used by the network plugin which manages the cluster network. The spec side represents the desired state requested by the network plugin, and the status side represents the current state that this CRD's controller has executed. No users will have permission to modify it, and if a cluster-admin decides to edit it for some reason, their changes will be overwritten the next time the network plugin reconciles the object. Note: the CR's name must specify the requested private IP address (can be IPv4 or IPv6).

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. EGRESSFIREWALL [K8S.OVN.ORG/V1]

Description

EgressFirewall describes the current egress firewall for a Namespace. Traffic from a pod to an IP address outside the cluster will be checked against each EgressFirewallRule in the pod's namespace's EgressFirewall, in order. If no rule matches (or no EgressFirewall is present) then the traffic will be allowed by default.

Type

object

1.7. EGRESSIP [K8S.OVN.ORG/V1]

Description

EgressIP is a CRD allowing the user to define a fixed source IP for all egress traffic originating from any pods which match the EgressIP resource according to its spec definition.

Type

object

1.8. EGRESSQOS [K8S.OVN.ORG/V1]

Description

EgressQoS is a CRD that allows the user to define a DSCP value for pods egress traffic on its namespace to specified CIDRs. Traffic from these pods will be checked against each EgressQoSRule in the namespace's EgressQoS, and if there is a match the traffic is marked with the relevant DSCP value.

Type

object

1.9. EGRESSSERVICE [K8S.OVN.ORG/V1]

Description

EgressService is a CRD that allows the user to request that the source IP of egress packets originating from all of the pods that are endpoints of the corresponding LoadBalancer Service would be its ingress IP. In addition, it allows the user to request that egress packets originating from all of the pods that are endpoints of the LoadBalancer service would use a different network than the main one.

Type

object

1.10. ENDPOINTS [V1]

Description

Endpoints is a collection of endpoints that implement the actual service. Example:

```
Name: "mysvc",
Subsets: [
  {
    Addresses: [{"ip": "10.10.1.1"}, {"ip": "10.10.2.2"}],
    Ports: [{"name": "a", "port": 8675}, {"name": "b", "port": 309}]
  },
  {
    Addresses: [{"ip": "10.10.3.3"}],
    Ports: [{"name": "a", "port": 93}, {"name": "b", "port": 76}]
  },
]
```

Endpoints is a legacy API and does not contain information about all Service features. Use `discoveryv1.EndpointSlice` for complete information about Service endpoints.

Deprecated: This API is deprecated in v1.33+. Use `discoveryv1.EndpointSlice`.

Type

object

1.11. ENDPOINTSlices [DISCOVERY.K8S.IO/V1]

Description

EndpointSlice represents a set of service endpoints. Most EndpointSlices are created by the EndpointSlice controller to represent the Pods selected by Service objects. For a given service there may be multiple EndpointSlice objects which must be joined to produce the full set of endpoints; you can find all of the slices for a given service by listing EndpointSlices in the service's namespace whose **kubernetes.io/service-name** label contains the service's name.

Type

object

1.12. EGRESSROUTER [NETWORK.OPERATOR.OPENSIFT.IO/V1]

Description

EgressRouter is a feature allowing the user to define an egress router that acts as a bridge between pods and external systems. The egress router runs a service that redirects egress traffic originating from a pod or a group of pods to a remote external system or multiple destinations as per configuration.

It is consumed by the cluster-network-operator. More specifically, given an EgressRouter CR with `<name>`, the CNO will create and manage: - A service called `<name>` - An egress pod called `<name>` - A NAD called `<name>`

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

EgressRouter is a single egressrouter pod configuration object.

Type

object

1.13. GRPCROUTE [GATEWAY.NETWORKING.K8S.IO/V1]

Description

GRPCRoute provides a way to route gRPC requests. This includes the capability to match requests by hostname, gRPC service, gRPC method, or HTTP/2 header. Filters can be used to specify additional processing steps. Backends specify where matching requests will be routed.

GRPCRoute falls under extended support within the Gateway API. Within the following specification, the word "MUST" indicates that an implementation supporting GRPCRoute must conform to the indicated requirement, but an implementation not supporting this route type need not follow the requirement unless explicitly indicated.

Implementations supporting **GRPCRoute** with the **HTTPS ProtocolType** MUST accept HTTP/2 connections without an initial upgrade from HTTP/1.1, i.e. via ALPN. If the implementation does not support this, then it MUST set the "Accepted" condition to "False" for the affected listener with a reason of "UnsupportedProtocol". Implementations MAY also accept HTTP/2 connections with an upgrade from HTTP/1.

Implementations supporting **GRPCRoute** with the **HTTP ProtocolType** MUST support HTTP/2 over cleartext TCP (h2c, <https://www.rfc-editor.org/rfc/rfc7540#section-3.1>) without an initial upgrade from HTTP/1.1, i.e. with prior knowledge (<https://www.rfc-editor.org/rfc/rfc7540#section-3.4>). If the implementation does not support this, then it MUST set the "Accepted" condition to "False" for the affected listener with a reason of "UnsupportedProtocol". Implementations MAY also accept HTTP/2 connections with an upgrade from HTTP/1, i.e. without prior knowledge.

Type

object

1.14. GATEWAY [GATEWAY.NETWORKING.K8S.IO/V1]

Description

Gateway represents an instance of a service-traffic handling infrastructure by binding Listeners to a set of IP addresses.

Type

object

1.15. GATEWAYCLASS [GATEWAY.NETWORKING.K8S.IO/V1]

Description

GatewayClass describes a class of Gateways available to the user for creating Gateway resources. It is recommended that this resource be used as a template for Gateways. This means that a Gateway is based on the state of the GatewayClass at the time it was created and changes to the GatewayClass or associated parameters are not propagated down to existing Gateways. This recommendation is intended to limit the blast radius of changes to GatewayClass or associated parameters. If implementations choose to propagate GatewayClass changes to existing Gateways, that MUST be clearly documented by the implementation.

Whenever one or more Gateways are using a GatewayClass, implementations SHOULD add the **gateway-exists-finalizer.gateway.networking.k8s.io** finalizer on the associated GatewayClass. This ensures that a GatewayClass associated with a Gateway is not deleted while in use.

GatewayClass is a Cluster level resource.

Type

object

1.16. HTTPROUTE [GATEWAY.NETWORKING.K8S.IO/V1]

Description

HTTPRoute provides a way to route HTTP requests. This includes the capability to match requests by hostname, path, header, or query param. Filters can be used to specify additional processing steps. Backends specify where matching requests should be routed.

Type

object

1.17. INGRESS [NETWORKING.K8S.IO/V1]

Description

Ingress is a collection of rules that allow inbound connections to reach the endpoints defined by a backend. An Ingress can be configured to give services externally-reachable urls, load balance traffic, terminate SSL, offer name based virtual hosting etc.

Type

object

1.18. INGRESSCLASS [NETWORKING.K8S.IO/V1]

Description

IngressClass represents the class of the Ingress, referenced by the Ingress Spec. The **ingressclass.kubernetes.io/is-default-class** annotation can be used to indicate that an IngressClass should be considered default. When a single IngressClass resource has this annotation set to true, new Ingress resources without a class specified will be assigned this default class.

Type

object

1.19. IPADDRESS [NETWORKING.K8S.IO/V1]

Description

IPAddress represents a single IP of a single IP Family. The object is designed to be used by APIs that operate on IP addresses. The object is used by the Service core API for allocation of IP addresses. An IP address can be represented in different formats, to guarantee the uniqueness of the IP, the name of the object is the IP address in canonical format, four decimal digits separated by dots suppressing leading zeros for IPv4 and the representation defined by RFC 5952 for IPv6. Valid: 192.168.1.5 or 2001:db8::1 or 2001:db8:aaaa:bbbb:cccc:dddd:eeee:1 Invalid: 10.01.2.3 or 2001:db8:0:0:0:0:1

Type

object

1.20. IPAMCLAIM [K8S.CNI.CNCF.IO/V1ALPHA1]

Description

IPAMClaim is the Schema for the IPAMClaim API

Type

object

1.21. IPPOL [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]

Description

IPPool is the Schema for the ippools API

Type

object

1.22. MULTINETWORKPOLICY [K8S.CNI.CNCF.IO/V1BETA1]

Description

MultiNetworkPolicy is a CRD schema to provide NetworkPolicy mechanism for net-attach-def which is specified by the Network Plumbing Working Group. MultiNetworkPolicy is identical to Kubernetes NetworkPolicy, See: <https://kubernetes.io/docs/concepts/services-networking/network-policies/> .

Type

object

1.23. NETWORKATTACHMENTDEFINITION [K8S.CNI.CNCF.IO/V1]

Description

NetworkAttachmentDefinition is a CRD schema specified by the Network Plumbing Working Group to express the intent for attaching pods to one or more logical or physical networks. More information available at: <https://github.com/k8snetworkplumbingwg/multi-net-spec>

Type

object

1.24. NETWORKPOLICY [NETWORKING.K8S.IO/V1]

Description

NetworkPolicy describes what network traffic is allowed for a set of Pods

Type

object

1.25. NODESLICEPOOL [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]

Description

NodeSlicePool is the Schema for the nodesliceippools API

Type

object

1.26. OVERLAPPINGRANGEIPRESERVATION [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]

Description

OverlappingRangeIPReservation is the Schema for the OverlappingRangeIPReservations API

Type

object

1.27. PODNETWORKCONNECTIVITYCHECK [CONTROLPLANE.OPERATOR.OPENSIFT.IO/V1ALPHA1]

Description

PodNetworkConnectivityCheck

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

1.28. REFERENCEGRANT [GATEWAY.NETWORKING.K8S.IO/V1BETA1]

Description

ReferenceGrant identifies kinds of resources in other namespaces that are trusted to reference the specified kinds of resources in the same namespace as the policy.

Each ReferenceGrant can be used to represent a unique trust relationship. Additional Reference Grants can be used to add to the set of trusted sources of inbound references for the namespace they are defined within.

All cross-namespace references in Gateway API (with the exception of cross-namespace Gateway-route attachment) require a ReferenceGrant.

ReferenceGrant is a form of runtime verification allowing users to assert which cross-namespace object references are permitted. Implementations that support ReferenceGrant **MUST NOT** permit cross-namespace references which have no grant, and **MUST** respond to the removal of a grant by revoking the access that the grant allowed.

Type

object

1.29. ROUTE [ROUTE.OPENSIFT.IO/V1]

Description

A route allows developers to expose services through an HTTP(S) aware load balancing and proxy layer via a public DNS entry. The route may further specify TLS options and a certificate, or specify a public CNAME that the router should also accept for HTTP and HTTPS traffic. An administrator typically configures their router to be visible outside the cluster firewall, and may also add additional security, caching, or traffic controls on the service content. Routers usually talk directly to the service endpoints.

Once a route is created, the **host** field may not be changed. Generally, routers use the oldest route with a given host when resolving conflicts.

Routers are subject to additional customization and may support additional controls via the annotations field.

Because administrators may configure multiple routers, the route status field is used to return

information to clients about the names and states of the route under each router. If a client chooses a duplicate name, for instance, the route status conditions are used to indicate the route cannot be chosen.

To enable HTTP/2 ALPN on a route it requires a custom (non-wildcard) certificate. This prevents connection coalescing by clients, notably web browsers. We do not support HTTP/2 ALPN on routes that use the default certificate because of the risk of connection re-use/coalescing. Routes that do not have their own custom certificate will not be HTTP/2 ALPN-enabled on either the frontend or the backend.

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

Type

object

1.30. SERVICE [V1]

Description

Service is a named abstraction of software service (for example, mysql) consisting of local port (for example 3306) that the proxy listens on, and the selector that determines which pods will answer requests sent through the proxy.

Type

object

1.31. SERVICECIDR [NETWORKING.K8S.IO/V1]

Description

ServiceCIDR defines a range of IP addresses using CIDR format (e.g. 192.168.0.0/24 or 2001:db2::/64). This range is used to allocate ClusterIPs to Service objects.

Type

object

1.32. IPADDRESSCLAIM [IPAM.CLUSTER.X-K8S.IO/V1BETA1]

Description

IPAddressClaim is the Schema for the ipaddressclaim API.

Type

object

1.33. USERDEFINEDNETWORK [K8S.OVN.ORG/V1]

Description

UserDefinedNetwork describe network request for a Namespace.

Type

object

CHAPTER 2. CLUSTERUSERDEFINEDNETWORK

[K8S.OVN.ORG/V1]

Description

ClusterUserDefinedNetwork describe network request for a shared network across namespaces.

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	ClusterUserDefinedNetworkSpec defines the desired state of ClusterUserDefinedNetwork.

Property	Type	Description
status	object	ClusterUserDefinedNetworkStatus contains the observed status of the ClusterUserDefinedNetwork.

2.1.1. .spec

Description

ClusterUserDefinedNetworkSpec defines the desired state of ClusterUserDefinedNetwork.

Type

object

Required

- **namespaceSelector**
- **network**

Property	Type	Description
namespaceSelector	object	NamespaceSelector Label selector for which namespace network should be available for.
network	object	Network is the user-defined-network spec

2.1.2. .spec.namespaceSelector

Description

NamespaceSelector Label selector for which namespace network should be available for.

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.

2.1.3. .spec.namespaceSelector.matchExpressions

Description

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

Type

array

2.1.4. .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.

2.1.5. .spec.network

Description

Network is the user-defined-network spec

Type

object

Required

- **topology**

Property	Type	Description
layer2	object	Layer2 is the Layer2 topology configuration.
layer3	object	Layer3 is the Layer3 topology configuration.
localnet	object	Localnet is the Localnet topology configuration.

Property	Type	Description
topology	string	Topology describes network configuration. Allowed values are "Layer3", "Layer2" and "Localnet". Layer3 topology creates a layer 2 segment per node, each with a different subnet. Layer 3 routing is used to interconnect node subnets. Layer2 topology creates one logical switch shared by all nodes. Localnet topology is based on layer 2 topology, but also allows connecting to an existent (configured) physical network to provide north-south traffic to the workloads.

2.1.6. .spec.network.layer2

Description

Layer2 is the Layer2 topology configuration.

Type

object

Required

- **role**

Property	Type	Description
ipam	object	IPAM section contains IPAM-related configuration for the network.

Property	Type	Description
joinSubnets	array (string)	JoinSubnets are used inside the OVN network topology. Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed. This field is only allowed for "Primary" network. It is not recommended to set this field without explicit need and understanding of the OVN network topology. When omitted, the platform will choose a reasonable default which is subject to change over time.
mtu	integer	MTU is the maximum transmission unit for a network. MTU is optional, if not provided, the globally configured value in OVN-Kubernetes (defaults to 1400) is used for the network.
role	string	Role describes the network role in the pod. Allowed value is "Secondary". Secondary network is only assigned to pods that use k8s.v1.cni.cncf.io/networks annotation to select given network.
subnets	array (string)	Subnets are used for the pod network across the cluster. Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed. The format should match standard CIDR notation (for example, "10.128.0.0/16"). This field must be omitted if ipam.mode is Disabled.

2.1.7. .spec.network.layer2.ipam

Description

IPAM section contains IPAM-related configuration for the network.

Type
object

Property	Type	Description
lifecycle	string	Lifecycle controls IP addresses management lifecycle. The only allowed value is Persistent. When set, the IP addresses assigned by OVN Kubernetes will be persisted in an ipamclaims.k8s.cni.cncf.io object. These IP addresses will be reused by other pods if requested. Only supported when mode is Enabled.
mode	string	Mode controls how much of the IP configuration will be managed by OVN. Enabled means OVN-Kubernetes will apply IP configuration to the SDN infrastructure and it will also assign IPs from the selected subnet to the individual pods. Disabled means OVN-Kubernetes will only assign MAC addresses and provide layer 2 communication, letting users configure IP addresses for the pods. Disabled is only available for Secondary networks. By disabling IPAM, any Kubernetes features that rely on selecting pods by IP will no longer function (such as network policy, services, etc). Additionally, IP port security will also be disabled for interfaces attached to this network. Defaults to Enabled.

2.1.8. .spec.network.layer3

Description

Layer3 is the Layer3 topology configuration.

Type
object

Required

- **role**

- **subnets**

Property	Type	Description
joinSubnets	array (string)	JoinSubnets are used inside the OVN network topology. Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed. This field is only allowed for "Primary" network. It is not recommended to set this field without explicit need and understanding of the OVN network topology. When omitted, the platform will choose a reasonable default which is subject to change over time.
mtu	integer	MTU is the maximum transmission unit for a network. MTU is optional, if not provided, the globally configured value in OVN-Kubernetes (defaults to 1400) is used for the network.
role	string	Role describes the network role in the pod. Allowed values are "Primary" and "Secondary". Primary network is automatically assigned to every pod created in the same namespace. Secondary network is only assigned to pods that use k8s.v1.cni.cncf.io/networks annotation to select given network.
subnets	array	Subnets are used for the pod network across the cluster. Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed. Given subnet is split into smaller subnets for every node.
subnets[]	object	

2.1.9. .spec.network.layer3.subnets

Description

Subnets are used for the pod network across the cluster.

Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed.

Given subnet is split into smaller subnets for every node.

Type

array

2.1.10. .spec.network.layer3.subnets[]

Description

Type

object

Required

- **cidr**

Property	Type	Description
cidr	string	CIDR specifies L3Subnet, which is split into smaller subnets for every node.
hostSubnet	integer	HostSubnet specifies the subnet size for every node. When not set, it will be assigned automatically.

2.1.11. .spec.network.localnet

Description

Localnet is the Localnet topology configuration.

Type

object

Required

- **physicalNetworkName**
- **role**

Property	Type	Description

Property	Type	Description
excludeSubnets	array (string)	excludeSubnets is a list of CIDRs to be removed from the specified CIDRs in subnets. The CIDRs in this list must be in range of at least one subnet specified in subnets. excludeSubnets is optional. When omitted no IP address is excluded and all IP addresses specified in subnets are subject to assignment. The format should match standard CIDR notation (for example, "10.128.0.0/16"). This field must be omitted if subnets is unset or ipam.mode is Disabled. When physicalNetworkName points to OVS bridge mapping of a network with reserved IP addresses (which shouldn't be assigned by OVN-Kubernetes), the specified CIDRs will not be assigned. For example: Given: subnets: "10.0.0.0/24", excludeSubnets: "10.0.0.200/30", the following addresses will not be assigned to pods: 10.0.0.201, 10.0.0.202.

Property	Type	Description
ipam	object	ipam configurations for the network. ipam is optional. When omitted, subnets must be specified. When ipam.mode is Disabled, subnets must be omitted. ipam.mode controls how much of the IP configuration will be managed by OVN. When Enabled, OVN-Kubernetes will apply IP configuration to the SDN infra and assign IPs from the selected subnet to the pods. When Disabled, OVN-Kubernetes only assigns MAC addresses, and provides layer2 communication, and enables users to configure IP addresses on the pods. ipam.lifecycle controls IP addresses management lifecycle. When set to 'Persistent', the assigned IP addresses will be persisted in ipamclaims.k8s.cni.cncf.io object. Useful for VMs, IP address will be persistent after restarts and migrations. Supported when ipam.mode is Enabled.
mtu	integer	mtu is the maximum transmission unit for a network. mtu is optional. When omitted, the configured value in OVN-Kubernetes (defaults to 1500 for localnet topology) is used for the network. Minimum value for IPv4 subnet is 576, and for IPv6 subnet is 1280. Maximum value is 65536. In a scenario physicalNetworkName points to OVS bridge mapping of a network configured with certain MTU settings, this field enables configuring the same MTU on pod interface, having the pod MTU aligned with the network MTU. Misaligned MTU across the stack (e.g.: pod has MTU X, node NIC has MTU Y), could result in network disruptions and bad performance.

Property	Type	Description
physicalNetworkName	string	physicalNetworkName points to the OVS bridge-mapping's network-name configured in the nodes, required. Min length is 1, max length is 253, cannot contain , or : characters. In case OVS bridge-mapping is defined by Kubernetes-nmstate with NodeNetworkConfigurationPolicy (NNCP), this field should point to the NNCP spec.desiredState.ovn.bridge-mappings item's localnet value.
role	string	role describes the network role in the pod, required. Controls whether the pod interface will act as primary or secondary. Localnet topology supports Secondary only. The network will be assigned to pods that have the k8s.v1.cni.cncf.io/networks annotation in place pointing to subject.

Property	Type	Description
subnets	array (string)	subnets is a list of subnets used for pods in this localnet network across the cluster. The list may be either 1 IPv4 subnet, 1 IPv6 subnet, or 1 of each IP family. When set, OVN-Kubernetes assigns an IP address from the specified CIDRs to the connected pod, eliminating the need for manual IP assignment or reliance on an external IPAM service (e.g., a DHCP server). subnets is optional. When omitted OVN-Kubernetes won't assign IP address automatically. Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed. The format should match standard CIDR notation (for example, "10.128.0.0/16"). This field must be omitted if ipam.mode is Disabled . When physicalNetworkName points to the OVS bridge mapping of a network that provides IPAM services (e.g., a DHCP server), ipam.mode should be set to Disabled. This turns off OVN-Kubernetes IPAM and avoids conflicts with the existing IPAM services on this localnet network.
vlan	object	vlan configuration for the network. vlan.mode is the VLAN mode. When "Access" is set, OVN-Kubernetes configures the network logical switch port in access mode. vlan.access is the access VLAN configuration. vlan.access.id is the VLAN ID (VID) to be set on the network logical switch port. vlan is optional, when omitted the underlying network default VLAN will be used (usually 1). When set, OVN-Kubernetes will apply VLAN configuration to the SDN infra and to the connected pods.

2.1.12. .spec.network.localnet.ipam

Description

ipam configurations for the network. ipam is optional. When omitted, **subnets** must be specified. When **ipam.mode** is **Disabled**, **subnets** must be omitted. **ipam.mode** controls how much of the IP configuration will be managed by OVN. When **Enabled**, OVN-Kubernetes will apply IP configuration to the SDN infra and assign IPs from the selected subnet to the pods. When **Disabled**, OVN-Kubernetes only assigns MAC addresses, and provides layer2 communication, and enables users to configure IP addresses on the pods. **ipam.lifecycle** controls IP addresses management lifecycle. When set to 'Persistent', the assigned IP addresses will be persisted in **ipamclaims.k8s.cni.cncf.io** object. Useful for VMs, IP address will be persistent after restarts and migrations. Supported when **ipam.mode** is **Enabled**.

Type

object

Property	Type	Description
lifecycle	string	Lifecycle controls IP addresses management lifecycle. The only allowed value is Persistent. When set, the IP addresses assigned by OVN Kubernetes will be persisted in an ipamclaims.k8s.cni.cncf.io object. These IP addresses will be reused by other pods if requested. Only supported when mode is Enabled.
mode	string	Mode controls how much of the IP configuration will be managed by OVN. Enabled means OVN-Kubernetes will apply IP configuration to the SDN infrastructure and it will also assign IPs from the selected subnet to the individual pods. Disabled means OVN-Kubernetes will only assign MAC addresses and provide layer 2 communication, letting users configure IP addresses for the pods. Disabled is only available for Secondary networks. By disabling IPAM, any Kubernetes features that rely on selecting pods by IP will no longer function (such as network policy, services, etc). Additionally, IP port security will also be disabled for interfaces attached to this network. Defaults to Enabled.

2.1.13. .spec.network.localnet.vlan

Description

vlan configuration for the network. vlan.mode is the VLAN mode. When "Access" is set, OVN-Kubernetes configures the network logical switch port in access mode. vlan.access is the access VLAN configuration. vlan.access.id is the VLAN ID (VID) to be set on the network logical switch port. vlan is optional, when omitted the underlying network default VLAN will be used (usually **1**). When set, OVN-Kubernetes will apply VLAN configuration to the SDN infra and to the connected pods.

Type

object

Required

- **mode**

Property	Type	Description
access	object	Access is the access VLAN configuration
mode	string	mode describe the network VLAN mode. Allowed value is "Access". Access sets the network logical switch port in access mode, according to the config.

2.1.14. .spec.network.localnet.vlan.access

Description

Access is the access VLAN configuration

Type

object

Required

- **id**

Property	Type	Description
id	integer	id is the VLAN ID (VID) to be set for the network. id should be higher than 0 and lower than 4095.

2.1.15. .status

Description

ClusterUserDefinedNetworkStatus contains the observed status of the ClusterUserDefinedNetwork.

Type
object

Property	Type	Description
conditions	array	Conditions slice of condition objects indicating details about ClusterUserDefineNetwork status.
conditions[]	object	Condition contains details for one aspect of the current state of this API Resource.

2.1.16. .status.conditions

Description

Conditions slice of condition objects indicating details about ClusterUserDefineNetwork status.

Type
array

2.1.17. .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.

Property	Type	Description
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.

2.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/k8s.ovn.org/v1/clusteruserdefinednetworks**
 - **DELETE**: delete collection of ClusterUserDefinedNetwork
 - **GET**: list objects of kind ClusterUserDefinedNetwork
 - **POST**: create a ClusterUserDefinedNetwork
- **/apis/k8s.ovn.org/v1/clusteruserdefinednetworks/{name}**
 - **DELETE**: delete a ClusterUserDefinedNetwork
 - **GET**: read the specified ClusterUserDefinedNetwork
 - **PATCH**: partially update the specified ClusterUserDefinedNetwork

- **PUT**: replace the specified ClusterUserDefinedNetwork
- **/apis/k8s.ovn.org/v1/clusteruserdefinednetworks/{name}/status**
 - **GET**: read status of the specified ClusterUserDefinedNetwork
 - **PATCH**: partially update status of the specified ClusterUserDefinedNetwork
 - **PUT**: replace status of the specified ClusterUserDefinedNetwork

2.2.1. /apis/k8s.ovn.org/v1/clusteruserdefinednetworks

HTTP method

DELETE

Description

delete collection of ClusterUserDefinedNetwork

Table 2.1. HTTP responses

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

HTTP method

GET

Description

list objects of kind ClusterUserDefinedNetwork

Table 2.2. HTTP responses

HTTP code	Reponse body
200 - OK	ClusterUserDefinedNetworkList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a ClusterUserDefinedNetwork

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	ClusterUserDefined Network schema	

Table 2.5. HTTP responses

HTTP code	Response body
200 - OK	ClusterUserDefinedNetwork schema
201 - Created	ClusterUserDefinedNetwork schema
202 - Accepted	ClusterUserDefinedNetwork schema
401 - Unauthorized	Empty

2.2.2. /apis/k8s.ovn.org/v1/clusteruserdefinednetworks/{name}

Table 2.6. Global path parameters

Parameter	Type	Description
name	string	name of the ClusterUserDefinedNetwork

HTTP method

DELETE

Description

delete a ClusterUserDefinedNetwork

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 ClusterUserDefinedNetwork

Table 2.9. HTTP responses

HTTP code	Reponse body
200 - OK	ClusterUserDefinedNetwork schema
401 - Unauthorized	Empty

HTTP method

PATCH**Description**

partially update the specified ClusterUserDefinedNetwork

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	Response body
200 - OK	ClusterUserDefinedNetwork schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified ClusterUserDefinedNetwork

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	ClusterUserDefinedNetwork schema	

Table 2.14. HTTP responses

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

2.2.3. /apis/k8s.ovn.org/v1/clusteruserdefinednetworks/{name}/status

Table 2.15. Global path parameters

Parameter	Type	Description
name	string	name of the ClusterUserDefinedNetwork

HTTP method**GET****Description**

read status of the specified ClusterUserDefinedNetwork

Table 2.16. HTTP responses

HTTP code	Reponse body
200 - OK	ClusterUserDefinedNetwork schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified ClusterUserDefinedNetwork

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	Response body
200 - OK	ClusterUserDefinedNetwork schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified ClusterUserDefinedNetwork

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	ClusterUserDefinedNetwork schema	

Table 2.21. HTTP responses

HTTP code	Response body
200 - OK	ClusterUserDefinedNetwork schema
201 - Created	ClusterUserDefinedNetwork schema
401 - Unauthorized	Empty

CHAPTER 3. ADMINNETWORKPOLICY [POLICY.NETWORKING.K8S.IO/V1ALPHA1]

Description

AdminNetworkPolicy is a cluster level resource that is part of the AdminNetworkPolicy API.

Type

object

Required

- **metadata**
- **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
spec	object	Specification of the desired behavior of AdminNetworkPolicy.

Property	Type	Description
status	object	Status is the status to be reported by the implementation.

3.1.1. .spec

Description

Specification of the desired behavior of AdminNetworkPolicy.

Type

object

Required

- **priority**
- **subject**

Property	Type	Description
egress	array	Egress is the list of Egress rules to be applied to the selected pods. A total of 100 rules will be allowed in each ANP instance. The relative precedence of egress rules within a single ANP object (all of which share the priority) will be determined by the order in which the rule is written. Thus, a rule that appears at the top of the egress rules would take the highest precedence. ANPs with no egress rules do not affect egress traffic. Support: Core
egress[]	object	AdminNetworkPolicyEgressRule describes an action to take on a particular set of traffic originating from pods selected by a AdminNetworkPolicy's Subject field. <network-policy-api:experimental:validation>

Property	Type	Description
ingress	array	Ingress is the list of Ingress rules to be applied to the selected pods. A total of 100 rules will be allowed in each ANP instance. The relative precedence of ingress rules within a single ANP object (all of which share the priority) will be determined by the order in which the rule is written. Thus, a rule that appears at the top of the ingress rules would take the highest precedence. ANPs with no ingress rules do not affect ingress traffic. Support: Core
ingress[]	object	AdminNetworkPolicyIngressRule describes an action to take on a particular set of traffic destined for pods selected by an AdminNetworkPolicy's Subject field.
priority	integer	Priority is a value from 0 to 1000. Rules with lower priority values have higher precedence, and are checked before rules with higher priority values. All AdminNetworkPolicy rules have higher precedence than NetworkPolicy or BaselineAdminNetworkPolicy rules. The behavior is undefined if two ANP objects have same priority. Support: Core
subject	object	Subject defines the pods to which this AdminNetworkPolicy applies. Note that host-networked pods are not included in subject selection. Support: Core

3.1.2. .spec.egress

Description

Egress is the list of Egress rules to be applied to the selected pods. A total of 100 rules will be allowed

in each ANP instance. The relative precedence of egress rules within a single ANP object (all of which share the priority) will be determined by the order in which the rule is written. Thus, a rule that appears at the top of the egress rules would take the highest precedence. ANPs with no egress rules do not affect egress traffic.

Support: Core

Type

array

3.1.3. .spec.egress[]

Description

AdminNetworkPolicyEgressRule describes an action to take on a particular set of traffic originating from pods selected by a AdminNetworkPolicy's Subject field. <network-policy-api:experimental:validation>

Type

object

Required

- **action**
- **to**

Property	Type	Description
action	string	Action specifies the effect this rule will have on matching traffic. Currently the following actions are supported: Allow: allows the selected traffic (even if it would otherwise have been denied by NetworkPolicy) Deny: denies the selected traffic Pass: instructs the selected traffic to skip any remaining ANP rules, and then pass execution to any NetworkPolicies that select the pod. If the pod is not selected by any NetworkPolicies then execution is passed to any BaselineAdminNetworkPolicies that select the pod. Support: Core

Property	Type	Description
name	string	Name is an identifier for this rule, that may be no more than 100 characters in length. This field should be used by the implementation to help improve observability, readability and error-reporting for any applied AdminNetworkPolicies. Support: Core
ports	array	Ports allows for matching traffic based on port and protocols. This field is a list of destination ports for the outgoing egress traffic. If Ports is not set then the rule does not filter traffic via port. Support: Core
ports[]	object	AdminNetworkPolicyPort describes how to select network ports on pod(s). Exactly one field must be set.
to	array	To is the List of destinations whose traffic this rule applies to. If any AdminNetworkPolicyEgressPeer matches the destination of outgoing traffic then the specified action is applied. This field must be defined and contain at least one item. Support: Core
to[]	object	AdminNetworkPolicyEgressPeer defines a peer to allow traffic to. Exactly one of the selector pointers must be set for a given peer. If a consumer observes none of its fields are set, they must assume an unknown option has been specified and fail closed.

3.1.4. .spec.egress[].ports

Description

Ports allows for matching traffic based on port and protocols. This field is a list of destination ports for the outgoing egress traffic. If Ports is not set then the rule does not filter traffic via port.

Support: Core

Type

array

3.1.5. .spec.egress[].ports[]

Description

AdminNetworkPolicyPort describes how to select network ports on pod(s). Exactly one field must be set.

Type

object

Property	Type	Description
namedPort	string	NamedPort selects a port on a pod(s) based on name. Support: Extended <network-policy-api:experimental>
portNumber	object	Port selects a port on a pod(s) based on number. Support: Core
portRange	object	PortRange selects a port range on a pod(s) based on provided start and end values. Support: Core

3.1.6. .spec.egress[].ports[].portNumber

Description

Port selects a port on a pod(s) based on number.

Support: Core

Type

object

Required

- **port**
- **protocol**

Property	Type	Description
port	integer	Number defines a network port value. Support: Core
protocol	string	Protocol is the network protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP. Support: Core

3.1.7. .spec.egress[].ports[].portRange

Description

PortRange selects a port range on a pod(s) based on provided start and end values.

Support: Core

Type

object

Required

- **end**
- **start**

Property	Type	Description
end	integer	End defines a network port that is the end of a port range, the End value must be greater than Start. Support: Core
protocol	string	Protocol is the network protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP. Support: Core
start	integer	Start defines a network port that is the start of a port range, the Start value must be less than End. Support: Core

3.1.8. .spec.egress[].to

Description

To is the List of destinations whose traffic this rule applies to. If any AdminNetworkPolicyEgressPeer matches the destination of outgoing traffic then the specified action is applied. This field must be defined and contain at least one item.

Support: Core

Type

array

3.1.9. .spec.egress[].to[]

Description

AdminNetworkPolicyEgressPeer defines a peer to allow traffic to. Exactly one of the selector pointers must be set for a given peer. If a consumer observes none of its fields are set, they must assume an unknown option has been specified and fail closed.

Type

object

Property	Type	Description
namespaces	object	Namespaces defines a way to select all pods within a set of Namespaces. Note that host-networked pods are not included in this type of peer. Support: Core

Property	Type	Description
networks	array (string)	Networks defines a way to select peers via CIDR blocks. This is intended for representing entities that live outside the cluster, which can't be selected by pods, namespaces and nodes peers, but note that cluster-internal traffic will be checked against the rule as well. So if you Allow or Deny traffic to "0.0.0.0/0", that will allow or deny all IPv4 pod-to-pod traffic as well. If you don't want that, add a rule that Passes all pod traffic before the Networks rule. Each item in Networks should be provided in the CIDR format and should be IPv4 or IPv6, for example "10.0.0.0/8" or "fd00::/8". Networks can have upto 25 CIDRs specified. Support: Extended <network-policy-api:experimental>
nodes	object	Nodes defines a way to select a set of nodes in the cluster. This field follows standard label selector semantics; if present but empty, it selects all Nodes. Support: Extended <network-policy-api:experimental>
pods	object	Pods defines a way to select a set of pods in a set of namespaces. Note that host-networked pods are not included in this type of peer. Support: Core

3.1.10. .spec.egress[].to[].namespaces

Description

Namespaces defines a way to select all pods within a set of Namespaces. Note that host-networked pods are not included in this type of peer.

Support: Core

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.

3.1.11. .spec.egress[].to[].namespaces.matchExpressions

Description

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

Type

array

3.1.12. .spec.egress[].to[].namespaces.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.

3.1.13. .spec.egress[.to[]].nodes

Description

Nodes defines a way to select a set of nodes in the cluster. This field follows standard label selector semantics; if present but empty, it selects all Nodes.

Support: Extended

<network-policy-api:experimental>

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.

3.1.14. .spec.egress[.to[].nodes.matchExpressions

Description

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

Type

array

3.1.15. .spec.egress[.to[].nodes.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.

3.1.16. .spec.egress[].to[].pods

Description

Pods defines a way to select a set of pods in a set of namespaces. Note that host-networked pods are not included in this type of peer.

Support: Core

Type

object

Required

- **namespaceSelector**
- **podSelector**

Property	Type	Description
namespaceSelector	object	NamespaceSelector follows standard label selector semantics; if empty, it selects all Namespaces.
podSelector	object	PodSelector is used to explicitly select pods within a namespace; if empty, it selects all Pods.

3.1.17. .spec.egress[].to[].pods.namespaceSelector

Description

NamespaceSelector follows standard label selector semantics; if empty, it selects 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.

3.1.18. .spec.egress[.to[]].pods.namespaceSelector.matchExpressions

Description

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

Type

array

3.1.19. .spec.egress[.to[]].pods.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.

3.1.20. .spec.egress[].to[].pods.podSelector

Description

PodSelector is used to explicitly select pods within a namespace; if empty, it selects all 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.

3.1.21. .spec.egress[].to[].pods.podSelector.matchExpressions

Description

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

Type

array

3.1.22. .spec.egress[].to[].pods.podSelector.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.

3.1.23. .spec.ingress

Description

Ingress is the list of Ingress rules to be applied to the selected pods. A total of 100 rules will be allowed in each ANP instance. The relative precedence of ingress rules within a single ANP object (all of which share the priority) will be determined by the order in which the rule is written. Thus, a rule that appears at the top of the ingress rules would take the highest precedence. ANPs with no ingress rules do not affect ingress traffic.

Support: Core

Type

array

3.1.24. .spec.ingress[]

Description

AdminNetworkPolicyIngressRule describes an action to take on a particular set of traffic destined for pods selected by an AdminNetworkPolicy's Subject field.

Type

object

Required

- **action**
- **from**

Property	Type	Description
action	string	Action specifies the effect this rule will have on matching traffic. Currently the following actions are supported: Allow: allows the selected traffic (even if it would otherwise have been denied by NetworkPolicy) Deny: denies the selected traffic Pass: instructs the selected traffic to skip any remaining ANP rules, and then pass execution to any NetworkPolicies that select the pod. If the pod is not selected by any NetworkPolicies then execution is passed to any BaselineAdminNetworkPolicies that select the pod. Support: Core
from	array	From is the list of sources whose traffic this rule applies to. If any AdminNetworkPolicyIngressPeer matches the source of incoming traffic then the specified action is applied. This field must be defined and contain at least one item. Support: Core

Property	Type	Description
from[]	object	AdminNetworkPolicyIngressPeer defines an in-cluster peer to allow traffic from. Exactly one of the selector pointers must be set for a given peer. If a consumer observes none of its fields are set, they must assume an unknown option has been specified and fail closed.
name	string	Name is an identifier for this rule, that may be no more than 100 characters in length. This field should be used by the implementation to help improve observability, readability and error-reporting for any applied AdminNetworkPolicies. Support: Core
ports	array	Ports allows for matching traffic based on port and protocols. This field is a list of ports which should be matched on the pods selected for this policy i.e the subject of the policy. So it matches on the destination port for the ingress traffic. If Ports is not set then the rule does not filter traffic via port. Support: Core
ports[]	object	AdminNetworkPolicyPort describes how to select network ports on pod(s). Exactly one field must be set.

3.1.25. .spec.ingress[].from

Description

From is the list of sources whose traffic this rule applies to. If any AdminNetworkPolicyIngressPeer matches the source of incoming traffic then the specified action is applied. This field must be defined and contain at least one item.

Support: Core

Type

array

3.1.26. .spec.ingress[].from[]

Description

AdminNetworkPolicyIngressPeer defines an in-cluster peer to allow traffic from. Exactly one of the selector pointers must be set for a given peer. If a consumer observes none of its fields are set, they must assume an unknown option has been specified and fail closed.

Type

object

Property	Type	Description
namespaces	object	Namespaces defines a way to select all pods within a set of Namespaces. Note that host-networked pods are not included in this type of peer. Support: Core
pods	object	Pods defines a way to select a set of pods in a set of namespaces. Note that host-networked pods are not included in this type of peer. Support: Core

3.1.27. .spec.ingress[].from[].namespaces**Description**

Namespaces defines a way to select all pods within a set of Namespaces. Note that host-networked pods are not included in this type of peer.

Support: Core

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.

3.1.28. .spec.ingress[].from[].namespaces.matchExpressions

Description

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

Type

array

3.1.29. .spec.ingress[].from[].namespaces.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.

3.1.30. .spec.ingress[].from[].pods

Description

Pods defines a way to select a set of pods in a set of namespaces. Note that host-networked pods are not included in this type of peer.

Support: Core

Type

object

Required

- **namespaceSelector**
- **podSelector**

Property	Type	Description
namespaceSelector	object	NamespaceSelector follows standard label selector semantics; if empty, it selects all Namespaces.
podSelector	object	PodSelector is used to explicitly select pods within a namespace; if empty, it selects all Pods.

3.1.31. .spec.ingress[].from[].pods.namespaceSelector

Description

NamespaceSelector follows standard label selector semantics; if empty, it selects 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.

3.1.32. .spec.ingress[].from[].pods.namespaceSelector.matchExpressions

Description

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

Type

array

3.1.33. .spec.ingress[].from[].pods.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.

3.1.34. .spec.ingress[].from[].pods.podSelector

Description

PodSelector is used to explicitly select pods within a namespace; if empty, it selects all 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.

3.1.35. .spec.ingress[].from[].pods.podSelector.matchExpressions

Description

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

Type

array

3.1.36. .spec.ingress[].from[].pods.podSelector.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.

3.1.37. .spec.ingress[].ports

Description

Ports allows for matching traffic based on port and protocols. This field is a list of ports which should be matched on the pods selected for this policy i.e the subject of the policy. So it matches on the destination port for the ingress traffic. If Ports is not set then the rule does not filter traffic via port.
Support: Core

Type

array

3.1.38. .spec.ingress[].ports[]

Description

AdminNetworkPolicyPort describes how to select network ports on pod(s). Exactly one field must be set.

Type

object

Property	Type	Description
namedPort	string	NamedPort selects a port on a pod(s) based on name. Support: Extended <network-policy-api:experimental>
portNumber	object	Port selects a port on a pod(s) based on number. Support: Core
portRange	object	PortRange selects a port range on a pod(s) based on provided start and end values. Support: Core

3.1.39. .spec.ingress[].ports[].portNumber

Description

Port selects a port on a pod(s) based on number.
Support: Core

Type

object

Required

- **port**
- **protocol**

Property	Type	Description
port	integer	Number defines a network port value. Support: Core

Property	Type	Description
protocol	string	Protocol is the network protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP. Support: Core

3.1.40. .spec.ingress[].ports[].portRange

Description

PortRange selects a port range on a pod(s) based on provided start and end values.

Support: Core

Type

object

Required

- **end**
- **start**

Property	Type	Description
end	integer	End defines a network port that is the end of a port range, the End value must be greater than Start. Support: Core
protocol	string	Protocol is the network protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP. Support: Core
start	integer	Start defines a network port that is the start of a port range, the Start value must be less than End. Support: Core

3.1.41. .spec.subject

Description

Subject defines the pods to which this AdminNetworkPolicy applies. Note that host-networked pods are not included in subject selection.

Support: Core

Type

object

Property	Type	Description
namespaces	object	Namespaces is used to select pods via namespace selectors.
pods	object	Pods is used to select pods via namespace AND pod selectors.

3.1.42. .spec.subject.namespaces

Description

Namespaces is used to select pods via namespace 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.
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.

3.1.43. .spec.subject.namespaces.matchExpressions

Description

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

Type

array

3.1.44. .spec.subject.namespaces.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.

3.1.45. .spec.subject.pods

Description

Pods is used to select pods via namespace AND pod selectors.

Type

object

Required

- **namespaceSelector**
- **podSelector**

Property	Type	Description
namespaceSelector	object	NamespaceSelector follows standard label selector semantics; if empty, it selects all Namespaces.
podSelector	object	PodSelector is used to explicitly select pods within a namespace; if empty, it selects all Pods.

3.1.46. .spec.subject.pods.namespaceSelector

Description

NamespaceSelector follows standard label selector semantics; if empty, it selects 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.

3.1.47. .spec.subject.pods.namespaceSelector.matchExpressions

Description

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

Type

array

3.1.48. .spec.subject.pods.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.
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.

3.1.49. .spec.subject.pods.podSelector**Description**

PodSelector is used to explicitly select pods within a namespace; if empty, it selects all 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.

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.

3.1.50. .spec.subject.pods.podSelector.matchExpressions

Description

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

Type

array

3.1.51. .spec.subject.pods.podSelector.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.

3.1.52. .status

Description

Status is the status to be reported by the implementation.

Type

object

Required

- **conditions**

Property	Type	Description
conditions	array	

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, <pre> 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 }</pre>

3.1.53. .status.conditions

Description

Type

array

3.1.54. .status.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.

Property	Type	Description
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. - -- 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)

3.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/policy.networking.k8s.io/v1alpha1/adminnetworkpolicies**
 - **DELETE**: delete collection of AdminNetworkPolicy
 - **GET**: list objects of kind AdminNetworkPolicy
 - **POST**: create an AdminNetworkPolicy
- **/apis/policy.networking.k8s.io/v1alpha1/adminnetworkpolicies/{name}**
 - **DELETE**: delete an AdminNetworkPolicy
 - **GET**: read the specified AdminNetworkPolicy
 - **PATCH**: partially update the specified AdminNetworkPolicy
 - **PUT**: replace the specified AdminNetworkPolicy
- **/apis/policy.networking.k8s.io/v1alpha1/adminnetworkpolicies/{name}/status**
 - **GET**: read status of the specified AdminNetworkPolicy

- **PATCH**: partially update status of the specified AdminNetworkPolicy
- **PUT**: replace status of the specified AdminNetworkPolicy

3.2.1. /apis/policy.networking.k8s.io/v1alpha1/adminnetworkpolicies

HTTP method

DELETE

Description

delete collection of AdminNetworkPolicy

Table 3.1. HTTP responses

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

HTTP method

GET

Description

list objects of kind AdminNetworkPolicy

Table 3.2. HTTP responses

HTTP code	Reponse body
200 - OK	AdminNetworkPolicyList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an AdminNetworkPolicy

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	AdminNetworkPolicy schema	

Table 3.5. HTTP responses

HTTP code	Response body
200 - OK	AdminNetworkPolicy schema
201 - Created	AdminNetworkPolicy schema
202 - Accepted	AdminNetworkPolicy schema
401 - Unauthorized	Empty

3.2.2. /apis/policy.networking.k8s.io/v1alpha1/adminnetworkpolicies/{name}

Table 3.6. Global path parameters

Parameter	Type	Description
name	string	name of the AdminNetworkPolicy

HTTP method**DELETE****Description**

delete an AdminNetworkPolicy

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	Response body
200 - OK	Status schema
202 - Accepted	Status schema
401 - Unauthorized	Empty

HTTP method**GET****Description**

read the specified AdminNetworkPolicy

Table 3.9. HTTP responses

HTTP code	Response body
200 - OK	AdminNetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified AdminNetworkPolicy

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	Response body
200 - OK	AdminNetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified AdminNetworkPolicy

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	AdminNetworkPolicy schema	

Table 3.14. HTTP responses

HTTP code	Response body
200 - OK	AdminNetworkPolicy schema
201 - Created	AdminNetworkPolicy schema
401 - Unauthorized	Empty

3.2.3. /apis/policy.networking.k8s.io/v1alpha1/adminnetworkpolicies/{name}/status

Table 3.15. Global path parameters

Parameter	Type	Description
name	string	name of the AdminNetworkPolicy

HTTP method

GET**Description**

read status of the specified AdminNetworkPolicy

Table 3.16. HTTP responses

HTTP code	Response body
200 - OK	AdminNetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified AdminNetworkPolicy

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	Response body
200 - OK	AdminNetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified AdminNetworkPolicy

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	AdminNetworkPolicy schema	

Table 3.21. HTTP responses

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

CHAPTER 4. ADMINPOLICYBASEDEXTERNALROUTE [K8S.OVN.ORG/V1]

Description

AdminPolicyBasedExternalRoute is a CRD allowing the cluster administrators to configure policies for external gateway IPs to be applied to all the pods contained in selected namespaces. Egress traffic from the pods that belong to the selected namespaces to outside the cluster is routed through these external gateway IPs.

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	AdminPolicyBasedExternalRouteSpec defines the desired state of AdminPolicyBasedExternalRoute
status	object	AdminPolicyBasedRouteStatus contains the observed status of the AdminPolicyBased route types.

4.1.1. .spec

Description

AdminPolicyBasedExternalRouteSpec defines the desired state of AdminPolicyBasedExternalRoute

Type

object

Required

- **from**
- **nextHops**

Property	Type	Description
from	object	From defines the selectors that will determine the target namespaces to this CR.
nextHops	object	NextHops defines two types of hops: Static and Dynamic. Each hop defines at least one external gateway IP.

4.1.2. .spec.from

Description

From defines the selectors that will determine the target namespaces to this CR.

Type

object

Required

- **namespaceSelector**

Property	Type	Description
namespaceSelector	object	NamespaceSelector defines a selector to be used to determine which namespaces will be targeted by this CR

4.1.3. .spec.from.namespaceSelector

Description

NamespaceSelector defines a selector to be used to determine which namespaces will be targeted by this CR

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.

4.1.4. .spec.from.namespaceSelector.matchExpressions

Description

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

Type

array

4.1.5. .spec.from.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.
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.

4.1.6. .spec.nextHops

Description

NextHops defines two types of hops: Static and Dynamic. Each hop defines at least one external gateway IP.

Type**object**

Property	Type	Description
dynamic	array	DynamicHops defines a slices of DynamicHop. This field is optional.

Property	Type	Description
dynamic[]	object	DynamicHop defines the configuration for a dynamic external gateway interface. These interfaces are wrapped around a pod object that resides inside the cluster. The field NetworkAttachmentName captures the name of the multus network name to use when retrieving the gateway IP to use. The PodSelector and the NamespaceSelector are mandatory fields.
static	array	StaticHops defines a slice of StaticHop. This field is optional.
static[]	object	StaticHop defines the configuration of a static IP that acts as an external Gateway Interface. IP field is mandatory.

4.1.7. .spec.nextHops.dynamic

Description

DynamicHops defines a slices of DynamicHop. This field is optional.

Type

array

4.1.8. .spec.nextHops.dynamic[]

Description

DynamicHop defines the configuration for a dynamic external gateway interface. These interfaces are wrapped around a pod object that resides inside the cluster. The field NetworkAttachmentName captures the name of the multus network name to use when retrieving the gateway IP to use. The PodSelector and the NamespaceSelector are mandatory fields.

Type

object

Required

- **namespaceSelector**
- **podSelector**

Property	Type	Description
bfdEnabled	boolean	BFDEnabled determines if the interface implements the Bidirectional Forward Detection protocol. Defaults to false.
namespaceSelector	object	NamespaceSelector defines a selector to filter the namespaces where the pod gateways are located.
networkAttachmentName	string	NetworkAttachmentName determines the multus network name to use when retrieving the pod IPs that will be used as the gateway IP. When this field is empty, the logic assumes that the pod is configured with HostNetwork and is using the node's IP as gateway.
podSelector	object	PodSelector defines the selector to filter the pods that are external gateways.

4.1.9. .spec.nextHops.dynamic[].namespaceSelector

Description

NamespaceSelector defines a selector to filter the namespaces where the pod gateways are located.

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.

4.1.10. .spec.nextHops.dynamic[].namespaceSelector.matchExpressions

Description

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

Type

array

4.1.11. .spec.nextHops.dynamic[].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.

4.1.12. .spec.nextHops.dynamic[].podSelector

Description

PodSelector defines the selector to filter the pods that are external gateways.

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.

4.1.13. .spec.nextHops.dynamic[].podSelector.matchExpressions

Description

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

Type

array

4.1.14. .spec.nextHops.dynamic[].podSelector.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.

4.1.15. .spec.nextHops.static

Description

StaticHops defines a slice of StaticHop. This field is optional.

Type

array

4.1.16. .spec.nextHops.static[]

Description

StaticHop defines the configuration of a static IP that acts as an external Gateway Interface. IP field is mandatory.

Type

object

Required

- **ip**

Property	Type	Description
bfdEnabled	boolean	BFDEnabled determines if the interface implements the Bidirectional Forward Detection protocol. Defaults to false.
ip	string	IP defines the static IP to be used for egress traffic. The IP can be either IPv4 or IPv6.

4.1.17. .status

Description

AdminPolicyBasedRouteStatus contains the observed status of the AdminPolicyBased route types.

Type

object

Property	Type	Description
lastTransitionTime	string	Captures the time when the last change was applied.
messages	array (string)	An array of Human-readable messages indicating details about the status of the object.
status	string	A concise indication of whether the AdminPolicyBasedRoute resource is applied with success

4.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/k8s.ovn.org/v1/adminpolicybasedexternalroutes**
 - **DELETE**: delete collection of AdminPolicyBasedExternalRoute
 - **GET**: list objects of kind AdminPolicyBasedExternalRoute
 - **POST**: create an AdminPolicyBasedExternalRoute
- **/apis/k8s.ovn.org/v1/adminpolicybasedexternalroutes/{name}**
 - **DELETE**: delete an AdminPolicyBasedExternalRoute

- **GET**: read the specified AdminPolicyBasedExternalRoute
- **PATCH**: partially update the specified AdminPolicyBasedExternalRoute
- **PUT**: replace the specified AdminPolicyBasedExternalRoute
- **/apis/k8s.ovn.org/v1/adminpolicybasedexternalroutes/{name}/status**
 - **GET**: read status of the specified AdminPolicyBasedExternalRoute
 - **PATCH**: partially update status of the specified AdminPolicyBasedExternalRoute
 - **PUT**: replace status of the specified AdminPolicyBasedExternalRoute

4.2.1. /apis/k8s.ovn.org/v1/adminpolicybasedexternalroutes

HTTP method

DELETE

Description

delete collection of AdminPolicyBasedExternalRoute

Table 4.1. HTTP responses

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

HTTP method

GET

Description

list objects of kind AdminPolicyBasedExternalRoute

Table 4.2. HTTP responses

HTTP code	Reponse body
200 - OK	AdminPolicyBasedExternalRouteList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an AdminPolicyBasedExternalRoute

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	AdminPolicyBasedExternalRoute schema	

Table 4.5. HTTP responses

HTTP code	Response body
200 - OK	AdminPolicyBasedExternalRoute schema
201 - Created	AdminPolicyBasedExternalRoute schema
202 - Accepted	AdminPolicyBasedExternalRoute schema
401 - Unauthorized	Empty

4.2.2. /apis/k8s.ovn.org/v1/adminpolicybasedexternalroutes/{name}

Table 4.6. Global path parameters

Parameter	Type	Description
name	string	name of the AdminPolicyBasedExternalRoute

HTTP method

DELETE

Description

delete an AdminPolicyBasedExternalRoute

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	Response body
200 - OK	Status schema
202 - Accepted	Status schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified AdminPolicyBasedExternalRoute

Table 4.9. HTTP responses

HTTP code	Response body
200 - OK	AdminPolicyBasedExternalRoute schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified AdminPolicyBasedExternalRoute

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	Response body
200 - OK	AdminPolicyBasedExternalRoute schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified AdminPolicyBasedExternalRoute

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	AdminPolicyBasedExternalRoute schema	

Table 4.14. HTTP responses

HTTP code	Response body
200 - OK	AdminPolicyBasedExternalRoute schema
201 - Created	AdminPolicyBasedExternalRoute schema
401 - Unauthorized	Empty

4.2.3. /apis/k8s.ovn.org/v1/adminpolicybasedexternalroutes/{name}/status

Table 4.15. Global path parameters

Parameter	Type	Description
name	string	name of the AdminPolicyBasedExternalRoute

HTTP method**GET****Description**

read status of the specified AdminPolicyBasedExternalRoute

Table 4.16. HTTP responses

HTTP code	Reponse body
200 - OK	AdminPolicyBasedExternalRoute schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified AdminPolicyBasedExternalRoute

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	Response body
200 - OK	AdminPolicyBasedExternalRoute schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified AdminPolicyBasedExternalRoute

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	AdminPolicyBasedExternalRoute schema	

Table 4.21. HTTP responses

HTTP code	Response body
200 - OK	AdminPolicyBasedExternalRoute schema
201 - Created	AdminPolicyBasedExternalRoute schema
401 - Unauthorized	Empty

CHAPTER 5. BASELINEADMINNETWORKPOLICY [POLICY.NETWORKING.K8S.IO/V1ALPHA1]

Description

BaselineAdminNetworkPolicy is a cluster level resource that is part of the AdminNetworkPolicy API.

Type

object

Required

- **metadata**
- **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	Specification of the desired behavior of BaselineAdminNetworkPolicy.

Property	Type	Description
status	object	Status is the status to be reported by the implementation.

5.1.1. .spec

Description

Specification of the desired behavior of BaselineAdminNetworkPolicy.

Type

object

Required

- **subject**

Property	Type	Description
egress	array	Egress is the list of Egress rules to be applied to the selected pods if they are not matched by any AdminNetworkPolicy or NetworkPolicy rules. A total of 100 Egress rules will be allowed in each BANP instance. The relative precedence of egress rules within a single BANP object will be determined by the order in which the rule is written. Thus, a rule that appears at the top of the egress rules would take the highest precedence. BANPs with no egress rules do not affect egress traffic. Support: Core
egress[]	object	BaselineAdminNetworkPolicyEgressRule describes an action to take on a particular set of traffic originating from pods selected by a BaselineAdminNetworkPolicy's Subject field. <network-policy-api:experimental:validation>

Property	Type	Description
ingress	array	Ingress is the list of Ingress rules to be applied to the selected pods if they are not matched by any AdminNetworkPolicy or NetworkPolicy rules. A total of 100 Ingress rules will be allowed in each BANP instance. The relative precedence of ingress rules within a single BANP object will be determined by the order in which the rule is written. Thus, a rule that appears at the top of the ingress rules would take the highest precedence. BANPs with no ingress rules do not affect ingress traffic. Support: Core
ingress[]	object	BaselineAdminNetworkPolicyIngressRule describes an action to take on a particular set of traffic destined for pods selected by a BaselineAdminNetworkPolicy's Subject field.
subject	object	Subject defines the pods to which this BaselineAdminNetworkPolicy applies. Note that host-networked pods are not included in subject selection. Support: Core

5.1.2. .spec.egress

Description

Egress is the list of Egress rules to be applied to the selected pods if they are not matched by any AdminNetworkPolicy or NetworkPolicy rules. A total of 100 Egress rules will be allowed in each BANP instance. The relative precedence of egress rules within a single BANP object will be determined by the order in which the rule is written. Thus, a rule that appears at the top of the egress rules would take the highest precedence. BANPs with no egress rules do not affect egress traffic.

Support: Core

Type

array

5.1.3. .spec.egress[]

Description

BaselineAdminNetworkPolicyEgressRule describes an action to take on a particular set of traffic originating from pods selected by a BaselineAdminNetworkPolicy's Subject field. <network-policy-api:experimental:validation>

Type

object

Required

- **action**
- **to**

Property	Type	Description
action	string	Action specifies the effect this rule will have on matching traffic. Currently the following actions are supported: Allow: allows the selected traffic Deny: denies the selected traffic Support: Core
name	string	Name is an identifier for this rule, that may be no more than 100 characters in length. This field should be used by the implementation to help improve observability, readability and error-reporting for any applied BaselineAdminNetworkPolicies. Support: Core
ports	array	Ports allows for matching traffic based on port and protocols. This field is a list of destination ports for the outgoing egress traffic. If Ports is not set then the rule does not filter traffic via port.
ports[]	object	AdminNetworkPolicyPort describes how to select network ports on pod(s). Exactly one field must be set.

Property	Type	Description
to	array	To is the list of destinations whose traffic this rule applies to. If any AdminNetworkPolicyEgressPeer matches the destination of outgoing traffic then the specified action is applied. This field must be defined and contain at least one item. Support: Core
to[]	object	AdminNetworkPolicyEgressPeer defines a peer to allow traffic to. Exactly one of the selector pointers must be set for a given peer. If a consumer observes none of its fields are set, they must assume an unknown option has been specified and fail closed.

5.1.4. .spec.egress[].ports

Description

Ports allows for matching traffic based on port and protocols. This field is a list of destination ports for the outgoing egress traffic. If Ports is not set then the rule does not filter traffic via port.

Type

array

5.1.5. .spec.egress[].ports[]

Description

AdminNetworkPolicyPort describes how to select network ports on pod(s). Exactly one field must be set.

Type

object

Property	Type	Description
namedPort	string	NamedPort selects a port on a pod(s) based on name. Support: Extended <network-policy-api:experimental>

Property	Type	Description
portNumber	object	Port selects a port on a pod(s) based on number. Support: Core
portRange	object	PortRange selects a port range on a pod(s) based on provided start and end values. Support: Core

5.1.6. .spec.egress[].ports[].portNumber

Description

Port selects a port on a pod(s) based on number.
Support: Core

Type

object

Required

- **port**
- **protocol**

Property	Type	Description
port	integer	Number defines a network port value. Support: Core
protocol	string	Protocol is the network protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP. Support: Core

5.1.7. .spec.egress[].ports[].portRange

Description

PortRange selects a port range on a pod(s) based on provided start and end values.
Support: Core

Type**object****Required**

- **end**
- **start**

Property	Type	Description
end	integer	End defines a network port that is the end of a port range, the End value must be greater than Start. Support: Core
protocol	string	Protocol is the network protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP. Support: Core
start	integer	Start defines a network port that is the start of a port range, the Start value must be less than End. Support: Core

5.1.8. .spec.egress[].to**Description**

To is the list of destinations whose traffic this rule applies to. If any AdminNetworkPolicyEgressPeer matches the destination of outgoing traffic then the specified action is applied. This field must be defined and contain at least one item.

Support: Core

Type**array****5.1.9. .spec.egress[].to[]****Description**

AdminNetworkPolicyEgressPeer defines a peer to allow traffic to. Exactly one of the selector pointers must be set for a given peer. If a consumer observes none of its fields are set, they must assume an unknown option has been specified and fail closed.

Type**object**

Property	Type	Description
namespaces	object	Namespaces defines a way to select all pods within a set of Namespaces. Note that host-networked pods are not included in this type of peer. Support: Core
networks	array (string)	Networks defines a way to select peers via CIDR blocks. This is intended for representing entities that live outside the cluster, which can't be selected by pods, namespaces and nodes peers, but note that cluster-internal traffic will be checked against the rule as well. So if you Allow or Deny traffic to "0.0.0.0/0", that will allow or deny all IPv4 pod-to-pod traffic as well. If you don't want that, add a rule that Passes all pod traffic before the Networks rule. Each item in Networks should be provided in the CIDR format and should be IPv4 or IPv6, for example "10.0.0.0/8" or "fd00::/8". Networks can have upto 25 CIDRs specified. Support: Extended <network-policy-api:experimental>
nodes	object	Nodes defines a way to select a set of nodes in the cluster. This field follows standard label selector semantics; if present but empty, it selects all Nodes. Support: Extended <network-policy-api:experimental>

Property	Type	Description
pods	object	Pods defines a way to select a set of pods in a set of namespaces. Note that host-networked pods are not included in this type of peer. Support: Core

5.1.10. .spec.egress[.to[].namespaces

Description

Namespaces defines a way to select all pods within a set of Namespaces. Note that host-networked pods are not included in this type of peer.

Support: Core

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.

5.1.11. .spec.egress[.to[].namespaces.matchExpressions

Description

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

Type

array

5.1.12. .spec.egress[].to[].namespaces.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.

5.1.13. .spec.egress[].to[].nodes

Description

Nodes defines a way to select a set of nodes in the cluster. This field follows standard label selector semantics; if present but empty, it selects all Nodes.

Support: Extended

<network-policy-api:experimental>

Type

object

Property	Type	Description

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.

5.1.14. .spec.egress[].to[].nodes.matchExpressions

Description

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

Type

array

5.1.15. .spec.egress[].to[].nodes.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.

5.1.16. .spec.egress[].to[].pods

Description

Pods defines a way to select a set of pods in a set of namespaces. Note that host-networked pods are not included in this type of peer.

Support: Core

Type

object

Required

- **namespaceSelector**
- **podSelector**

Property	Type	Description
namespaceSelector	object	NamespaceSelector follows standard label selector semantics; if empty, it selects all Namespaces.
podSelector	object	PodSelector is used to explicitly select pods within a namespace; if empty, it selects all Pods.

5.1.17. .spec.egress[].to[].pods.namespaceSelector

Description

NamespaceSelector follows standard label selector semantics; if empty, it selects 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.

5.1.18. .spec.egress[.to[].pods.namespaceSelector.matchExpressions

Description

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

Type

array

5.1.19. .spec.egress[.to[].pods.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.

5.1.20. .spec.egress[].to[].pods.podSelector

Description

PodSelector is used to explicitly select pods within a namespace; if empty, it selects all 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.

5.1.21. .spec.egress[].to[].pods.podSelector.matchExpressions

Description

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

Type

array

5.1.22. .spec.egress[].to[].pods.podSelector.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.

5.1.23. .spec.ingress

Description

Ingress is the list of Ingress rules to be applied to the selected pods if they are not matched by any AdminNetworkPolicy or NetworkPolicy rules. A total of 100 Ingress rules will be allowed in each BANP instance. The relative precedence of ingress rules within a single BANP object will be determined by the order in which the rule is written. Thus, a rule that appears at the top of the ingress rules would take the highest precedence. BANPs with no ingress rules do not affect ingress traffic.

Support: Core

Type

array

5.1.24. .spec.ingress[]

Description

BaselineAdminNetworkPolicyIngressRule describes an action to take on a particular set of traffic destined for pods selected by a BaselineAdminNetworkPolicy's Subject field.

Type

object

Required

- **action**
- **from**

Property	Type	Description
action	string	Action specifies the effect this rule will have on matching traffic. Currently the following actions are supported: Allow: allows the selected traffic Deny: denies the selected traffic Support: Core
from	array	From is the list of sources whose traffic this rule applies to. If any AdminNetworkPolicyIngressPeer matches the source of incoming traffic then the specified action is applied. This field must be defined and contain at least one item. Support: Core
from[]	object	AdminNetworkPolicyIngressPeer defines an in-cluster peer to allow traffic from. Exactly one of the selector pointers must be set for a given peer. If a consumer observes none of its fields are set, they must assume an unknown option has been specified and fail closed.

Property	Type	Description
name	string	Name is an identifier for this rule, that may be no more than 100 characters in length. This field should be used by the implementation to help improve observability, readability and error-reporting for any applied BaselineAdminNetworkPolicies. Support: Core
ports	array	Ports allows for matching traffic based on port and protocols. This field is a list of ports which should be matched on the pods selected for this policy i.e the subject of the policy. So it matches on the destination port for the ingress traffic. If Ports is not set then the rule does not filter traffic via port. Support: Core
ports[]	object	AdminNetworkPolicyPort describes how to select network ports on pod(s). Exactly one field must be set.

5.1.25. .spec.ingress[].from

Description

From is the list of sources whose traffic this rule applies to. If any AdminNetworkPolicyIngressPeer matches the source of incoming traffic then the specified action is applied. This field must be defined and contain at least one item.

Support: Core

Type

array

5.1.26. .spec.ingress[].from[]

Description

AdminNetworkPolicyIngressPeer defines an in-cluster peer to allow traffic from. Exactly one of the selector pointers must be set for a given peer. If a consumer observes none of its fields are set, they must assume an unknown option has been specified and fail closed.

Type

object

Property	Type	Description
namespaces	object	Namespaces defines a way to select all pods within a set of Namespaces. Note that host-networked pods are not included in this type of peer. Support: Core
pods	object	Pods defines a way to select a set of pods in a set of namespaces. Note that host-networked pods are not included in this type of peer. Support: Core

5.1.27. .spec.ingress[].from[].namespaces

Description

Namespaces defines a way to select all pods within a set of Namespaces. Note that host-networked pods are not included in this type of peer.

Support: Core

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.

5.1.28. .spec.ingress[].from[].namespaces.matchExpressions

Description

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

Type

array

5.1.29. .spec.ingress[].from[].namespaces.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.

5.1.30. .spec.ingress[].from[].pods**Description**

Pods defines a way to select a set of pods in a set of namespaces. Note that host-networked pods are not included in this type of peer.

Support: Core

Type

object

Required

- **namespaceSelector**

- **podSelector**

Property	Type	Description
namespaceSelector	object	NamespaceSelector follows standard label selector semantics; if empty, it selects all Namespaces.
podSelector	object	PodSelector is used to explicitly select pods within a namespace; if empty, it selects all Pods.

5.1.31. .spec.ingress[].from[].pods.namespaceSelector

Description

NamespaceSelector follows standard label selector semantics; if empty, it selects 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.

5.1.32. .spec.ingress[].from[].pods.namespaceSelector.matchExpressions

Description

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

Type

array

5.1.33. .spec.ingress[].from[].pods.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.
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.ingress[].from[].pods.podSelector

Description

PodSelector is used to explicitly select pods within a namespace; if empty, it selects all 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.

5.1.35. .spec.ingress[].from[].pods.podSelector.matchExpressions

Description

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

Type

array

5.1.36. .spec.ingress[].from[].pods.podSelector.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.

5.1.37. .spec.ingress[].ports

Description

Ports allows for matching traffic based on port and protocols. This field is a list of ports which should be matched on the pods selected for this policy i.e the subject of the policy. So it matches on the destination port for the ingress traffic. If Ports is not set then the rule does not filter traffic via port.
Support: Core

Type

array

5.1.38. .spec.ingress[].ports[]

Description

AdminNetworkPolicyPort describes how to select network ports on pod(s). Exactly one field must be set.

Type

object

Property	Type	Description
namedPort	string	NamedPort selects a port on a pod(s) based on name. Support: Extended <network-policy-api:experimental>
portNumber	object	Port selects a port on a pod(s) based on number. Support: Core

Property	Type	Description
portRange	object	PortRange selects a port range on a pod(s) based on provided start and end values. Support: Core

5.1.39. .spec.ingress[].ports[].portNumber

Description

Port selects a port on a pod(s) based on number.
Support: Core

Type

object

Required

- **port**
- **protocol**

Property	Type	Description
port	integer	Number defines a network port value. Support: Core
protocol	string	Protocol is the network protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP. Support: Core

5.1.40. .spec.ingress[].ports[].portRange

Description

PortRange selects a port range on a pod(s) based on provided start and end values.
Support: Core

Type

object

Required

- **end**

- **start**

Property	Type	Description
end	integer	End defines a network port that is the end of a port range, the End value must be greater than Start. Support: Core
protocol	string	Protocol is the network protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP. Support: Core
start	integer	Start defines a network port that is the start of a port range, the Start value must be less than End. Support: Core

5.1.41. .spec.subject

Description

Subject defines the pods to which this BaselineAdminNetworkPolicy applies. Note that host-networked pods are not included in subject selection.

Support: Core

Type

object

Property	Type	Description
namespaces	object	Namespaces is used to select pods via namespace selectors.
pods	object	Pods is used to select pods via namespace AND pod selectors.

5.1.42. .spec.subject.namespaces

Description

Namespaces is used to select pods via namespace 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.
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.

5.1.43. .spec.subject.namespaces.matchExpressions

Description

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

Type

array

5.1.44. .spec.subject.namespaces.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.

5.1.45. .spec.subject.pods

Description

Pods is used to select pods via namespace AND pod selectors.

Type

object

Required

- **namespaceSelector**
- **podSelector**

Property	Type	Description
namespaceSelector	object	NamespaceSelector follows standard label selector semantics; if empty, it selects all Namespaces.
podSelector	object	PodSelector is used to explicitly select pods within a namespace; if empty, it selects all Pods.

5.1.46. .spec.subject.pods.namespaceSelector

Description

NamespaceSelector follows standard label selector semantics; if empty, it selects 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.

5.1.47. .spec.subject.pods.namespaceSelector.matchExpressions

Description

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

Type

array

5.1.48. .spec.subject.pods.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.

5.1.49. .spec.subject.pods.podSelector

Description

PodSelector is used to explicitly select pods within a namespace; if empty, it selects all 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.

5.1.50. .spec.subject.pods.podSelector.matchExpressions

Description

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

Type

array

5.1.51. .spec.subject.pods.podSelector.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.

5.1.52. .status

Description

Status is the status to be reported by the implementation.

Type

object

Required

- **conditions**

Property	Type	Description
conditions	array	

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, <pre> 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 }</pre>

5.1.53. .status.conditions

Description

Type

array

5.1.54. .status.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.

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. -- 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)

5.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/policy.networking.k8s.io/v1alpha1/baselineadminnetworkpolicies**
 - **DELETE**: delete collection of BaselineAdminNetworkPolicy
 - **GET**: list objects of kind BaselineAdminNetworkPolicy
 - **POST**: create a BaselineAdminNetworkPolicy
- **/apis/policy.networking.k8s.io/v1alpha1/baselineadminnetworkpolicies/{name}**
 - **DELETE**: delete a BaselineAdminNetworkPolicy
 - **GET**: read the specified BaselineAdminNetworkPolicy
 - **PATCH**: partially update the specified BaselineAdminNetworkPolicy
 - **PUT**: replace the specified BaselineAdminNetworkPolicy
- **/apis/policy.networking.k8s.io/v1alpha1/baselineadminnetworkpolicies/{name}/status**
 - **GET**: read status of the specified BaselineAdminNetworkPolicy
 - **PATCH**: partially update status of the specified BaselineAdminNetworkPolicy
 - **PUT**: replace status of the specified BaselineAdminNetworkPolicy

5.2.1. /apis/policy.networking.k8s.io/v1alpha1/baselineadminnetworkpolicies

HTTP method

DELETE

Description

delete collection of BaselineAdminNetworkPolicy

Table 5.1. HTTP responses

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

HTTP method

GET

Description

list objects of kind BaselineAdminNetworkPolicy

Table 5.2. HTTP responses

HTTP code	Reponse body
200 - OK	BaselineAdminNetworkPolicyList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a BaselineAdminNetworkPolicy

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	BaselineAdminNetworkPolicy schema	

Table 5.5. HTTP responses

HTTP code	Response body
200 - OK	BaselineAdminNetworkPolicy schema
201 - Created	BaselineAdminNetworkPolicy schema
202 - Accepted	BaselineAdminNetworkPolicy schema
401 - Unauthorized	Empty

5.2.2. /apis/policy.networking.k8s.io/v1alpha1/baselineadminnetworkpolicies/{name}

Table 5.6. Global path parameters

Parameter	Type	Description
name	string	name of the BaselineAdminNetworkPolicy

HTTP method**DELETE****Description**

delete a BaselineAdminNetworkPolicy

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 BaselineAdminNetworkPolicy

Table 5.9. HTTP responses

HTTP code	Reponse body
200 - OK	BaselineAdminNetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified BaselineAdminNetworkPolicy

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	Response body
200 - OK	BaselineAdminNetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified BaselineAdminNetworkPolicy

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	BaselineAdminNetworkPolicy schema	

Table 5.14. HTTP responses

HTTP code	Response body
200 - OK	BaselineAdminNetworkPolicy schema
201 - Created	BaselineAdminNetworkPolicy schema
401 - Unauthorized	Empty

5.2.3. /apis/policy.networking.k8s.io/v1alpha1/baselineadminnetworkpolicies/{name}/

Table 5.15. Global path parameters

Parameter	Type	Description
name	string	name of the BaselineAdminNetworkPolicy

HTTP method

GET**Description**

read status of the specified BaselineAdminNetworkPolicy

Table 5.16. HTTP responses

HTTP code	Response body
200 - OK	BaselineAdminNetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified BaselineAdminNetworkPolicy

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	Response body
200 - OK	BaselineAdminNetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified BaselineAdminNetworkPolicy

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	BaselineAdminNetworkPolicy schema	

Table 5.21. HTTP responses

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

CHAPTER 6. CLOUDPRIVATEIPCONFIG [CLOUD.NETWORK.OPENSIFT.IO/V1]

Description

CloudPrivateIPConfig performs an assignment of a private IP address to the primary NIC associated with cloud VMs. This is done by specifying the IP and Kubernetes node which the IP should be assigned to. This CRD is intended to be used by the network plugin which manages the cluster network. The spec side represents the desired state requested by the network plugin, and the status side represents the current state that this CRD's controller has executed. No users will have permission to modify it, and if a cluster-admin decides to edit it for some reason, their changes will be overwritten the next time the network plugin reconciles the object. Note: the CR's name must specify the requested private IP address (can be IPv4 or IPv6).

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

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 definition of the desired private IP request.
status	object	status is the observed status of the desired private IP request. Read-only.

6.1.1. .spec

Description

spec is the definition of the desired private IP request.

Type

object

Property	Type	Description
node	string	node is the node name, as specified by the Kubernetes field: node.metadata.name

6.1.2. .status

Description

status is the observed status of the desired private IP request. Read-only.

Type

object

Required

- **conditions**

Property	Type	Description
conditions	array	condition is the assignment condition of the private IP and its status

Property	Type	Description
conditions[]	object	Condition contains details for one aspect of the current state of this API Resource.
node	string	node is the node name, as specified by the Kubernetes field: node.metadata.name

6.1.3. .status.conditions

Description

condition is the assignment condition of the private IP and its status

Type

array

6.1.4. .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.

Property	Type	Description
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.

6.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/cloud.network.openshift.io/v1/cloudprivateipconfigs**
 - **DELETE**: delete collection of CloudPrivateIPConfig
 - **GET**: list objects of kind CloudPrivateIPConfig
 - **POST**: create a CloudPrivateIPConfig
- **/apis/cloud.network.openshift.io/v1/cloudprivateipconfigs/{name}**
 - **DELETE**: delete a CloudPrivateIPConfig

- **GET**: read the specified CloudPrivateIPConfig
- **PATCH**: partially update the specified CloudPrivateIPConfig
- **PUT**: replace the specified CloudPrivateIPConfig
- **/apis/cloud.network.openshift.io/v1/cloudprivateipconfigs/{name}/status**
 - **GET**: read status of the specified CloudPrivateIPConfig
 - **PATCH**: partially update status of the specified CloudPrivateIPConfig
 - **PUT**: replace status of the specified CloudPrivateIPConfig

6.2.1. /apis/cloud.network.openshift.io/v1/cloudprivateipconfigs

HTTP method

DELETE

Description

delete collection of CloudPrivateIPConfig

Table 6.1. HTTP responses

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

HTTP method

GET

Description

list objects of kind CloudPrivateIPConfig

Table 6.2. HTTP responses

HTTP code	Reponse body
200 - OK	CloudPrivateIPConfigList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a CloudPrivateIPConfig

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
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	CloudPrivateIPConfig schema	

Table 6.5. HTTP responses

HTTP code	Response body
200 - OK	CloudPrivateIPConfig schema
201 - Created	CloudPrivateIPConfig schema
202 - Accepted	CloudPrivateIPConfig schema
401 - Unauthorized	Empty

6.2.2. /apis/cloud.network.openshift.io/v1/cloudprivateipconfigs/{name}

Table 6.6. Global path parameters

Parameter	Type	Description
name	string	name of the CloudPrivateIPConfig

HTTP method

DELETE

Description

delete a CloudPrivateIPConfig

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 CloudPrivateIPConfig

Table 6.9. HTTP responses

HTTP code	Reponse body
200 - OK	CloudPrivateIPConfig schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified CloudPrivateIPConfig

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	Response body
200 - OK	CloudPrivateIPConfig schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified CloudPrivateIPConfig

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
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	CloudPrivateIPConfig schema	

Table 6.14. HTTP responses

HTTP code	Response body
200 - OK	CloudPrivateIPConfig schema
201 - Created	CloudPrivateIPConfig schema
401 - Unauthorized	Empty

6.2.3. /apis/cloud.network.openshift.io/v1/cloudprivateipconfigs/{name}/status

Table 6.15. Global path parameters

Parameter	Type	Description
name	string	name of the CloudPrivateIPConfig

HTTP method**GET****Description**

read status of the specified CloudPrivateIPConfig

Table 6.16. HTTP responses

HTTP code	Reponse body
200 - OK	CloudPrivateIPConfig schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified CloudPrivateIPConfig

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

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 6.18. HTTP responses

HTTP code	Response body
200 - OK	CloudPrivateIPConfig schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified CloudPrivateIPConfig

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

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 6.20. Body parameters

Parameter	Type	Description
body	CloudPrivateIPConfig schema	

Table 6.21. HTTP responses

HTTP code	Response body
200 - OK	CloudPrivateIPConfig schema
201 - Created	CloudPrivateIPConfig schema
401 - Unauthorized	Empty

CHAPTER 7. EGRESSFIREWALL [K8S.OVN.ORG/V1]

Description

EgressFirewall describes the current egress firewall for a Namespace. Traffic from a pod to an IP address outside the cluster will be checked against each EgressFirewallRule in the pod's namespace's EgressFirewall, in order. If no rule matches (or no EgressFirewall is present) then the traffic will be allowed by default.

Type

object

Required

- **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
spec	object	Specification of the desired behavior of EgressFirewall.

Property	Type	Description
status	object	Observed status of EgressFirewall

7.1.1. .spec

Description

Specification of the desired behavior of EgressFirewall.

Type

object

Required

- **egress**

Property	Type	Description
egress	array	a collection of egress firewall rule objects
egress[]	object	EgressFirewallRule is a single egressfirewall rule object

7.1.2. .spec.egress

Description

a collection of egress firewall rule objects

Type

array

7.1.3. .spec.egress[]

Description

EgressFirewallRule is a single egressfirewall rule object

Type

object

Required

- **to**
- **type**

Property	Type	Description

Property	Type	Description
ports	array	ports specify what ports and protocols the rule applies to
ports[]	object	EgressFirewallPort specifies the port to allow or deny traffic to
to	object	to is the target that traffic is allowed/denied to
type	string	type marks this as an "Allow" or "Deny" rule

7.1.4. .spec.egress[].ports

Description

ports specify what ports and protocols the rule applies to

Type

array

7.1.5. .spec.egress[].ports[]

Description

EgressFirewallPort specifies the port to allow or deny traffic to

Type

object

Required

- **port**
- **protocol**

Property	Type	Description
port	integer	port that the traffic must match
protocol	string	protocol (tcp, udp, sctp) that the traffic must match.

7.1.6. .spec.egress[].to

Description

to is the target that traffic is allowed/denied to

Type

object

Property	Type	Description
cidrSelector	string	cidrSelector is the CIDR range to allow/deny traffic to. If this is set, dnsName and nodeSelector must be unset.
dnsName	string	dnsName is the domain name to allow/deny traffic to. If this is set, cidrSelector and nodeSelector must be unset. For a wildcard DNS name, the " will match only one label. Additionally, only a single " can be used at the beginning of the wildcard DNS name. For example, '*.example.com' will match 'sub1.example.com' but won't match 'sub2.sub1.example.com'.
nodeSelector	object	nodeSelector will allow/deny traffic to the Kubernetes node IP of selected nodes. If this is set, cidrSelector and DNSName must be unset.

7.1.7. .spec.egress[].to.nodeSelector

Description

nodeSelector will allow/deny traffic to the Kubernetes node IP of selected nodes. If this is set, cidrSelector and DNSName must be unset.

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.

7.1.8. .spec.egress[].to.nodeSelector.matchExpressions

Description

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

Type

array

7.1.9. .spec.egress[].to.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.

7.1.10. .status

Description

Observed status of EgressFirewall

Type

object

Property	Type	Description
messages	array (string)	
status	string	

7.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/k8s.ovn.org/v1/egressfirewalls**
 - **GET**: list objects of kind EgressFirewall
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/egressfirewalls**
 - **DELETE**: delete collection of EgressFirewall
 - **GET**: list objects of kind EgressFirewall
 - **POST**: create an EgressFirewall
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/egressfirewalls/{name}**
 - **DELETE**: delete an EgressFirewall
 - **GET**: read the specified EgressFirewall
 - **PATCH**: partially update the specified EgressFirewall

- **PUT**: replace the specified EgressFirewall
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/egressfirewalls/{name}/status**
 - **GET**: read status of the specified EgressFirewall
 - **PATCH**: partially update status of the specified EgressFirewall
 - **PUT**: replace status of the specified EgressFirewall

7.2.1. /apis/k8s.ovn.org/v1/egressfirewalls

HTTP method

GET

Description

list objects of kind EgressFirewall

Table 7.1. HTTP responses

HTTP code	Reponse body
200 - OK	EgressFirewallList schema
401 - Unauthorized	Empty

7.2.2. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressfirewalls

HTTP method

DELETE

Description

delete collection of EgressFirewall

Table 7.2. HTTP responses

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

HTTP method

GET

Description

list objects of kind EgressFirewall

Table 7.3. HTTP responses

HTTP code	Response body
200 - OK	EgressFirewallList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an EgressFirewall

Table 7.4. 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 7.5. Body parameters

Parameter	Type	Description
body	EgressFirewall schema	

Table 7.6. HTTP responses

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

7.2.3. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressfirewalls/{name}

Table 7.7. Global path parameters

Parameter	Type	Description
name	string	name of the EgressFirewall

HTTP method

DELETE

Description

delete an EgressFirewall

Table 7.8. 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 7.9. HTTP responses

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

HTTP method

GET

Description

read the specified EgressFirewall

Table 7.10. HTTP responses

HTTP code	Response body
200 - OK	EgressFirewall schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified EgressFirewall

Table 7.11. 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 7.12. HTTP responses

HTTP code	Reponse body
200 - OK	EgressFirewall schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified EgressFirewall

Table 7.13. 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 7.14. Body parameters

Parameter	Type	Description
body	EgressFirewall schema	

Table 7.15. HTTP responses

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

7.2.4. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressfirewalls/{name}/status

Table 7.16. Global path parameters

Parameter	Type	Description
name	string	name of the EgressFirewall

HTTP method

GET

Description

read status of the specified EgressFirewall

Table 7.17. HTTP responses

HTTP code	Reponse body
200 - OK	EgressFirewall schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified EgressFirewall

Table 7.18. 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 7.19. HTTP responses

HTTP code	Response body
200 - OK	EgressFirewall schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified EgressFirewall

Table 7.20. 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 7.21. Body parameters

Parameter	Type	Description
body	EgressFirewall schema	

Table 7.22. HTTP responses

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

CHAPTER 8. EGRESSIP [K8S.OVN.ORG/V1]

Description

EgressIP is a CRD allowing the user to define a fixed source IP for all egress traffic originating from any pods which match the EgressIP resource according to its spec definition.

Type

object

Required

- **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
spec	object	Specification of the desired behavior of EgressIP.
status	object	Observed status of EgressIP. Read-only.

8.1.1. .spec

Description

Specification of the desired behavior of EgressIP.

Type

object

Required

- **egressIPs**
- **namespaceSelector**

Property	Type	Description
egressIPs	array (string)	EgressIPs is the list of egress IP addresses requested. Can be IPv4 and/or IPv6. This field is mandatory.
namespaceSelector	object	NamespaceSelector applies the egress IP only to the namespace(s) whose label matches this definition. This field is mandatory.
podSelector	object	PodSelector applies the egress IP only to the pods whose label matches this definition. This field is optional, and in case it is not set: results in the egress IP being applied to all pods in the namespace(s) matched by the NamespaceSelector. In case it is set: is intersected with the NamespaceSelector, thus applying the egress IP to the pods (in the namespace(s) already matched by the NamespaceSelector) which match this pod selector.

8.1.2. .spec.namespaceSelector

Description

NamespaceSelector applies the egress IP only to the namespace(s) whose label matches this definition. This field is mandatory.

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.

8.1.3. .spec.namespaceSelector.matchExpressions

Description

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

Type

array

8.1.4. .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.

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.

8.1.5. .spec.podSelector

Description

PodSelector applies the egress IP only to the pods whose label matches this definition. This field is optional, and in case it is not set: results in the egress IP being applied to all pods in the namespace(s) matched by the NamespaceSelector. In case it is set: is intersected with the NamespaceSelector, thus applying the egress IP to the pods (in the namespace(s) already matched by the NamespaceSelector) which match this pod selector.

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.

8.1.6. .spec.podSelector.matchExpressions

Description

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

Type

array

8.1.7. .spec.podSelector.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.

8.1.8. .status**Description**

Observed status of EgressIP. Read-only.

Type

object

Required

- **items**

Property	Type	Description
items	array	The list of assigned egress IPs and their corresponding node assignment.
items[]	object	The per node status, for those egress IPs who have been assigned.

8.1.9. .status.items

Description

The list of assigned egress IPs and their corresponding node assignment.

Type

array

8.1.10. .status.items[]

Description

The per node status, for those egress IPs who have been assigned.

Type

object

Required

- **egressIP**
- **node**

Property	Type	Description
egressIP	string	Assigned egress IP
node	string	Assigned node name

8.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/k8s.ovn.org/v1/egressips**
 - **DELETE**: delete collection of EgressIP
 - **GET**: list objects of kind EgressIP
 - **POST**: create an EgressIP

- **/apis/k8s.ovn.org/v1/egressips/{name}**
 - **DELETE**: delete an EgressIP
 - **GET**: read the specified EgressIP
 - **PATCH**: partially update the specified EgressIP
 - **PUT**: replace the specified EgressIP

8.2.1. /apis/k8s.ovn.org/v1/egressips

HTTP method

DELETE

Description

delete collection of EgressIP

Table 8.1. HTTP responses

HTTP code	Response body
200 - OK	Status schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind EgressIP

Table 8.2. HTTP responses

HTTP code	Response body
200 - OK	EgressIPList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an EgressIP

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	EgressIP schema	

Table 8.5. HTTP responses

HTTP code	Response body
200 - OK	EgressIP schema
201 - Created	EgressIP schema
202 - Accepted	EgressIP schema
401 - Unauthorized	Empty

8.2.2. /apis/k8s.ovn.org/v1/egressips/{name}

Table 8.6. Global path parameters

Parameter	Type	Description
name	string	name of the EgressIP

HTTP method

DELETE

Description

delete an EgressIP

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	Response body
200 - OK	Status schema
202 - Accepted	Status schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified EgressIP

Table 8.9. HTTP responses

HTTP code	Response body
200 - OK	EgressIP schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified EgressIP

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	Response body
200 - OK	EgressIP schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified EgressIP

Table 8.12. 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 8.13. Body parameters

Parameter	Type	Description
body	EgressIP schema	

Table 8.14. HTTP responses

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

CHAPTER 9. EGRESSQOS [K8S.OVN.ORG/V1]

Description

EgressQoS is a CRD that allows the user to define a DSCP value for pods egress traffic on its namespace to specified CIDRs. Traffic from these pods will be checked against each EgressQoSRule in the namespace's EgressQoS, and if there is a match the traffic is marked with the relevant DSCP value.

Type

object

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
spec	object	EgressQoSSpec defines the desired state of EgressQoS
status	object	EgressQoSStatus defines the observed state of EgressQoS

9.1.1. .spec

Description

EgressQoSSpec defines the desired state of EgressQoS

Type

object

Required

- **egress**

Property	Type	Description
egress	array	a collection of Egress QoS rule objects
egress[]	object	

9.1.2. .spec.egress

Description

a collection of Egress QoS rule objects

Type

array

9.1.3. .spec.egress[]

Description

Type

object

Required

- **dscp**

Property	Type	Description
dscp	integer	DSCP marking value for matching pods' traffic.
dstCIDR	string	DstCIDR specifies the destination's CIDR. Only traffic heading to this CIDR will be marked with the DSCP value. This field is optional, and in case it is not set the rule is applied to all egress traffic regardless of the destination.

Property	Type	Description
podSelector	object	PodSelector applies the QoS rule only to the pods in the namespace whose label matches this definition. This field is optional, and in case it is not set results in the rule being applied to all pods in the namespace.

9.1.4. .spec.egress[].podSelector

Description

PodSelector applies the QoS rule only to the pods in the namespace whose label matches this definition. This field is optional, and in case it is not set results in the rule being applied to all pods in the namespace.

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.

9.1.5. .spec.egress[].podSelector.matchExpressions

Description

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

Type

array

9.1.6. .spec.egress[].podSelector.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.

9.1.7. .status

Description

EgressQoSStatus defines the observed state of EgressQoS

Type

object

Property	Type	Description
conditions	array	An array of condition objects indicating details about status of EgressQoS object.

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 <code>.status.conditions</code>. For example, <pre> 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 }</pre>
status	string	A concise indication of whether the EgressQoS resource is applied with success.

9.1.8. `.status.conditions`

Description

An array of condition objects indicating details about status of EgressQoS object.

Type

array

9.1.9. `.status.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.

Property	Type	Description
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. -- 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)

9.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/k8s.ovn.org/v1/egressqoses**
 - **GET**: list objects of kind EgressQoS
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/egressqoses**
 - **DELETE**: delete collection of EgressQoS
 - **GET**: list objects of kind EgressQoS
 - **POST**: create an EgressQoS
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/egressqoses/{name}**
 - **DELETE**: delete an EgressQoS
 - **GET**: read the specified EgressQoS
 - **PATCH**: partially update the specified EgressQoS
 - **PUT**: replace the specified EgressQoS

- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/egressqoses/{name}/status**
 - **GET**: read status of the specified EgressQoS
 - **PATCH**: partially update status of the specified EgressQoS
 - **PUT**: replace status of the specified EgressQoS

9.2.1. /apis/k8s.ovn.org/v1/egressqoses

HTTP method

GET

Description

list objects of kind EgressQoS

Table 9.1. HTTP responses

HTTP code	Reponse body
200 - OK	EgressQoSList schema
401 - Unauthorized	Empty

9.2.2. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressqoses

HTTP method

DELETE

Description

delete collection of EgressQoS

Table 9.2. HTTP responses

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

HTTP method

GET

Description

list objects of kind EgressQoS

Table 9.3. HTTP responses

HTTP code	Reponse body

HTTP code	Response body
200 - OK	EgressQoSList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an EgressQoS

Table 9.4. 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 9.5. Body parameters

Parameter	Type	Description
body	EgressQoS schema	

Table 9.6. HTTP responses

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

9.2.3. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressqoses/{name}

Table 9.7. Global path parameters

Parameter	Type	Description
name	string	name of the EgressQoS

HTTP method

DELETE

Description

delete an EgressQoS

Table 9.8. 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 9.9. HTTP responses

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

HTTP method

GET

Description

read the specified EgressQoS

Table 9.10. HTTP responses

HTTP code	Response body
200 - OK	EgressQoS schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified EgressQoS

Table 9.11. 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 9.12. HTTP responses

HTTP code	Response body
200 - OK	EgressQoS schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified EgressQoS

Table 9.13. 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 9.14. Body parameters

Parameter	Type	Description
body	EgressQoS schema	

Table 9.15. HTTP responses

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

9.2.4. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressqoses/{name}/status

Table 9.16. Global path parameters

Parameter	Type	Description
name	string	name of the EgressQoS

HTTP method

GET

Description

read status of the specified EgressQoS

Table 9.17. HTTP responses

HTTP code	Reponse body
200 - OK	EgressQoS schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified EgressQoS

Table 9.18. 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 9.19. HTTP responses

HTTP code	Response body
200 - OK	EgressQoS schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified EgressQoS

Table 9.20. 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 9.21. Body parameters

Parameter	Type	Description
body	EgressQoS schema	

Table 9.22. HTTP responses

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

CHAPTER 10. EGRESSSERVICE [K8S.OVN.ORG/V1]

Description

EgressService is a CRD that allows the user to request that the source IP of egress packets originating from all of the pods that are endpoints of the corresponding LoadBalancer Service would be its ingress IP. In addition, it allows the user to request that egress packets originating from all of the pods that are endpoints of the LoadBalancer service would use a different network than the main one.

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	EgressServiceSpec defines the desired state of EgressService
status	object	EgressServiceStatus defines the observed state of EgressService

10.1.1. .spec

Description

EgressServiceSpec defines the desired state of EgressService

Type

object

Property	Type	Description
network	string	The network which this service should send egress and corresponding ingress replies to. This is typically implemented as VRF mapping, representing a numeric id or string name of a routing table which by omission uses the default host routing.
nodeSelector	object	Allows limiting the nodes that can be selected to handle the service's traffic when <code>sourceIPBy=LoadBalancerIP</code> . When present only a node whose labels match the specified selectors can be selected for handling the service's traffic. When it is not specified any node in the cluster can be chosen to manage the service's traffic.
sourceIPBy	string	Determines the source IP of egress traffic originating from the pods backing the LoadBalancer Service. When LoadBalancerIP the source IP is set to its LoadBalancer ingress IP. When Network the source IP is set according to the interface of the Network, leveraging the masquerade rules that are already in place. Typically these rules specify SNAT to the IP of the outgoing interface, which means the packet will typically leave with the IP of the node.

10.1.2. .spec.nodeSelector

Description

Allows limiting the nodes that can be selected to handle the service's traffic when `sourceIPBy=LoadBalancerIP`. When present only a node whose labels match the specified selectors can be selected for handling the service's traffic. When it is not specified any node in the cluster can

be chosen to manage the service's traffic.

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.

10.1.3. .spec.nodeSelector.matchExpressions

Description

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

Type

array

10.1.4. .spec.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.
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.

10.1.5. .status

Description

EgressServiceStatus defines the observed state of EgressService

Type

object

Required

- **host**

Property	Type	Description
host	string	The name of the node selected to handle the service's traffic. In case sourceIPBy=Network the field will be set to "ALL".

10.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/k8s.ovn.org/v1/egressservices**
 - **GET**: list objects of kind EgressService
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/egressservices**
 - **DELETE**: delete collection of EgressService

- **GET**: list objects of kind EgressService
- **POST**: create an EgressService
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/egressservices/{name}**
 - **DELETE**: delete an EgressService
 - **GET**: read the specified EgressService
 - **PATCH**: partially update the specified EgressService
 - **PUT**: replace the specified EgressService
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/egressservices/{name}/status**
 - **GET**: read status of the specified EgressService
 - **PATCH**: partially update status of the specified EgressService
 - **PUT**: replace status of the specified EgressService

10.2.1. /apis/k8s.ovn.org/v1/egressservices

HTTP method

GET

Description

list objects of kind EgressService

Table 10.1. HTTP responses

HTTP code	Reponse body
200 - OK	EgressServiceList schema
401 - Unauthorized	Empty

10.2.2. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressservices

HTTP method

DELETE

Description

delete collection of EgressService

Table 10.2. 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 EgressService

Table 10.3. HTTP responses

HTTP code	Reponse body
200 - OK	EgressServiceList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an EgressService

Table 10.4. 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

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 10.5. Body parameters

Parameter	Type	Description
body	EgressService schema	

Table 10.6. HTTP responses

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

10.2.3. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressservices/{name}

Table 10.7. Global path parameters

Parameter	Type	Description
name	string	name of the EgressService

HTTP method**DELETE****Description**

delete an EgressService

Table 10.8. 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 10.9. HTTP responses

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

HTTP method**GET****Description**

read the specified EgressService

Table 10.10. HTTP responses

HTTP code	Reponse body
200 - OK	EgressService schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified EgressService

Table 10.11. 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 10.12. HTTP responses

HTTP code	Response body
200 - OK	EgressService schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified EgressService

Table 10.13. 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 10.14. Body parameters

Parameter	Type	Description
body	EgressService schema	

Table 10.15. HTTP responses

HTTP code	Response body
200 - OK	EgressService schema
201 - Created	EgressService schema
401 - Unauthorized	Empty

10.2.4. /apis/k8s.ovn.org/v1/namespaces/{namespace}/egressservices/{name}/statu

Table 10.16. Global path parameters

Parameter	Type	Description
name	string	name of the EgressService

HTTP method**GET****Description**

read status of the specified EgressService

Table 10.17. HTTP responses

HTTP code	Reponse body
200 - OK	EgressService schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified EgressService

Table 10.18. 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 10.19. HTTP responses

HTTP code	Response body
200 - OK	EgressService schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified EgressService

Table 10.20. 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 10.21. Body parameters

Parameter	Type	Description
body	EgressService schema	

Table 10.22. HTTP responses

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

CHAPTER 11. ENDPOINTS [V1]

Description

Endpoints is a collection of endpoints that implement the actual service. Example:

```
Name: "mysvc",
Subsets: [
  {
    Addresses: [{"ip": "10.10.1.1"}, {"ip": "10.10.2.2"}],
    Ports: [{"name": "a", "port": 8675}, {"name": "b", "port": 309}]
  },
  {
    Addresses: [{"ip": "10.10.3.3"}],
    Ports: [{"name": "a", "port": 93}, {"name": "b", "port": 76}]
  },
]
```

Endpoints is a legacy API and does not contain information about all Service features. Use `discoveryv1.EndpointSlice` for complete information about Service endpoints.

Deprecated: This API is deprecated in v1.33+. Use `discoveryv1.EndpointSlice`.

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

Property	Type	Description
metadata	ObjectMeta	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
subsets	array	The set of all endpoints is the union of all subsets. Addresses are placed into subsets according to the IPs they share. A single address with multiple ports, some of which are ready and some of which are not (because they come from different containers) will result in the address being displayed in different subsets for the different ports. No address will appear in both Addresses and NotReadyAddresses in the same subset. Sets of addresses and ports that comprise a service.
subsets[]	object	EndpointSubset is a group of addresses with a common set of ports. The expanded set of endpoints is the Cartesian product of Addresses x Ports. For example, given: <pre>{ Addresses: [{ "ip": "10.10.1.1"}, { "ip": "10.10.2.2"}], Ports: [{ "name": "a", "port": 8675}, { "name": "b", "port": 309}] }</pre> The resulting set of endpoints can be viewed as: <pre>a: [10.10.1.1:8675, 10.10.2.2:8675], b: [10.10.1.1:309, 10.10.2.2:309]</pre> Deprecated: This API is deprecated in v1.33+.

11.1.1. .subsets

Description

The set of all endpoints is the union of all subsets. Addresses are placed into subsets according to the IPs they share. A single address with multiple ports, some of which are ready and some of which are not (because they come from different containers) will result in the address being displayed in

different subsets for the different ports. No address will appear in both `Addresses` and `NotReadyAddresses` in the same subset. Sets of addresses and ports that comprise a service.

Type

array

11.1.2. `.subsets[]`

Description

`EndpointSubset` is a group of addresses with a common set of ports. The expanded set of endpoints is the Cartesian product of `Addresses` x `Ports`. For example, given:

```
{
  Addresses: [{"ip": "10.10.1.1"}, {"ip": "10.10.2.2"}],
  Ports:     [{"name": "a", "port": 8675}, {"name": "b", "port": 309}]
}
```

The resulting set of endpoints can be viewed as:

```
a: [ 10.10.1.1:8675, 10.10.2.2:8675 ],
b: [ 10.10.1.1:309, 10.10.2.2:309 ]
```

Deprecated: This API is deprecated in v1.33+.

Type

object

Property	Type	Description
addresses	array	IP addresses which offer the related ports that are marked as ready. These endpoints should be considered safe for load balancers and clients to utilize.
addresses[]	object	<code>EndpointAddress</code> is a tuple that describes single IP address. Deprecated: This API is deprecated in v1.33+.
notReadyAddresses	array	IP addresses which offer the related ports but are not currently marked as ready because they have not yet finished starting, have recently failed a readiness check, or have recently failed a liveness check.
notReadyAddresses[]	object	<code>EndpointAddress</code> is a tuple that describes single IP address. Deprecated: This API is deprecated in v1.33+.

Property	Type	Description
ports	array	Port numbers available on the related IP addresses.
ports[]	object	EndpointPort is a tuple that describes a single port. Deprecated: This API is deprecated in v1.33+.

11.1.3. .subsets[].addresses

Description

IP addresses which offer the related ports that are marked as ready. These endpoints should be considered safe for load balancers and clients to utilize.

Type

array

11.1.4. .subsets[].addresses[]

Description

EndpointAddress is a tuple that describes single IP address. Deprecated: This API is deprecated in v1.33+.

Type

object

Required

- **ip**

Property	Type	Description
hostname	string	The Hostname of this endpoint
ip	string	The IP of this endpoint. May not be loopback (127.0.0.0/8 or ::1), link-local (169.254.0.0/16 or fe80::/10), or link-local multicast (224.0.0.0/24 or ff02::/16).

Property	Type	Description
nodeName	string	Optional: Node hosting this endpoint. This can be used to determine endpoints local to a node.
targetRef	object	ObjectReference contains enough information to let you inspect or modify the referred object.

11.1.5. .subsets[].addresses[].targetRef

Description

ObjectReference contains enough information to let you inspect or modify the referred object.

Type

object

Property	Type	Description
apiVersion	string	API version of the referent.
fieldPath	string	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as <code>desiredState.manifest.containers[2]</code> . For example, if the object reference is to a container within a pod, this would take on a value like: <code>"spec.containers{name}"</code> (where "name" refers to the name of the container that triggered the event) or if no container name is specified <code>"spec.containers[2]"</code> (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.
kind	string	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds

Property	Type	Description
name	string	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
namespace	string	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/
resourceVersion	string	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency
uid	string	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids

11.1.6. .subsets[].notReadyAddresses

Description

IP addresses which offer the related ports but are not currently marked as ready because they have not yet finished starting, have recently failed a readiness check, or have recently failed a liveness check.

Type

array

11.1.7. .subsets[].notReadyAddresses[]

Description

EndpointAddress is a tuple that describes single IP address. Deprecated: This API is deprecated in v1.33+.

Type

object

Required

- **ip**

Property	Type	Description
hostname	string	The Hostname of this endpoint
ip	string	The IP of this endpoint. May not be loopback (127.0.0.0/8 or ::1), link-local (169.254.0.0/16 or fe80::/10), or link-local multicast (224.0.0.0/24 or ff02::/16).
nodeName	string	Optional: Node hosting this endpoint. This can be used to determine endpoints local to a node.
targetRef	object	ObjectReference contains enough information to let you inspect or modify the referred object.

11.1.8. .subsets[].notReadyAddresses[].targetRef

Description

ObjectReference contains enough information to let you inspect or modify the referred object.

Type

object

Property	Type	Description
apiVersion	string	API version of the referent.
fieldPath	string	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as <code>desiredState.manifest.containers[2]</code> . For example, if the object reference is to a container within a pod, this would take on a value like: <code>"spec.containers{name}"</code> (where "name" refers to the name of the container that triggered the event) or if no container name is specified <code>"spec.containers[2]"</code> (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object.

Property	Type	Description
kind	string	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
name	string	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
namespace	string	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/
resourceVersion	string	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency
uid	string	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids

11.1.9. .subsets[].ports

Description

Port numbers available on the related IP addresses.

Type

array

11.1.10. .subsets[].ports[]

Description

EndpointPort is a tuple that describes a single port. Deprecated: This API is deprecated in v1.33+.

Type

object

Required

- **port**

Property	Type	Description
appProtocol	string	The application protocol for this port. This is used as a hint for implementations to offer richer behavior for protocols that they understand. This field follows standard Kubernetes label syntax. Valid values are either: * Un-prefixed protocol names - reserved for IANA standard service names (as per RFC-6335 and https://www.iana.org/assignments/service-names/). * Kubernetes-defined prefixed names: * 'kubernetes.io/h2c' - HTTP/2 prior knowledge over cleartext as described in https://www.rfc-editor.org/rfc/rfc9113.html#name-starting-http-2-with-prior- * 'kubernetes.io/ws' - WebSocket over cleartext as described in https://www.rfc-editor.org/rfc/rfc6455 * 'kubernetes.io/wss' - WebSocket over TLS as described in https://www.rfc-editor.org/rfc/rfc6455 * Other protocols should use implementation-defined prefixed names such as mycompany.com/my-custom-protocol.
name	string	The name of this port. This must match the 'name' field in the corresponding ServicePort. Must be a DNS_LABEL. Optional only if one port is defined.
port	integer	The port number of the endpoint.

Property	Type	Description
protocol	string	The IP protocol for this port. Must be UDP, TCP, or SCTP. Default is TCP. Possible enum values: - "SCTP" is the SCTP protocol. - "TCP" is the TCP protocol. - "UDP" is the UDP protocol.

11.2. API ENDPOINTS

The following API endpoints are available:

- **/api/v1/endpoints**
 - **GET**: list or watch objects of kind Endpoints
- **/api/v1/watch/endpoints**
 - **GET**: watch individual changes to a list of Endpoints. deprecated: use the 'watch' parameter with a list operation instead.
- **/api/v1/namespaces/{namespace}/endpoints**
 - **DELETE**: delete collection of Endpoints
 - **GET**: list or watch objects of kind Endpoints
 - **POST**: create Endpoints
- **/api/v1/watch/namespaces/{namespace}/endpoints**
 - **GET**: watch individual changes to a list of Endpoints. deprecated: use the 'watch' parameter with a list operation instead.
- **/api/v1/namespaces/{namespace}/endpoints/{name}**
 - **DELETE**: delete Endpoints
 - **GET**: read the specified Endpoints
 - **PATCH**: partially update the specified Endpoints
 - **PUT**: replace the specified Endpoints
- **/api/v1/watch/namespaces/{namespace}/endpoints/{name}**
 - **GET**: watch changes to an object of kind Endpoints. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

11.2.1. /api/v1/endpoints

HTTP method**GET****Description**

list or watch objects of kind Endpoints

Table 11.1. HTTP responses

HTTP code	Reponse body
200 - OK	EndpointsList schema
401 - Unauthorized	Empty

11.2.2. /api/v1/watch/endpoints**HTTP method****GET****Description**

watch individual changes to a list of Endpoints. deprecated: use the 'watch' parameter with a list operation instead.

Table 11.2. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

11.2.3. /api/v1/namespaces/{namespace}/endpoints**HTTP method****DELETE****Description**

delete collection of Endpoints

Table 11.3. 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

Table 11.4. HTTP responses

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

HTTP method**GET****Description**

list or watch objects of kind Endpoints

Table 11.5. HTTP responses

HTTP code	Reponse body
200 - OK	EndpointsList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create Endpoints

Table 11.6. 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 11.7. Body parameters

Parameter	Type	Description
body	Endpoints schema	

Table 11.8. HTTP responses

HTTP code	Response body
200 - OK	Endpoints schema
201 - Created	Endpoints schema
202 - Accepted	Endpoints schema
401 - Unauthorized	Empty

11.2.4. /api/v1/watch/namespaces/{namespace}/endpoints

HTTP method

GET

Description

watch individual changes to a list of Endpoints. deprecated: use the 'watch' parameter with a list operation instead.

Table 11.9. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

11.2.5. /api/v1/namespaces/{namespace}/endpoints/{name}

Table 11.10. Global path parameters

Parameter	Type	Description
name	string	name of the Endpoints

HTTP method

DELETE

Description

delete Endpoints

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

Table 11.12. HTTP responses

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

HTTP method

GET

Description

read the specified Endpoints

Table 11.13. HTTP responses

HTTP code	Response body
200 - OK	Endpoints schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Endpoints

Table 11.14. 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 11.15. HTTP responses

HTTP code	Response body
200 - OK	Endpoints schema
201 - Created	Endpoints schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified Endpoints

Table 11.16. 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 11.17. Body parameters

Parameter	Type	Description
body	Endpoints schema	

Table 11.18. HTTP responses

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

11.2.6. /api/v1/watch/namespaces/{namespace}/endpoints/{name}

Table 11.19. Global path parameters

Parameter	Type	Description
name	string	name of the Endpoints

HTTP method

GET

Description

watch changes to an object of kind Endpoints. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 11.20. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

CHAPTER 12. ENDPOINTS LICE [DISCOVERY.K8S.IO/V1]

Description

EndpointSlice represents a set of service endpoints. Most EndpointSlices are created by the EndpointSlice controller to represent the Pods selected by Service objects. For a given service there may be multiple EndpointSlice objects which must be joined to produce the full set of endpoints; you can find all of the slices for a given service by listing EndpointSlices in the service's namespace whose **kubernetes.io/service-name** label contains the service's name.

Type

object

Required

- **addressType**
- **endpoints**

12.1. SPECIFICATION

Property	Type	Description
addressType	string	addressType specifies the type of address carried by this EndpointSlice. All addresses in this slice must be the same type. This field is immutable after creation. The following address types are currently supported: * IPv4: Represents an IPv4 Address. * IPv6: Represents an IPv6 Address. * FQDN: Represents a Fully Qualified Domain Name. (Deprecated) The EndpointSlice controller only generates, and kube-proxy only processes, slices of addressType "IPv4" and "IPv6". No semantics are defined for the "FQDN" type. Possible enum values: - "FQDN" represents a FQDN. - "IPv4" represents an IPv4 Address. - "IPv6" represents an IPv6 Address.

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
endpoints	array	endpoints is a list of unique endpoints in this slice. Each slice may include a maximum of 1000 endpoints.
endpoints[]	object	Endpoint represents a single logical "backend" implementing a service.
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.
ports	array	ports specifies the list of network ports exposed by each endpoint in this slice. Each port must have a unique name. Each slice may include a maximum of 100 ports. Services always have at least 1 port, so EndpointSlices generated by the EndpointSlice controller will likewise always have at least 1 port. EndpointSlices used for other purposes may have an empty ports list.
ports[]	object	EndpointPort represents a Port used by an EndpointSlice

12.1.1. .endpoints

Description

endpoints is a list of unique endpoints in this slice. Each slice may include a maximum of 1000 endpoints.

Type

array

12.1.2. .endpoints[]

Description

Endpoint represents a single logical "backend" implementing a service.

Type

object

Required

- **addresses**

Property	Type	Description
addresses	array (string)	addresses of this endpoint. For EndpointSlices of addressType "IPv4" or "IPv6", the values are IP addresses in canonical form. The syntax and semantics of other addressType values are not defined. This must contain at least one address but no more than 100. EndpointSlices generated by the EndpointSlice controller will always have exactly 1 address. No semantics are defined for additional addresses beyond the first, and kube-proxy does not look at them.
conditions	object	EndpointConditions represents the current condition of an endpoint.

Property	Type	Description
deprecatedTopology	object (string)	deprecatedTopology contains topology information part of the v1beta1 API. This field is deprecated, and will be removed when the v1beta1 API is removed (no sooner than kubernetes v1.24). While this field can hold values, it is not writable through the v1 API, and any attempts to write to it will be silently ignored. Topology information can be found in the zone and nodeName fields instead.
hints	object	EndpointHints provides hints describing how an endpoint should be consumed.
hostname	string	hostname of this endpoint. This field may be used by consumers of endpoints to distinguish endpoints from each other (e.g. in DNS names). Multiple endpoints which use the same hostname should be considered fungible (e.g. multiple A values in DNS). Must be lowercase and pass DNS Label (RFC 1123) validation.
nodeName	string	nodeName represents the name of the Node hosting this endpoint. This can be used to determine endpoints local to a Node.
targetRef	ObjectReference	targetRef is a reference to a Kubernetes object that represents this endpoint.
zone	string	zone is the name of the Zone this endpoint exists in.

12.1.3. .endpoints[].conditions

Description

EndpointConditions represents the current condition of an endpoint.

Type

object

Property	Type	Description
ready	boolean	ready indicates that this endpoint is ready to receive traffic, according to whatever system is managing the endpoint. A nil value should be interpreted as "true". In general, an endpoint should be marked ready if it is serving and not terminating, though this can be overridden in some cases, such as when the associated Service has set the publishNotReadyAddresses flag.
serving	boolean	serving indicates that this endpoint is able to receive traffic, according to whatever system is managing the endpoint. For endpoints backed by pods, the EndpointSlice controller will mark the endpoint as serving if the pod's Ready condition is True. A nil value should be interpreted as "true".
terminating	boolean	terminating indicates that this endpoint is terminating. A nil value should be interpreted as "false".

12.1.4. .endpoints[].hints

Description

EndpointHints provides hints describing how an endpoint should be consumed.

Type

object

Property	Type	Description
forNodes	array	forNodes indicates the node(s) this endpoint should be consumed by when using topology aware routing. May contain a maximum of 8 entries. This is an Alpha feature and is only used when the PreferSameTrafficDistribution feature gate is enabled.

Property	Type	Description
forNodes[]	object	ForNode provides information about which nodes should consume this endpoint.
forZones	array	forZones indicates the zone(s) this endpoint should be consumed by when using topology aware routing. May contain a maximum of 8 entries.
forZones[]	object	ForZone provides information about which zones should consume this endpoint.

12.1.5. .endpoints[].hints.forNodes

Description

forNodes indicates the node(s) this endpoint should be consumed by when using topology aware routing. May contain a maximum of 8 entries. This is an Alpha feature and is only used when the PreferSameTrafficDistribution feature gate is enabled.

Type

array

12.1.6. .endpoints[].hints.forNodes[]

Description

ForNode provides information about which nodes should consume this endpoint.

Type

object

Required

- **name**

Property	Type	Description
name	string	name represents the name of the node.

12.1.7. .endpoints[].hints.forZones

Description

forZones indicates the zone(s) this endpoint should be consumed by when using topology aware routing. May contain a maximum of 8 entries.

Type**array**

12.1.8. .endpoints[].hints.forZones[]

Description

ForZone provides information about which zones should consume this endpoint.

Type**object****Required**

- **name**

Property	Type	Description
name	string	name represents the name of the zone.

12.1.9. .ports

Description

ports specifies the list of network ports exposed by each endpoint in this slice. Each port must have a unique name. Each slice may include a maximum of 100 ports. Services always have at least 1 port, so EndpointSlices generated by the EndpointSlice controller will likewise always have at least 1 port. EndpointSlices used for other purposes may have an empty ports list.

Type**array**

12.1.10. .ports[]

Description

EndpointPort represents a Port used by an EndpointSlice

Type**object**

Property	Type	Description

Property	Type	Description
appProtocol	string	The application protocol for this port. This is used as a hint for implementations to offer richer behavior for protocols that they understand. This field follows standard Kubernetes label syntax. Valid values are either: * Un-prefixed protocol names - reserved for IANA standard service names (as per RFC-6335 and https://www.iana.org/assignments/service-names/). * Kubernetes-defined prefixed names: * 'kubernetes.io/h2c' - HTTP/2 prior knowledge over cleartext as described in https://www.rfc-editor.org/rfc/rfc9113.html#name-starting-http-2-with-prior- * 'kubernetes.io/ws' - WebSocket over cleartext as described in https://www.rfc-editor.org/rfc/rfc6455 * 'kubernetes.io/wss' - WebSocket over TLS as described in https://www.rfc-editor.org/rfc/rfc6455 * Other protocols should use implementation-defined prefixed names such as mycompany.com/my-custom-protocol.
name	string	name represents the name of this port. All ports in an EndpointSlice must have a unique name. If the EndpointSlice is derived from a Kubernetes service, this corresponds to the Service.ports[].name. Name must either be an empty string or pass DNS_LABEL validation: * must be no more than 63 characters long. * must consist of lower case alphanumeric characters or '-'. * * must start and end with an alphanumeric character. Default is empty string.

Property	Type	Description
port	integer	port represents the port number of the endpoint. If the EndpointSlice is derived from a Kubernetes service, this must be set to the service's target port. EndpointSlices used for other purposes may have a nil port.
protocol	string	protocol represents the IP protocol for this port. Must be UDP, TCP, or SCTP. Default is TCP. Possible enum values: - "SCTP" is the SCTP protocol. - "TCP" is the TCP protocol. - "UDP" is the UDP protocol.

12.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/discovery.k8s.io/v1/endpointslices**
 - **GET**: list or watch objects of kind EndpointSlice
- **/apis/discovery.k8s.io/v1/watch/endpointslices**
 - **GET**: watch individual changes to a list of EndpointSlice. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/discovery.k8s.io/v1/namespaces/{namespace}/endpointslices**
 - **DELETE**: delete collection of EndpointSlice
 - **GET**: list or watch objects of kind EndpointSlice
 - **POST**: create an EndpointSlice
- **/apis/discovery.k8s.io/v1/watch/namespaces/{namespace}/endpointslices**
 - **GET**: watch individual changes to a list of EndpointSlice. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/discovery.k8s.io/v1/namespaces/{namespace}/endpointslices/{name}**
 - **DELETE**: delete an EndpointSlice
 - **GET**: read the specified EndpointSlice
 - **PATCH**: partially update the specified EndpointSlice

- **PUT**: replace the specified EndpointSlice
- **/apis/discovery.k8s.io/v1/watch/namespaces/{namespace}/endpointslices/{name}**
 - **GET**: watch changes to an object of kind EndpointSlice. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

12.2.1. /apis/discovery.k8s.io/v1/endpointslices

HTTP method

GET

Description

list or watch objects of kind EndpointSlice

Table 12.1. HTTP responses

HTTP code	Reponse body
200 - OK	EndpointSliceList schema
401 - Unauthorized	Empty

12.2.2. /apis/discovery.k8s.io/v1/watch/endpointslices

HTTP method

GET

Description

watch individual changes to a list of EndpointSlice. deprecated: use the 'watch' parameter with a list operation instead.

Table 12.2. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

12.2.3. /apis/discovery.k8s.io/v1/namespaces/{namespace}/endpointslices

HTTP method

DELETE

Description

delete collection of EndpointSlice

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

Table 12.4. HTTP responses

HTTP code	Response body
200 - OK	Status schema
401 - Unauthorized	Empty

HTTP method**GET****Description**

list or watch objects of kind EndpointSlice

Table 12.5. HTTP responses

HTTP code	Response body
200 - OK	EndpointSliceList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an EndpointSlice

Table 12.6. 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 12.7. Body parameters

Parameter	Type	Description
body	EndpointSlice schema	

Table 12.8. HTTP responses

HTTP code	Response body
200 - OK	EndpointSlice schema
201 - Created	EndpointSlice schema
202 - Accepted	EndpointSlice schema
401 - Unauthorized	Empty

12.2.4. /apis/discovery.k8s.io/v1/watch/namespaces/{namespace}/endpointslices

HTTP method

GET

Description

watch individual changes to a list of EndpointSlice. deprecated: use the 'watch' parameter with a list operation instead.

Table 12.9. HTTP responses

HTTP code	Response body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

12.2.5. /apis/discovery.k8s.io/v1/namespaces/{namespace}/endpointslices/{name}

Table 12.10. Global path parameters

Parameter	Type	Description
name	string	name of the EndpointSlice

HTTP method

DELETE

Description

delete an EndpointSlice

Table 12.11. 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

Table 12.12. HTTP responses

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

HTTP method

GET

Description

read the specified EndpointSlice

Table 12.13. HTTP responses

HTTP code	Response body
200 - OK	EndpointSlice schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified EndpointSlice

Table 12.14. 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 12.15. HTTP responses

HTTP code	Response body
200 - OK	EndpointSlice schema
201 - Created	EndpointSlice schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified EndpointSlice

Table 12.16. 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 12.17. Body parameters

Parameter	Type	Description
body	EndpointSlice schema	

Table 12.18. HTTP responses

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

12.2.6. /apis/discovery.k8s.io/v1/watch/namespaces/{namespace}/endpointslices/{name}

Table 12.19. Global path parameters

Parameter	Type	Description
name	string	name of the EndpointSlice

HTTP method

GET

Description

watch changes to an object of kind EndpointSlice. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 12.20. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

CHAPTER 13. EGRESSROUTER [NETWORK.OPERATOR.OPENSIFT.IO/V1]

Description

EgressRouter is a feature allowing the user to define an egress router that acts as a bridge between pods and external systems. The egress router runs a service that redirects egress traffic originating from a pod or a group of pods to a remote external system or multiple destinations as per configuration.

It is consumed by the cluster-network-operator. More specifically, given an EgressRouter CR with <name>, the CNO will create and manage:

- A service called <name>
- An egress pod called <name>
- A NAD called <name>

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

EgressRouter is a single egressrouter pod configuration object.

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

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	Specification of the desired egress router.
status	object	Observed status of EgressRouter.

13.1.1. .spec

Description

Specification of the desired egress router.

Type

object

Required

- **addresses**
- **mode**
- **networkInterface**

Property	Type	Description
addresses	array	List of IP addresses to configure on the pod's secondary interface.
addresses[]	object	EgressRouterAddress contains a pair of IP CIDR and gateway to be configured on the router's interface
mode	string	mode depicts the mode that is used for the egress router. The default mode is "Redirect" and is the only supported mode currently.

Property	Type	Description
networkInterface	object	Specification of interface to create/use. The default is macvlan. Currently only macvlan is supported.
redirect	object	redirect represents the configuration parameters specific to redirect mode.

13.1.2. .spec.addresses

Description

List of IP addresses to configure on the pod's secondary interface.

Type

array

13.1.3. .spec.addresses[]

Description

EgressRouterAddress contains a pair of IP CIDR and gateway to be configured on the router's interface

Type

object

Required

- **ip**

Property	Type	Description
gateway	string	IP address of the next-hop gateway, if it cannot be automatically determined. Can be IPv4 or IPv6.
ip	string	ip is the address to configure on the router's interface. Can be IPv4 or IPv6.

13.1.4. .spec.networkInterface

Description

Specification of interface to create/use. The default is macvlan. Currently only macvlan is supported.

Type

object

Property	Type	Description
macvlan	object	Arguments specific to the interfaceType macvlan

13.1.5. .spec.networkInterface.macvlan**Description**

Arguments specific to the interfaceType macvlan

Type

object

Required

- **mode**

Property	Type	Description
master	string	Name of the master interface. Need not be specified if it can be inferred from the IP address.
mode	string	mode depicts the mode that is used for the macvlan interface; one of Bridge Private VEPA Passthru. The default mode is "Bridge".

13.1.6. .spec.redirect**Description**

redirect represents the configuration parameters specific to redirect mode.

Type

object

Property	Type	Description

Property	Type	Description
fallbackIP	string	fallbackIP specifies the remote destination's IP address. Can be IPv4 or IPv6. If no redirect rules are specified, all traffic from the router are redirected to this IP. If redirect rules are specified, then any connections on any other port (undefined in the rules) on the router will be redirected to this IP. If redirect rules are specified and no fallback IP is provided, connections on other ports will simply be rejected.
redirectRules	array	List of L4RedirectRules that define the DNAT redirection from the pod to the destination in redirect mode.
redirectRules[]	object	L4RedirectRule defines a DNAT redirection from a given port to a destination IP and port.

13.1.7. .spec.redirect.redirectRules

Description

List of L4RedirectRules that define the DNAT redirection from the pod to the destination in redirect mode.

Type

array

13.1.8. .spec.redirect.redirectRules[]

Description

L4RedirectRule defines a DNAT redirection from a given port to a destination IP and port.

Type

object

Required

- **destinationIP**
- **port**
- **protocol**

Property	Type	Description
destinationIP	string	IP specifies the remote destination's IP address. Can be IPv4 or IPv6.
port	integer	port is the port number to which clients should send traffic to be redirected.
protocol	string	protocol can be TCP, SCTP or UDP.
targetPort	integer	targetPort allows specifying the port number on the remote destination to which the traffic gets redirected to. If unspecified, the value from "Port" is used.

13.1.9. .status

Description

Observed status of EgressRouter.

Type

object

Required

- **conditions**

Property	Type	Description
conditions	array	Observed status of the egress router
conditions[]	object	EgressRouterStatusCondition represents the state of the egress router's managed and monitored components.

13.1.10. .status.conditions

Description

Observed status of the egress router

Type

array

13.1.11. .status.conditions[]

Description

EgressRouterStatusCondition represents the state of the egress router's managed and monitored components.

Type

object

Required

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

Property	Type	Description
lastTransitionTime	..	lastTransitionTime is the time of the last update to the current status property.
message	string	message provides additional information about the current condition. This is only to be consumed by humans. It may contain Line Feed characters (U+000A), which should be rendered as new lines.
reason	string	reason is the CamelCase reason for the condition's current status.
status	string	status of the condition, one of True, False, Unknown.
type	string	type specifies the aspect reported by this condition; one of Available, Progressing, Degraded

13.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/network.operator.openshift.io/v1/egressrouters**
 - **GET**: list objects of kind EgressRouter
- **/apis/network.operator.openshift.io/v1/namespaces/{namespace}/egressrouters**
 - **DELETE**: delete collection of EgressRouter
 - **GET**: list objects of kind EgressRouter
 - **POST**: create an EgressRouter

- **/apis/network.operator.openshift.io/v1/namespaces/{namespace}/egressrouters/{name}**
 - **DELETE**: delete an EgressRouter
 - **GET**: read the specified EgressRouter
 - **PATCH**: partially update the specified EgressRouter
 - **PUT**: replace the specified EgressRouter
- **/apis/network.operator.openshift.io/v1/namespaces/{namespace}/egressrouters/{name}/status**
 - **GET**: read status of the specified EgressRouter
 - **PATCH**: partially update status of the specified EgressRouter
 - **PUT**: replace status of the specified EgressRouter

13.2.1. /apis/network.operator.openshift.io/v1/egressrouters

HTTP method

GET

Description

list objects of kind EgressRouter

Table 13.1. HTTP responses

HTTP code	Response body
200 - OK	EgressRouterList schema
401 - Unauthorized	Empty

13.2.2. /apis/network.operator.openshift.io/v1/namespaces/{namespace}/egressrouters

HTTP method

DELETE

Description

delete collection of EgressRouter

Table 13.2. HTTP responses

HTTP code	Response body
200 - OK	Status schema
401 - Unauthorized	Empty

HTTP method

GET**Description**

list objects of kind EgressRouter

Table 13.3. HTTP responses

HTTP code	Reponse body
200 - OK	EgressRouterList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an EgressRouter

Table 13.4. 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 13.5. Body parameters

Parameter	Type	Description
body	EgressRouter schema	

Table 13.6. HTTP responses

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

13.2.3. /apis/network.operator.openshift.io/v1/namespaces/{namespace}/egressrouter

Table 13.7. Global path parameters

Parameter	Type	Description
name	string	name of the EgressRouter

HTTP method

DELETE

Description

delete an EgressRouter

Table 13.8. 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 13.9. 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 EgressRouter

Table 13.10. HTTP responses

HTTP code	Reponse body
200 - OK	EgressRouter schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified EgressRouter

Table 13.11. 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 13.12. HTTP responses

HTTP code	Response body
200 - OK	EgressRouter schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified EgressRouter

Table 13.13. 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 13.14. Body parameters

Parameter	Type	Description
body	EgressRouter schema	

Table 13.15. HTTP responses

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

13.2.4. /apis/network.operator.openshift.io/v1/namespaces/{namespace}/egressrout

Table 13.16. Global path parameters

Parameter	Type	Description
name	string	name of the EgressRouter

HTTP method

GET

Description

read status of the specified EgressRouter

Table 13.17. HTTP responses

HTTP code	Response body
200 - OK	EgressRouter schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified EgressRouter

Table 13.18. 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 13.19. HTTP responses

HTTP code	Reponse body
200 - OK	EgressRouter schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified EgressRouter

Table 13.20. 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 13.21. Body parameters

Parameter	Type	Description
body	EgressRouter schema	

Table 13.22. HTTP responses

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

CHAPTER 14. GRPCROUTE

[GATEWAY.NETWORKING.K8S.IO/V1]

Description

GRPCRoute provides a way to route gRPC requests. This includes the capability to match requests by hostname, gRPC service, gRPC method, or HTTP/2 header. Filters can be used to specify additional processing steps. Backends specify where matching requests will be routed.

GRPCRoute falls under extended support within the Gateway API. Within the following specification, the word "MUST" indicates that an implementation supporting GRPCRoute must conform to the indicated requirement, but an implementation not supporting this route type need not follow the requirement unless explicitly indicated.

Implementations supporting **GRPCRoute** with the **HTTPS ProtocolType** MUST accept HTTP/2 connections without an initial upgrade from HTTP/1.1, i.e. via ALPN. If the implementation does not support this, then it MUST set the "Accepted" condition to "False" for the affected listener with a reason of "UnsupportedProtocol". Implementations MAY also accept HTTP/2 connections with an upgrade from HTTP/1.

Implementations supporting **GRPCRoute** with the **HTTP ProtocolType** MUST support HTTP/2 over cleartext TCP (h2c, <https://www.rfc-editor.org/rfc/rfc7540#section-3.1>) without an initial upgrade from HTTP/1.1, i.e. with prior knowledge (<https://www.rfc-editor.org/rfc/rfc7540#section-3.4>). If the implementation does not support this, then it MUST set the "Accepted" condition to "False" for the affected listener with a reason of "UnsupportedProtocol". Implementations MAY also accept HTTP/2 connections with an upgrade from HTTP/1, i.e. without prior knowledge.

Type

object

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

Property	Type	Description
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 defines the desired state of GRPCRoute.
status	object	Status defines the current state of GRPCRoute.

14.1.1. .spec

Description

Spec defines the desired state of GRPCRoute.

Type

object

Property	Type	Description
hostnames	array (string)	Hostnames defines a set of hostnames to match against the GRPC Host header to select a GRPCRoute to process the request. This matches the RFC 1123 definition of a hostname with 2 notable exceptions: 1. IPs are not allowed. 2. A hostname may be prefixed with a wildcard label (*). The wildcard label MUST appear by itself as the first label.

Property	Type	Description
		If a hostname is specified by both the Listener and GRPCRoute, there MUST be at least one intersecting hostname for the GRPCRoute to be attached to the Listener. For example: * A Listener with test.example.com as the hostname matches GRPCRoutes that have either not specified any hostnames, or have specified at least one of test.example.com or *.example.com. * A Listener with *.example.com as the hostname matches GRPCRoutes that have either not specified any hostnames or have specified at least one hostname that matches the Listener hostname. For example, test.example.com and *.example.com would both match. On the other hand, example.com and test.example.net would not match. Hostnames that are prefixed with a wildcard label (*.) are interpreted as a suffix match. That means that a match for *.example.com would match both test.example.com, and foo.test.example.com, but not example.com. If both the Listener and GRPCRoute have specified hostnames, any GRPCRoute hostnames that do not match the Listener hostname MUST be ignored. For example, if a Listener specified *.example.com, and the GRPCRoute specified test.example.com and test.example.net, test.example.net MUST NOT be considered for a match. If both the Listener and GRPCRoute have specified hostnames, and none match with the criteria above, then the GRPCRoute MUST NOT be accepted by the implementation. The implementation MUST raise

Property	Type	Description
		an 'Accepted' Condition with a status of 'False' in the corresponding RouteParentStatus. If a Route (A) of type HTTPRoute or GRPCRoute is attached to a Listener and that listener already has another Route (B) of the other type attached and the intersection of the hostnames of A and B is non-empty, then the implementation MUST accept exactly one of these two routes, determined by the following criteria, in order: * The oldest Route based on creation timestamp. * The Route appearing first in alphabetical order by "{namespace}/{name}". The rejected Route MUST raise an 'Accepted' condition with a status of 'False' in the corresponding RouteParentStatus. Support: Core
parentRefs	array	ParentRefs references the resources (usually Gateways) that a Route wants to be attached to. Note that the referenced parent resource needs to allow this for the attachment to be complete. For Gateways, that means the Gateway needs to allow attachment from Routes of this kind and namespace. For Services, that means the Service must either be in the same namespace for a "producer" route, or the mesh implementation must support and allow "consumer" routes for the referenced Service. ReferenceGrant is not applicable for governing ParentRefs to Services - it is not possible to create a "producer" route for a Service in a different namespace from the Route. There are two kinds of parent resources with "Core" support:

Property	Type	* Gateway (Gateway conformance profile) * Service Description
		(Mesh conformance profile, ClusterIP Services only) This API may be extended in the future to support additional kinds of parent resources. ParentRefs must be <i>distinct</i>. This means either that: * They select different objects. If this is the case, then parentRef entries are distinct. In terms of fields, this means that the multi-part key defined by group, kind, namespace, and name must be unique across all parentRef entries in the Route. * They do not select different objects, but for each optional field used, each ParentRef that selects the same object must set the same set of optional fields to different values. If one ParentRef sets a combination of optional fields, all must set the same combination. Some examples: * If one ParentRef sets sectionName, all ParentRefs referencing the same object must also set sectionName. * If one ParentRef sets port, all ParentRefs referencing the same object must also set port. * If one ParentRef sets sectionName and port, all ParentRefs referencing the same object must also set sectionName and port. It is possible to separately reference multiple distinct objects that may be collapsed by an implementation. For example, some implementations may choose to merge compatible Gateway Listeners together. If that is the case, the list of routes attached to those resources should also be merged. Note that for ParentRefs that cross namespace boundaries, there are specific rules. Cross-

Property	Type	Description
		namespace references are only valid if they are explicitly allowed by something in the namespace they are referring to. For example, Gateway has the AllowedRoutes field, and ReferenceGrant provides a generic way to enable other kinds of cross-namespace reference.
parentRefs[]	object	ParentReference identifies an API object (usually a Gateway) that can be considered a parent of this resource (usually a route). There are two kinds of parent resources with "Core" support: * Gateway (Gateway conformance profile) * Service (Mesh conformance profile, ClusterIP Services only) This API may be extended in the future to support additional kinds of parent resources. The API object must be valid in the cluster; the Group and Kind must be registered in the cluster for this reference to be valid.
rules	array	Rules are a list of GRPC matchers, filters and actions.
rules[]	object	GRPCRouteRule defines the semantics for matching a gRPC request based on conditions (matches), processing it (filters), and forwarding the request to an API object (backendRefs).

14.1.2. .spec.parentRefs

Description

ParentRefs references the resources (usually Gateways) that a Route wants to be attached to. Note that the referenced parent resource needs to allow this for the attachment to be complete. For Gateways, that means the Gateway needs to allow attachment from Routes of this kind and namespace. For Services, that means the Service must either be in the same namespace for a "producer" route, or the mesh implementation must support and allow "consumer" routes for the referenced Service. ReferenceGrant is not applicable for governing ParentRefs to Services – it is not possible to create a "producer" route for a Service in a different namespace from the Route. There are two kinds of parent resources with "Core" support:

- Gateway (Gateway conformance profile)

- Service (Mesh conformance profile, ClusterIP Services only)

This API may be extended in the future to support additional kinds of parent resources.

ParentRefs must be *distinct*. This means either that:

- They select different objects. If this is the case, then parentRef entries are distinct. In terms of fields, this means that the multi-part key defined by **group**, **kind**, **namespace**, and **name** must be unique across all parentRef entries in the Route.
- They do not select different objects, but for each optional field used, each ParentRef that selects the same object must set the same set of optional fields to different values. If one ParentRef sets a combination of optional fields, all must set the same combination.

Some examples:

- If one ParentRef sets **sectionName**, all ParentRefs referencing the same object must also set **sectionName**.
- If one ParentRef sets **port**, all ParentRefs referencing the same object must also set **port**.
- If one ParentRef sets **sectionName** and **port**, all ParentRefs referencing the same object must also set **sectionName** and **port**.

It is possible to separately reference multiple distinct objects that may be collapsed by an implementation. For example, some implementations may choose to merge compatible Gateway Listeners together. If that is the case, the list of routes attached to those resources should also be merged.

Note that for ParentRefs that cross namespace boundaries, there are specific rules. Cross-namespace references are only valid if they are explicitly allowed by something in the namespace they are referring to. For example, Gateway has the AllowedRoutes field, and ReferenceGrant provides a generic way to enable other kinds of cross-namespace reference.

Type

array

14.1.3. .spec.parentRefs[]

Description

ParentReference identifies an API object (usually a Gateway) that can be considered a parent of this resource (usually a route). There are two kinds of parent resources with "Core" support:

- Gateway (Gateway conformance profile)
- Service (Mesh conformance profile, ClusterIP Services only)

This API may be extended in the future to support additional kinds of parent resources.

The API object must be valid in the cluster; the Group and Kind must be registered in the cluster for this reference to be valid.

Type

object

Required

- **name**

Property	Type	Description
group	string	Group is the group of the referent. When unspecified, "gateway.networking.k8s.io" is inferred. To set the core API group (such as for a "Service" kind referent), Group must be explicitly set to "" (empty string). Support: Core
kind	string	Kind is kind of the referent. There are two kinds of parent resources with "Core" support: * Gateway (Gateway conformance profile) * Service (Mesh conformance profile, ClusterIP Services only) Support for other resources is Implementation-Specific.
name	string	Name is the name of the referent. Support: Core
namespace	string	Namespace is the namespace of the referent. When unspecified, this refers to the local namespace of the Route. Note that there are specific rules for ParentRefs which cross namespace boundaries. Cross-namespace references are only valid if they are explicitly allowed by something in the namespace they are referring to. For example: Gateway has the AllowedRoutes field, and ReferenceGrant provides a generic way to enable any other kind of cross-namespace reference. Support: Core

Property	Type	Description
port	integer	Port is the network port this Route targets. It can be interpreted differently based on the type of parent resource. When the parent resource is a Gateway, this targets all listeners listening on the specified port that also support this kind of Route (and select this Route). It's not recommended to set Port unless the networking behaviors specified in a Route must apply to a specific port as opposed to a listener(s) whose port(s) may be changed. When both Port and SectionName are specified, the name and port of the selected listener must match both specified values. Implementations MAY choose to support other parent resources. Implementations supporting other types of parent resources MUST clearly document how/if Port is interpreted. For the purpose of status, an attachment is considered successful as long as the parent resource accepts it partially. For example, Gateway listeners can restrict which Routes can attach to them by Route kind, namespace, or hostname. If 1 of 2 Gateway listeners accept attachment from the referencing Route, the Route MUST be considered successfully attached. If no Gateway listeners accept attachment from this Route, the Route MUST be considered detached from the Gateway. Support: Extended

Property	Type	Description
sectionName	string	SectionName is the name of a section within the target resource. In the following resources, SectionName is interpreted as the following: * Gateway: Listener name. When both Port (experimental) and SectionName are specified, the name and port of the selected listener must match both specified values. * Service: Port name. When both Port (experimental) and SectionName are specified, the name and port of the selected listener must match both specified values. Implementations MAY choose to support attaching Routes to other resources. If that is the case, they MUST clearly document how SectionName is interpreted. When unspecified (empty string), this will reference the entire resource. For the purpose of status, an attachment is considered successful if at least one section in the parent resource accepts it. For example, Gateway listeners can restrict which Routes can attach to them by Route kind, namespace, or hostname. If 1 of 2 Gateway listeners accept attachment from the referencing Route, the Route MUST be considered successfully attached. If no Gateway listeners accept attachment from this Route, the Route MUST be considered detached from the Gateway. Support: Core

14.1.4. .spec.rules

Description

Rules are a list of GRPC matchers, filters and actions.

Type

array

14.1.5. .spec.rules[]

Description

GRPCRouteRule defines the semantics for matching a gRPC request based on conditions (matches), processing it (filters), and forwarding the request to an API object (backendRefs).

Type

object

Property	Type	Description

Property	Type	Description
backendRefs	array	BackendRefs defines the backend(s) where matching requests should be sent. Failure behavior here depends on how many BackendRefs are specified and how many are invalid. If all entries in BackendRefs are invalid, and there are also no filters specified in this route rule, all traffic which matches this rule MUST receive an UNAVAILABLE status. See the GRPCBackendRef definition for the rules about what makes a single GRPCBackendRef invalid. When a GRPCBackendRef is invalid, UNAVAILABLE statuses MUST be returned for requests that would have otherwise been routed to an invalid backend. If multiple backends are specified, and some are invalid, the proportion of requests that would otherwise have been routed to an invalid backend MUST receive an UNAVAILABLE status. For example, if two backends are specified with equal weights, and one is invalid, 50 percent of traffic MUST receive an UNAVAILABLE status. Implementations may choose how that 50 percent is determined. Support: Core for Kubernetes Service Support: Implementation-specific for any other resource Support for weight: Core

Property	Type	Description
backendRefs[]	object	GRPCBackendRef defines how a GRPCRoute forwards a gRPC request. Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details.

Property	Type	Description
filters	array	Filters define the filters that are applied to requests that match this rule. The effects of ordering of multiple behaviors are currently unspecified. This can change in the future based on feedback during the alpha stage. Conformance-levels at this level are defined based on the type of filter: - ALL core filters MUST be supported by all implementations that support GRPCRoute. - Implementers are encouraged to support extended filters. - Implementation-specific custom filters have no API guarantees across implementations. Specifying the same filter multiple times is not supported unless explicitly indicated in the filter. If an implementation cannot support a combination of filters, it must clearly document that limitation. In cases where incompatible or unsupported filters are specified and cause the Accepted condition to be set to status False, implementations may use the IncompatibleFilters reason to specify this configuration error. Support: Core

Property	Type	Description
filters[]	object	GRPCRouteFilter defines processing steps that must be completed during the request or response lifecycle. GRPCRouteFilters are meant as an extension point to express processing that may be done in Gateway implementations. Some examples include request or response modification, implementing authentication strategies, rate-limiting, and traffic shaping. API guarantee/conformance is defined based on the type of the filter.
matches	array	Matches define conditions used for matching the rule against incoming gRPC requests. Each match is independent, i.e. this rule will be matched if any one of the matches is satisfied. For example, take the following matches configuration: <pre>matches: - method: service: foo.bar headers: values: version: 2 - method: service: foo.bar.v2</pre> For a request to match against this rule, it MUST satisfy EITHER of the two conditions: - service of foo.bar AND contains the header version: 2 - service of foo.bar.v2 See the documentation for GRPCRouteMatch on how to specify multiple match conditions to be ANDed together. If no matches are specified, the implementation MUST match every gRPC request. Proxy or Load Balancer routing configuration generated from GRPCRoutes MUST prioritize rules based on the following criteria, continuing on ties. Merging MUST not be done

Property	Type	Description
		between GRPCRoutes and HTTPRoutes. Precedence MUST be given to the rule with the largest number of: * Characters in a matching non-wildcard hostname. * Characters in a matching hostname. * Characters in a matching service. * Characters in a matching method. * Header matches. If ties still exist across multiple Routes, matching precedence MUST be determined in order of the following criteria, continuing on ties: * The oldest Route based on creation timestamp. * The Route appearing first in alphabetical order by "{namespace}/{name}". If ties still exist within the Route that has been given precedence, matching precedence MUST be granted to the first matching rule meeting the above criteria.
matches[]	object	GRPCRouteMatch defines the predicate used to match requests to a given action. Multiple match types are ANDed together, i.e. the match will evaluate to true only if all conditions are satisfied. For example, the match below will match a gRPC request only if its service is foo AND it contains the version: v1 header: <pre>matches: - method: type: Exact service: "foo" headers: - name: "version" value "v1"</pre>

14.1.6. .spec.rules[].backendRefs

Description

BackendRefs defines the backend(s) where matching requests should be sent.

Failure behavior here depends on how many BackendRefs are specified and how many are invalid.

If **all** entries in BackendRefs are invalid, and there are also no filters specified in this route rule, **all** traffic which matches this rule MUST receive an **UNAVAILABLE** status.

See the GRPCBackendRef definition for the rules about what makes a single GRPCBackendRef invalid.

When a GRPCBackendRef is invalid, **UNAVAILABLE** statuses MUST be returned for requests that would have otherwise been routed to an invalid backend. If multiple backends are specified, and some are invalid, the proportion of requests that would otherwise have been routed to an invalid backend MUST receive an **UNAVAILABLE** status.

For example, if two backends are specified with equal weights, and one is invalid, 50 percent of traffic MUST receive an **UNAVAILABLE** status. Implementations may choose how that 50 percent is determined.

Support: Core for Kubernetes Service

Support: Implementation-specific for any other resource

Support for weight: Core

Type

array

14.1.7. .spec.rules[].backendRefs[]

Description

GRPCBackendRef defines how a GRPCRoute forwards a gRPC request.

Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details.

Type

object

Required

- **name**

Property	Type	Description
filters	array	Filters defined at this level MUST be executed if and only if the request is being forwarded to the backend defined here. Support: Implementation-specific (For broader support of filters, use the Filters field in GRPCRouteRule.)

Property	Type	Description
filters[]	object	GRPCRouteFilter defines processing steps that must be completed during the request or response lifecycle. GRPCRouteFilters are meant as an extension point to express processing that may be done in Gateway implementations. Some examples include request or response modification, implementing authentication strategies, rate-limiting, and traffic shaping. API guarantee/conformance is defined based on the type of the filter.
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is the Kubernetes resource kind of the referent. For example "Service". Defaults to "Service" when not specified. ExternalName services can refer to CNAME DNS records that may live outside of the cluster and as such are difficult to reason about in terms of conformance. They also may not be safe to forward to (see CVE-2021-25740 for more information). Implementations SHOULD NOT support ExternalName Services. Support: Core (Services with a type other than ExternalName) Support: Implementation-specific (Services with type ExternalName)
name	string	Name is the name of the referent.

Property	Type	Description
namespace	string	Namespace is the namespace of the backend. When unspecified, the local namespace is inferred. Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. Support: Core
port	integer	Port specifies the destination port number to use for this resource. Port is required when the referent is a Kubernetes Service. In this case, the port number is the service port number, not the target port. For other resources, destination port might be derived from the referent resource or this field.
weight	integer	Weight specifies the proportion of requests forwarded to the referenced backend. This is computed as $\text{weight} / (\text{sum of all weights in this BackendRefs list})$. For non-zero values, there may be some epsilon from the exact proportion defined here depending on the precision an implementation supports. Weight is not a percentage and the sum of weights does not need to equal 100. If only one backend is specified and it has a weight greater than 0, 100% of the traffic is forwarded to that backend. If weight is set to 0, no traffic should be forwarded for this entry. If unspecified, weight defaults to 1. Support for this field varies based on the context where used.

14.1.8. .spec.rules[].backendRefs[].filters

Description

Filters defined at this level MUST be executed if and only if the request is being forwarded to the backend defined here.

Support: Implementation-specific (For broader support of filters, use the Filters field in GRPCRouteRule.)

Type

array

14.1.9. .spec.rules[].backendRefs[].filters[]

Description

GRPCRouteFilter defines processing steps that must be completed during the request or response lifecycle. GRPCRouteFilters are meant as an extension point to express processing that may be done in Gateway implementations. Some examples include request or response modification, implementing authentication strategies, rate-limiting, and traffic shaping. API guarantee/conformance is defined based on the type of the filter.

Type

object

Required

- **type**

Property	Type	Description
extensionRef	object	ExtensionRef is an optional, implementation-specific extension to the "filter" behavior. For example, resource "myroutefilter" in group "networking.example.net"). ExtensionRef MUST NOT be used for core and extended filters. Support: Implementation-specific This filter can be used multiple times within the same rule.
requestHeaderModifier	object	RequestHeaderModifier defines a schema for a filter that modifies request headers. Support: Core

Property	Type	Description
requestMirror	object	RequestMirror defines a schema for a filter that mirrors requests. Requests are sent to the specified destination, but responses from that destination are ignored. This filter can be used multiple times within the same rule. Note that not all implementations will be able to support mirroring to multiple backends. Support: Extended
responseHeaderModifier	object	ResponseHeaderModifier defines a schema for a filter that modifies response headers. Support: Extended

Property	Type	Description
type	string	Type identifies the type of filter to apply. As with other API fields, types are classified into three conformance levels: - Core: Filter types and their corresponding configuration defined by "Support: Core" in this package, e.g. "RequestHeaderModifier". All implementations supporting GRPCRoute MUST support core filters. - Extended: Filter types and their corresponding configuration defined by "Support: Extended" in this package, e.g. "RequestMirror". Implementers are encouraged to support extended filters. - Implementation-specific: Filters that are defined and supported by specific vendors. In the future, filters showing convergence in behavior across multiple implementations will be considered for inclusion in extended or core conformance levels. Filter-specific configuration for such filters is specified using the ExtensionRef field. Type MUST be set to "ExtensionRef" for custom filters. Implementers are encouraged to define custom implementation types to extend the core API with implementation-specific behavior. If a reference to a custom filter type cannot be resolved, the filter MUST NOT be skipped. Instead, requests that would have been processed by that filter MUST receive a HTTP error response.

14.1.10. .spec.rules[].backendRefs[].filters[].extensionRef

Description

ExtensionRef is an optional, implementation-specific extension to the "filter" behavior. For example, resource "myroutefilter" in group "networking.example.net"). ExtensionRef MUST NOT be used for core and extended filters.

Support: Implementation-specific

This filter can be used multiple times within the same rule.

Type

object

Required

- **group**
- **kind**
- **name**

Property	Type	Description
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is kind of the referent. For example "HTTPRoute" or "Service".
name	string	Name is the name of the referent.

14.1.11. .spec.rules[].backendRefs[].filters[].requestHeaderModifier

Description

RequestHeaderModifier defines a schema for a filter that modifies request headers.

Support: Core

Type

object

Property	Type	Description

Property	Type	Description
add	array	Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name. Input: GET /foo HTTP/1.1 my-header: foo Config: add: - name: "my-header" value: "bar,baz" Output: GET /foo HTTP/1.1 my-header: foo,bar,baz
add[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.
remove	array (string)	Remove the given header(s) from the HTTP request before the action. The value of Remove is a list of HTTP header names. Note that the header names are case-insensitive (see https://datatracker.ietf.org/doc/html/rfc2616#section-4.2). Input: GET /foo HTTP/1.1 my-header1: foo my-header2: bar my-header3: baz Config: remove: ["my-header1", "my-header3"] Output: GET /foo HTTP/1.1 my-header2: bar
set	array	Set overwrites the request with the given header (name, value) before the action. Input: GET /foo HTTP/1.1 my-header: foo Config: set: - name: "my-header" value: "bar" Output: GET /foo HTTP/1.1 my-header: bar

Property	Type	Description
set[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

14.1.12. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.add

Description

Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name.

Input: GET /foo HTTP/1.1 my-header: foo

Config: add: - name: "my-header" value: "bar,baz"

Output: GET /foo HTTP/1.1 my-header: foo,bar,baz

Type

array

14.1.13. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.add[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

14.1.14. `.spec.rules[].backendRefs[].filters[].requestHeaderModifier.set`

Description

Set overwrites the request with the given header (name, value) before the action.

Input: GET /foo HTTP/1.1 my-header: foo

Config: set: - name: "my-header" value: "bar"

Output: GET /foo HTTP/1.1 my-header: bar

Type

array

14.1.15. `.spec.rules[].backendRefs[].filters[].requestHeaderModifier.set[]`

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

14.1.16. .spec.rules[].backendRefs[].filters[].requestMirror

Description

RequestMirror defines a schema for a filter that mirrors requests. Requests are sent to the specified destination, but responses from that destination are ignored.

This filter can be used multiple times within the same rule. Note that not all implementations will be able to support mirroring to multiple backends.

Support: Extended

Type

object

Required

- **backendRef**

Property	Type	Description

Property	Type	Description
backendRef	object	BackendRef references a resource where mirrored requests are sent. Mirrored requests must be sent only to a single destination endpoint within this BackendRef, irrespective of how many endpoints are present within this BackendRef. If the referent cannot be found, this BackendRef is invalid and must be dropped from the Gateway. The controller must ensure the "ResolvedRefs" condition on the Route status is set to status: False and not configure this backend in the underlying implementation. If there is a cross-namespace reference to an existing object that is not allowed by a ReferenceGrant, the controller must ensure the "ResolvedRefs" condition on the Route is set to status: False, with the "RefNotPermitted" reason and not configure this backend in the underlying implementation. In either error case, the Message of the ResolvedRefs Condition should be used to provide more detail about the problem. Support: Extended for Kubernetes Service Support: Implementation-specific for any other resource
fraction	object	Fraction represents the fraction of requests that should be mirrored to BackendRef. Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

Property	Type	Description
percent	integer	Percent represents the percentage of requests that should be mirrored to BackendRef. Its minimum value is 0 (indicating 0% of requests) and its maximum value is 100 (indicating 100% of requests). Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

14.1.17. .spec.rules[].backendRefs[].filters[].requestMirror.backendRef

Description

BackendRef references a resource where mirrored requests are sent.

Mirrored requests must be sent only to a single destination endpoint within this BackendRef, irrespective of how many endpoints are present within this BackendRef.

If the referent cannot be found, this BackendRef is invalid and must be dropped from the Gateway. The controller must ensure the "ResolvedRefs" condition on the Route status is set to **status: False** and not configure this backend in the underlying implementation.

If there is a cross-namespace reference to an **existing** object that is not allowed by a ReferenceGrant, the controller must ensure the "ResolvedRefs" condition on the Route is set to **status: False**, with the "RefNotPermitted" reason and not configure this backend in the underlying implementation.

In either error case, the Message of the **ResolvedRefs** Condition should be used to provide more detail about the problem.

Support: Extended for Kubernetes Service

Support: Implementation-specific for any other resource

Type

object

Required

- **name**

Property	Type	Description

Property	Type	Description
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is the Kubernetes resource kind of the referent. For example "Service". Defaults to "Service" when not specified. ExternalName services can refer to CNAME DNS records that may live outside of the cluster and as such are difficult to reason about in terms of conformance. They also may not be safe to forward to (see CVE-2021-25740 for more information). Implementations SHOULD NOT support ExternalName Services. Support: Core (Services with a type other than ExternalName) Support: Implementation-specific (Services with type ExternalName)
name	string	Name is the name of the referent.
namespace	string	Namespace is the namespace of the backend. When unspecified, the local namespace is inferred. Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. Support: Core

Property	Type	Description
port	integer	Port specifies the destination port number to use for this resource. Port is required when the referent is a Kubernetes Service. In this case, the port number is the service port number, not the target port. For other resources, destination port might be derived from the referent resource or this field.

14.1.18. .spec.rules[].backendRefs[].filters[].requestMirror.fraction

Description

Fraction represents the fraction of requests that should be mirrored to BackendRef.

Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

Type

object

Required

- **numerator**

Property	Type	Description
denominator	integer	
numerator	integer	

14.1.19. .spec.rules[].backendRefs[].filters[].responseHeaderModifier

Description

ResponseHeaderModifier defines a schema for a filter that modifies response headers.

Support: Extended

Type

object

Property	Type	Description

Property	Type	Description
add	array	Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name. Input: GET /foo HTTP/1.1 my-header: foo Config: add: - name: "my-header" value: "bar,baz" Output: GET /foo HTTP/1.1 my-header: foo,bar,baz
add[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.
remove	array (string)	Remove the given header(s) from the HTTP request before the action. The value of Remove is a list of HTTP header names. Note that the header names are case-insensitive (see https://datatracker.ietf.org/doc/html/rfc2616#section-4.2). Input: GET /foo HTTP/1.1 my-header1: foo my-header2: bar my-header3: baz Config: remove: ["my-header1", "my-header3"] Output: GET /foo HTTP/1.1 my-header2: bar
set	array	Set overwrites the request with the given header (name, value) before the action. Input: GET /foo HTTP/1.1 my-header: foo Config: set: - name: "my-header" value: "bar" Output: GET /foo HTTP/1.1 my-header: bar

Property	Type	Description
set[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

14.1.20. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.add

Description

Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name.

Input: GET /foo HTTP/1.1 my-header: foo

Config: add: - name: "my-header" value: "bar,baz"

Output: GET /foo HTTP/1.1 my-header: foo,bar,baz

Type

array

14.1.21. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.add[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

14.1.22. `.spec.rules[].backendRefs[].filters[].responseHeaderModifier.set`

Description

Set overwrites the request with the given header (name, value) before the action.

Input: GET /foo HTTP/1.1 my-header: foo

Config: set: - name: "my-header" value: "bar"

Output: GET /foo HTTP/1.1 my-header: bar

Type

array

14.1.23. `.spec.rules[].backendRefs[].filters[].responseHeaderModifier.set[]`

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

14.1.24. .spec.rules[].filters

Description

Filters define the filters that are applied to requests that match this rule.

The effects of ordering of multiple behaviors are currently unspecified. This can change in the future based on feedback during the alpha stage.

Conformance-levels at this level are defined based on the type of filter:

- ALL core filters MUST be supported by all implementations that support GRPCRoute.
- Implementers are encouraged to support extended filters.
- Implementation-specific custom filters have no API guarantees across implementations.

Specifying the same filter multiple times is not supported unless explicitly indicated in the filter.

If an implementation cannot support a combination of filters, it must clearly document that limitation. In cases where incompatible or unsupported filters are specified and cause the **Accepted** condition to be set to status **False**, implementations may use the **IncompatibleFilters** reason to specify this configuration error.

Support: Core

Type

array

14.1.25. .spec.rules[].filters[]

Description

GRPCRouteFilter defines processing steps that must be completed during the request or response lifecycle. GRPCRouteFilters are meant as an extension point to express processing that may be done in Gateway implementations. Some examples include request or response modification, implementing authentication strategies, rate-limiting, and traffic shaping. API guarantee/conformance is defined based on the type of the filter.

Type

object

Required

- **type**

Property	Type	Description
extensionRef	object	ExtensionRef is an optional, implementation-specific extension to the "filter" behavior. For example, resource "myroutefilter" in group "networking.example.net"). ExtensionRef MUST NOT be used for core and extended filters. Support: Implementation-specific This filter can be used multiple times within the same rule.
requestHeaderModifier	object	RequestHeaderModifier defines a schema for a filter that modifies request headers. Support: Core
requestMirror	object	RequestMirror defines a schema for a filter that mirrors requests. Requests are sent to the specified destination, but responses from that destination are ignored. This filter can be used multiple times within the same rule. Note that not all implementations will be able to support mirroring to multiple backends. Support: Extended

Property	Type	Description
responseHeaderModifier	object	ResponseHeaderModifier defines a schema for a filter that modifies response headers. Support: Extended
type	string	Type identifies the type of filter to apply. As with other API fields, types are classified into three conformance levels: - Core: Filter types and their corresponding configuration defined by "Support: Core" in this package, e.g. "RequestHeaderModifier". All implementations supporting GRPCRoute MUST support core filters. - Extended: Filter types and their corresponding configuration defined by "Support: Extended" in this package, e.g. "RequestMirror". Implementers are encouraged to support extended filters. - Implementation-specific: Filters that are defined and supported by specific vendors. In the future, filters showing convergence in behavior across multiple implementations will be considered for inclusion in extended or core conformance levels. Filter-specific configuration for such filters is specified using the ExtensionRef field. Type MUST be set to "ExtensionRef" for custom filters. Implementers are encouraged to define custom implementation types to extend the core API with implementation-specific behavior. If a reference to a custom filter type cannot be resolved, the filter MUST NOT be skipped. Instead, requests that would have been processed by that filter MUST receive a HTTP error response.

14.1.26. .spec.rules[].filters[].extensionRef

Description

ExtensionRef is an optional, implementation-specific extension to the "filter" behavior. For example, resource "myroutefilter" in group "networking.example.net"). ExtensionRef MUST NOT be used for core and extended filters.

Support: Implementation-specific

This filter can be used multiple times within the same rule.

Type

object

Required

- **group**
- **kind**
- **name**

Property	Type	Description
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is kind of the referent. For example "HTTPRoute" or "Service".
name	string	Name is the name of the referent.

14.1.27. .spec.rules[].filters[].requestHeaderModifier

Description

RequestHeaderModifier defines a schema for a filter that modifies request headers.

Support: Core

Type

object

Property	Type	Description

Property	Type	Description
add	array	Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name. Input: GET /foo HTTP/1.1 my-header: foo Config: add: - name: "my-header" value: "bar,baz" Output: GET /foo HTTP/1.1 my-header: foo,bar,baz
add[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.
remove	array (string)	Remove the given header(s) from the HTTP request before the action. The value of Remove is a list of HTTP header names. Note that the header names are case-insensitive (see https://datatracker.ietf.org/doc/html/rfc2616#section-4.2). Input: GET /foo HTTP/1.1 my-header1: foo my-header2: bar my-header3: baz Config: remove: ["my-header1", "my-header3"] Output: GET /foo HTTP/1.1 my-header2: bar
set	array	Set overwrites the request with the given header (name, value) before the action. Input: GET /foo HTTP/1.1 my-header: foo Config: set: - name: "my-header" value: "bar" Output: GET /foo HTTP/1.1 my-header: bar

Property	Type	Description
set[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

14.1.28. .spec.rules[].filters[].requestHeaderModifier.add

Description

Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name.

Input: GET /foo HTTP/1.1 my-header: foo

Config: add: - name: "my-header" value: "bar,baz"

Output: GET /foo HTTP/1.1 my-header: foo,bar,baz

Type

array

14.1.29. .spec.rules[].filters[].requestHeaderModifier.add[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

14.1.30. .spec.rules[].filters[].requestHeaderModifier.set

Description

Set overwrites the request with the given header (name, value) before the action.

Input: GET /foo HTTP/1.1 my-header: foo

Config: set: - name: "my-header" value: "bar"

Output: GET /foo HTTP/1.1 my-header: bar

Type

array

14.1.31. .spec.rules[].filters[].requestHeaderModifier.set[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

14.1.32. .spec.rules[].filters[].requestMirror

Description

RequestMirror defines a schema for a filter that mirrors requests. Requests are sent to the specified destination, but responses from that destination are ignored.

This filter can be used multiple times within the same rule. Note that not all implementations will be able to support mirroring to multiple backends.

Support: Extended

Type

object

Required

- **backendRef**

Property	Type	Description

Property	Type	Description
backendRef	object	BackendRef references a resource where mirrored requests are sent. Mirrored requests must be sent only to a single destination endpoint within this BackendRef, irrespective of how many endpoints are present within this BackendRef. If the referent cannot be found, this BackendRef is invalid and must be dropped from the Gateway. The controller must ensure the "ResolvedRefs" condition on the Route status is set to status: False and not configure this backend in the underlying implementation. If there is a cross-namespace reference to an existing object that is not allowed by a ReferenceGrant, the controller must ensure the "ResolvedRefs" condition on the Route is set to status: False, with the "RefNotPermitted" reason and not configure this backend in the underlying implementation. In either error case, the Message of the ResolvedRefs Condition should be used to provide more detail about the problem. Support: Extended for Kubernetes Service Support: Implementation-specific for any other resource
fraction	object	Fraction represents the fraction of requests that should be mirrored to BackendRef. Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

Property	Type	Description
percent	integer	Percent represents the percentage of requests that should be mirrored to BackendRef. Its minimum value is 0 (indicating 0% of requests) and its maximum value is 100 (indicating 100% of requests). Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

14.1.33. .spec.rules[].filters[].requestMirror.backendRef

Description

BackendRef references a resource where mirrored requests are sent.

Mirrored requests must be sent only to a single destination endpoint within this BackendRef, irrespective of how many endpoints are present within this BackendRef.

If the referent cannot be found, this BackendRef is invalid and must be dropped from the Gateway. The controller must ensure the "ResolvedRefs" condition on the Route status is set to **status: False** and not configure this backend in the underlying implementation.

If there is a cross-namespace reference to an **existing** object that is not allowed by a ReferenceGrant, the controller must ensure the "ResolvedRefs" condition on the Route is set to **status: False**, with the "RefNotPermitted" reason and not configure this backend in the underlying implementation.

In either error case, the Message of the **ResolvedRefs** Condition should be used to provide more detail about the problem.

Support: Extended for Kubernetes Service

Support: Implementation-specific for any other resource

Type

object

Required

- **name**

Property	Type	Description

Property	Type	Description
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is the Kubernetes resource kind of the referent. For example "Service". Defaults to "Service" when not specified. ExternalName services can refer to CNAME DNS records that may live outside of the cluster and as such are difficult to reason about in terms of conformance. They also may not be safe to forward to (see CVE-2021-25740 for more information). Implementations SHOULD NOT support ExternalName Services. Support: Core (Services with a type other than ExternalName) Support: Implementation-specific (Services with type ExternalName)
name	string	Name is the name of the referent.
namespace	string	Namespace is the namespace of the backend. When unspecified, the local namespace is inferred. Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. Support: Core

Property	Type	Description
port	integer	Port specifies the destination port number to use for this resource. Port is required when the referent is a Kubernetes Service. In this case, the port number is the service port number, not the target port. For other resources, destination port might be derived from the referent resource or this field.

14.1.34. .spec.rules[].filters[].requestMirror.fraction

Description

Fraction represents the fraction of requests that should be mirrored to BackendRef.

Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

Type

object

Required

- **numerator**

Property	Type	Description
denominator	integer	
numerator	integer	

14.1.35. .spec.rules[].filters[].responseHeaderModifier

Description

ResponseHeaderModifier defines a schema for a filter that modifies response headers.

Support: Extended

Type

object

Property	Type	Description

Property	Type	Description
add	array	Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name. Input: GET /foo HTTP/1.1 my-header: foo Config: add: - name: "my-header" value: "bar,baz" Output: GET /foo HTTP/1.1 my-header: foo,bar,baz
add[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.
remove	array (string)	Remove the given header(s) from the HTTP request before the action. The value of Remove is a list of HTTP header names. Note that the header names are case-insensitive (see https://datatracker.ietf.org/doc/html/rfc2616#section-4.2). Input: GET /foo HTTP/1.1 my-header1: foo my-header2: bar my-header3: baz Config: remove: ["my-header1", "my-header3"] Output: GET /foo HTTP/1.1 my-header2: bar
set	array	Set overwrites the request with the given header (name, value) before the action. Input: GET /foo HTTP/1.1 my-header: foo Config: set: - name: "my-header" value: "bar" Output: GET /foo HTTP/1.1 my-header: bar

Property	Type	Description
set[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

14.1.36. .spec.rules[].filters[].responseHeaderModifier.add

Description

Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name.

Input: GET /foo HTTP/1.1 my-header: foo

Config: add: - name: "my-header" value: "bar,baz"

Output: GET /foo HTTP/1.1 my-header: foo,bar,baz

Type

array

14.1.37. .spec.rules[].filters[].responseHeaderModifier.add[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

14.1.38. .spec.rules[].filters[].responseHeaderModifier.set

Description

Set overwrites the request with the given header (name, value) before the action.

Input: GET /foo HTTP/1.1 my-header: foo

Config: set: - name: "my-header" value: "bar"

Output: GET /foo HTTP/1.1 my-header: bar

Type

array

14.1.39. .spec.rules[].filters[].responseHeaderModifier.set[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

14.1.40. .spec.rules[].matches

Description

Matches define conditions used for matching the rule against incoming gRPC requests. Each match is independent, i.e. this rule will be matched if **any** one of the matches is satisfied.

For example, take the following matches configuration:

```
matches: - method: service: foo.bar headers: values: version: 2 - method: service: foo.bar.v2
```

For a request to match against this rule, it MUST satisfy EITHER of the two conditions:

- service of foo.bar AND contains the header **version: 2**
- service of foo.bar.v2

See the documentation for `GRPCRouteMatch` on how to specify multiple match conditions to be ANDed together.

If no matches are specified, the implementation MUST match every gRPC request.

Proxy or Load Balancer routing configuration generated from `GRPCRoutes` MUST prioritize rules based on the following criteria, continuing on ties. Merging MUST not be done between `GRPCRoutes` and `HTTPRoutes`. Precedence MUST be given to the rule with the largest number of:

- Characters in a matching non-wildcard hostname.
- Characters in a matching hostname.
- Characters in a matching service.
- Characters in a matching method.

- Header matches.

If ties still exist across multiple Routes, matching precedence **MUST** be determined in order of the following criteria, continuing on ties:

- The oldest Route based on creation timestamp.
- The Route appearing first in alphabetical order by "{namespace}/{name}".

If ties still exist within the Route that has been given precedence, matching precedence **MUST** be granted to the first matching rule meeting the above criteria.

Type

array

14.1.41. .spec.rules[].matches[]

Description

GRPCRouteMatch defines the predicate used to match requests to a given action. Multiple match types are ANDed together, i.e. the match will evaluate to true only if all conditions are satisfied. For example, the match below will match a gRPC request only if its service is **foo** AND it contains the **version: v1** header:

```
matches: - method: type: Exact service: "foo" headers: - name: "version" value "v1"
```

Type

object

Property	Type	Description
headers	array	Headers specifies gRPC request header matchers. Multiple match values are ANDed together, meaning, a request MUST match all the specified headers to select the route.
headers[]	object	GRPCHeaderMatch describes how to select a gRPC route by matching gRPC request headers.
method	object	Method specifies a gRPC request service/method matcher. If this field is not specified, all services and methods will match.

14.1.42. .spec.rules[].matches[].headers

Description

Headers specifies gRPC request header matchers. Multiple match values are ANDed together, meaning, a request **MUST** match all the specified headers to select the route.

Type

array**14.1.43. .spec.rules[].matches[].headers[]**

Description

GRPCHeaderMatch describes how to select a gRPC route by matching gRPC request headers.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the gRPC Header to be matched. If multiple entries specify equivalent header names, only the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
type	string	Type specifies how to match against the value of the header.
value	string	Value is the value of the gRPC Header to be matched.

14.1.44. .spec.rules[].matches[].method

Description

Method specifies a gRPC request service/method matcher. If this field is not specified, all services and methods will match.

Type

object

Property	Type	Description

Property	Type	Description
method	string	Value of the method to match against. If left empty or omitted, will match all services. At least one of Service and Method MUST be a non-empty string.
service	string	Value of the service to match against. If left empty or omitted, will match any service. At least one of Service and Method MUST be a non-empty string.
type	string	Type specifies how to match against the service and/or method. Support: Core (Exact with service and method specified) Support: Implementation-specific (Exact with method specified but no service specified) Support: Implementation-specific (RegularExpression)

14.1.45. .status

Description

Status defines the current state of GRPCRoute.

Type

object

Required

- **parents**

Property	Type	Description

Property	Type	Description
parents	array	Parents is a list of parent resources (usually Gateways) that are associated with the route, and the status of the route with respect to each parent. When this route attaches to a parent, the controller that manages the parent must add an entry to this list when the controller first sees the route and should update the entry as appropriate when the route or gateway is modified. Note that parent references that cannot be resolved by an implementation of this API will not be added to this list. Implementations of this API can only populate Route status for the Gateways/parent resources they are responsible for. A maximum of 32 Gateways will be represented in this list. An empty list means the route has not been attached to any Gateway.
parents[]	object	RouteParentStatus describes the status of a route with respect to an associated Parent.

14.1.46. .status.parents

Description

Parents is a list of parent resources (usually Gateways) that are associated with the route, and the status of the route with respect to each parent. When this route attaches to a parent, the controller that manages the parent must add an entry to this list when the controller first sees the route and should update the entry as appropriate when the route or gateway is modified.

Note that parent references that cannot be resolved by an implementation of this API will not be added to this list. Implementations of this API can only populate Route status for the Gateways/parent resources they are responsible for.

A maximum of 32 Gateways will be represented in this list. An empty list means the route has not been attached to any Gateway.

Type

array

14.1.47. .status.parents[]

Description

RouteParentStatus describes the status of a route with respect to an associated Parent.

Type

object

Required

- **controllerName**
- **parentRef**

Property	Type	Description
conditions	array	Conditions describes the status of the route with respect to the Gateway. Note that the route's availability is also subject to the Gateway's own status conditions and listener status. If the Route's ParentRef specifies an existing Gateway that supports Routes of this kind AND that Gateway's controller has sufficient access, then that Gateway's controller MUST set the "Accepted" condition on the Route, to indicate whether the route has been accepted or rejected by the Gateway, and why. A Route MUST be considered "Accepted" if at least one of the Route's rules is implemented by the Gateway. There are a number of cases where the "Accepted" condition may not be set due to lack of controller visibility, that includes when: * The Route refers to a nonexistent parent. * The Route is of a type that the controller does not support. * The Route is in a namespace the controller does not have access to.
conditions[]	object	Condition contains details for one aspect of the current state of this API Resource.

Property	Type	Description
controllerName	string	ControllerName is a domain/path string that indicates the name of the controller that wrote this status. This corresponds with the controllerName field on GatewayClass. Example: "example.net/gateway-controller". The format of this field is DOMAIN "/" PATH, where DOMAIN and PATH are valid Kubernetes names (https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names). Controllers MUST populate this field when writing status. Controllers should ensure that entries to status populated with their ControllerName are cleaned up when they are no longer necessary.
parentRef	object	ParentRef corresponds with a ParentRef in the spec that this RouteParentStatus struct describes the status of.

14.1.48. .status.parents[].conditions

Description

Conditions describes the status of the route with respect to the Gateway. Note that the route's availability is also subject to the Gateway's own status conditions and listener status.

If the Route's ParentRef specifies an existing Gateway that supports Routes of this kind AND that Gateway's controller has sufficient access, then that Gateway's controller MUST set the "Accepted" condition on the Route, to indicate whether the route has been accepted or rejected by the Gateway, and why.

A Route MUST be considered "Accepted" if at least one of the Route's rules is implemented by the Gateway.

There are a number of cases where the "Accepted" condition may not be set due to lack of controller visibility, that includes when:

- The Route refers to a nonexistent parent.
- The Route is of a type that the controller does not support.

- The Route is in a namespace the controller does not have access to.

Type

array

14.1.49. .status.parents[].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.

Property	Type	Description
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.

14.1.50. .status.parents[].parentRef

Description

ParentRef corresponds with a ParentRef in the spec that this RouteParentStatus struct describes the status of.

Type

object

Required

- **name**

Property	Type	Description
group	string	Group is the group of the referent. When unspecified, "gateway.networking.k8s.io" is inferred. To set the core API group (such as for a "Service" kind referent), Group must be explicitly set to "" (empty string). Support: Core

Property	Type	Description
kind	string	Kind is kind of the referent. There are two kinds of parent resources with "Core" support: * Gateway (Gateway conformance profile) * Service (Mesh conformance profile, ClusterIP Services only) Support for other resources is Implementation-Specific.
name	string	Name is the name of the referent. Support: Core
namespace	string	Namespace is the namespace of the referent. When unspecified, this refers to the local namespace of the Route. Note that there are specific rules for ParentRefs which cross namespace boundaries. Cross-namespace references are only valid if they are explicitly allowed by something in the namespace they are referring to. For example: Gateway has the AllowedRoutes field, and ReferenceGrant provides a generic way to enable any other kind of cross-namespace reference. Support: Core

Property	Type	Description
port	integer	Port is the network port this Route targets. It can be interpreted differently based on the type of parent resource. When the parent resource is a Gateway, this targets all listeners listening on the specified port that also support this kind of Route (and select this Route). It's not recommended to set Port unless the networking behaviors specified in a Route must apply to a specific port as opposed to a listener(s) whose port(s) may be changed. When both Port and SectionName are specified, the name and port of the selected listener must match both specified values. Implementations MAY choose to support other parent resources. Implementations supporting other types of parent resources MUST clearly document how/if Port is interpreted. For the purpose of status, an attachment is considered successful as long as the parent resource accepts it partially. For example, Gateway listeners can restrict which Routes can attach to them by Route kind, namespace, or hostname. If 1 of 2 Gateway listeners accept attachment from the referencing Route, the Route MUST be considered successfully attached. If no Gateway listeners accept attachment from this Route, the Route MUST be considered detached from the Gateway. Support: Extended

Property	Type	Description
sectionName	string	SectionName is the name of a section within the target resource. In the following resources, SectionName is interpreted as the following: * Gateway: Listener name. When both Port (experimental) and SectionName are specified, the name and port of the selected listener must match both specified values. * Service: Port name. When both Port (experimental) and SectionName are specified, the name and port of the selected listener must match both specified values. Implementations MAY choose to support attaching Routes to other resources. If that is the case, they MUST clearly document how SectionName is interpreted. When unspecified (empty string), this will reference the entire resource. For the purpose of status, an attachment is considered successful if at least one section in the parent resource accepts it. For example, Gateway listeners can restrict which Routes can attach to them by Route kind, namespace, or hostname. If 1 of 2 Gateway listeners accept attachment from the referencing Route, the Route MUST be considered successfully attached. If no Gateway listeners accept attachment from this Route, the Route MUST be considered detached from the Gateway. Support: Core

14.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/gateway.networking.k8s.io/v1/grpcroutes**
 - **GET**: list objects of kind GRPCRoute

- **/apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/grpcroutes**
 - **DELETE**: delete collection of GRPCRoute
 - **GET**: list objects of kind GRPCRoute
 - **POST**: create a GRPCRoute
- **/apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/grpcroutes/{name}**
 - **DELETE**: delete a GRPCRoute
 - **GET**: read the specified GRPCRoute
 - **PATCH**: partially update the specified GRPCRoute
 - **PUT**: replace the specified GRPCRoute
- **/apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/grpcroutes/{name}/status**
 - **GET**: read status of the specified GRPCRoute
 - **PATCH**: partially update status of the specified GRPCRoute
 - **PUT**: replace status of the specified GRPCRoute

14.2.1. /apis/gateway.networking.k8s.io/v1/grpcroutes

HTTP method

GET

Description

list objects of kind GRPCRoute

Table 14.1. HTTP responses

HTTP code	Reponse body
200 - OK	GRPCRouteList schema
401 - Unauthorized	Empty

14.2.2. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/grpcroutes

HTTP method

DELETE

Description

delete collection of GRPCRoute

Table 14.2. HTTP responses

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

HTTP method**GET****Description**

list objects of kind GRPCRoute

Table 14.3. HTTP responses

HTTP code	Reponse body
200 - OK	GRPCRouteList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create a GRPCRoute

Table 14.4. 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

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 14.5. Body parameters

Parameter	Type	Description
body	GRPCRoute schema	

Table 14.6. HTTP responses

HTTP code	Response body
200 - OK	GRPCRoute schema
201 - Created	GRPCRoute schema
202 - Accepted	GRPCRoute schema
401 - Unauthorized	Empty

14.2.3. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/grpcroutes/{name}

Table 14.7. Global path parameters

Parameter	Type	Description
name	string	name of the GRPCRoute

HTTP method**DELETE****Description**

delete a GRPCRoute

Table 14.8. 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 14.9. HTTP responses

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

HTTP method**GET****Description**

read the specified GRPCRoute

Table 14.10. HTTP responses

HTTP code	Reponse body
200 - OK	GRPCRoute schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified GRPCRoute

Table 14.11. 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 14.12. HTTP responses

HTTP code	Response body
200 - OK	GRPCRoute schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified GRPCRoute

Table 14.13. 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 14.14. Body parameters

Parameter	Type	Description
body	GRPCRoute schema	

Table 14.15. HTTP responses

HTTP code	Response body
200 - OK	GRPCRoute schema
201 - Created	GRPCRoute schema
401 - Unauthorized	Empty

14.2.4. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/grpcroutes/{n

Table 14.16. Global path parameters

Parameter	Type	Description
name	string	name of the GRPCRoute

HTTP method

GET

Description

read status of the specified GRPCRoute

Table 14.17. HTTP responses

HTTP code	Response body
200 - OK	GRPCRoute schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified GRPCRoute

Table 14.18. 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 14.19. HTTP responses

HTTP code	Response body
200 - OK	GRPCRoute schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified GRPCRoute

Table 14.20. 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 14.21. Body parameters

Parameter	Type	Description
body	GRPCRoute schema	

Table 14.22. HTTP responses

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

CHAPTER 15. GATEWAY [GATEWAY.NETWORKING.K8S.IO/V1]

Description

Gateway represents an instance of a service-traffic handling infrastructure by binding Listeners to a set of IP addresses.

Type

object

Required

- **spec**

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
metadata	ObjectMeta	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	object	Spec defines the desired state of Gateway.
status	object	Status defines the current state of Gateway.

15.1.1. .spec

Description

Spec defines the desired state of Gateway.

Type

object

Required

- **gatewayClassName**
- **listeners**

Property	Type	Description
addresses	array	Addresses requested for this Gateway. This is optional and behavior can depend on the implementation. If a value is set in the spec and the requested address is invalid or unavailable, the implementation MUST indicate this in the associated entry in <code>GatewayStatus.Addresses</code>. The <code>Addresses</code> field represents a request for the address(es) on the "outside of the Gateway", that traffic bound for this Gateway will use. This could be the IP address or hostname of an external load balancer or other networking infrastructure, or some other address that traffic will be sent to. If no <code>Addresses</code> are specified, the implementation MAY schedule the Gateway in an implementation-specific manner, assigning an appropriate set of <code>Addresses</code>. The implementation MUST bind all <code>Listeners</code> to every <code>GatewayAddress</code> that it assigns to the Gateway and add a corresponding entry in <code>GatewayStatus.Addresses</code>. Support: Extended

Property	Type	Description
addresses[]	object	GatewayAddress describes an address that can be bound to a Gateway.
gatewayClassName	string	GatewayClassName used for this Gateway. This is the name of a GatewayClass resource.
infrastructure	object	Infrastructure defines infrastructure level attributes about this Gateway instance. Support: Extended
listeners	array	Listeners associated with this Gateway. Listeners define logical endpoints that are bound on this Gateway's addresses. At least one Listener MUST be specified. Distinct Listeners Each Listener in a set of Listeners (for example, in a single Gateway) MUST be <i>distinct</i>, in that a traffic flow MUST be able to be assigned to exactly one listener. (This section uses "set of Listeners" rather than "Listeners in a single Gateway" because implementations MAY merge configuration from multiple Gateways onto a single data plane, and these rules <i>also</i> apply in that case). Practically, this means that each listener in a set MUST have a unique combination of Port, Protocol, and, if supported by the protocol, Hostname. Some combinations of port, protocol, and TLS settings are considered Core support and MUST be supported by implementations based on the objects they support: HTTPRoute

Property	Type	Description
		1. HTTPRoute, Port: 80, Protocol: HTTP, HTTPRoute, Port: 443, Protocol: HTTPS, TLS Mode: Terminate, TLS keypair provided TLSRoute 1. TLSRoute, Port: 443, Protocol: TLS, TLS Mode: Passthrough "Distinct" Listeners have the following property: The implementation can match inbound requests to a single distinct Listener. When multiple Listeners share values for fields (for example, two Listeners with the same Port value), the implementation can match requests to only one of the Listeners using other Listener fields. When multiple listeners have the same value for the Protocol field, then each of the Listeners with matching Protocol values MUST have different values for other fields. The set of fields that MUST be different for a Listener differs per protocol. The following rules define the rules for what fields MUST be considered for Listeners to be distinct with each protocol currently defined in the Gateway API spec. The set of listeners that all share a protocol value MUST have <i>different</i> values for <i>at least one</i> of these fields to be distinct: * HTTP, HTTPS, TLS: Port, Hostname * TCP, UDP: Port One very important rule to call out involves what happens when an implementation: * Supports TCP protocol Listeners, as well as HTTP, HTTPS, or TLS protocol Listeners, and * sees HTTP,

Property	Type	Description
		HTTPS, or TLS protocols with the same port as one with TCP Protocol. In this case all the Listeners that share a port with the TCP Listener are not distinct and so MUST NOT be accepted. If an implementation does not support TCP Protocol Listeners, then the previous rule does not apply, and the TCP Listeners SHOULD NOT be accepted. Note that the tls field is not used for determining if a listener is distinct, because Listeners that <i>only</i> differ on TLS config will still conflict in all cases. # Listeners that are distinct only by Hostname When the Listeners are distinct based only on Hostname, inbound request hostnames MUST match from the most specific to least specific Hostname values to choose the correct Listener and its associated set of Routes. Exact matches MUST be processed before wildcard matches, and wildcard matches MUST be processed before fallback (empty Hostname value) matches. For example, "foo.example.com" takes precedence over "*.example.com", and "*.example.com" takes precedence over "". Additionally, if there are multiple wildcard entries, more specific wildcard entries must be processed before less specific wildcard entries. For example, "*.foo.example.com" takes precedence over "*.example.com". The precise definition here is that the higher the number of dots in the hostname to the right of the

Property	Type	Description
		wildcard character, the higher the precedence. The wildcard character will match any number of characters <i>and dots</i> to the left, however, so "*.example.com" will match both "foo.bar.example.com" and "bar.example.com". Handling indistinct Listeners If a set of Listeners contains Listeners that are not distinct, then those Listeners are <i>Conflicted</i>, and the implementation MUST set the "Conflicted" condition in the Listener Status to "True". The words "indistinct" and "conflicted" are considered equivalent for the purpose of this documentation. Implementations MAY choose to accept a Gateway with some Conflicted Listeners only if they only accept the partial Listener set that contains no Conflicted Listeners. Specifically, an implementation MAY accept a partial Listener set subject to the following rules: * The implementation MUST NOT pick one conflicting Listener as the winner. ALL indistinct Listeners must not be accepted for processing. * At least one distinct Listener MUST be present, or else the Gateway effectively contains <i>no</i> Listeners, and must be rejected from processing as a whole. The implementation MUST set a "ListenersNotValid" condition on the Gateway Status when the Gateway contains Conflicted Listeners whether or not they accept the Gateway. That Condition SHOULD clearly indicate in the Message which Listeners are conflicted, and which are Accepted. Additionally,

Property	Type	Description
		the Listener status for those listeners SHOULD indicate which Listeners are conflicted and not Accepted. General Listener behavior Note that, for all distinct Listeners, requests SHOULD match at most one Listener. For example, if Listeners are defined for "foo.example.com" and "*.example.com", a request to "foo.example.com" SHOULD only be routed using routes attached to the "foo.example.com" Listener (and not the "*.example.com" Listener). This concept is known as "Listener Isolation", and it is an Extended feature of Gateway API. Implementations that do not support Listener Isolation MUST clearly document this, and MUST NOT claim support for the GatewayHTTPListenerIsolation feature. Implementations that <i>do</i> support Listener Isolation SHOULD claim support for the Extended GatewayHTTPListenerIsolation feature and pass the associated conformance tests. Compatible Listeners A Gateway's Listeners are considered <i>compatible</i> if: 1. They are distinct. 2. The implementation can serve them in compliance with the Addresses requirement that all Listeners are available on all assigned addresses. Compatible combinations in Extended support are expected to vary across implementations. A combination that is compatible for one implementation may not be compatible for another. For example, an implementation that cannot serve both TCP and

Property	Type	Description
		UDP listeners on the same address, or cannot mix HTTPS and generic TLS listens on the same port would not consider those cases compatible, even though they are distinct. Implementations MAY merge separate Gateways onto a single set of Addresses if all Listeners across all Gateways are compatible. In a future release the MinItems=1 requirement MAY be dropped.
listeners[]	object	Support: Core Listener embodies the concept of a logical endpoint where a Gateway accepts network connections.

15.1.2. .spec.addresses

Description

Addresses requested for this Gateway. This is optional and behavior can depend on the implementation. If a value is set in the spec and the requested address is invalid or unavailable, the implementation MUST indicate this in the associated entry in GatewayStatus.Addresses.

The Addresses field represents a request for the address(es) on the "outside of the Gateway", that traffic bound for this Gateway will use. This could be the IP address or hostname of an external load balancer or other networking infrastructure, or some other address that traffic will be sent to.

If no Addresses are specified, the implementation MAY schedule the Gateway in an implementation-specific manner, assigning an appropriate set of Addresses.

The implementation MUST bind all Listeners to every GatewayAddress that it assigns to the Gateway and add a corresponding entry in GatewayStatus.Addresses.

Support: Extended

Type

array

15.1.3. .spec.addresses[]

Description

GatewayAddress describes an address that can be bound to a Gateway.

Type

object

Required

- **value**

Property	Type	Description
type	string	Type of the address.
value	string	Value of the address. The validity of the values will depend on the type and support by the controller. Examples: 1.2.3.4, 128::1, my-ip-address.

15.1.4. .spec.infrastructure

Description

Infrastructure defines infrastructure level attributes about this Gateway instance.

Support: Extended

Type

object

Property	Type	Description
annotations	object (string)	Annotations that SHOULD be applied to any resources created in response to this Gateway. For implementations creating other Kubernetes objects, this should be the metadata.annotations field on resources. For other implementations, this refers to any relevant (implementation specific) "annotations" concepts. An implementation may chose to add additional implementation-specific annotations as they see fit. Support: Extended

Property	Type	Description
labels	object (string)	Labels that SHOULD be applied to any resources created in response to this Gateway. For implementations creating other Kubernetes objects, this should be the metadata.labels field on resources. For other implementations, this refers to any relevant (implementation specific) "labels" concepts. An implementation may chose to add additional implementation-specific labels as they see fit. If an implementation maps these labels to Pods, or any other resource that would need to be recreated when labels change, it SHOULD clearly warn about this behavior in documentation. Support: Extended

Property	Type	Description
parametersRef	object	ParametersRef is a reference to a resource that contains the configuration parameters corresponding to the Gateway. This is optional if the controller does not require any additional configuration. This follows the same semantics as GatewayClass's parametersRef, but on a per-Gateway basis The Gateway's GatewayClass may provide its own parametersRef. When both are specified, the merging behavior is implementation specific. It is generally recommended that GatewayClass provides defaults that can be overridden by a Gateway. If the referent cannot be found, refers to an unsupported kind, or when the data within that resource is malformed, the Gateway SHOULD be rejected with the "Accepted" status condition set to "False" and an "InvalidParameters" reason. Support: Implementation-specific

15.1.5. .spec.infrastructure.parametersRef

Description

ParametersRef is a reference to a resource that contains the configuration parameters corresponding to the Gateway. This is optional if the controller does not require any additional configuration.

This follows the same semantics as GatewayClass's **parametersRef**, but on a per-Gateway basis

The Gateway's GatewayClass may provide its own **parametersRef**. When both are specified, the merging behavior is implementation specific. It is generally recommended that GatewayClass provides defaults that can be overridden by a Gateway.

If the referent cannot be found, refers to an unsupported kind, or when the data within that resource is malformed, the Gateway SHOULD be rejected with the "Accepted" status condition set to "False" and an "InvalidParameters" reason.

Support: Implementation-specific

Type**object****Required**

- **group**
- **kind**
- **name**

Property	Type	Description
group	string	Group is the group of the referent.
kind	string	Kind is kind of the referent.
name	string	Name is the name of the referent.

15.1.6. .spec.listeners**Description**

Listeners associated with this Gateway. Listeners define logical endpoints that are bound on this Gateway's addresses. At least one Listener **MUST** be specified.

Distinct Listeners

Each Listener in a set of Listeners (for example, in a single Gateway) **MUST** be *distinct*, in that a traffic flow **MUST** be able to be assigned to exactly one listener. (This section uses "set of Listeners" rather than "Listeners in a single Gateway" because implementations **MAY** merge configuration from multiple Gateways onto a single data plane, and these rules *also* apply in that case).

Practically, this means that each listener in a set **MUST** have a unique combination of Port, Protocol, and, if supported by the protocol, Hostname.

Some combinations of port, protocol, and TLS settings are considered Core support and **MUST** be supported by implementations based on the objects they support:

HTTPRoute

1. HTTPRoute, Port: 80, Protocol: HTTP
2. HTTPRoute, Port: 443, Protocol: HTTPS, TLS Mode: Terminate, TLS keypair provided

TLSRoute

1. TLSRoute, Port: 443, Protocol: TLS, TLS Mode: Passthrough

"Distinct" Listeners have the following property:

The implementation can match inbound requests to a single distinct Listener

When multiple Listeners share values for fields (for example, two Listeners with the same Port value), the implementation can match requests to only one of the Listeners using other Listener fields.

When multiple listeners have the same value for the Protocol field, then each of the Listeners with matching Protocol values **MUST** have different values for other fields.

The set of fields that **MUST** be different for a Listener differs per protocol. The following rules define the rules for what fields **MUST** be considered for Listeners to be distinct with each protocol currently defined in the Gateway API spec.

The set of listeners that all share a protocol value **MUST** have *different* values for *at least one* of these fields to be distinct:

- **HTTP, HTTPS, TLS:** Port, Hostname
- **TCP, UDP:** Port

One **very** important rule to call out involves what happens when an implementation:

- Supports TCP protocol Listeners, as well as HTTP, HTTPS, or TLS protocol Listeners, and
- sees HTTP, HTTPS, or TLS protocols with the same **port** as one with TCP Protocol.

In this case all the Listeners that share a port with the TCP Listener are not distinct and so **MUST NOT** be accepted.

If an implementation does not support TCP Protocol Listeners, then the previous rule does not apply, and the TCP Listeners **SHOULD NOT** be accepted.

Note that the **tls** field is not used for determining if a listener is distinct, because Listeners that *only* differ on TLS config will still conflict in all cases.

Listeners that are distinct only by Hostname

When the Listeners are distinct based only on Hostname, inbound request hostnames **MUST** match from the most specific to least specific Hostname values to choose the correct Listener and its associated set of Routes.

Exact matches **MUST** be processed before wildcard matches, and wildcard matches **MUST** be processed before fallback (empty Hostname value) matches. For example, **"foo.example.com"** takes precedence over **"*.example.com"**, and **"*.example.com"** takes precedence over **""**.

Additionally, if there are multiple wildcard entries, more specific wildcard entries must be processed before less specific wildcard entries. For example, **"*.foo.example.com"** takes precedence over **"*.example.com"**.

The precise definition here is that the higher the number of dots in the hostname to the right of the wildcard character, the higher the precedence.

The wildcard character will match any number of characters *and dots* to the left, however, so **"*.example.com"** will match both **"foo.bar.example.com"** and **"bar.example.com"**.

Handling indistinct Listeners

If a set of Listeners contains Listeners that are not distinct, then those Listeners are *Conflicted*, and the implementation **MUST** set the "Conflicted" condition in the Listener Status to "True".

The words "indistinct" and "conflicted" are considered equivalent for the purpose of this documentation.

Implementations MAY choose to accept a Gateway with some Conflicted Listeners only if they only accept the partial Listener set that contains no Conflicted Listeners.

Specifically, an implementation MAY accept a partial Listener set subject to the following rules:

- The implementation **MUST NOT** pick one conflicting Listener as the winner. ALL indistinct Listeners must not be accepted for processing.
- At least one distinct Listener **MUST** be present, or else the Gateway effectively contains *no* Listeners, and must be rejected from processing as a whole.

The implementation **MUST** set a "ListenersNotValid" condition on the Gateway Status when the Gateway contains Conflicted Listeners whether or not they accept the Gateway. That Condition **SHOULD** clearly indicate in the Message which Listeners are conflicted, and which are Accepted. Additionally, the Listener status for those listeners **SHOULD** indicate which Listeners are conflicted and not Accepted.

General Listener behavior

Note that, for all distinct Listeners, requests **SHOULD** match at most one Listener. For example, if Listeners are defined for "foo.example.com" and ".example.com", **a request to "foo.example.com" *SHOULD only be routed using routes attached to the "foo.example.com" Listener (and not the ".example.com" Listener.***

This concept is known as "Listener Isolation", and it is an Extended feature of Gateway API. Implementations that do not support Listener Isolation **MUST** clearly document this, and **MUST NOT** claim support for the **GatewayHTTPListenerIsolation** feature.

Implementations that *do* support Listener Isolation **SHOULD** claim support for the Extended **GatewayHTTPListenerIsolation** feature and pass the associated conformance tests.

Compatible Listeners

A Gateway's Listeners are considered *compatible* if:

1. They are distinct.
2. The implementation can serve them in compliance with the Addresses requirement that all Listeners are available on all assigned addresses.

Compatible combinations in Extended support are expected to vary across implementations. A combination that is compatible for one implementation may not be compatible for another.

For example, an implementation that cannot serve both TCP and UDP listeners on the same address, or cannot mix HTTPS and generic TLS listens on the same port would not consider those cases compatible, even though they are distinct.

Implementations MAY merge separate Gateways onto a single set of Addresses if all Listeners across all Gateways are compatible.

In a future release the MinItems=1 requirement MAY be dropped.

Support: Core

Type

array

15.1.7. .spec.listeners[]

Description

Listener embodies the concept of a logical endpoint where a Gateway accepts network connections.

Type

object

Required

- **name**
- **port**
- **protocol**

Property	Type	Description

Property	Type	Description
allowedRoutes	object	AllowedRoutes defines the types of routes that MAY be attached to a Listener and the trusted namespaces where those Route resources MAY be present. Although a client request may match multiple route rules, only one rule may ultimately receive the request. Matching precedence MUST be determined in order of the following criteria: * The most specific match as defined by the Route type. * The oldest Route based on creation timestamp. For example, a Route with a creation timestamp of "2020-09-08 01:02:03" is given precedence over a Route with a creation timestamp of "2020-09-08 01:02:04". * If everything else is equivalent, the Route appearing first in alphabetical order (namespace/name) should be given precedence. For example, foo/bar is given precedence over foo/baz. All valid rules within a Route attached to this Listener should be implemented. Invalid Route rules can be ignored (sometimes that will mean the full Route). If a Route rule transitions from valid to invalid, support for that Route rule should be dropped to ensure consistency. For example, even if a filter specified by a Route rule is invalid, the rest of the rules within that Route should still be supported. Support: Core
hostname	string	Hostname specifies the virtual hostname to match for protocol types that define this concept. When unspecified, all hostnames are matched. This field is ignored for protocols that don't require hostname based matching.

Property	Type	Description
		Implementations MUST apply hostname matching appropriately for each of the following protocols: * TLS: The Listener Hostname MUST match the SNI. * HTTP: The Listener Hostname MUST match the Host header of the request. * HTTPS: The Listener Hostname SHOULD match both the SNI and Host header. Note that this does not require the SNI and Host header to be the same. The semantics of this are described in more detail below. To ensure security, Section 11.1 of RFC-6066 emphasizes that server implementations that rely on SNI hostname matching MUST also verify hostnames within the application protocol. Section 9.1.2 of RFC-7540 provides a mechanism for servers to reject the reuse of a connection by responding with the HTTP 421 Misdirected Request status code. This indicates that the origin server has rejected the request because it appears to have been misdirected. To detect misdirected requests, Gateways SHOULD match the authority of the requests with all the SNI hostname(s) configured across all the Gateway Listeners on the same port and protocol: * If another Listener has an exact match or more specific wildcard entry, the Gateway SHOULD return a 421. * If the current Listener (selected by SNI matching during ClientHello) does not match the Host: * If another Listener does match the Host the Gateway SHOULD return a 421. * If no other Listener matches the Host, the Gateway MUST return a 404. For HTTPRoute and TLSRoute resources, there is an interaction

Property	Type	Description
		with the spec.hostnames array. When both listener and route specify hostnames, there MUST be an intersection between the values for a Route to be accepted. For more information, refer to the Route specific Hostnames documentation. Hostnames that are prefixed with a wildcard label (*.*) are interpreted as a suffix match. That means that a match for *.example.com would match both test.example.com, and foo.test.example.com, but not example.com. Support: Core
name	string	Name is the name of the Listener. This name MUST be unique within a Gateway. Support: Core
port	integer	Port is the network port. Multiple listeners may use the same port, subject to the Listener compatibility rules. Support: Core
protocol	string	Protocol specifies the network protocol this listener expects to receive. Support: Core

Property	Type	Description
tls	object	TLS is the TLS configuration for the Listener. This field is required if the Protocol field is "HTTPS" or "TLS". It is invalid to set this field if the Protocol field is "HTTP", "TCP", or "UDP". The association of SNIs to Certificate defined in GatewayTLSConfig is defined based on the Hostname field for this listener. The GatewayClass MUST use the longest matching SNI out of all available certificates for any TLS handshake. Support: Core

15.1.8. .spec.listeners[].allowedRoutes

Description

AllowedRoutes defines the types of routes that MAY be attached to a Listener and the trusted namespaces where those Route resources MAY be present.

Although a client request may match multiple route rules, only one rule may ultimately receive the request. Matching precedence MUST be determined in order of the following criteria:

- The most specific match as defined by the Route type.
- The oldest Route based on creation timestamp. For example, a Route with a creation timestamp of "2020-09-08 01:02:03" is given precedence over a Route with a creation timestamp of "2020-09-08 01:02:04".
- If everything else is equivalent, the Route appearing first in alphabetical order (namespace/name) should be given precedence. For example, foo/bar is given precedence over foo/baz.

All valid rules within a Route attached to this Listener should be implemented. Invalid Route rules can be ignored (sometimes that will mean the full Route). If a Route rule transitions from valid to invalid, support for that Route rule should be dropped to ensure consistency. For example, even if a filter specified by a Route rule is invalid, the rest of the rules within that Route should still be supported.

Support: Core

Type

object

Property	Type	Description
kinds	array	Kinds specifies the groups and kinds of Routes that are allowed to bind to this Gateway Listener. When unspecified or empty, the kinds of Routes selected are determined using the Listener protocol. A RouteGroupKind MUST correspond to kinds of Routes that are compatible with the application protocol specified in the Listener's Protocol field. If an implementation does not support or recognize this resource type, it MUST set the "ResolvedRefs" condition to False for this Listener with the "InvalidRouteKinds" reason. Support: Core
kinds[]	object	RouteGroupKind indicates the group and kind of a Route resource.
namespaces	object	Namespaces indicates namespaces from which Routes may be attached to this Listener. This is restricted to the namespace of this Gateway by default. Support: Core

15.1.9. .spec.listeners[].allowedRoutes.kinds

Description

Kinds specifies the groups and kinds of Routes that are allowed to bind to this Gateway Listener. When unspecified or empty, the kinds of Routes selected are determined using the Listener protocol. A RouteGroupKind MUST correspond to kinds of Routes that are compatible with the application protocol specified in the Listener's Protocol field. If an implementation does not support or recognize this resource type, it MUST set the "ResolvedRefs" condition to False for this Listener with the "InvalidRouteKinds" reason.

Support: Core

Type

array

15.1.10. .spec.listeners[].allowedRoutes.kinds[]

Description

RouteGroupKind indicates the group and kind of a Route resource.

Type

object

Required

- **kind**

Property	Type	Description
group	string	Group is the group of the Route.
kind	string	Kind is the kind of the Route.

15.1.11. .spec.listeners[].allowedRoutes.namespaces

Description

Namespaces indicates namespaces from which Routes may be attached to this Listener. This is restricted to the namespace of this Gateway by default.

Support: Core

Type

object

Property	Type	Description
from	string	From indicates where Routes will be selected for this Gateway. Possible values are: * All: Routes in all namespaces may be used by this Gateway. * Selector: Routes in namespaces selected by the selector may be used by this Gateway. * Same: Only Routes in the same namespace may be used by this Gateway. Support: Core

Property	Type	Description
selector	object	Selector must be specified when From is set to "Selector". In that case, only Routes in Namespaces matching this Selector will be selected by this Gateway. This field is ignored for other values of "From". Support: Core

15.1.12. .spec.listeners[].allowedRoutes.namespaces.selector

Description

Selector must be specified when From is set to "Selector". In that case, only Routes in Namespaces matching this Selector will be selected by this Gateway. This field is ignored for other values of "From".

Support: Core

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.

15.1.13. .spec.listeners[].allowedRoutes.namespaces.selector.matchExpressions

Description

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

Type

array

15.1.14. .spec.listeners[].allowedRoutes.namespaces.selector.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.

15.1.15. .spec.listeners[].tls

Description

TLS is the TLS configuration for the Listener. This field is required if the Protocol field is "HTTPS" or "TLS". It is invalid to set this field if the Protocol field is "HTTP", "TCP", or "UDP".

The association of SNIs to Certificate defined in GatewayTLSConfig is defined based on the Hostname field for this listener.

The GatewayClass MUST use the longest matching SNI out of all available certificates for any TLS handshake.

Support: Core

Type object

Property	Type	Description
certificateRefs	array	CertificateRefs contains a series of references to Kubernetes objects that contains TLS certificates and private keys. These certificates are used to establish a TLS handshake for requests that match the hostname of the associated listener. A single CertificateRef to a Kubernetes Secret has "Core" support. Implementations MAY choose to support attaching multiple certificates to a Listener, but this behavior is implementation-specific. References to a resource in different namespace are invalid UNLESS there is a ReferenceGrant in the target namespace that allows the certificate to be attached. If a ReferenceGrant does not allow this reference, the "ResolvedRefs" condition MUST be set to False for this listener with the "RefNotPermitted" reason. This field is required to have at least one element when the mode is set to "Terminate" (default) and is optional otherwise. CertificateRefs can reference to standard Kubernetes resources, i.e. Secret, or implementation-specific custom resources. Support: Core - A single reference to a Kubernetes Secret of type <code>kubernetes.io/tls</code> Support: Implementation-specific (More than one reference or other resource types)

Property	Type	Description
certificateRefs[]	object	SecretObjectReference identifies an API object including its namespace, defaulting to Secret. The API object must be valid in the cluster; the Group and Kind must be registered in the cluster for this reference to be valid. References to objects with invalid Group and Kind are not valid, and must be rejected by the implementation, with appropriate Conditions set on the containing object.
mode	string	Mode defines the TLS behavior for the TLS session initiated by the client. There are two possible modes: - Terminate: The TLS session between the downstream client and the Gateway is terminated at the Gateway. This mode requires certificates to be specified in some way, such as populating the certificateRefs field. - - Passthrough: The TLS session is NOT terminated by the Gateway. This implies that the Gateway can't decipher the TLS stream except for the ClientHello message of the TLS protocol. The certificateRefs field is ignored in this mode. Support: Core

Property	Type	Description
options	object (string)	Options are a list of key/value pairs to enable extended TLS configuration for each implementation. For example, configuring the minimum TLS version or supported cipher suites. A set of common keys MAY be defined by the API in the future. To avoid any ambiguity, implementation-specific definitions MUST use domain-prefixed names, such as example.com/my-custom-option. Un-prefixed names are reserved for key names defined by Gateway API. Support: Implementation-specific

15.1.16. .spec.listeners[].tls.certificateRefs

Description

CertificateRefs contains a series of references to Kubernetes objects that contains TLS certificates and private keys. These certificates are used to establish a TLS handshake for requests that match the hostname of the associated listener.

A single CertificateRef to a Kubernetes Secret has "Core" support. Implementations MAY choose to support attaching multiple certificates to a Listener, but this behavior is implementation-specific.

References to a resource in different namespace are invalid UNLESS there is a ReferenceGrant in the target namespace that allows the certificate to be attached. If a ReferenceGrant does not allow this reference, the "ResolvedRefs" condition MUST be set to False for this listener with the "RefNotPermitted" reason.

This field is required to have at least one element when the mode is set to "Terminate" (default) and is optional otherwise.

CertificateRefs can reference to standard Kubernetes resources, i.e. Secret, or implementation-specific custom resources.

Support: Core - A single reference to a Kubernetes Secret of type `kubernetes.io/tls`

Support: Implementation-specific (More than one reference or other resource types)

Type

array

15.1.17. .spec.listeners[].tls.certificateRefs[]

Description

SecretObjectReference identifies an API object including its namespace, defaulting to Secret. The API object must be valid in the cluster; the Group and Kind must be registered in the cluster for this reference to be valid.

References to objects with invalid Group and Kind are not valid, and must be rejected by the implementation, with appropriate Conditions set on the containing object.

Type

object

Required

- **name**

Property	Type	Description
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is kind of the referent. For example "Secret".
name	string	Name is the name of the referent.
namespace	string	Namespace is the namespace of the referenced object. When unspecified, the local namespace is inferred. Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. Support: Core

15.1.18. .status

Description

Status defines the current state of Gateway.

Type

object

Property	Type	Description
addresses	array	Addresses lists the network addresses that have been bound to the Gateway. This list may differ from the addresses provided in the spec under some conditions: * no addresses are specified, all addresses are dynamically assigned * a combination of specified and dynamic addresses are assigned * a specified address was unusable (e.g. already in use)
addresses[]	object	GatewayStatusAddress describes a network address that is bound to a Gateway.
conditions	array	Conditions describe the current conditions of the Gateway. Implementations should prefer to express Gateway conditions using the GatewayConditionType and GatewayConditionReason constants so that operators and tools can converge on a common vocabulary to describe Gateway state. Known condition types are: * "Accepted" * "Programmed" * "Ready"
conditions[]	object	Condition contains details for one aspect of the current state of this API Resource.
listeners	array	Listeners provide status for each unique listener port defined in the Spec.
listeners[]	object	ListenerStatus is the status associated with a Listener.

15.1.19. .status.addresses

Description

Addresses lists the network addresses that have been bound to the Gateway.

This list may differ from the addresses provided in the spec under some conditions:

- no addresses are specified, all addresses are dynamically assigned
- a combination of specified and dynamic addresses are assigned
- a specified address was unusable (e.g. already in use)

Type

array

15.1.20. .status.addresses[]

Description

GatewayStatusAddress describes a network address that is bound to a Gateway.

Type

object

Required

- value

Property	Type	Description
type	string	Type of the address.
value	string	Value of the address. The validity of the values will depend on the type and support by the controller. Examples: 1.2.3.4 , 128::1 , my-ip-address .

15.1.21. .status.conditions

Description

Conditions describe the current conditions of the Gateway.

Implementations should prefer to express Gateway conditions using the **GatewayConditionType** and **GatewayConditionReason** constants so that operators and tools can converge on a common vocabulary to describe Gateway state.

Known condition types are:

- "Accepted"
- "Programmed"
- "Ready"

Type

array

15.1.22. .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.

Property	Type	Description
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.

15.1.23. .status.listeners

Description

Listeners provide status for each unique listener port defined in the Spec.

Type

array

15.1.24. .status.listeners[]

Description

ListenerStatus is the status associated with a Listener.

Type

object

Required

- **attachedRoutes**
- **conditions**
- **name**
- **supportedKinds**

Property	Type	Description

Property	Type	Description
attachedRoutes	integer	AttachedRoutes represents the total number of Routes that have been successfully attached to this Listener. Successful attachment of a Route to a Listener is based solely on the combination of the AllowedRoutes field on the corresponding Listener and the Route's ParentRefs field. A Route is successfully attached to a Listener when it is selected by the Listener's AllowedRoutes field AND the Route has a valid ParentRef selecting the whole Gateway resource or a specific Listener as a parent resource (more detail on attachment semantics can be found in the documentation on the various Route kinds ParentRefs fields). Listener or Route status does not impact successful attachment, i.e. the AttachedRoutes field count MUST be set for Listeners with condition Accepted: false and MUST count successfully attached Routes that may themselves have Accepted: false conditions. Uses for this field include troubleshooting Route attachment and measuring blast radius/impact of changes to a Listener.
conditions	array	Conditions describe the current condition of this listener.
conditions[]	object	Condition contains details for one aspect of the current state of this API Resource.
name	string	Name is the name of the Listener that this status corresponds to.

Property	Type	Description
supportedKinds	array	SupportedKinds is the list indicating the Kinds supported by this listener. This MUST represent the kinds an implementation supports for that Listener configuration. If kinds are specified in Spec that are not supported, they MUST NOT appear in this list and an implementation MUST set the "ResolvedRefs" condition to "False" with the "InvalidRouteKinds" reason. If both valid and invalid Route kinds are specified, the implementation MUST reference the valid Route kinds that have been specified.
supportedKinds[]	object	RouteGroupKind indicates the group and kind of a Route resource.

15.1.25. .status.listeners[].conditions

Description

Conditions describe the current condition of this listener.

Type

array

15.1.26. .status.listeners[].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.

15.1.27. .status.listeners[].supportedKinds

Description

SupportedKinds is the list indicating the Kinds supported by this listener. This MUST represent the kinds an implementation supports for that Listener configuration.

If kinds are specified in Spec that are not supported, they MUST NOT appear in this list and an implementation MUST set the "ResolvedRefs" condition to "False" with the "InvalidRouteKinds" reason. If both valid and invalid Route kinds are specified, the implementation MUST reference the valid Route kinds that have been specified.

Type
array

15.1.28. .status.listeners[].supportedKinds[]

Description

RouteGroupKind indicates the group and kind of a Route resource.

Type
object

Required

- **kind**

Property	Type	Description
group	string	Group is the group of the Route.
kind	string	Kind is the kind of the Route.

15.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/gateway.networking.k8s.io/v1/gateways**
 - **GET**: list objects of kind Gateway
- **/apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/gateways**
 - **DELETE**: delete collection of Gateway
 - **GET**: list objects of kind Gateway
 - **POST**: create a Gateway
- **/apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/gateways/{name}**
 - **DELETE**: delete a Gateway
 - **GET**: read the specified Gateway
 - **PATCH**: partially update the specified Gateway
 - **PUT**: replace the specified Gateway
- **/apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/gateways/{name}/status**
 - **GET**: read status of the specified Gateway
 - **PATCH**: partially update status of the specified Gateway

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

15.2.1. /apis/gateway.networking.k8s.io/v1/gateways

HTTP method

GET

Description

list objects of kind Gateway

Table 15.1. HTTP responses

HTTP code	Reponse body
200 - OK	GatewayList schema
401 - Unauthorized	Empty

15.2.2. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/gateways

HTTP method

DELETE

Description

delete collection of Gateway

Table 15.2. HTTP responses

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

HTTP method

GET

Description

list objects of kind Gateway

Table 15.3. HTTP responses

HTTP code	Reponse body
200 - OK	GatewayList schema
401 - Unauthorized	Empty

HTTP method

POST**Description**

create a Gateway

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	Gateway schema	

Table 15.6. HTTP responses

HTTP code	Response body
200 - OK	Gateway schema
201 - Created	Gateway schema
202 - Accepted	Gateway schema

HTTP code	Reponse body
401 - Unauthorized	Empty

15.2.3. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/gateways/{name}

Table 15.7. Global path parameters

Parameter	Type	Description
name	string	name of the Gateway

HTTP method

DELETE

Description

delete a Gateway

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 Gateway

Table 15.10. HTTP responses

HTTP code	Response body
200 - OK	Gateway schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified Gateway

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	Response body
200 - OK	Gateway schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Gateway

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	Gateway schema	

Table 15.15. HTTP responses

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

15.2.4. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/gateways/{name}

Table 15.16. Global path parameters

Parameter	Type	Description
name	string	name of the Gateway

HTTP method

GET

Description

read status of the specified Gateway

Table 15.17. HTTP responses

HTTP code	Response body
200 - OK	Gateway schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Gateway

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	Response body
200 - OK	Gateway schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified Gateway

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	Gateway schema	

Table 15.22. HTTP responses

HTTP code	Response body
200 - OK	Gateway schema
201 - Created	Gateway schema
401 - Unauthorized	Empty

CHAPTER 16. GATEWAYCLASS

[GATEWAY.NETWORKING.K8S.IO/V1]

Description

GatewayClass describes a class of Gateways available to the user for creating Gateway resources. It is recommended that this resource be used as a template for Gateways. This means that a Gateway is based on the state of the GatewayClass at the time it was created and changes to the GatewayClass or associated parameters are not propagated down to existing Gateways. This recommendation is intended to limit the blast radius of changes to GatewayClass or associated parameters. If implementations choose to propagate GatewayClass changes to existing Gateways, that **MUST** be clearly documented by the implementation.

Whenever one or more Gateways are using a GatewayClass, implementations **SHOULD** add the **gateway-exists-finalizer.gateway.networking.k8s.io** finalizer on the associated GatewayClass. This ensures that a GatewayClass associated with a Gateway is not deleted while in use.

GatewayClass is a Cluster level resource.

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

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 defines the desired state of GatewayClass.
status	object	Status defines the current state of GatewayClass. Implementations MUST populate status on all GatewayClass resources which specify their controller name.

16.1.1. .spec

Description

Spec defines the desired state of GatewayClass.

Type

object

Required

- **controllerName**

Property	Type	Description
controllerName	string	ControllerName is the name of the controller that is managing Gateways of this class. The value of this field MUST be a domain prefixed path. Example: "example.net/gateway-controller". This field is not mutable and cannot be empty. Support: Core
description	string	Description helps describe a GatewayClass with more details.

Property	Type	Description
parametersRef	object	ParametersRef is a reference to a resource that contains the configuration parameters corresponding to the GatewayClass. This is optional if the controller does not require any additional configuration. ParametersRef can reference a standard Kubernetes resource, i.e. ConfigMap, or an implementation-specific custom resource. The resource can be cluster-scoped or namespace-scoped. If the referent cannot be found, refers to an unsupported kind, or when the data within that resource is malformed, the GatewayClass SHOULD be rejected with the "Accepted" status condition set to "False" and an "InvalidParameters" reason. A Gateway for this GatewayClass may provide its own parametersRef. When both are specified, the merging behavior is implementation specific. It is generally recommended that GatewayClass provides defaults that can be overridden by a Gateway. Support: Implementation-specific

16.1.2. .spec.parametersRef

Description

ParametersRef is a reference to a resource that contains the configuration parameters corresponding to the GatewayClass. This is optional if the controller does not require any additional configuration.

ParametersRef can reference a standard Kubernetes resource, i.e. ConfigMap, or an implementation-specific custom resource. The resource can be cluster-scoped or namespace-scoped.

If the referent cannot be found, refers to an unsupported kind, or when the data within that resource is malformed, the GatewayClass SHOULD be rejected with the "Accepted" status condition set to "False" and an "InvalidParameters" reason.

A Gateway for this GatewayClass may provide its own **parametersRef**. When both are specified, the merging behavior is implementation specific. It is generally recommended that GatewayClass provides defaults that can be overridden by a Gateway.

Support: Implementation-specific

Type

object

Required

- **group**
- **kind**
- **name**

Property	Type	Description
group	string	Group is the group of the referent.
kind	string	Kind is kind of the referent.
name	string	Name is the name of the referent.
namespace	string	Namespace is the namespace of the referent. This field is required when referring to a Namespace-scoped resource and MUST be unset when referring to a Cluster-scoped resource.

16.1.3. .status

Description

Status defines the current state of GatewayClass.

Implementations **MUST** populate status on all GatewayClass resources which specify their controller name.

Type

object

Property	Type	Description

Property	Type	Description
conditions	array	Conditions is the current status from the controller for this GatewayClass. Controllers should prefer to publish conditions using values of GatewayClassConditionType for the type of each Condition.
conditions[]	object	Condition contains details for one aspect of the current state of this API Resource.

16.1.4. .status.conditions

Description

Conditions is the current status from the controller for this GatewayClass.

Controllers should prefer to publish conditions using values of GatewayClassConditionType for the type of each Condition.

Type

array

16.1.5. .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

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.

16.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/gateway.networking.k8s.io/v1/gatewayclasses**
 - **DELETE:** delete collection of GatewayClass
 - **GET:** list objects of kind GatewayClass

- **POST**: create a GatewayClass
- **/apis/gateway.networking.k8s.io/v1/gatewayclasses/{name}**
 - **DELETE**: delete a GatewayClass
 - **GET**: read the specified GatewayClass
 - **PATCH**: partially update the specified GatewayClass
 - **PUT**: replace the specified GatewayClass
- **/apis/gateway.networking.k8s.io/v1/gatewayclasses/{name}/status**
 - **GET**: read status of the specified GatewayClass
 - **PATCH**: partially update status of the specified GatewayClass
 - **PUT**: replace status of the specified GatewayClass

16.2.1. /apis/gateway.networking.k8s.io/v1/gatewayclasses

HTTP method

DELETE

Description

delete collection of GatewayClass

Table 16.1. HTTP responses

HTTP code	Response body
200 - OK	Status schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind GatewayClass

Table 16.2. HTTP responses

HTTP code	Response body
200 - OK	GatewayClassList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a GatewayClass

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	GatewayClass schema	

Table 16.5. HTTP responses

HTTP code	Response body
200 - OK	GatewayClass schema
201 - Created	GatewayClass schema
202 - Accepted	GatewayClass schema

HTTP code	Reponse body
401 - Unauthorized	Empty

16.2.2. /apis/gateway.networking.k8s.io/v1/gatewayclasses/{name}

Table 16.6. Global path parameters

Parameter	Type	Description
name	string	name of the GatewayClass

HTTP method

DELETE

Description

delete a GatewayClass

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 GatewayClass

Table 16.9. HTTP responses

HTTP code	Reponse body
200 - OK	GatewayClass schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified GatewayClass

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	GatewayClass schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified GatewayClass

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	GatewayClass schema	

Table 16.14. HTTP responses

HTTP code	Response body
200 - OK	GatewayClass schema
201 - Created	GatewayClass schema

HTTP code	Reponse body
401 - Unauthorized	Empty

16.2.3. /apis/gateway.networking.k8s.io/v1/gatewayclasses/{name}/status

Table 16.15. Global path parameters

Parameter	Type	Description
name	string	name of the GatewayClass

HTTP method

GET

Description

read status of the specified GatewayClass

Table 16.16. HTTP responses

HTTP code	Reponse body
200 - OK	GatewayClass schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified GatewayClass

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	Response body
200 - OK	GatewayClass schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified GatewayClass

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	GatewayClass schema	

Table 16.21. HTTP responses

HTTP code	Response body
200 - OK	GatewayClass schema
201 - Created	GatewayClass schema
401 - Unauthorized	Empty

CHAPTER 17. HTTPROUTE

[GATEWAY.NETWORKING.K8S.IO/V1]

Description

HTTPRoute provides a way to route HTTP requests. This includes the capability to match requests by hostname, path, header, or query param. Filters can be used to specify additional processing steps. Backends specify where matching requests should be routed.

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
spec	object	Spec defines the desired state of HTTPRoute.

Property	Type	Description
status	object	Status defines the current state of HTTPRoute.

17.1.1. .spec

Description

Spec defines the desired state of HTTPRoute.

Type

object

Property	Type	Description
hostnames	array (string)	Hostnames defines a set of hostnames that should match against the HTTP Host header to select a HTTPRoute used to process the request. Implementations MUST ignore any port value specified in the HTTP Host header while performing a match and (absent of any applicable header modification configuration) MUST forward this header unmodified to the backend. Valid values for Hostnames are determined by RFC 1123 definition of a hostname with 2 notable exceptions: 1. IPs are not allowed. 2. A hostname may be prefixed with a wildcard label (*). The wildcard label must appear by itself as the first label. If a hostname is specified by both the Listener and HTTPRoute, there must be at least one intersecting hostname for the HTTPRoute to be attached to the Listener. For example: * A Listener with test.example.com as the hostname matches HTTPRoutes that have either not specified any hostnames, or have specified at least one of test.example.com

Property	Type	Description
		or *.example.com. * A Listener with *.example.com as the hostname matches HTTPRoutes that have either not specified any hostnames or have specified at least one hostname that matches the Listener hostname. For example, *.example.com, test.example.com, and foo.test.example.com would all match. On the other hand, example.com and test.example.net would not match. Hostnames that are prefixed with a wildcard label (*) are interpreted as a suffix match. That means that a match for *.example.com would match both test.example.com, and foo.test.example.com, but not example.com. If both the Listener and HTTPRoute have specified hostnames, any HTTPRoute hostnames that do not match the Listener hostname MUST be ignored. For example, if a Listener specified *.example.com, and the HTTPRoute specified test.example.com and test.example.net, test.example.net must not be considered for a match. If both the Listener and HTTPRoute have specified hostnames, and none match with the criteria above, then the HTTPRoute is not accepted. The implementation must raise an 'Accepted' Condition with a status of False in the corresponding RouteParentStatus. In the event that multiple HTTPRoutes specify intersecting hostnames (e.g. overlapping wildcard matching and exact matching hostnames), precedence must be given to rules from the HTTPRoute with the largest number of:

Property	Type	Description
		* Characters in a matching non-wildcard hostname. * Characters in a matching hostname. If ties exist across multiple Routes, the matching precedence rules for HTTPRouteMatches takes over. Support: Core
parentRefs	array	ParentRefs references the resources (usually Gateways) that a Route wants to be attached to. Note that the referenced parent resource needs to allow this for the attachment to be complete. For Gateways, that means the Gateway needs to allow attachment from Routes of this kind and namespace. For Services, that means the Service must either be in the same namespace for a "producer" route, or the mesh implementation must support and allow "consumer" routes for the referenced Service. ReferenceGrant is not applicable for governing ParentRefs to Services - it is not possible to create a "producer" route for a Service in a different namespace from the Route. There are two kinds of parent resources with "Core" support: * Gateway (Gateway conformance profile) * Service (Mesh conformance profile, ClusterIP Services only) This API may be extended in the future to support additional kinds of parent resources. ParentRefs must be <i>distinct</i>. This means either that: * They select different objects. If this is the case, then parentRef entries are distinct. In terms of fields, this means that the multi-part key defined by group, kind, namespace, and name must be unique across all parentRef

Property	Type	Description
		entries in the Route. * They do not select different objects, but for each optional field used, each ParentRef that selects the same object must set the same set of optional fields to different values. If one ParentRef sets a combination of optional fields, all must set the same combination. Some examples: * If one ParentRef sets sectionName, all ParentRefs referencing the same object must also set sectionName. * If one ParentRef sets port, all ParentRefs referencing the same object must also set port. * If one ParentRef sets sectionName and port, all ParentRefs referencing the same object must also set sectionName and port. It is possible to separately reference multiple distinct objects that may be collapsed by an implementation. For example, some implementations may choose to merge compatible Gateway Listeners together. If that is the case, the list of routes attached to those resources should also be merged. Note that for ParentRefs that cross namespace boundaries, there are specific rules. Cross-namespace references are only valid if they are explicitly allowed by something in the namespace they are referring to. For example, Gateway has the AllowedRoutes field, and ReferenceGrant provides a generic way to enable other kinds of cross-namespace reference.

Property	Type	Description
parentRefs[]	object	ParentReference identifies an API object (usually a Gateway) that can be considered a parent of this resource (usually a route). There are two kinds of parent resources with "Core" support: * Gateway (Gateway conformance profile) * Service (Mesh conformance profile, ClusterIP Services only) This API may be extended in the future to support additional kinds of parent resources. The API object must be valid in the cluster; the Group and Kind must be registered in the cluster for this reference to be valid.
rules	array	Rules are a list of HTTP matchers, filters and actions.
rules[]	object	HTTPRouteRule defines semantics for matching an HTTP request based on conditions (matches), processing it (filters), and forwarding the request to an API object (backendRefs).

17.1.2. .spec.parentRefs

Description

ParentRefs references the resources (usually Gateways) that a Route wants to be attached to. Note that the referenced parent resource needs to allow this for the attachment to be complete. For Gateways, that means the Gateway needs to allow attachment from Routes of this kind and namespace. For Services, that means the Service must either be in the same namespace for a "producer" route, or the mesh implementation must support and allow "consumer" routes for the referenced Service. ReferenceGrant is not applicable for governing ParentRefs to Services - it is not possible to create a "producer" route for a Service in a different namespace from the Route.

There are two kinds of parent resources with "Core" support:

- Gateway (Gateway conformance profile)
- Service (Mesh conformance profile, ClusterIP Services only)

This API may be extended in the future to support additional kinds of parent resources.

ParentRefs must be *distinct*. This means either that:

- They select different objects. If this is the case, then parentRef entries are distinct. In terms of fields, this means that the multi-part key defined by **group**, **kind**, **namespace**, and **name** must be unique across all parentRef entries in the Route.
- They do not select different objects, but for each optional field used, each ParentRef that selects the same object must set the same set of optional fields to different values. If one ParentRef sets a combination of optional fields, all must set the same combination.

Some examples:

- If one ParentRef sets **sectionName**, all ParentRefs referencing the same object must also set **sectionName**.
- If one ParentRef sets **port**, all ParentRefs referencing the same object must also set **port**.
- If one ParentRef sets **sectionName** and **port**, all ParentRefs referencing the same object must also set **sectionName** and **port**.

It is possible to separately reference multiple distinct objects that may be collapsed by an implementation. For example, some implementations may choose to merge compatible Gateway Listeners together. If that is the case, the list of routes attached to those resources should also be merged.

Note that for ParentRefs that cross namespace boundaries, there are specific rules. Cross-namespace references are only valid if they are explicitly allowed by something in the namespace they are referring to. For example, Gateway has the AllowedRoutes field, and ReferenceGrant provides a generic way to enable other kinds of cross-namespace reference.

Type

array

17.1.3. .spec.parentRefs[]

Description

ParentReference identifies an API object (usually a Gateway) that can be considered a parent of this resource (usually a route). There are two kinds of parent resources with "Core" support:

- Gateway (Gateway conformance profile)
- Service (Mesh conformance profile, ClusterIP Services only)

This API may be extended in the future to support additional kinds of parent resources.

The API object must be valid in the cluster; the Group and Kind must be registered in the cluster for this reference to be valid.

Type

object

Required

- **name**

Property	Type	Description
group	string	Group is the group of the referent. When unspecified, "gateway.networking.k8s.io" is inferred. To set the core API group (such as for a "Service" kind referent), Group must be explicitly set to "" (empty string). Support: Core
kind	string	Kind is kind of the referent. There are two kinds of parent resources with "Core" support: * Gateway (Gateway conformance profile) * Service (Mesh conformance profile, ClusterIP Services only) Support for other resources is Implementation-Specific.
name	string	Name is the name of the referent. Support: Core
namespace	string	Namespace is the namespace of the referent. When unspecified, this refers to the local namespace of the Route. Note that there are specific rules for ParentRefs which cross namespace boundaries. Cross-namespace references are only valid if they are explicitly allowed by something in the namespace they are referring to. For example: Gateway has the AllowedRoutes field, and ReferenceGrant provides a generic way to enable any other kind of cross-namespace reference. Support: Core

Property	Type	Description
port	integer	Port is the network port this Route targets. It can be interpreted differently based on the type of parent resource. When the parent resource is a Gateway, this targets all listeners listening on the specified port that also support this kind of Route (and select this Route). It's not recommended to set Port unless the networking behaviors specified in a Route must apply to a specific port as opposed to a listener(s) whose port(s) may be changed. When both Port and SectionName are specified, the name and port of the selected listener must match both specified values. Implementations MAY choose to support other parent resources. Implementations supporting other types of parent resources MUST clearly document how/if Port is interpreted. For the purpose of status, an attachment is considered successful as long as the parent resource accepts it partially. For example, Gateway listeners can restrict which Routes can attach to them by Route kind, namespace, or hostname. If 1 of 2 Gateway listeners accept attachment from the referencing Route, the Route MUST be considered successfully attached. If no Gateway listeners accept attachment from this Route, the Route MUST be considered detached from the Gateway. Support: Extended

Property	Type	Description
sectionName	string	SectionName is the name of a section within the target resource. In the following resources, SectionName is interpreted as the following: * Gateway: Listener name. When both Port (experimental) and SectionName are specified, the name and port of the selected listener must match both specified values. * Service: Port name. When both Port (experimental) and SectionName are specified, the name and port of the selected listener must match both specified values. Implementations MAY choose to support attaching Routes to other resources. If that is the case, they MUST clearly document how SectionName is interpreted. When unspecified (empty string), this will reference the entire resource. For the purpose of status, an attachment is considered successful if at least one section in the parent resource accepts it. For example, Gateway listeners can restrict which Routes can attach to them by Route kind, namespace, or hostname. If 1 of 2 Gateway listeners accept attachment from the referencing Route, the Route MUST be considered successfully attached. If no Gateway listeners accept attachment from this Route, the Route MUST be considered detached from the Gateway. Support: Core

17.1.4. .spec.rules

Description

Rules are a list of HTTP matchers, filters and actions.

Type

array

17.1.5. .spec.rules[]

Description

HTTPRouteRule defines semantics for matching an HTTP request based on conditions (matches), processing it (filters), and forwarding the request to an API object (backendRefs).

Type

object

Property	Type	Description
backendRefs	array	BackendRefs defines the backend(s) where matching requests should be sent. Failure behavior here depends on how many BackendRefs are specified and how many are invalid. If all entries in BackendRefs are invalid, and there are also no filters specified in this route rule, all traffic which matches this rule MUST receive a 500 status code. See the HTTPBackendRef definition for the rules about what makes a single HTTPBackendRef invalid. When a HTTPBackendRef is invalid, 500 status codes MUST be returned for requests that would have otherwise been routed to an invalid backend. If multiple backends are specified, and some are invalid, the proportion of requests that would otherwise have been routed to an invalid backend MUST receive a 500 status code. For example, if two backends are specified with equal weights, and one is invalid, 50 percent of traffic must receive a 500. Implementations may choose how that 50 percent is determined. When a HTTPBackendRef refers to a Service that has no ready endpoints, implementations SHOULD return a 503 for requests to that backend instead. If an implementation chooses to

Property	Type	Description
		do this, all of the above rules for 500 responses MUST also apply for responses that return a 503. Support: Core for Kubernetes Service Support: Extended for Kubernetes ServiceImport Support: Implementation-specific for any other resource Support for weight: Core
backendRefs[]	object	HTTPBackendRef defines how a HTTPRoute forwards a HTTP request. Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details.
filters	array	Filters define the filters that are applied to requests that match this rule. Wherever possible, implementations SHOULD implement filters in the order they are specified. Implementations MAY choose to implement this ordering strictly, rejecting any combination or order of filters that cannot be supported. If implementations choose a strict interpretation of filter ordering, they MUST clearly document that behavior. To reject an invalid combination or order of filters, implementations SHOULD consider the Route Rules with this configuration invalid. If all Route Rules in a Route are invalid, the entire Route would be considered invalid. If only a portion of Route

Property	Type	Description
		Rules are invalid, implementations MUST set the "PartiallyInvalid" condition for the Route. Conformance-levels at this level are defined based on the type of filter: - ALL core filters MUST be supported by all implementations. - Implementers are encouraged to support extended filters. - Implementation-specific custom filters have no API guarantees across implementations. Specifying the same filter multiple times is not supported unless explicitly indicated in the filter. All filters are expected to be compatible with each other except for the URLRewrite and RequestRedirect filters, which may not be combined. If an implementation cannot support other combinations of filters, they must clearly document that limitation. In cases where incompatible or unsupported filters are specified and cause the Accepted condition to be set to status False, implementations may use the IncompatibleFilters reason to specify this configuration error. Support: Core
filters[]	object	HTTPRouteFilter defines processing steps that must be completed during the request or response lifecycle. HTTPRouteFilters are meant as an extension point to express processing that may be done in Gateway implementations. Some examples include request or response modification, implementing authentication strategies, rate-limiting, and traffic shaping. API guarantee/conformance is defined based on the type of the filter.

matches Property	array Type	Matches define conditions used for matching the rule against Description
		incoming HTTP requests. Each match is independent, i.e. this rule will be matched if any one of the matches is satisfied. For example, take the following matches configuration: <pre>matches: - path: value: "/foo" headers: - name: "version" value: "v2" - path: value: "/v2/foo"</pre> For a request to match against this rule, a request must satisfy EITHER of the two conditions: - path prefixed with /foo AND contains the header version: v2 - path prefix of /v2/foo See the documentation for HTTPRouteMatch on how to specify multiple match conditions that should be ANDed together. If no matches are specified, the default is a prefix path match on "/", which has the effect of matching every HTTP request. Proxy or Load Balancer routing configuration generated from HTTPRoutes MUST prioritize matches based on the following criteria, continuing on ties. Across all rules specified on applicable Routes, precedence must be given to the match having: * "Exact" path match. * "Prefix" path match with largest number of characters. * Method match. * Largest number of header matches. * Largest number of query param matches. Note: The precedence of RegularExpression path matches are implementation-specific. If ties still exist across multiple Routes, matching precedence MUST be determined in order of the following criteria, continuing on ties:

Property	Type	Description
		* The oldest Route based on creation timestamp. * The Route appearing first in alphabetical order by "{namespace}/{name}". If ties still exist within an HTTPRoute, matching precedence MUST be granted to the FIRST matching rule (in list order) with a match meeting the above criteria. When no rules matching a request have been successfully attached to the parent a request is coming from, a HTTP 404 status code MUST be returned.
matches[]	object	HTTPRouteMatch defines the predicate used to match requests to a given action. Multiple match types are ANDed together, i.e. the match will evaluate to true only if all conditions are satisfied. For example, the match below will match a HTTP request only if its path starts with /foo AND it contains the version: v1 header: match: path: value: "/foo" headers: - name: "version" value "v1"
timeouts	object	Timeouts defines the timeouts that can be configured for an HTTP request. Support: Extended

17.1.6. .spec.rules[].backendRefs

Description

BackendRefs defines the backend(s) where matching requests should be sent.

Failure behavior here depends on how many BackendRefs are specified and how many are invalid.

If **all** entries in BackendRefs are invalid, and there are also no filters specified in this route rule, **all** traffic which matches this rule MUST receive a 500 status code.

See the HTTPBackendRef definition for the rules about what makes a single HTTPBackendRef invalid.

When a HTTPBackendRef is invalid, 500 status codes MUST be returned for requests that would have otherwise been routed to an invalid backend. If multiple backends are specified, and some are invalid, the proportion of requests that would otherwise have been routed to an invalid backend MUST receive a 500 status code.

For example, if two backends are specified with equal weights, and one is invalid, 50 percent of traffic must receive a 500. Implementations may choose how that 50 percent is determined.

When a HTTPBackendRef refers to a Service that has no ready endpoints, implementations SHOULD return a 503 for requests to that backend instead. If an implementation chooses to do this, all of the above rules for 500 responses MUST also apply for responses that return a 503.

Support: Core for Kubernetes Service

Support: Extended for Kubernetes ServiceImport

Support: Implementation-specific for any other resource

Support for weight: Core

Type

array

17.1.7. .spec.rules[].backendRefs[]

Description

HTTPBackendRef defines how a HTTPRoute forwards a HTTP request.

Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details.

Type

object

Required

- **name**

Property	Type	Description
filters	array	Filters defined at this level should be executed if and only if the request is being forwarded to the backend defined here. Support: Implementation-specific (For broader support of filters, use the Filters field in HTTPRouteRule.)

Property	Type	Description
filters[]	object	HTTPRouteFilter defines processing steps that must be completed during the request or response lifecycle. HTTPRouteFilters are meant as an extension point to express processing that may be done in Gateway implementations. Some examples include request or response modification, implementing authentication strategies, rate-limiting, and traffic shaping. API guarantee/conformance is defined based on the type of the filter.
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is the Kubernetes resource kind of the referent. For example "Service". Defaults to "Service" when not specified. ExternalName services can refer to CNAME DNS records that may live outside of the cluster and as such are difficult to reason about in terms of conformance. They also may not be safe to forward to (see CVE-2021-25740 for more information). Implementations SHOULD NOT support ExternalName Services. Support: Core (Services with a type other than ExternalName) Support: Implementation-specific (Services with type ExternalName)
name	string	Name is the name of the referent.

Property	Type	Description
namespace	string	Namespace is the namespace of the backend. When unspecified, the local namespace is inferred. Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. Support: Core
port	integer	Port specifies the destination port number to use for this resource. Port is required when the referent is a Kubernetes Service. In this case, the port number is the service port number, not the target port. For other resources, destination port might be derived from the referent resource or this field.

Property	Type	Description
weight	integer	Weight specifies the proportion of requests forwarded to the referenced backend. This is computed as $\text{weight} / (\text{sum of all weights in this BackendRefs list})$. For non-zero values, there may be some epsilon from the exact proportion defined here depending on the precision an implementation supports. Weight is not a percentage and the sum of weights does not need to equal 100. If only one backend is specified and it has a weight greater than 0, 100% of the traffic is forwarded to that backend. If weight is set to 0, no traffic should be forwarded for this entry. If unspecified, weight defaults to 1. Support for this field varies based on the context where used.

17.1.8. .spec.rules[].backendRefs[].filters

Description

Filters defined at this level should be executed if and only if the request is being forwarded to the backend defined here.

Support: Implementation-specific (For broader support of filters, use the Filters field in HTTPRouteRule.)

Type

array

17.1.9. .spec.rules[].backendRefs[].filters[]

Description

HTTPRouteFilter defines processing steps that must be completed during the request or response lifecycle. HTTPRouteFilters are meant as an extension point to express processing that may be done in Gateway implementations. Some examples include request or response modification, implementing authentication strategies, rate-limiting, and traffic shaping. API guarantee/conformance is defined based on the type of the filter.

Type

object

Required

- **type**

Property	Type	Description
extensionRef	object	ExtensionRef is an optional, implementation-specific extension to the "filter" behavior. For example, resource "myroutefilter" in group "networking.example.net"). ExtensionRef MUST NOT be used for core and extended filters. This filter can be used multiple times within the same rule. Support: Implementation-specific
requestHeaderModifier	object	RequestHeaderModifier defines a schema for a filter that modifies request headers. Support: Core
requestMirror	object	RequestMirror defines a schema for a filter that mirrors requests. Requests are sent to the specified destination, but responses from that destination are ignored. This filter can be used multiple times within the same rule. Note that not all implementations will be able to support mirroring to multiple backends. Support: Extended
requestRedirect	object	RequestRedirect defines a schema for a filter that responds to the request with an HTTP redirection. Support: Core
responseHeaderModifier	object	ResponseHeaderModifier defines a schema for a filter that modifies response headers. Support: Extended

Property	Type	Description
type	string	Type identifies the type of filter to apply. As with other API fields, types are classified into three conformance levels: - Core: Filter types and their corresponding configuration defined by "Support: Core" in this package, e.g. "RequestHeaderModifier". All implementations must support core filters. - Extended: Filter types and their corresponding configuration defined by "Support: Extended" in this package, e.g. "RequestMirror". Implementers are encouraged to support extended filters. - Implementation-specific: Filters that are defined and supported by specific vendors. In the future, filters showing convergence in behavior across multiple implementations will be considered for inclusion in extended or core conformance levels. Filter-specific configuration for such filters is specified using the ExtensionRef field. Type should be set to "ExtensionRef" for custom filters. Implementers are encouraged to define custom implementation types to extend the core API with implementation-specific behavior. If a reference to a custom filter type cannot be resolved, the filter MUST NOT be skipped. Instead, requests that would have been processed by that filter MUST receive a HTTP error response. Note that values may be added to this enum, implementations must ensure that unknown values will not cause a crash. Unknown values here must result in the implementation setting the Accepted Condition for the Route

Property	Type	Description
urlRewrite	object	URLRewrite defines a schema for a filter that modifies a request during forwarding. Support: Extended

17.1.10. .spec.rules[].backendRefs[].filters[].extensionRef

Description

ExtensionRef is an optional, implementation-specific extension to the "filter" behavior. For example, resource "myroutefilter" in group "networking.example.net"). ExtensionRef MUST NOT be used for core and extended filters.

This filter can be used multiple times within the same rule.

Support: Implementation-specific

Type

object

Required

- **group**
- **kind**
- **name**

Property	Type	Description
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is kind of the referent. For example "HTTPRoute" or "Service".
name	string	Name is the name of the referent.

17.1.11. .spec.rules[].backendRefs[].filters[].requestHeaderModifier

Description

RequestHeaderModifier defines a schema for a filter that modifies request headers.

Support: Core

Type

object

Property	Type	Description
add	array	Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name. Input: GET /foo HTTP/1.1 my-header: foo Config: add: - name: "my-header" value: "bar,baz" Output: GET /foo HTTP/1.1 my-header: foo,bar,baz
add[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.
remove	array (string)	Remove the given header(s) from the HTTP request before the action. The value of Remove is a list of HTTP header names. Note that the header names are case-insensitive (see https://datatracker.ietf.org/doc/html/rfc2616#section-4.2). Input: GET /foo HTTP/1.1 my-header1: foo my-header2: bar my-header3: baz Config: remove: ["my-header1", "my-header3"] Output: GET /foo HTTP/1.1 my-header2: bar
set	array	Set overwrites the request with the given header (name, value) before the action. Input: GET /foo HTTP/1.1 my-header: foo Config: set: - name: "my-header" value: "bar" Output: GET /foo HTTP/1.1 my-header: bar

Property	Type	Description
set[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

17.1.12. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.add

Description

Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name.

Input: GET /foo HTTP/1.1 my-header: foo

Config: add: - name: "my-header" value: "bar,baz"

Output: GET /foo HTTP/1.1 my-header: foo,bar,baz

Type

array

17.1.13. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.add[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

17.1.14. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.set

Description

Set overwrites the request with the given header (name, value) before the action.

Input: GET /foo HTTP/1.1 my-header: foo

Config: set: - name: "my-header" value: "bar"

Output: GET /foo HTTP/1.1 my-header: bar

Type

array

17.1.15. .spec.rules[].backendRefs[].filters[].requestHeaderModifier.set[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

17.1.16. .spec.rules[].backendRefs[].filters[].requestMirror

Description

RequestMirror defines a schema for a filter that mirrors requests. Requests are sent to the specified destination, but responses from that destination are ignored.

This filter can be used multiple times within the same rule. Note that not all implementations will be able to support mirroring to multiple backends.

Support: Extended

Type

object

Required

- **backendRef**

Property	Type	Description

Property	Type	Description
backendRef	object	BackendRef references a resource where mirrored requests are sent. Mirrored requests must be sent only to a single destination endpoint within this BackendRef, irrespective of how many endpoints are present within this BackendRef. If the referent cannot be found, this BackendRef is invalid and must be dropped from the Gateway. The controller must ensure the "ResolvedRefs" condition on the Route status is set to status: False and not configure this backend in the underlying implementation. If there is a cross-namespace reference to an existing object that is not allowed by a ReferenceGrant, the controller must ensure the "ResolvedRefs" condition on the Route is set to status: False, with the "RefNotPermitted" reason and not configure this backend in the underlying implementation. In either error case, the Message of the ResolvedRefs Condition should be used to provide more detail about the problem. Support: Extended for Kubernetes Service Support: Implementation-specific for any other resource
fraction	object	Fraction represents the fraction of requests that should be mirrored to BackendRef. Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

Property	Type	Description
percent	integer	Percent represents the percentage of requests that should be mirrored to BackendRef. Its minimum value is 0 (indicating 0% of requests) and its maximum value is 100 (indicating 100% of requests). Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

17.1.17. .spec.rules[].backendRefs[].filters[].requestMirror.backendRef

Description

BackendRef references a resource where mirrored requests are sent.

Mirrored requests must be sent only to a single destination endpoint within this BackendRef, irrespective of how many endpoints are present within this BackendRef.

If the referent cannot be found, this BackendRef is invalid and must be dropped from the Gateway. The controller must ensure the "ResolvedRefs" condition on the Route status is set to **status: False** and not configure this backend in the underlying implementation.

If there is a cross-namespace reference to an **existing** object that is not allowed by a ReferenceGrant, the controller must ensure the "ResolvedRefs" condition on the Route is set to **status: False**, with the "RefNotPermitted" reason and not configure this backend in the underlying implementation.

In either error case, the Message of the **ResolvedRefs** Condition should be used to provide more detail about the problem.

Support: Extended for Kubernetes Service

Support: Implementation-specific for any other resource

Type

object

Required

- **name**

Property	Type	Description

Property	Type	Description
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is the Kubernetes resource kind of the referent. For example "Service". Defaults to "Service" when not specified. ExternalName services can refer to CNAME DNS records that may live outside of the cluster and as such are difficult to reason about in terms of conformance. They also may not be safe to forward to (see CVE-2021-25740 for more information). Implementations SHOULD NOT support ExternalName Services. Support: Core (Services with a type other than ExternalName) Support: Implementation-specific (Services with type ExternalName)
name	string	Name is the name of the referent.
namespace	string	Namespace is the namespace of the backend. When unspecified, the local namespace is inferred. Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. Support: Core

Property	Type	Description
port	integer	Port specifies the destination port number to use for this resource. Port is required when the referent is a Kubernetes Service. In this case, the port number is the service port number, not the target port. For other resources, destination port might be derived from the referent resource or this field.

17.1.18. .spec.rules[].backendRefs[].filters[].requestMirror.fraction

Description

Fraction represents the fraction of requests that should be mirrored to BackendRef.

Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

Type

object

Required

- **numerator**

Property	Type	Description
denominator	integer	
numerator	integer	

17.1.19. .spec.rules[].backendRefs[].filters[].requestRedirect

Description

RequestRedirect defines a schema for a filter that responds to the request with an HTTP redirection.

Support: Core

Type

object

Property	Type	Description

Property	Type	Description
hostname	string	Hostname is the hostname to be used in the value of the Location header in the response. When empty, the hostname in the Host header of the request is used. Support: Core
path	object	Path defines parameters used to modify the path of the incoming request. The modified path is then used to construct the Location header. When empty, the request path is used as-is. Support: Extended

Property	Type	Description
port	integer	Port is the port to be used in the value of the Location header in the response. If no port is specified, the redirect port MUST be derived using the following rules: * If redirect scheme is not-empty, the redirect port MUST be the well-known port associated with the redirect scheme. Specifically "http" to port 80 and "https" to port 443. If the redirect scheme does not have a well-known port, the listener port of the Gateway SHOULD be used. * If redirect scheme is empty, the redirect port MUST be the Gateway Listener port. Implementations SHOULD NOT add the port number in the 'Location' header in the following cases: * A Location header that will use HTTP (whether that is determined via the Listener protocol or the Scheme field) <i>and</i> use port 80. * A Location header that will use HTTPS (whether that is determined via the Listener protocol or the Scheme field) <i>and</i> use port 443. Support: Extended

Property	Type	Description
scheme	string	Scheme is the scheme to be used in the value of the Location header in the response. When empty, the scheme of the request is used. Scheme redirects can affect the port of the redirect, for more information, refer to the documentation for the port field of this filter. Note that values may be added to this enum, implementations must ensure that unknown values will not cause a crash. Unknown values here must result in the implementation setting the Accepted Condition for the Route to status: False, with a Reason of UnsupportedValue. Support: Extended
statusCode	integer	StatusCode is the HTTP status code to be used in response. Note that values may be added to this enum, implementations must ensure that unknown values will not cause a crash. Unknown values here must result in the implementation setting the Accepted Condition for the Route to status: False, with a Reason of UnsupportedValue. Support: Core

17.1.20. .spec.rules[].backendRefs[].filters[].requestRedirect.path

Description

Path defines parameters used to modify the path of the incoming request. The modified path is then used to construct the **Location** header. When empty, the request path is used as-is.

Support: Extended

Type

object

Required

- type

Property	Type	Description
replaceFullPath	string	ReplaceFullPath specifies the value with which to replace the full path of a request during a rewrite or redirect.
replacePrefixMatch	string	ReplacePrefixMatch specifies the value with which to replace the prefix match of a request during a rewrite or redirect. For example, a request to <code>/foo/bar</code> with a prefix match of <code>/foo</code> and a ReplacePrefixMatch of <code>/xyz</code> would be modified to <code>/xyz/bar</code>. Note that this matches the behavior of the PathPrefix match type. This matches full path elements. A path element refers to the list of labels in the path split by the <code>/</code> separator. When specified, a trailing <code>/</code> is ignored. For example, the paths <code>/abc</code>, <code>/abc/</code>, and <code>/abc/def</code> would all match the prefix <code>/abc</code>, but the path <code>/abcd</code> would not. ReplacePrefixMatch is only compatible with a PathPrefix HTTPRouteMatch. Using any other HTTPRouteMatch type on the same HTTPRouteRule will result in the implementation setting the Accepted Condition for the Route to status: False. Request Path Prefix Match Replace Prefix Modified Path

Property	Type	Description
type	string	Type defines the type of path modifier. Additional types may be added in a future release of the API. Note that values may be added to this enum, implementations must ensure that unknown values will not cause a crash. Unknown values here must result in the implementation setting the Accepted Condition for the Route to status: False, with a Reason of UnsupportedValue.

17.1.21. .spec.rules[].backendRefs[].filters[].responseHeaderModifier

Description

ResponseHeaderModifier defines a schema for a filter that modifies response headers.
Support: Extended

Type

object

Property	Type	Description
add	array	Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name. Input: GET /foo HTTP/1.1 my-header: foo Config: add: - name: "my-header" value: "bar,baz" Output: GET /foo HTTP/1.1 my-header: foo,bar,baz
add[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Property	Type	Description
remove	array (string)	Remove the given header(s) from the HTTP request before the action. The value of Remove is a list of HTTP header names. Note that the header names are case-insensitive (see https://datatracker.ietf.org/doc/html/rfc2616#section-4.2). Input: GET /foo HTTP/1.1 my-header1: foo my-header2: bar my-header3: baz Config: remove: ["my-header1", "my-header3"] Output: GET /foo HTTP/1.1 my-header2: bar
set	array	Set overwrites the request with the given header (name, value) before the action. Input: GET /foo HTTP/1.1 my-header: foo Config: set: - name: "my-header" value: "bar" Output: GET /foo HTTP/1.1 my-header: bar
set[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

17.1.22. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.add

Description

Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name.

Input: GET /foo HTTP/1.1 my-header: foo

Config: add: - name: "my-header" value: "bar,baz"

Output: GET /foo HTTP/1.1 my-header: foo,bar,baz

Type

array

17.1.23. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.add[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

17.1.24. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.set

Description

Set overwrites the request with the given header (name, value) before the action.

Input: GET /foo HTTP/1.1 my-header: foo

Config: set: - name: "my-header" value: "bar"

Output: GET /foo HTTP/1.1 my-header: bar

Type

array

17.1.25. .spec.rules[].backendRefs[].filters[].responseHeaderModifier.set[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

17.1.26. .spec.rules[].backendRefs[].filters[].urlRewrite**Description**

URLRewrite defines a schema for a filter that modifies a request during forwarding.

Support: Extended

Type

object

Property	Type	Description
hostname	string	Hostname is the value to be used to replace the Host header value during forwarding. Support: Extended

Property	Type	Description
path	object	Path defines a path rewrite. Support: Extended

17.1.27. .spec.rules[].backendRefs[].filters[].urlRewrite.path

Description

Path defines a path rewrite.

Support: Extended

Type

object

Required

- **type**

Property	Type	Description
replaceFullPath	string	ReplaceFullPath specifies the value with which to replace the full path of a request during a rewrite or redirect.

Property	Type	Description
replacePrefixMatch	string	ReplacePrefixMatch specifies the value with which to replace the prefix match of a request during a rewrite or redirect. For example, a request to <code>/foo/bar</code> with a prefix match of <code>/foo</code> and a ReplacePrefixMatch of <code>/xyz</code> would be modified to <code>/xyz/bar</code>. Note that this matches the behavior of the PathPrefix match type. This matches full path elements. A path element refers to the list of labels in the path split by the <code>/</code> separator. When specified, a trailing <code>/</code> is ignored. For example, the paths <code>/abc</code>, <code>/abc/</code>, and <code>/abc/def</code> would all match the prefix <code>/abc</code>, but the path <code>/abcd</code> would not. ReplacePrefixMatch is only compatible with a PathPrefix HTTPRouteMatch. Using any other HTTPRouteMatch type on the same HTTPRouteRule will result in the implementation setting the Accepted Condition for the Route to status: False. Request Path Prefix Match Replace Prefix Modified Path
type	string	Type defines the type of path modifier. Additional types may be added in a future release of the API. Note that values may be added to this enum, implementations must ensure that unknown values will not cause a crash. Unknown values here must result in the implementation setting the Accepted Condition for the Route to status: False, with a Reason of UnsupportedValue.

17.1.28. .spec.rules[].filters

Description

Filters define the filters that are applied to requests that match this rule.

Wherever possible, implementations **SHOULD** implement filters in the order they are specified.

Implementations **MAY** choose to implement this ordering strictly, rejecting any combination or order of filters that cannot be supported. If implementations choose a strict interpretation of filter ordering, they **MUST** clearly document that behavior.

To reject an invalid combination or order of filters, implementations **SHOULD** consider the Route Rules with this configuration invalid. If all Route Rules in a Route are invalid, the entire Route would be considered invalid. If only a portion of Route Rules are invalid, implementations **MUST** set the "PartiallyInvalid" condition for the Route.

Conformance-levels at this level are defined based on the type of filter:

- ALL core filters **MUST** be supported by all implementations.
- Implementers are encouraged to support extended filters.
- Implementation-specific custom filters have no API guarantees across implementations.

Specifying the same filter multiple times is not supported unless explicitly indicated in the filter.

All filters are expected to be compatible with each other except for the URLRewrite and RequestRedirect filters, which may not be combined. If an implementation cannot support other combinations of filters, they must clearly document that limitation. In cases where incompatible or unsupported filters are specified and cause the **Accepted** condition to be set to status **False**, implementations may use the **IncompatibleFilters** reason to specify this configuration error.

Support: Core

Type

array

17.1.29. .spec.rules[].filters[]

Description

HTTPRouteFilter defines processing steps that must be completed during the request or response lifecycle. HTTPRouteFilters are meant as an extension point to express processing that may be done in Gateway implementations. Some examples include request or response modification, implementing authentication strategies, rate-limiting, and traffic shaping. API guarantee/conformance is defined based on the type of the filter.

Type

object

Required

- **type**

Property	Type	Description

Property	Type	Description
extensionRef	object	ExtensionRef is an optional, implementation-specific extension to the "filter" behavior. For example, resource "myroutefilter" in group "networking.example.net"). ExtensionRef MUST NOT be used for core and extended filters. This filter can be used multiple times within the same rule. Support: Implementation-specific
requestHeaderModifier	object	RequestHeaderModifier defines a schema for a filter that modifies request headers. Support: Core
requestMirror	object	RequestMirror defines a schema for a filter that mirrors requests. Requests are sent to the specified destination, but responses from that destination are ignored. This filter can be used multiple times within the same rule. Note that not all implementations will be able to support mirroring to multiple backends. Support: Extended
requestRedirect	object	RequestRedirect defines a schema for a filter that responds to the request with an HTTP redirection. Support: Core
responseHeaderModifier	object	ResponseHeaderModifier defines a schema for a filter that modifies response headers. Support: Extended

Property	Type	Description
type	string	Type identifies the type of filter to apply. As with other API fields, types are classified into three conformance levels: - Core: Filter types and their corresponding configuration defined by "Support: Core" in this package, e.g. "RequestHeaderModifier". All implementations must support core filters. - Extended: Filter types and their corresponding configuration defined by "Support: Extended" in this package, e.g. "RequestMirror". Implementers are encouraged to support extended filters. - Implementation-specific: Filters that are defined and supported by specific vendors. In the future, filters showing convergence in behavior across multiple implementations will be considered for inclusion in extended or core conformance levels. Filter-specific configuration for such filters is specified using the ExtensionRef field. Type should be set to "ExtensionRef" for custom filters. Implementers are encouraged to define custom implementation types to extend the core API with implementation-specific behavior. If a reference to a custom filter type cannot be resolved, the filter MUST NOT be skipped. Instead, requests that would have been processed by that filter MUST receive a HTTP error response. Note that values may be added to this enum, implementations must ensure that unknown values will not cause a crash. Unknown values here must result in the implementation setting the Accepted Condition for the Route

Property	Type	Description
urlRewrite	object	URLRewrite defines a schema for a filter that modifies a request during forwarding. Support: Extended

17.1.30. .spec.rules[].filters[].extensionRef

Description

ExtensionRef is an optional, implementation-specific extension to the "filter" behavior. For example, resource "myroutefilter" in group "networking.example.net"). ExtensionRef MUST NOT be used for core and extended filters.

This filter can be used multiple times within the same rule.

Support: Implementation-specific

Type

object

Required

- **group**
- **kind**
- **name**

Property	Type	Description
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is kind of the referent. For example "HTTPRoute" or "Service".
name	string	Name is the name of the referent.

17.1.31. .spec.rules[].filters[].requestHeaderModifier

Description

RequestHeaderModifier defines a schema for a filter that modifies request headers.

Support: Core

Type

object

Property	Type	Description
add	array	Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name. Input: GET /foo HTTP/1.1 my-header: foo Config: add: - name: "my-header" value: "bar,baz" Output: GET /foo HTTP/1.1 my-header: foo,bar,baz
add[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.
remove	array (string)	Remove the given header(s) from the HTTP request before the action. The value of Remove is a list of HTTP header names. Note that the header names are case-insensitive (see https://datatracker.ietf.org/doc/html/rfc2616#section-4.2). Input: GET /foo HTTP/1.1 my-header1: foo my-header2: bar my-header3: baz Config: remove: ["my-header1", "my-header3"] Output: GET /foo HTTP/1.1 my-header2: bar
set	array	Set overwrites the request with the given header (name, value) before the action. Input: GET /foo HTTP/1.1 my-header: foo Config: set: - name: "my-header" value: "bar" Output: GET /foo HTTP/1.1 my-header: bar

Property	Type	Description
set[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

17.1.32. .spec.rules[].filters[].requestHeaderModifier.add

Description

Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name.

Input: GET /foo HTTP/1.1 my-header: foo

Config: add: - name: "my-header" value: "bar,baz"

Output: GET /foo HTTP/1.1 my-header: foo,bar,baz

Type

array

17.1.33. .spec.rules[].filters[].requestHeaderModifier.add[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

17.1.34. .spec.rules[].filters[].requestHeaderModifier.set

Description

Set overwrites the request with the given header (name, value) before the action.

Input: GET /foo HTTP/1.1 my-header: foo

Config: set: - name: "my-header" value: "bar"

Output: GET /foo HTTP/1.1 my-header: bar

Type

array

17.1.35. .spec.rules[].filters[].requestHeaderModifier.set[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

17.1.36. .spec.rules[].filters[].requestMirror

Description

RequestMirror defines a schema for a filter that mirrors requests. Requests are sent to the specified destination, but responses from that destination are ignored.

This filter can be used multiple times within the same rule. Note that not all implementations will be able to support mirroring to multiple backends.

Support: Extended

Type

object

Required

- **backendRef**

Property	Type	Description

Property	Type	Description
backendRef	object	BackendRef references a resource where mirrored requests are sent. Mirrored requests must be sent only to a single destination endpoint within this BackendRef, irrespective of how many endpoints are present within this BackendRef. If the referent cannot be found, this BackendRef is invalid and must be dropped from the Gateway. The controller must ensure the "ResolvedRefs" condition on the Route status is set to status: False and not configure this backend in the underlying implementation. If there is a cross-namespace reference to an existing object that is not allowed by a ReferenceGrant, the controller must ensure the "ResolvedRefs" condition on the Route is set to status: False, with the "RefNotPermitted" reason and not configure this backend in the underlying implementation. In either error case, the Message of the ResolvedRefs Condition should be used to provide more detail about the problem. Support: Extended for Kubernetes Service Support: Implementation-specific for any other resource
fraction	object	Fraction represents the fraction of requests that should be mirrored to BackendRef. Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

Property	Type	Description
percent	integer	Percent represents the percentage of requests that should be mirrored to BackendRef. Its minimum value is 0 (indicating 0% of requests) and its maximum value is 100 (indicating 100% of requests). Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

17.1.37. .spec.rules[].filters[].requestMirror.backendRef

Description

BackendRef references a resource where mirrored requests are sent.

Mirrored requests must be sent only to a single destination endpoint within this BackendRef, irrespective of how many endpoints are present within this BackendRef.

If the referent cannot be found, this BackendRef is invalid and must be dropped from the Gateway. The controller must ensure the "ResolvedRefs" condition on the Route status is set to **status: False** and not configure this backend in the underlying implementation.

If there is a cross-namespace reference to an **existing** object that is not allowed by a ReferenceGrant, the controller must ensure the "ResolvedRefs" condition on the Route is set to **status: False**, with the "RefNotPermitted" reason and not configure this backend in the underlying implementation.

In either error case, the Message of the **ResolvedRefs** Condition should be used to provide more detail about the problem.

Support: Extended for Kubernetes Service

Support: Implementation-specific for any other resource

Type

object

Required

- **name**

Property	Type	Description

Property	Type	Description
group	string	Group is the group of the referent. For example, "gateway.networking.k8s.io". When unspecified or empty string, core API group is inferred.
kind	string	Kind is the Kubernetes resource kind of the referent. For example "Service". Defaults to "Service" when not specified. ExternalName services can refer to CNAME DNS records that may live outside of the cluster and as such are difficult to reason about in terms of conformance. They also may not be safe to forward to (see CVE-2021-25740 for more information). Implementations SHOULD NOT support ExternalName Services. Support: Core (Services with a type other than ExternalName) Support: Implementation-specific (Services with type ExternalName)
name	string	Name is the name of the referent.
namespace	string	Namespace is the namespace of the backend. When unspecified, the local namespace is inferred. Note that when a namespace different than the local namespace is specified, a ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. Support: Core

Property	Type	Description
port	integer	Port specifies the destination port number to use for this resource. Port is required when the referent is a Kubernetes Service. In this case, the port number is the service port number, not the target port. For other resources, destination port might be derived from the referent resource or this field.

17.1.38. .spec.rules[].filters[].requestMirror.fraction

Description

Fraction represents the fraction of requests that should be mirrored to BackendRef.

Only one of Fraction or Percent may be specified. If neither field is specified, 100% of requests will be mirrored.

Type

object

Required

- **numerator**

Property	Type	Description
denominator	integer	
numerator	integer	

17.1.39. .spec.rules[].filters[].requestRedirect

Description

RequestRedirect defines a schema for a filter that responds to the request with an HTTP redirection.

Support: Core

Type

object

Property	Type	Description

Property	Type	Description
hostname	string	Hostname is the hostname to be used in the value of the Location header in the response. When empty, the hostname in the Host header of the request is used. Support: Core
path	object	Path defines parameters used to modify the path of the incoming request. The modified path is then used to construct the Location header. When empty, the request path is used as-is. Support: Extended

Property	Type	Description
port	integer	Port is the port to be used in the value of the Location header in the response. If no port is specified, the redirect port MUST be derived using the following rules: * If redirect scheme is not-empty, the redirect port MUST be the well-known port associated with the redirect scheme. Specifically "http" to port 80 and "https" to port 443. If the redirect scheme does not have a well-known port, the listener port of the Gateway SHOULD be used. * If redirect scheme is empty, the redirect port MUST be the Gateway Listener port. Implementations SHOULD NOT add the port number in the 'Location' header in the following cases: * A Location header that will use HTTP (whether that is determined via the Listener protocol or the Scheme field) <i>and</i> use port 80. * A Location header that will use HTTPS (whether that is determined via the Listener protocol or the Scheme field) <i>and</i> use port 443. Support: Extended

Property	Type	Description
scheme	string	Scheme is the scheme to be used in the value of the Location header in the response. When empty, the scheme of the request is used. Scheme redirects can affect the port of the redirect, for more information, refer to the documentation for the port field of this filter. Note that values may be added to this enum, implementations must ensure that unknown values will not cause a crash. Unknown values here must result in the implementation setting the Accepted Condition for the Route to status: False, with a Reason of UnsupportedValue. Support: Extended
statusCode	integer	StatusCode is the HTTP status code to be used in response. Note that values may be added to this enum, implementations must ensure that unknown values will not cause a crash. Unknown values here must result in the implementation setting the Accepted Condition for the Route to status: False, with a Reason of UnsupportedValue. Support: Core

17.1.40. .spec.rules[].filters[].requestRedirect.path

Description

Path defines parameters used to modify the path of the incoming request. The modified path is then used to construct the **Location** header. When empty, the request path is used as-is.

Support: Extended

Type

object

Required

- type

Property	Type	Description
replaceFullPath	string	ReplaceFullPath specifies the value with which to replace the full path of a request during a rewrite or redirect.
replacePrefixMatch	string	ReplacePrefixMatch specifies the value with which to replace the prefix match of a request during a rewrite or redirect. For example, a request to <code>/foo/bar</code> with a prefix match of <code>/foo</code> and a ReplacePrefixMatch of <code>/xyz</code> would be modified to <code>/xyz/bar</code>. Note that this matches the behavior of the PathPrefix match type. This matches full path elements. A path element refers to the list of labels in the path split by the <code>/</code> separator. When specified, a trailing <code>/</code> is ignored. For example, the paths <code>/abc</code>, <code>/abc/</code>, and <code>/abc/def</code> would all match the prefix <code>/abc</code>, but the path <code>/abcd</code> would not. ReplacePrefixMatch is only compatible with a PathPrefix HTTPRouteMatch. Using any other HTTPRouteMatch type on the same HTTPRouteRule will result in the implementation setting the Accepted Condition for the Route to status: False. Request Path Prefix Match Replace Prefix Modified Path

Property	Type	Description
type	string	Type defines the type of path modifier. Additional types may be added in a future release of the API. Note that values may be added to this enum, implementations must ensure that unknown values will not cause a crash. Unknown values here must result in the implementation setting the Accepted Condition for the Route to status: False, with a Reason of UnsupportedValue.

17.1.41. .spec.rules[].filters[].responseHeaderModifier

Description

ResponseHeaderModifier defines a schema for a filter that modifies response headers.
Support: Extended

Type

object

Property	Type	Description
add	array	Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name. Input: GET /foo HTTP/1.1 my-header: foo Config: add: - name: "my-header" value: "bar,baz" Output: GET /foo HTTP/1.1 my-header: foo,bar,baz
add[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Property	Type	Description
remove	array (string)	Remove the given header(s) from the HTTP request before the action. The value of Remove is a list of HTTP header names. Note that the header names are case-insensitive (see https://datatracker.ietf.org/doc/html/rfc2616#section-4.2). Input: GET /foo HTTP/1.1 my-header1: foo my-header2: bar my-header3: baz Config: remove: ["my-header1", "my-header3"] Output: GET /foo HTTP/1.1 my-header2: bar
set	array	Set overwrites the request with the given header (name, value) before the action. Input: GET /foo HTTP/1.1 my-header: foo Config: set: - name: "my-header" value: "bar" Output: GET /foo HTTP/1.1 my-header: bar
set[]	object	HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

17.1.42. .spec.rules[].filters[].responseHeaderModifier.add

Description

Add adds the given header(s) (name, value) to the request before the action. It appends to any existing values associated with the header name.

Input: GET /foo HTTP/1.1 my-header: foo

Config: add: - name: "my-header" value: "bar,baz"

Output: GET /foo HTTP/1.1 my-header: foo,bar,baz

Type

array

17.1.43. .spec.rules[].filters[].responseHeaderModifier.add[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

17.1.44. .spec.rules[].filters[].responseHeaderModifier.set

Description

Set overwrites the request with the given header (name, value) before the action.

Input: GET /foo HTTP/1.1 my-header: foo

Config: set: - name: "my-header" value: "bar"

Output: GET /foo HTTP/1.1 my-header: bar

Type

array

17.1.45. .spec.rules[].filters[].responseHeaderModifier.set[]

Description

HTTPHeader represents an HTTP Header name and value as defined by RFC 7230.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent.
value	string	Value is the value of HTTP Header to be matched.

17.1.46. .spec.rules[].filters[].urlRewrite**Description**

URLRewrite defines a schema for a filter that modifies a request during forwarding.

Support: Extended

Type

object

Property	Type	Description
hostname	string	Hostname is the value to be used to replace the Host header value during forwarding. Support: Extended

Property	Type	Description
path	object	Path defines a path rewrite. Support: Extended

17.1.47. .spec.rules[].filters[].urlRewrite.path

Description

Path defines a path rewrite.
Support: Extended

Type

object

Required

- **type**

Property	Type	Description
replaceFullPath	string	ReplaceFullPath specifies the value with which to replace the full path of a request during a rewrite or redirect.

Property	Type	Description
replacePrefixMatch	string	ReplacePrefixMatch specifies the value with which to replace the prefix match of a request during a rewrite or redirect. For example, a request to <code>/foo/bar</code> with a prefix match of <code>/foo</code> and a ReplacePrefixMatch of <code>/xyz</code> would be modified to <code>/xyz/bar</code>. Note that this matches the behavior of the PathPrefix match type. This matches full path elements. A path element refers to the list of labels in the path split by the <code>/</code> separator. When specified, a trailing <code>/</code> is ignored. For example, the paths <code>/abc</code>, <code>/abc/</code>, and <code>/abc/def</code> would all match the prefix <code>/abc</code>, but the path <code>/abcd</code> would not. ReplacePrefixMatch is only compatible with a PathPrefix HTTPRouteMatch. Using any other HTTPRouteMatch type on the same HTTPRouteRule will result in the implementation setting the Accepted Condition for the Route to status: False. Request Path Prefix Match Replace Prefix Modified Path
type	string	Type defines the type of path modifier. Additional types may be added in a future release of the API. Note that values may be added to this enum, implementations must ensure that unknown values will not cause a crash. Unknown values here must result in the implementation setting the Accepted Condition for the Route to status: False, with a Reason of UnsupportedValue.

17.1.48. .spec.rules[].matches

Description

Matches define conditions used for matching the rule against incoming HTTP requests. Each match is independent, i.e. this rule will be matched if **any** one of the matches is satisfied.

For example, take the following matches configuration:

matches: - path: value: "/foo" headers: - name: "version" value: "v2" - path: value: "/v2/foo"

For a request to match against this rule, a request must satisfy EITHER of the two conditions:

- path prefixed with **/foo** AND contains the header **version: v2**
- path prefix of **/v2/foo**

See the documentation for HTTPRouteMatch on how to specify multiple match conditions that should be ANDed together.

If no matches are specified, the default is a prefix path match on "/", which has the effect of matching every HTTP request.

Proxy or Load Balancer routing configuration generated from HTTPRoutes MUST prioritize matches based on the following criteria, continuing on ties. Across all rules specified on applicable Routes, precedence must be given to the match having:

- "Exact" path match.
- "Prefix" path match with largest number of characters.
- Method match.
- Largest number of header matches.
- Largest number of query param matches.

Note: The precedence of RegularExpression path matches are implementation-specific.

If ties still exist across multiple Routes, matching precedence MUST be determined in order of the following criteria, continuing on ties:

- The oldest Route based on creation timestamp.
- The Route appearing first in alphabetical order by "{namespace}/{name}".

If ties still exist within an HTTPRoute, matching precedence MUST be granted to the FIRST matching rule (in list order) with a match meeting the above criteria.

When no rules matching a request have been successfully attached to the parent a request is coming from, a HTTP 404 status code MUST be returned.

Type

array

17.1.49. .spec.rules[].matches[]

Description

HTTPRouteMatch defines the predicate used to match requests to a given action. Multiple match types are ANDed together, i.e. the match will evaluate to true only if all conditions are satisfied.

For example, the match below will match a HTTP request only if its path starts with **/foo** AND it contains the **version: v1** header:

match:

```
path:
  value: "/foo"
headers:
  - name: "version"
    value "v1"
```

Type

object

Property	Type	Description
headers	array	Headers specifies HTTP request header matchers. Multiple match values are ANDed together, meaning, a request must match all the specified headers to select the route.
headers[]	object	HTTPHeaderMatch describes how to select a HTTP route by matching HTTP request headers.
method	string	Method specifies HTTP method matcher. When specified, this route will be matched only if the request has the specified method. Support: Extended
path	object	Path specifies a HTTP request path matcher. If this field is not specified, a default prefix match on the "/" path is provided.
queryParams	array	QueryParams specifies HTTP query parameter matchers. Multiple match values are ANDed together, meaning, a request must match all the specified query parameters to select the route. Support: Extended
queryParams[]	object	HTTPQueryParamMatch describes how to select a HTTP route by matching HTTP query parameters.

17.1.50. .spec.rules[].matches[].headers

Description

Headers specifies HTTP request header matchers. Multiple match values are ANDed together, meaning, a request must match all the specified headers to select the route.

Type

array

17.1.51. .spec.rules[].matches[].headers[]

Description

HTTPHeaderMatch describes how to select a HTTP route by matching HTTP request headers.

Type

object

Required

- **name**
- **value**

Property	Type	Description
name	string	Name is the name of the HTTP Header to be matched. Name matching MUST be case-insensitive. (See https://tools.ietf.org/html/rfc7230#section-3.2). If multiple entries specify equivalent header names, only the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent header name MUST be ignored. Due to the case-insensitivity of header names, "foo" and "Foo" are considered equivalent. When a header is repeated in an HTTP request, it is implementation-specific behavior as to how this is represented. Generally, proxies should follow the guidance from the RFC: https://www.rfc-editor.org/rfc/rfc7230.html#section-3.2.2 regarding processing a repeated header, with special handling for "Set-Cookie".

Property	Type	Description
type	string	Type specifies how to match against the value of the header. Support: Core (Exact) Support: Implementation-specific (RegularExpression) Since RegularExpression HeaderMatchType has implementation-specific conformance, implementations can support POSIX, PCRE or any other dialects of regular expressions. Please read the implementation's documentation to determine the supported dialect.
value	string	Value is the value of HTTP Header to be matched.

17.1.52. .spec.rules[].matches[].path

Description

Path specifies a HTTP request path matcher. If this field is not specified, a default prefix match on the "/" path is provided.

Type

object

Property	Type	Description
type	string	Type specifies how to match against the path Value. Support: Core (Exact, PathPrefix) Support: Implementation-specific (RegularExpression)
value	string	Value of the HTTP path to match against.

17.1.53. .spec.rules[].matches[].queryParams

Description

QueryParams specifies HTTP query parameter matchers. Multiple match values are ANDed together, meaning, a request must match all the specified query parameters to select the route.

Support: Extended

Type

array

17.1.54. .spec.rules[].matches[].queryParams[]

Description

HTTPQueryParamMatch describes how to select a HTTP route by matching HTTP query parameters.

Type

object

Required

- **name**
- **value**

Property	Type	Description

Property	Type	Description
name	string	Name is the name of the HTTP query param to be matched. This must be an exact string match. (See https://tools.ietf.org/html/rfc7230#section-2.7.3). If multiple entries specify equivalent query param names, only the first entry with an equivalent name MUST be considered for a match. Subsequent entries with an equivalent query param name MUST be ignored. If a query param is repeated in an HTTP request, the behavior is purposely left undefined, since different data planes have different capabilities. However, it is recommended that implementations should match against the first value of the param if the data plane supports it, as this behavior is expected in other load balancing contexts outside of the Gateway API. Users SHOULD NOT route traffic based on repeated query params to guard themselves against potential differences in the implementations.

Property	Type	Description
type	string	Type specifies how to match against the value of the query parameter. Support: Extended (Exact) Support: Implementation-specific (RegularExpression) Since RegularExpression QueryParamMatchType has Implementation-specific conformance, implementations can support POSIX, PCRE or any other dialects of regular expressions. Please read the implementation's documentation to determine the supported dialect.
value	string	Value is the value of HTTP query param to be matched.

17.1.55. .spec.rules[].timeouts

Description

Timeouts defines the timeouts that can be configured for an HTTP request.

Support: Extended

Type

object

Property	Type	Description

Property	Type	Description
backendRequest	string	BackendRequest specifies a timeout for an individual request from the gateway to a backend. This covers the time from when the request first starts being sent from the gateway to when the full response has been received from the backend. Setting a timeout to the zero duration (e.g. "0s") SHOULD disable the timeout completely. Implementations that cannot completely disable the timeout MUST instead interpret the zero duration as the longest possible value to which the timeout can be set. An entire client HTTP transaction with a gateway, covered by the Request timeout, may result in more than one call from the gateway to the destination backend, for example, if automatic retries are supported. The value of BackendRequest must be a Gateway API Duration string as defined by GEP-2257. When this field is unspecified, its behavior is implementation-specific; when specified, the value of BackendRequest must be no more than the value of the Request timeout (since the Request timeout encompasses the BackendRequest timeout). Support: Extended

Property	Type	Description
request	string	Request specifies the maximum duration for a gateway to respond to an HTTP request. If the gateway has not been able to respond before this deadline is met, the gateway MUST return a timeout error. For example, setting the rules.timeouts.request field to the value 10s in an HTTPRoute will cause a timeout if a client request is taking longer than 10 seconds to complete. Setting a timeout to the zero duration (e.g. "0s") SHOULD disable the timeout completely. Implementations that cannot completely disable the timeout MUST instead interpret the zero duration as the longest possible value to which the timeout can be set. This timeout is intended to cover as close to the whole request-response transaction as possible although an implementation MAY choose to start the timeout after the entire request stream has been received instead of immediately after the transaction is initiated by the client. The value of Request is a Gateway API Duration string as defined by GEP-2257. When this field is unspecified, request timeout behavior is implementation-specific. Support: Extended

17.1.56. .status

Description

Status defines the current state of HTTPRoute.

Type

object

Required

- **parents**

Property	Type	Description
parents	array	Parents is a list of parent resources (usually Gateways) that are associated with the route, and the status of the route with respect to each parent. When this route attaches to a parent, the controller that manages the parent must add an entry to this list when the controller first sees the route and should update the entry as appropriate when the route or gateway is modified. Note that parent references that cannot be resolved by an implementation of this API will not be added to this list. Implementations of this API can only populate Route status for the Gateways/parent resources they are responsible for. A maximum of 32 Gateways will be represented in this list. An empty list means the route has not been attached to any Gateway.
parents[]	object	RouteParentStatus describes the status of a route with respect to an associated Parent.

17.1.57. .status.parents**Description**

Parents is a list of parent resources (usually Gateways) that are associated with the route, and the status of the route with respect to each parent. When this route attaches to a parent, the controller that manages the parent must add an entry to this list when the controller first sees the route and should update the entry as appropriate when the route or gateway is modified.

Note that parent references that cannot be resolved by an implementation of this API will not be added to this list. Implementations of this API can only populate Route status for the Gateways/parent resources they are responsible for.

A maximum of 32 Gateways will be represented in this list. An empty list means the route has not been attached to any Gateway.

Type

array**17.1.58. .status.parents[]****Description**

RouteParentStatus describes the status of a route with respect to an associated Parent.

Type**object****Required**

- **controllerName**
- **parentRef**

Property	Type	Description
conditions	array	Conditions describes the status of the route with respect to the Gateway. Note that the route's availability is also subject to the Gateway's own status conditions and listener status. If the Route's ParentRef specifies an existing Gateway that supports Routes of this kind AND that Gateway's controller has sufficient access, then that Gateway's controller MUST set the "Accepted" condition on the Route, to indicate whether the route has been accepted or rejected by the Gateway, and why. A Route MUST be considered "Accepted" if at least one of the Route's rules is implemented by the Gateway. There are a number of cases where the "Accepted" condition may not be set due to lack of controller visibility, that includes when: * The Route refers to a nonexistent parent. * The Route is of a type that the controller does not support. * The Route is in a namespace the controller does not have access to.

Property	Type	Description
conditions[]	object	Condition contains details for one aspect of the current state of this API Resource.
controllerName	string	ControllerName is a domain/path string that indicates the name of the controller that wrote this status. This corresponds with the controllerName field on GatewayClass. Example: "example.net/gateway-controller". The format of this field is DOMAIN "/" PATH, where DOMAIN and PATH are valid Kubernetes names (https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names). Controllers MUST populate this field when writing status. Controllers should ensure that entries to status populated with their ControllerName are cleaned up when they are no longer necessary.
parentRef	object	ParentRef corresponds with a ParentRef in the spec that this RouteParentStatus struct describes the status of.

17.1.59. .status.parents[].conditions

Description

Conditions describes the status of the route with respect to the Gateway. Note that the route's availability is also subject to the Gateway's own status conditions and listener status.

If the Route's ParentRef specifies an existing Gateway that supports Routes of this kind AND that Gateway's controller has sufficient access, then that Gateway's controller MUST set the "Accepted" condition on the Route, to indicate whether the route has been accepted or rejected by the Gateway, and why.

A Route MUST be considered "Accepted" if at least one of the Route's rules is implemented by the Gateway.

There are a number of cases where the "Accepted" condition may not be set due to lack of controller visibility, that includes when:

- The Route refers to a nonexistent parent.
- The Route is of a type that the controller does not support.
- The Route is in a namespace the controller does not have access to.

Type

array

17.1.60. `.status.parents[].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 <code>.metadata.generation</code> that the condition was set based upon. For instance, if <code>.metadata.generation</code> is currently 12, but the <code>.status.conditions[x].observedGeneration</code> is 9, the condition is out of date with respect to the current state of the instance.

Property	Type	Description
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.

17.1.61. .status.parents[].parentRef

Description

ParentRef corresponds with a ParentRef in the spec that this RouteParentStatus struct describes the status of.

Type

object

Required

- **name**

Property	Type	Description
group	string	Group is the group of the referent. When unspecified, "gateway.networking.k8s.io" is inferred. To set the core API group (such as for a "Service" kind referent), Group must be explicitly set to "" (empty string). Support: Core

Property	Type	Description
kind	string	Kind is kind of the referent. There are two kinds of parent resources with "Core" support: * Gateway (Gateway conformance profile) * Service (Mesh conformance profile, ClusterIP Services only) Support for other resources is Implementation-Specific.
name	string	Name is the name of the referent. Support: Core
namespace	string	Namespace is the namespace of the referent. When unspecified, this refers to the local namespace of the Route. Note that there are specific rules for ParentRefs which cross namespace boundaries. Cross-namespace references are only valid if they are explicitly allowed by something in the namespace they are referring to. For example: Gateway has the AllowedRoutes field, and ReferenceGrant provides a generic way to enable any other kind of cross-namespace reference. Support: Core

Property	Type	Description
port	integer	Port is the network port this Route targets. It can be interpreted differently based on the type of parent resource. When the parent resource is a Gateway, this targets all listeners listening on the specified port that also support this kind of Route (and select this Route). It's not recommended to set Port unless the networking behaviors specified in a Route must apply to a specific port as opposed to a listener(s) whose port(s) may be changed. When both Port and SectionName are specified, the name and port of the selected listener must match both specified values. Implementations MAY choose to support other parent resources. Implementations supporting other types of parent resources MUST clearly document how/if Port is interpreted. For the purpose of status, an attachment is considered successful as long as the parent resource accepts it partially. For example, Gateway listeners can restrict which Routes can attach to them by Route kind, namespace, or hostname. If 1 of 2 Gateway listeners accept attachment from the referencing Route, the Route MUST be considered successfully attached. If no Gateway listeners accept attachment from this Route, the Route MUST be considered detached from the Gateway. Support: Extended

Property	Type	Description
sectionName	string	SectionName is the name of a section within the target resource. In the following resources, SectionName is interpreted as the following: * Gateway: Listener name. When both Port (experimental) and SectionName are specified, the name and port of the selected listener must match both specified values. * Service: Port name. When both Port (experimental) and SectionName are specified, the name and port of the selected listener must match both specified values. Implementations MAY choose to support attaching Routes to other resources. If that is the case, they MUST clearly document how SectionName is interpreted. When unspecified (empty string), this will reference the entire resource. For the purpose of status, an attachment is considered successful if at least one section in the parent resource accepts it. For example, Gateway listeners can restrict which Routes can attach to them by Route kind, namespace, or hostname. If 1 of 2 Gateway listeners accept attachment from the referencing Route, the Route MUST be considered successfully attached. If no Gateway listeners accept attachment from this Route, the Route MUST be considered detached from the Gateway. Support: Core

17.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/gateway.networking.k8s.io/v1/httproutes**
 - **GET**: list objects of kind HTTPRoute

- **/apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/httproutes**
 - **DELETE**: delete collection of HTTPRoute
 - **GET**: list objects of kind HTTPRoute
 - **POST**: create a HTTPRoute
- **/apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/httproutes/{name}**
 - **DELETE**: delete a HTTPRoute
 - **GET**: read the specified HTTPRoute
 - **PATCH**: partially update the specified HTTPRoute
 - **PUT**: replace the specified HTTPRoute
- **/apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/httproutes/{name}/status**
 - **GET**: read status of the specified HTTPRoute
 - **PATCH**: partially update status of the specified HTTPRoute
 - **PUT**: replace status of the specified HTTPRoute

17.2.1. /apis/gateway.networking.k8s.io/v1/httproutes

HTTP method

GET

Description

list objects of kind HTTPRoute

Table 17.1. HTTP responses

HTTP code	Reponse body
200 - OK	HTTPRouteList schema
401 - Unauthorized	Empty

17.2.2. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/httproutes

HTTP method

DELETE

Description

delete collection of HTTPRoute

Table 17.2. HTTP responses

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

HTTP method**GET****Description**

list objects of kind HTTPRoute

Table 17.3. HTTP responses

HTTP code	Reponse body
200 - OK	HTTPRouteList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create a HTTPRoute

Table 17.4. 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

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 17.5. Body parameters

Parameter	Type	Description
body	HTTPRoute schema	

Table 17.6. HTTP responses

HTTP code	Response body
200 - OK	HTTPRoute schema
201 - Created	HTTPRoute schema
202 - Accepted	HTTPRoute schema
401 - Unauthorized	Empty

17.2.3. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/httproutes/{name}

Table 17.7. Global path parameters

Parameter	Type	Description
name	string	name of the HTTPRoute

HTTP method**DELETE****Description**

delete a HTTPRoute

Table 17.8. 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 17.9. HTTP responses

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

HTTP method**GET****Description**

read the specified HTTPRoute

Table 17.10. HTTP responses

HTTP code	Reponse body
200 - OK	HTTPRoute schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified HTTPRoute

Table 17.11. 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 17.12. HTTP responses

HTTP code	Response body
200 - OK	HTTPRoute schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified HTTPRoute

Table 17.13. 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 17.14. Body parameters

Parameter	Type	Description
body	HTTPRoute schema	

Table 17.15. HTTP responses

HTTP code	Response body
200 - OK	HTTPRoute schema
201 - Created	HTTPRoute schema
401 - Unauthorized	Empty

17.2.4. /apis/gateway.networking.k8s.io/v1/namespaces/{namespace}/httproutes/{name}

Table 17.16. Global path parameters

Parameter	Type	Description
name	string	name of the HTTPRoute

HTTP method

GET

Description

read status of the specified HTTPRoute

Table 17.17. HTTP responses

HTTP code	Response body
200 - OK	HTTPRoute schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified HTTPRoute

Table 17.18. 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 17.19. HTTP responses

HTTP code	Response body
200 - OK	HTTPRoute schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified HTTPRoute

Table 17.20. 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 17.21. Body parameters

Parameter	Type	Description
body	HTTPRoute schema	

Table 17.22. HTTP responses

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

CHAPTER 18. INGRESS [NETWORKING.K8S.IO/V1]

Description

Ingress is a collection of rules that allow inbound connections to reach the endpoints defined by a backend. An Ingress can be configured to give services externally-reachable urls, load balance traffic, terminate SSL, offer name based virtual hosting etc.

Type

object

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
spec	object	IngressSpec describes the Ingress the user wishes to exist.
status	object	IngressStatus describe the current state of the Ingress.

18.1.1. .spec

Description

IngressSpec describes the Ingress the user wishes to exist.

Type

object

Property	Type	Description
defaultBackend	object	IngressBackend describes all endpoints for a given service and port.
ingressClassName	string	ingressClassName is the name of an IngressClass cluster resource. Ingress controller implementations use this field to know whether they should be serving this Ingress resource, by a transitive connection (controller → IngressClass → Ingress resource). Although the kubernetes.io/ingress.class annotation (simple constant name) was never formally defined, it was widely supported by Ingress controllers to create a direct binding between Ingress controller and Ingress resources. Newly created Ingress resources should prefer using the field. However, even though the annotation is officially deprecated, for backwards compatibility reasons, ingress controllers should still honor that annotation if present.
rules	array	rules is a list of host rules used to configure the Ingress. If unspecified, or no rule matches, all traffic is sent to the default backend.
rules[]	object	IngressRule represents the rules mapping the paths under a specified host to the related backend services. Incoming requests are first evaluated for a host match, then routed to the backend associated with the matching IngressRuleValue.

Property	Type	Description
tls	array	tls represents the TLS configuration. Currently the Ingress only supports a single TLS port, 443. If multiple members of this list specify different hosts, they will be multiplexed on the same port according to the hostname specified through the SNI TLS extension, if the ingress controller fulfilling the ingress supports SNI.
tls[]	object	IngressTLS describes the transport layer security associated with an ingress.

18.1.2. .spec.defaultBackend

Description

IngressBackend describes all endpoints for a given service and port.

Type

object

Property	Type	Description
resource	TypedLocalObjectReference	resource is an ObjectRef to another Kubernetes resource in the namespace of the Ingress object. If resource is specified, a service.Name and service.Port must not be specified. This is a mutually exclusive setting with "Service".
service	object	IngressServiceBackend references a Kubernetes Service as a Backend.

18.1.3. .spec.defaultBackend.service

Description

IngressServiceBackend references a Kubernetes Service as a Backend.

Type

object

Required

- **name**

Property	Type	Description
name	string	name is the referenced service. The service must exist in the same namespace as the Ingress object.
port	object	ServiceBackendPort is the service port being referenced.

18.1.4. .spec.defaultBackend.service.port

Description

ServiceBackendPort is the service port being referenced.

Type

object

Property	Type	Description
name	string	name is the name of the port on the Service. This is a mutually exclusive setting with "Number".
number	integer	number is the numerical port number (e.g. 80) on the Service. This is a mutually exclusive setting with "Name".

18.1.5. .spec.rules

Description

rules is a list of host rules used to configure the Ingress. If unspecified, or no rule matches, all traffic is sent to the default backend.

Type

array

18.1.6. .spec.rules[]

Description

IngressRule represents the rules mapping the paths under a specified host to the related backend services. Incoming requests are first evaluated for a host match, then routed to the backend associated with the matching IngressRuleValue.

Type

object

Property	Type	Description
host	string	host is the fully qualified domain name of a network host, as defined by RFC 3986. Note the following deviations from the "host" part of the URI as defined in RFC 3986: 1. IPs are not allowed. Currently an IngressRuleValue can only apply to the IP in the Spec of the parent Ingress. 2. The <code>:</code> delimiter is not respected because ports are not allowed. Currently the port of an Ingress is implicitly <code>:80</code> for http and <code>:443</code> for https. Both these may change in the future. Incoming requests are matched against the host before the IngressRuleValue. If the host is unspecified, the Ingress routes all traffic based on the specified IngressRuleValue. host can be "precise" which is a domain name without the terminating dot of a network host (e.g. "foo.bar.com") or "wildcard", which is a domain name prefixed with a single wildcard label (e.g. ".foo.com"). The wildcard character " must appear by itself as the first DNS label and matches only a single label. You cannot have a wildcard label by itself (e.g. Host == "*"). Requests will be matched against the Host field in the following way: 1. If host is precise, the request matches this rule if the http host header is equal to Host. 2. If host is a wildcard, then the request matches this rule if the http host header is to equal to the suffix (removing the first label) of the wildcard rule.

Property	Type	Description
http	object	HTTPIngressRuleValue is a list of http selectors pointing to backends. In the example: <a href="http://<host>/<path>?<searchpart>">http://<host>/<path>?<searchpart> → backend where where parts of the url correspond to RFC 3986, this resource will be used to match against everything after the last '/' and before the first '?' or '#'.

18.1.7. .spec.rules[].http

Description

HTTPIngressRuleValue is a list of http selectors pointing to backends. In the example: <http://<host>/<path>?<searchpart>> → backend where where parts of the url correspond to RFC 3986, this resource will be used to match against everything after the last '/' and before the first '?' or '#'.

Type

object

Required

- **paths**

Property	Type	Description
paths	array	paths is a collection of paths that map requests to backends.
paths[]	object	HTTPIngressPath associates a path with a backend. Incoming urls matching the path are forwarded to the backend.

18.1.8. .spec.rules[].http.paths

Description

paths is a collection of paths that map requests to backends.

Type

array

18.1.9. .spec.rules[].http.paths[]

Description

HTTPIngressPath associates a path with a backend. Incoming urls matching the path are forwarded to the backend.

Type

object

Required

- **pathType**
- **backend**

Property	Type	Description
backend	object	IngressBackend describes all endpoints for a given service and port.
path	string	path is matched against the path of an incoming request. Currently it can contain characters disallowed from the conventional "path" part of a URL as defined by RFC 3986. Paths must begin with a '/' and must be present when using PathType with value "Exact" or "Prefix".
pathType	string	pathType determines the interpretation of the path matching. PathType can be one of the following values: * Exact: Matches the URL path exactly. * Prefix: Matches based on a URL path prefix split by '/'. Matching is done on a path element by element basis. A path element refers is the list of labels in the path split by the '/' separator. A request is a match for path p if every p is an element-wise prefix of p of the request path. Note that if the last element of the path is a substring of the last element in request path, it is not a match (e.g. /foo/bar matches /foo/bar/baz, but does not match /foo/barbaz). * ImplementationSpecific: Interpretation of the Path matching is up to the IngressClass. Implementations can treat this as a separate PathType or treat it identically to

Property	Type	Description
		Prefix or Exact path types. Implementations are required to support all path types. Possible enum values: - "Exact" matches the URL path exactly and with case sensitivity. - "ImplementationSpecific" matching is up to the IngressClass. Implementations can treat this as a separate PathType or treat it identically to Prefix or Exact path types. - "Prefix" matches based on a URL path prefix split by '/'. Matching is case sensitive and done on a path element by element basis. A path element refers to the list of labels in the path split by the '/' separator. A request is a match for path p if every p is an element-wise prefix of p of the request path. Note that if the last element of the path is a substring of the last element in request path, it is not a match (e.g. /foo/bar matches /foo/bar/baz, but does not match /foo/barbaz). If multiple matching paths exist in an Ingress spec, the longest matching path is given priority. Examples: - /foo/bar does not match requests to /foo/barbaz - /foo/bar matches request to /foo/bar and /foo/bar/baz - /foo and /foo/ both match requests to /foo and /foo/. If both paths are present in an Ingress spec, the longest matching path (/foo/) is given priority.

18.1.10. .spec.rules[].http.paths[].backend

Description

IngressBackend describes all endpoints for a given service and port.

Type

object

Property	Type	Description

Property	Type	Description
resource	TypedLocalObjectReference	resource is an ObjectRef to another Kubernetes resource in the namespace of the Ingress object. If resource is specified, a service.Name and service.Port must not be specified. This is a mutually exclusive setting with "Service".
service	object	IngressServiceBackend references a Kubernetes Service as a Backend.

18.1.11. .spec.rules[].http.paths[].backend.service

Description

IngressServiceBackend references a Kubernetes Service as a Backend.

Type

object

Required

- **name**

Property	Type	Description
name	string	name is the referenced service. The service must exist in the same namespace as the Ingress object.
port	object	ServiceBackendPort is the service port being referenced.

18.1.12. .spec.rules[].http.paths[].backend.service.port

Description

ServiceBackendPort is the service port being referenced.

Type

object

Property	Type	Description
name	string	name is the name of the port on the Service. This is a mutually exclusive setting with "Number".

Property	Type	Description
number	integer	number is the numerical port number (e.g. 80) on the Service. This is a mutually exclusive setting with "Name".

18.1.13. .spec.tls

Description

tls represents the TLS configuration. Currently the Ingress only supports a single TLS port, 443. If multiple members of this list specify different hosts, they will be multiplexed on the same port according to the hostname specified through the SNI TLS extension, if the ingress controller fulfilling the ingress supports SNI.

Type

array

18.1.14. .spec.tls[]

Description

IngressTLS describes the transport layer security associated with an ingress.

Type

object

Property	Type	Description
hosts	array (string)	hosts is a list of hosts included in the TLS certificate. The values in this list must match the name/s used in the tlsSecret. Defaults to the wildcard host setting for the loadbalancer controller fulfilling this Ingress, if left unspecified.
secretName	string	secretName is the name of the secret used to terminate TLS traffic on port 443. Field is left optional to allow TLS routing based on SNI hostname alone. If the SNI host in a listener conflicts with the "Host" header field used by an IngressRule, the SNI host is used for termination and value of the "Host" header is used for routing.

18.1.15. .status

Description

IngressStatus describe the current state of the Ingress.

Type

object

Property	Type	Description
loadBalancer	object	IngressLoadBalancerStatus represents the status of a load-balancer.

18.1.16. .status.loadBalancer

Description

IngressLoadBalancerStatus represents the status of a load-balancer.

Type

object

Property	Type	Description
ingress	array	ingress is a list containing ingress points for the load-balancer.
ingress[]	object	IngressLoadBalancerIngress represents the status of a load-balancer ingress point.

18.1.17. .status.loadBalancer.ingress

Description

ingress is a list containing ingress points for the load-balancer.

Type

array

18.1.18. .status.loadBalancer.ingress[]

Description

IngressLoadBalancerIngress represents the status of a load-balancer ingress point.

Type

object

Property	Type	Description
hostname	string	hostname is set for load-balancer ingress points that are DNS based.

Property	Type	Description
ip	string	ip is set for load-balancer ingress points that are IP based.
ports	array	ports provides information about the ports exposed by this LoadBalancer.
ports[]	object	IngressPortStatus represents the error condition of a service port

18.1.19. .status.loadBalancer.ingress[].ports

Description

ports provides information about the ports exposed by this LoadBalancer.

Type

array

18.1.20. .status.loadBalancer.ingress[].ports[]

Description

IngressPortStatus represents the error condition of a service port

Type

object

Required

- **port**
- **protocol**

Property	Type	Description
error	string	error is to record the problem with the service port The format of the error shall comply with the following rules: - built-in error values shall be specified in this file and those shall use CamelCase names - cloud provider specific error values must have names that comply with the format foo.example.com/CamelCase.
port	integer	port is the port number of the ingress port.

Property	Type	Description
protocol	string	protocol is the protocol of the ingress port. The supported values are: "TCP", "UDP", "SCTP" Possible enum values: - "SCTP" is the SCTP protocol. - "TCP" is the TCP protocol. - "UDP" is the UDP protocol.

18.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/networking.k8s.io/v1/ingresses**
 - **GET**: list or watch objects of kind Ingress
- **/apis/networking.k8s.io/v1/watch/ingresses**
 - **GET**: watch individual changes to a list of Ingress. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses**
 - **DELETE**: delete collection of Ingress
 - **GET**: list or watch objects of kind Ingress
 - **POST**: create an Ingress
- **/apis/networking.k8s.io/v1/watch/namespaces/{namespace}/ingresses**
 - **GET**: watch individual changes to a list of Ingress. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}**
 - **DELETE**: delete an Ingress
 - **GET**: read the specified Ingress
 - **PATCH**: partially update the specified Ingress
 - **PUT**: replace the specified Ingress
- **/apis/networking.k8s.io/v1/watch/namespaces/{namespace}/ingresses/{name}**
 - **GET**: watch changes to an object of kind Ingress. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.
- **/apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}/status**

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

18.2.1. /apis/networking.k8s.io/v1/ingresses

HTTP method

GET

Description

list or watch objects of kind Ingress

Table 18.1. HTTP responses

HTTP code	Reponse body
200 - OK	IngressList schema
401 - Unauthorized	Empty

18.2.2. /apis/networking.k8s.io/v1/watch/ingresses

HTTP method

GET

Description

watch individual changes to a list of Ingress. deprecated: use the 'watch' parameter with a list operation instead.

Table 18.2. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

18.2.3. /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses

HTTP method

DELETE

Description

delete collection of Ingress

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

Table 18.4. HTTP responses

HTTP code	Response body
200 - OK	Status schema
401 - Unauthorized	Empty

HTTP method**GET****Description**

list or watch objects of kind Ingress

Table 18.5. HTTP responses

HTTP code	Response body
200 - OK	IngressList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an Ingress

Table 18.6. 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 18.7. Body parameters

Parameter	Type	Description
body	Ingress schema	

Table 18.8. HTTP responses

HTTP code	Response body
200 - OK	Ingress schema
201 - Created	Ingress schema
202 - Accepted	Ingress schema
401 - Unauthorized	Empty

18.2.4. /apis/networking.k8s.io/v1/watch/namespaces/{namespace}/ingresses

HTTP method

GET

Description

watch individual changes to a list of Ingress. deprecated: use the 'watch' parameter with a list operation instead.

Table 18.9. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

18.2.5. /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}

Table 18.10. Global path parameters

Parameter	Type	Description
name	string	name of the Ingress

HTTP method

DELETE

Description

delete an Ingress

Table 18.11. 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

Table 18.12. HTTP responses

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

HTTP method

GET

Description

read the specified Ingress

Table 18.13. HTTP responses

HTTP code	Response body
200 - OK	Ingress schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified Ingress

Table 18.14. 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 18.15. HTTP responses

HTTP code	Response body
200 - OK	Ingress schema
201 - Created	Ingress schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified Ingress

Table 18.16. 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 18.17. Body parameters

Parameter	Type	Description
body	Ingress schema	

Table 18.18. HTTP responses

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

18.2.6. /apis/networking.k8s.io/v1/watch/namespaces/{namespace}/ingresses/{name}

Table 18.19. Global path parameters

Parameter	Type	Description
name	string	name of the Ingress

HTTP method

GET

Description

watch changes to an object of kind Ingress. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 18.20. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

18.2.7. /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}/status

Table 18.21. Global path parameters

Parameter	Type	Description
name	string	name of the Ingress

HTTP method

GET

Description

read status of the specified Ingress

Table 18.22. HTTP responses

HTTP code	Response body
200 - OK	Ingress schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified Ingress

Table 18.23. 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 18.24. HTTP responses

HTTP code	Response body
200 - OK	Ingress schema
201 - Created	Ingress schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified Ingress

Table 18.25. 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 18.26. Body parameters

Parameter	Type	Description
body	Ingress schema	

Table 18.27. HTTP responses

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

CHAPTER 19. INGRESSCLASS [NETWORKING.K8S.IO/V1]

Description

IngressClass represents the class of the Ingress, referenced by the Ingress Spec. The **ingressclass.kubernetes.io/is-default-class** annotation can be used to indicate that an IngressClass should be considered default. When a single IngressClass resource has this annotation set to true, new Ingress resources without a class specified will be assigned this default class.

Type

object

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
spec	object	IngressClassSpec provides information about the class of an Ingress.

19.1.1. .spec

Description

IngressClassSpec provides information about the class of an Ingress.

Type

object

Property	Type	Description
controller	string	controller refers to the name of the controller that should handle this class. This allows for different "flavors" that are controlled by the same controller. For example, you may have different parameters for the same implementing controller. This should be specified as a domain-prefixed path no more than 250 characters in length, e.g. "acme.io/ingress-controller". This field is immutable.
parameters	object	IngressClassParametersReference identifies an API object. This can be used to specify a cluster or namespace-scoped resource.

19.1.2. .spec.parameters

Description

IngressClassParametersReference identifies an API object. This can be used to specify a cluster or namespace-scoped resource.

Type

object

Required

- **kind**
- **name**

Property	Type	Description
apiGroup	string	apiGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
kind	string	kind is the type of resource being referenced.

Property	Type	Description
name	string	name is the name of resource being referenced.
namespace	string	namespace is the namespace of the resource being referenced. This field is required when scope is set to "Namespace" and must be unset when scope is set to "Cluster".
scope	string	scope represents if this refers to a cluster or namespace scoped resource. This may be set to "Cluster" (default) or "Namespace".

19.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/networking.k8s.io/v1/ingressclasses**
 - **DELETE**: delete collection of IngressClass
 - **GET**: list or watch objects of kind IngressClass
 - **POST**: create an IngressClass
- **/apis/networking.k8s.io/v1/watch/ingressclasses**
 - **GET**: watch individual changes to a list of IngressClass. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/networking.k8s.io/v1/ingressclasses/{name}**
 - **DELETE**: delete an IngressClass
 - **GET**: read the specified IngressClass
 - **PATCH**: partially update the specified IngressClass
 - **PUT**: replace the specified IngressClass
- **/apis/networking.k8s.io/v1/watch/ingressclasses/{name}**
 - **GET**: watch changes to an object of kind IngressClass. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

19.2.1. /apis/networking.k8s.io/v1/ingressclasses

HTTP method**DELETE****Description**

delete collection of IngressClass

Table 19.1. 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 19.2. HTTP responses

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

HTTP method**GET****Description**

list or watch objects of kind IngressClass

Table 19.3. HTTP responses

HTTP code	Reponse body
200 - OK	IngressClassList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an IngressClass

Table 19.4. 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 19.5. Body parameters

Parameter	Type	Description
body	IngressClass schema	

Table 19.6. HTTP responses

HTTP code	Response body
200 - OK	IngressClass schema
201 - Created	IngressClass schema
202 - Accepted	IngressClass schema
401 - Unauthorized	Empty

19.2.2. /apis/networking.k8s.io/v1/watch/ingressclasses

HTTP method**GET****Description**

watch individual changes to a list of IngressClass. deprecated: use the 'watch' parameter with a list operation instead.

Table 19.7. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

19.2.3. /apis/networking.k8s.io/v1/ingressclasses/{name}**Table 19.8. Global path parameters**

Parameter	Type	Description
name	string	name of the IngressClass

HTTP method**DELETE****Description**

delete an IngressClass

Table 19.9. 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 19.10. HTTP responses

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

HTTP method**GET****Description**

read the specified IngressClass

Table 19.11. HTTP responses

HTTP code	Response body
200 - OK	IngressClass schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified IngressClass

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. HTTP responses

HTTP code	Response body
200 - OK	IngressClass schema
201 - Created	IngressClass schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified IngressClass

Table 19.14. 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 19.15. Body parameters

Parameter	Type	Description
body	IngressClass schema	

Table 19.16. HTTP responses

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

19.2.4. /apis/networking.k8s.io/v1/watch/ingressclasses/{name}

Table 19.17. Global path parameters

Parameter	Type	Description
name	string	name of the IngressClass

HTTP method

GET

Description

watch changes to an object of kind IngressClass. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 19.18. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

CHAPTER 20. IPADDRESS [NETWORKING.K8S.IO/V1]

Description

IPAddress represents a single IP of a single IP Family. The object is designed to be used by APIs that operate on IP addresses. The object is used by the Service core API for allocation of IP addresses. An IP address can be represented in different formats, to guarantee the uniqueness of the IP, the name of the object is the IP address in canonical format, four decimal digits separated by dots suppressing leading zeros for IPv4 and the representation defined by RFC 5952 for IPv6. Valid: 192.168.1.5 or 2001:db8::1 or 2001:db8:aaaa:bbbb:cccc:dddd:eeee:1 Invalid: 10.01.2.3 or 2001:db8:0:0:0:0:1

Type

object

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	IPAddressSpec describe the attributes in an IP Address.

20.1.1. .spec

Description

IPAddressSpec describe the attributes in an IP Address.

Type

object

Required

- **parentRef**

Property	Type	Description
parentRef	object	ParentReference describes a reference to a parent object.

20.1.2. .spec.parentRef**Description**

ParentReference describes a reference to a parent object.

Type

object

Required

- **resource**
- **name**

Property	Type	Description
group	string	Group is the group of the object being referenced.
name	string	Name is the name of the object being referenced.
namespace	string	Namespace is the namespace of the object being referenced.
resource	string	Resource is the resource of the object being referenced.

20.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/networking.k8s.io/v1/ipaddresses**
 - **DELETE:** delete collection of IPAddress

- **GET**: list or watch objects of kind IPAddress
- **POST**: create an IPAddress
- **/apis/networking.k8s.io/v1/watch/ipaddresses**
 - **GET**: watch individual changes to a list of IPAddress. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/networking.k8s.io/v1/ipaddresses/{name}**
 - **DELETE**: delete an IPAddress
 - **GET**: read the specified IPAddress
 - **PATCH**: partially update the specified IPAddress
 - **PUT**: replace the specified IPAddress
- **/apis/networking.k8s.io/v1/watch/ipaddresses/{name}**
 - **GET**: watch changes to an object of kind IPAddress. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

20.2.1. /apis/networking.k8s.io/v1/ipaddresses

HTTP method

DELETE

Description

delete collection of IPAddress

Table 20.1. 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 20.2. HTTP responses

HTTP code	Response body
200 - OK	Status schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list or watch objects of kind IPAddress

Table 20.3. HTTP responses

HTTP code	Response body
200 - OK	IPAddressList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an IPAddress

Table 20.4. 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 20.5. Body parameters

Parameter	Type	Description
body	IPAddress schema	

Table 20.6. HTTP responses

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

20.2.2. /apis/networking.k8s.io/v1/watch/ipaddresses

HTTP method

GET

Description

watch individual changes to a list of IPAddress. deprecated: use the 'watch' parameter with a list operation instead.

Table 20.7. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

20.2.3. /apis/networking.k8s.io/v1/ipaddresses/{name}

Table 20.8. Global path parameters

Parameter	Type	Description
name	string	name of the IPAddress

HTTP method

DELETE

Description

delete an IPAddress

Table 20.9. 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 20.10. HTTP responses

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

HTTP method**GET****Description**

read the specified IPAddress

Table 20.11. HTTP responses

HTTP code	Reponse body
200 - OK	IPAddress schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified IPAddress

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. HTTP responses

HTTP code	Response body
200 - OK	IPAddress schema
201 - Created	IPAddress schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified IPAddress

Table 20.14. 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 20.15. Body parameters

Parameter	Type	Description
body	IPAddress schema	

Table 20.16. HTTP responses

HTTP code	Response body
200 - OK	IPAddress schema
201 - Created	IPAddress schema
401 - Unauthorized	Empty

20.2.4. /apis/networking.k8s.io/v1/watch/ipaddresses/{name}

Table 20.17. Global path parameters

Parameter	Type	Description
name	string	name of the IPAddress

HTTP method

GET

Description

watch changes to an object of kind IPAddress. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 20.18. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

CHAPTER 21. IPAMCLAIM [K8S.CNI.CNCF.IO/V1ALPHA1]

Description

IPAMClaim is the Schema for the IPAMClaim API

Type

object

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
spec	object	
status	object	IPAMClaimStatus contains the observed status of the IPAMClaim.

21.1.1. .spec

Description

Type**object****Required**

- **interface**
- **network**

Property	Type	Description
interface	string	The pod interface name for which this allocation was created
network	string	The network name for which this persistent allocation was created

21.1.2. .status**Description**

IPAMClaimStatus contains the observed status of the IPAMClaim.

Type**object****Required**

- **ips**

Property	Type	Description
ips	array (string)	The list of IP addresses (v4, v6) that were allocated for the pod interface

21.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/k8s.cni.cncf.io/v1alpha1/ipamclaims**
 - **GET**: list objects of kind IPAMClaim
- **/apis/k8s.cni.cncf.io/v1alpha1/namespaces/{namespace}/ipamclaims**
 - **DELETE**: delete collection of IPAMClaim
 - **GET**: list objects of kind IPAMClaim
 - **POST**: create an IPAMClaim

- **/apis/k8s.cni.cncf.io/v1alpha1/namespaces/{namespace}/ipamclaims/{name}**
 - **DELETE**: delete an IPAMClaim
 - **GET**: read the specified IPAMClaim
 - **PATCH**: partially update the specified IPAMClaim
 - **PUT**: replace the specified IPAMClaim
- **/apis/k8s.cni.cncf.io/v1alpha1/namespaces/{namespace}/ipamclaims/{name}/status**
 - **GET**: read status of the specified IPAMClaim
 - **PATCH**: partially update status of the specified IPAMClaim
 - **PUT**: replace status of the specified IPAMClaim

21.2.1. /apis/k8s.cni.cncf.io/v1alpha1/ipamclaims

HTTP method

GET

Description

list objects of kind IPAMClaim

Table 21.1. HTTP responses

HTTP code	Reponse body
200 - OK	IPAMClaimList schema
401 - Unauthorized	Empty

21.2.2. /apis/k8s.cni.cncf.io/v1alpha1/namespaces/{namespace}/ipamclaims

HTTP method

DELETE

Description

delete collection of IPAMClaim

Table 21.2. HTTP responses

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

HTTP method

GET**Description**

list objects of kind IPAMClaim

Table 21.3. HTTP responses

HTTP code	Response body
200 - OK	IPAMClaimList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an IPAMClaim

Table 21.4. 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 21.5. Body parameters

Parameter	Type	Description
body	IPAMClaim schema	

Table 21.6. HTTP responses

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

21.2.3. /apis/k8s.cni.cncf.io/v1alpha1/namespaces/{namespace}/ipamclaims/{name}

Table 21.7. Global path parameters

Parameter	Type	Description
name	string	name of the IPAMClaim

HTTP method

DELETE

Description

delete an IPAMClaim

Table 21.8. 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 21.9. 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 IPAMClaim

Table 21.10. HTTP responses

HTTP code	Reponse body
200 - OK	IPAMClaim schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified IPAMClaim

Table 21.11. 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 21.12. HTTP responses

HTTP code	Response body
200 - OK	IPAMClaim schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified IPAMClaim

Table 21.13. 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 21.14. Body parameters

Parameter	Type	Description
body	IPAMClaim schema	

Table 21.15. HTTP responses

HTTP code	Response body
200 - OK	IPAMClaim schema
201 - Created	IPAMClaim schema
401 - Unauthorized	Empty

21.2.4. /apis/k8s.cni.cncf.io/v1alpha1/namespaces/{namespace}/ipamclaims/{name}/:

Table 21.16. Global path parameters

Parameter	Type	Description
name	string	name of the IPAMClaim

HTTP method

GET

Description

read status of the specified IPAMClaim

Table 21.17. HTTP responses

HTTP code	Response body
200 - OK	IPAMClaim schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified IPAMClaim

Table 21.18. 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 21.19. HTTP responses

HTTP code	Response body
200 - OK	IPAMClaim schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified IPAMClaim

Table 21.20. 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 21.21. Body parameters

Parameter	Type	Description
body	IPAMClaim schema	

Table 21.22. HTTP responses

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

CHAPTER 22. IPPOOL

[WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]

Description

IPPool is the Schema for the ippools API

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	IPPoolSpec defines the desired state of IPPool

22.1.1. .spec

Description

IPPoolSpec defines the desired state of IPPool

Type

object

Required

- **allocations**
- **range**

Property	Type	Description
allocations	object	Allocations is the set of allocated IPs for the given range. Its `indices` are a direct mapping to the IP with the same index/offset for the pool's range.
allocations{}	object	IPAllocation represents metadata about the pod/container owner of a specific IP
range	string	Range is a RFC 4632/4291-style string that represents an IP address and prefix length in CIDR notation

22.1.2. .spec.allocations

Description

Allocations is the set of allocated IPs for the given range. Its `indices` are a direct mapping to the IP with the same index/offset for the pool's range.

Type

object

22.1.3. .spec.allocations{}

Description

IPAllocation represents metadata about the pod/container owner of a specific IP

Type

object

Required

- **id**
- **podref**

Property	Type	Description
id	string	

Property	Type	Description
ifname	string	
podref	string	

22.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/whereabouts.cni.cncf.io/v1alpha1/ippools**
 - **GET**: list objects of kind IPPool
- **/apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/ippools**
 - **DELETE**: delete collection of IPPool
 - **GET**: list objects of kind IPPool
 - **POST**: create an IPPool
- **/apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/ippools/{name}**
 - **DELETE**: delete an IPPool
 - **GET**: read the specified IPPool
 - **PATCH**: partially update the specified IPPool
 - **PUT**: replace the specified IPPool

22.2.1. /apis/whereabouts.cni.cncf.io/v1alpha1/ippools

HTTP method

GET

Description

list objects of kind IPPool

Table 22.1. HTTP responses

HTTP code	Reponse body
200 - OK	IPPoolList schema
401 - Unauthorized	Empty

22.2.2. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/ippools

HTTP method

DELETE**Description**

delete collection of IPPool

Table 22.2. HTTP responses

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

HTTP method**GET****Description**

list objects of kind IPPool

Table 22.3. HTTP responses

HTTP code	Reponse body
200 - OK	IPPoolList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an IPPool

Table 22.4. 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

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 22.5. Body parameters

Parameter	Type	Description
body	IPPool schema	

Table 22.6. HTTP responses

HTTP code	Response body
200 - OK	IPPool schema
201 - Created	IPPool schema
202 - Accepted	IPPool schema
401 - Unauthorized	Empty

22.2.3. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/ippools/{name}

Table 22.7. Global path parameters

Parameter	Type	Description
name	string	name of the IPPool

HTTP method**DELETE****Description**

delete an IPPool

Table 22.8. 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 22.9. HTTP responses

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

HTTP method**GET****Description**

read the specified IPPool

Table 22.10. HTTP responses

HTTP code	Response body
200 - OK	IPPool schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified IPPool

Table 22.11. 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 22.12. HTTP responses

HTTP code	Response body
200 - OK	IPPool schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified IPPool

Table 22.13. 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 22.14. Body parameters

Parameter	Type	Description
body	IPPool schema	

Table 22.15. HTTP responses

HTTP code	Response body
200 - OK	IPPool schema
201 - Created	IPPool schema
401 - Unauthorized	Empty

CHAPTER 23. MULTINETWORKPOLICY

[K8S.CNI.CNCF.IO/V1BETA1]

Description

MultiNetworkPolicy is a CRD schema to provide NetworkPolicy mechanism for net-attach-def which is specified by the Network Plumbing Working Group. MultiNetworkPolicy is identical to Kubernetes NetworkPolicy, See: <https://kubernetes.io/docs/concepts/services-networking/network-policies/>.

Type

object

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
spec	object	Specification of the desired behavior for this MultiNetworkPolicy.

23.1.1. .spec

Description

Specification of the desired behavior for this MultiNetworkPolicy.

Type

object

Required

- **podSelector**

Property	Type	Description
egress	array	List of egress rules to be applied to the selected pods. Outgoing traffic is allowed if there are no NetworkPolicies selecting the pod (and cluster policy otherwise allows the traffic), OR if the traffic matches at least one egress rule across all of the NetworkPolicy objects whose podSelector matches the pod. If this field is empty then this NetworkPolicy limits all outgoing traffic (and serves solely to ensure that the pods it selects are isolated by default). This field is beta-level in 1.8
egress[]	object	NetworkPolicyEgressRule describes a particular set of traffic that is allowed out of pods matched by a NetworkPolicySpec's podSelector. The traffic must match both ports and to. This type is beta-level in 1.8
ingress	array	List of ingress rules to be applied to the selected pods. Traffic is allowed to a pod if there are no NetworkPolicies selecting the pod (and cluster policy otherwise allows the traffic), OR if the traffic source is the pod's local node, OR if the traffic matches at least one ingress rule across all of the NetworkPolicy objects whose podSelector matches the pod. If this field is empty then this NetworkPolicy does not allow any traffic (and serves solely to ensure that the pods it selects are isolated by default)

Property	Type	Description
ingress[]	object	NetworkPolicyIngressRule describes a particular set of traffic that is allowed to the pods matched by a NetworkPolicySpec's podSelector. The traffic must match both ports and from.
podSelector	object	This is a label selector which selects Pods. This field follows standard label selector semantics; if present but empty, it selects all pods. If NamespaceSelector is also set, then the NetworkPolicyPeer as a whole selects the Pods matching PodSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects the Pods matching PodSelector in the policy's own Namespace.

Property	Type	Description
policyTypes	array (string)	List of rule types that the NetworkPolicy relates to. Valid options are 'Ingress', 'Egress', or 'Ingress,Egress'. If this field is not specified, it will default based on the existence of Ingress or Egress rules; policies that contain an Egress section are assumed to affect Egress, and all policies (whether or not they contain an Ingress section) are assumed to affect Ingress. If you want to write an egress-only policy, you must explicitly specify policyTypes ['Egress']. Likewise, if you want to write a policy that specifies that no egress is allowed, you must specify a policyTypes value that include 'Egress' (since such a policy would not include an Egress section and would otherwise default to just ['Ingress']). This field is beta-level in 1.8

23.1.2. .spec.egress

Description

List of egress rules to be applied to the selected pods. Outgoing traffic is allowed if there are no NetworkPolicies selecting the pod (and cluster policy otherwise allows the traffic), OR if the traffic matches at least one egress rule across all of the NetworkPolicy objects whose podSelector matches the pod. If this field is empty then this NetworkPolicy limits all outgoing traffic (and serves solely to ensure that the pods it selects are isolated by default). This field is beta-level in 1.8

Type

array

23.1.3. .spec.egress[]

Description

NetworkPolicyEgressRule describes a particular set of traffic that is allowed out of pods matched by a NetworkPolicySpec's podSelector. The traffic must match both ports and to. This type is beta-level in 1.8

Type

object

Property	Type	Description

Property	Type	Description
ports	array	List of destination ports for outgoing traffic. Each item in this list is combined using a logical OR. If this field is empty or missing, this rule matches all ports (traffic not restricted by port). If this field is present and contains at least one item, then this rule allows traffic only if the traffic matches at least one port in the list.
ports[]	object	NetworkPolicyPort describes a port to allow traffic on
to	array	List of destinations for outgoing traffic of pods selected for this rule. Items in this list are combined using a logical OR operation. If this field is empty or missing, this rule matches all destinations (traffic not restricted by destination). If this field is present and contains at least one item, this rule allows traffic only if the traffic matches at least one item in the to list.
to[]	object	NetworkPolicyPeer describes a peer to allow traffic from. Only certain combinations of fields are allowed

23.1.4. .spec.egress[].ports

Description

List of destination ports for outgoing traffic. Each item in this list is combined using a logical OR. If this field is empty or missing, this rule matches all ports (traffic not restricted by port). If this field is present and contains at least one item, then this rule allows traffic only if the traffic matches at least one port in the list.

Type

array

23.1.5. .spec.egress[].ports[]

Description

NetworkPolicyPort describes a port to allow traffic on

Type

object

Property	Type	Description
endPort	integer	If set, indicates that the range of ports from port to endPort, inclusive, should be allowed by the policy. This field cannot be defined if the port field is not defined or if the port field is defined as a named (string) port. The endPort must be equal or greater than port.
port	integer-or-string	The port on the given protocol. This can either be a numerical or named port on a pod. If this field is not provided, this matches all port names and numbers.
protocol	string	The protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP.

23.1.6. .spec.egress[].to**Description**

List of destinations for outgoing traffic of pods selected for this rule. Items in this list are combined using a logical OR operation. If this field is empty or missing, this rule matches all destinations (traffic not restricted by destination). If this field is present and contains at least one item, this rule allows traffic only if the traffic matches at least one item in the to list.

Type

array

23.1.7. .spec.egress[].to[]**Description**

NetworkPolicyPeer describes a peer to allow traffic from. Only certain combinations of fields are allowed

Type

object

Property	Type	Description
ipBlock	object	IPBlock defines policy on a particular IPBlock. If this field is set then neither of the other fields can be.

Property	Type	Description
namespaceSelector	object	Selects Namespaces using cluster-scoped labels. This field follows standard label selector semantics; if present but empty, it selects all namespaces. If PodSelector is also set, then the NetworkPolicyPeer as a whole selects the Pods matching PodSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects all Pods in the Namespaces selected by NamespaceSelector.
podSelector	object	This is a label selector which selects Pods. This field follows standard label selector semantics; if present but empty, it selects all pods. If NamespaceSelector is also set, then the NetworkPolicyPeer as a whole selects the Pods matching PodSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects the Pods matching PodSelector in the policy's own Namespace.

23.1.8. .spec.egress[].to[].ipBlock

Description

IPBlock defines policy on a particular IPBlock. If this field is set then neither of the other fields can be.

Type

object

Required

- **cidr**

Property	Type	Description
cidr	string	CIDR is a string representing the IP Block Valid examples are '192.168.1.1/24'

Property	Type	Description
except	array (string)	Except is a slice of CIDRs that should not be included within an IP Block Valid examples are '192.168.1.1/24' Except values will be rejected if they are outside the CIDR range

23.1.9. .spec.egress[].to[].namespaceSelector

Description

Selects Namespaces using cluster-scoped labels. This field follows standard label selector semantics; if present but empty, it selects all namespaces. If PodSelector is also set, then the NetworkPolicyPeer as a whole selects the Pods matching PodSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects all Pods in the Namespaces selected by NamespaceSelector.

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.

23.1.10. .spec.egress[].to[].namespaceSelector.matchExpressions

Description

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

Type

array

23.1.11. .spec.egress[].to[].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.
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.

23.1.12. .spec.egress[].to[].podSelector

Description

This is a label selector which selects Pods. This field follows standard label selector semantics; if present but empty, it selects all pods. If NamespaceSelector is also set, then the NetworkPolicyPeer as a whole selects the Pods matching PodSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects the Pods matching PodSelector in the policy's own Namespace.

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.

23.1.13. .spec.egress[].to[].podSelector.matchExpressions

Description

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

Type

array

23.1.14. .spec.egress[].to[].podSelector.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.

23.1.15. .spec.ingress

Description

List of ingress rules to be applied to the selected pods. Traffic is allowed to a pod if there are no NetworkPolicies selecting the pod (and cluster policy otherwise allows the traffic), OR if the traffic source is the pod's local node, OR if the traffic matches at least one ingress rule across all of the NetworkPolicy objects whose podSelector matches the pod. If this field is empty then this NetworkPolicy does not allow any traffic (and serves solely to ensure that the pods it selects are isolated by default)

Type

array

23.1.16. .spec.ingress[]

Description

NetworkPolicyIngressRule describes a particular set of traffic that is allowed to the pods matched by a NetworkPolicySpec's podSelector. The traffic must match both ports and from.

Type

object

Property	Type	Description
from	array	List of sources which should be able to access the pods selected for this rule. Items in this list are combined using a logical OR operation. If this field is empty or missing, this rule matches all sources (traffic not restricted by source). If this field is present and contains at least one item, this rule allows traffic only if the traffic matches at least one item in the from list.

Property	Type	Description
from[]	object	NetworkPolicyPeer describes a peer to allow traffic from. Only certain combinations of fields are allowed
ports	array	List of ports which should be made accessible on the pods selected for this rule. Each item in this list is combined using a logical OR. If this field is empty or missing, this rule matches all ports (traffic not restricted by port). If this field is present and contains at least one item, then this rule allows traffic only if the traffic matches at least one port in the list.
ports[]	object	NetworkPolicyPort describes a port to allow traffic on

23.1.17. .spec.ingress[].from

Description

List of sources which should be able to access the pods selected for this rule. Items in this list are combined using a logical OR operation. If this field is empty or missing, this rule matches all sources (traffic not restricted by source). If this field is present and contains at least one item, this rule allows traffic only if the traffic matches at least one item in the from list.

Type

array

23.1.18. .spec.ingress[].from[]

Description

NetworkPolicyPeer describes a peer to allow traffic from. Only certain combinations of fields are allowed

Type

object

Property	Type	Description
ipBlock	object	IPBlock defines policy on a particular IPBlock. If this field is set then neither of the other fields can be.

Property	Type	Description
namespaceSelector	object	Selects Namespaces using cluster-scoped labels. This field follows standard label selector semantics; if present but empty, it selects all namespaces. If PodSelector is also set, then the NetworkPolicyPeer as a whole selects the Pods matching PodSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects all Pods in the Namespaces selected by NamespaceSelector.
podSelector	object	This is a label selector which selects Pods. This field follows standard label selector semantics; if present but empty, it selects all pods. If NamespaceSelector is also set, then the NetworkPolicyPeer as a whole selects the Pods matching PodSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects the Pods matching PodSelector in the policy's own Namespace.

23.1.19. .spec.ingress[].from[].ipBlock

Description

IPBlock defines policy on a particular IPBlock. If this field is set then neither of the other fields can be.

Type

object

Required

- **cidr**

Property	Type	Description
cidr	string	CIDR is a string representing the IP Block Valid examples are '192.168.1.1/24'

Property	Type	Description
except	array (string)	Except is a slice of CIDRs that should not be included within an IP Block Valid examples are '192.168.1.1/24' Except values will be rejected if they are outside the CIDR range

23.1.20. .spec.ingress[].from[].namespaceSelector

Description

Selects Namespaces using cluster-scoped labels. This field follows standard label selector semantics; if present but empty, it selects all namespaces. If PodSelector is also set, then the NetworkPolicyPeer as a whole selects the Pods matching PodSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects all Pods in the Namespaces selected by NamespaceSelector.

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.

23.1.21. .spec.ingress[].from[].namespaceSelector.matchExpressions

Description

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

Type

array

23.1.22. .spec.ingress[].from[].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.
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.

23.1.23. .spec.ingress[].from[].podSelector

Description

This is a label selector which selects Pods. This field follows standard label selector semantics; if present but empty, it selects all pods. If NamespaceSelector is also set, then the NetworkPolicyPeer as a whole selects the Pods matching PodSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects the Pods matching PodSelector in the policy's own Namespace.

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.

23.1.24. .spec.ingress[].from[].podSelector.matchExpressions

Description

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

Type

array

23.1.25. .spec.ingress[].from[].podSelector.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.

23.1.26. .spec.ingress[].ports

Description

List of ports which should be made accessible on the pods selected for this rule. Each item in this list is combined using a logical OR. If this field is empty or missing, this rule matches all ports (traffic not restricted by port). If this field is present and contains at least one item, then this rule allows traffic only if the traffic matches at least one port in the list.

Type

array

23.1.27. .spec.ingress[].ports[]

Description

NetworkPolicyPort describes a port to allow traffic on

Type

object

Property	Type	Description
endPort	integer	If set, indicates that the range of ports from port to endPort, inclusive, should be allowed by the policy. This field cannot be defined if the port field is not defined or if the port field is defined as a named (string) port. The endPort must be equal or greater than port.

Property	Type	Description
port	integer-or-string	The port on the given protocol. This can either be a numerical or named port on a pod. If this field is not provided, this matches all port names and numbers.
protocol	string	The protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP.

23.1.28. .spec.podSelector

Description

This is a label selector which selects Pods. This field follows standard label selector semantics; if present but empty, it selects all pods. If NamespaceSelector is also set, then the NetworkPolicyPeer as a whole selects the Pods matching PodSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects the Pods matching PodSelector in the policy's own Namespace.

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)	

23.1.29. .spec.podSelector.matchExpressions

Description

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

Type

array

23.1.30. .spec.podSelector.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.

23.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/k8s.cni.cncf.io/v1beta1/multi-networkpolicies**
 - **GET**: list objects of kind MultiNetworkPolicy
- **/apis/k8s.cni.cncf.io/v1beta1/namespaces/{namespace}/multi-networkpolicies**
 - **DELETE**: delete collection of MultiNetworkPolicy
 - **GET**: list objects of kind MultiNetworkPolicy
 - **POST**: create a MultiNetworkPolicy
- **/apis/k8s.cni.cncf.io/v1beta1/namespaces/{namespace}/multi-networkpolicies/{name}**
 - **DELETE**: delete a MultiNetworkPolicy
 - **GET**: read the specified MultiNetworkPolicy
 - **PATCH**: partially update the specified MultiNetworkPolicy

- **PUT**: replace the specified MultiNetworkPolicy

23.2.1. /apis/k8s.cni.cncf.io/v1beta1/multi-networkpolicies

HTTP method

GET

Description

list objects of kind MultiNetworkPolicy

Table 23.1. HTTP responses

HTTP code	Reponse body
200 - OK	MultiNetworkPolicyList schema
401 - Unauthorized	Empty

23.2.2. /apis/k8s.cni.cncf.io/v1beta1/namespaces/{namespace}/multi-networkpolicies

HTTP method

DELETE

Description

delete collection of MultiNetworkPolicy

Table 23.2. HTTP responses

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

HTTP method

GET

Description

list objects of kind MultiNetworkPolicy

Table 23.3. HTTP responses

HTTP code	Reponse body
200 - OK	MultiNetworkPolicyList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create a MultiNetworkPolicy

Table 23.4. 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 23.5. Body parameters

Parameter	Type	Description
body	MultiNetworkPolicy schema	

Table 23.6. HTTP responses

HTTP code	Reponse body
200 - OK	MultiNetworkPolicy schema
201 - Created	MultiNetworkPolicy schema

HTTP code	Reponse body
202 - Accepted	MultiNetworkPolicy schema
401 - Unauthorized	Empty

23.2.3. /apis/k8s.cni.cncf.io/v1beta1/namespaces/{namespace}/multi-networkpolicies/{name}

Table 23.7. Global path parameters

Parameter	Type	Description
name	string	name of the MultiNetworkPolicy

HTTP method

DELETE

Description

delete a MultiNetworkPolicy

Table 23.8. 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 23.9. HTTP responses

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

HTTP method

GET

Description

read the specified MultiNetworkPolicy

Table 23.10. HTTP responses

HTTP code	Response body
200 - OK	MultiNetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified MultiNetworkPolicy

Table 23.11. 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 23.12. HTTP responses

HTTP code	Response body
200 - OK	MultiNetworkPolicy schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified MultiNetworkPolicy

Table 23.13. 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 23.14. Body parameters

Parameter	Type	Description
body	MultiNetworkPolicy schema	

Table 23.15. HTTP responses

HTTP code	Reponse body
200 - OK	MultiNetworkPolicy schema
201 - Created	MultiNetworkPolicy schema

HTTP code	Reponse body
401 - Unauthorized	Empty

CHAPTER 24. NETWORKATTACHMENTDEFINITION [K8S.CNI.CNCF.IO/V1]

Description

NetworkAttachmentDefinition is a CRD schema specified by the Network Plumbing Working Group to express the intent for attaching pods to one or more logical or physical networks. More information available at: <https://github.com/k8snetworkplumbingwg/multi-net-spec>

Type

object

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	NetworkAttachmentDefinition spec defines the desired state of a network attachment

24.1.1. .spec

Description

NetworkAttachmentDefinition spec defines the desired state of a network attachment

Type

object

Property	Type	Description
config	string	NetworkAttachmentDefinition config is a JSON-formatted CNI configuration

24.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/k8s.cni.cncf.io/v1/network-attachment-definitions**
 - **GET**: list objects of kind NetworkAttachmentDefinition
- **/apis/k8s.cni.cncf.io/v1/namespaces/{namespace}/network-attachment-definitions**
 - **DELETE**: delete collection of NetworkAttachmentDefinition
 - **GET**: list objects of kind NetworkAttachmentDefinition
 - **POST**: create a NetworkAttachmentDefinition
- **/apis/k8s.cni.cncf.io/v1/namespaces/{namespace}/network-attachment-definitions/{name}**
 - **DELETE**: delete a NetworkAttachmentDefinition
 - **GET**: read the specified NetworkAttachmentDefinition
 - **PATCH**: partially update the specified NetworkAttachmentDefinition
 - **PUT**: replace the specified NetworkAttachmentDefinition

24.2.1. /apis/k8s.cni.cncf.io/v1/network-attachment-definitions

HTTP method

GET

Description

list objects of kind NetworkAttachmentDefinition

Table 24.1. HTTP responses

HTTP code	Response body
200 - OK	NetworkAttachmentDefinitionList schema

HTTP code	Reponse body
401 - Unauthorized	Empty

24.2.2. /apis/k8s.cni.cncf.io/v1/namespaces/{namespace}/network-attachment-definitions

HTTP method

DELETE

Description

delete collection of NetworkAttachmentDefinition

Table 24.2. HTTP responses

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

HTTP method

GET

Description

list objects of kind NetworkAttachmentDefinition

Table 24.3. HTTP responses

HTTP code	Reponse body
200 - OK	NetworkAttachmentDefinitionList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a NetworkAttachmentDefinition

Table 24.4. 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 24.5. Body parameters

Parameter	Type	Description
body	NetworkAttachmentDefinition schema	

Table 24.6. HTTP responses

HTTP code	Response body
200 - OK	NetworkAttachmentDefinition schema
201 - Created	NetworkAttachmentDefinition schema
202 - Accepted	NetworkAttachmentDefinition schema
401 - Unauthorized	Empty

24.2.3. /apis/k8s.cni.cncf.io/v1/namespaces/{namespace}/network-attachment-definitions/{name}

Table 24.7. Global path parameters

Parameter	Type	Description
name	string	name of the NetworkAttachmentDefinition

HTTP method

DELETE

Description

delete a NetworkAttachmentDefinition

Table 24.8. 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 24.9. HTTP responses

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

HTTP method

GET

Description

read the specified NetworkAttachmentDefinition

Table 24.10. HTTP responses

HTTP code	Response body
200 - OK	NetworkAttachmentDefinition schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified NetworkAttachmentDefinition

Table 24.11. 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 24.12. HTTP responses

HTTP code	Response body
200 - OK	NetworkAttachmentDefinition schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified NetworkAttachmentDefinition

Table 24.13. 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 24.14. Body parameters

Parameter	Type	Description
body	NetworkAttachmentDefinition schema	

Table 24.15. HTTP responses

HTTP code	Response body
200 - OK	NetworkAttachmentDefinition schema
201 - Created	NetworkAttachmentDefinition schema
401 - Unauthorized	Empty

CHAPTER 25. NETWORKPOLICY [NETWORKING.K8S.IO/V1]

Description

NetworkPolicy describes what network traffic is allowed for a set of Pods

Type

object

25.1. 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	NetworkPolicySpec provides the specification of a NetworkPolicy

25.1.1. .spec

Description

NetworkPolicySpec provides the specification of a NetworkPolicy

Type

object

Required

- **podSelector**

Property	Type	Description
egress	array	egress is a list of egress rules to be applied to the selected pods. Outgoing traffic is allowed if there are no NetworkPolicies selecting the pod (and cluster policy otherwise allows the traffic), OR if the traffic matches at least one egress rule across all of the NetworkPolicy objects whose podSelector matches the pod. If this field is empty then this NetworkPolicy limits all outgoing traffic (and serves solely to ensure that the pods it selects are isolated by default). This field is beta-level in 1.8
egress[]	object	NetworkPolicyEgressRule describes a particular set of traffic that is allowed out of pods matched by a NetworkPolicySpec's podSelector. The traffic must match both ports and to. This type is beta-level in 1.8
ingress	array	ingress is a list of ingress rules to be applied to the selected pods. Traffic is allowed to a pod if there are no NetworkPolicies selecting the pod (and cluster policy otherwise allows the traffic), OR if the traffic source is the pod's local node, OR if the traffic matches at least one ingress rule across all of the NetworkPolicy objects whose podSelector matches the pod. If this field is empty then this NetworkPolicy does not allow any traffic (and serves solely to ensure that the pods it selects are isolated by default)

Property	Type	Description
ingress[]	object	NetworkPolicyIngressRule describes a particular set of traffic that is allowed to the pods matched by a NetworkPolicySpec's podSelector. The traffic must match both ports and from.
podSelector	LabelSelector	podSelector selects the pods to which this NetworkPolicy object applies. The array of ingress rules is applied to any pods selected by this field. Multiple network policies can select the same set of pods. In this case, the ingress rules for each are combined additively. This field is NOT optional and follows standard label selector semantics. An empty podSelector matches all pods in this namespace.
policyTypes	array (string)	policyTypes is a list of rule types that the NetworkPolicy relates to. Valid options are ["Ingress"], ["Egress"], or ["Ingress", "Egress"]. If this field is not specified, it will default based on the existence of ingress or egress rules; policies that contain an egress section are assumed to affect egress, and all policies (whether or not they contain an ingress section) are assumed to affect ingress. If you want to write an egress-only policy, you must explicitly specify policyTypes ["Egress"]. Likewise, if you want to write a policy that specifies that no egress is allowed, you must specify a policyTypes value that include "Egress" (since such a policy would not include an egress section and would otherwise default to just ["Ingress"]). This field is beta-level in 1.8

25.1.2. .spec.egress

Description

egress is a list of egress rules to be applied to the selected pods. Outgoing traffic is allowed if there are no NetworkPolicies selecting the pod (and cluster policy otherwise allows the traffic), OR if the traffic matches at least one egress rule across all of the NetworkPolicy objects whose podSelector matches the pod. If this field is empty then this NetworkPolicy limits all outgoing traffic (and serves solely to ensure that the pods it selects are isolated by default). This field is beta-level in 1.8

Type

array

25.1.3. .spec.egress[]

Description

NetworkPolicyEgressRule describes a particular set of traffic that is allowed out of pods matched by a NetworkPolicySpec's podSelector. The traffic must match both ports and to. This type is beta-level in 1.8

Type

object

Property	Type	Description
ports	array	ports is a list of destination ports for outgoing traffic. Each item in this list is combined using a logical OR. If this field is empty or missing, this rule matches all ports (traffic not restricted by port). If this field is present and contains at least one item, then this rule allows traffic only if the traffic matches at least one port in the list.
ports[]	object	NetworkPolicyPort describes a port to allow traffic on
to	array	to is a list of destinations for outgoing traffic of pods selected for this rule. Items in this list are combined using a logical OR operation. If this field is empty or missing, this rule matches all destinations (traffic not restricted by destination). If this field is present and contains at least one item, this rule allows traffic only if the traffic matches at least one item in the to list.
to[]	object	NetworkPolicyPeer describes a peer to allow traffic to/from. Only certain combinations of fields are allowed

25.1.4. .spec.egress[].ports

Description

ports is a list of destination ports for outgoing traffic. Each item in this list is combined using a logical OR. If this field is empty or missing, this rule matches all ports (traffic not restricted by port). If this field is present and contains at least one item, then this rule allows traffic only if the traffic matches at least one port in the list.

Type

array

25.1.5. .spec.egress[].ports[]

Description

NetworkPolicyPort describes a port to allow traffic on

Type

object

Property	Type	Description
endPort	integer	endPort indicates that the range of ports from port to endPort if set, inclusive, should be allowed by the policy. This field cannot be defined if the port field is not defined or if the port field is defined as a named (string) port. The endPort must be equal or greater than port.
port	IntOrString	port represents the port on the given protocol. This can either be a numerical or named port on a pod. If this field is not provided, this matches all port names and numbers. If present, only traffic on the specified protocol AND port will be matched.
protocol	string	protocol represents the protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP. Possible enum values: - "SCTP" is the SCTP protocol. - "TCP" is the TCP protocol. - "UDP" is the UDP protocol.

25.1.6. .spec.egress[].to

Description

to is a list of destinations for outgoing traffic of pods selected for this rule. Items in this list are combined using a logical OR operation. If this field is empty or missing, this rule matches all destinations (traffic not restricted by destination). If this field is present and contains at least one item, this rule allows traffic only if the traffic matches at least one item in the to list.

Type

array

25.1.7. .spec.egress[].to[]**Description**

NetworkPolicyPeer describes a peer to allow traffic to/from. Only certain combinations of fields are allowed

Type

object

Property	Type	Description
ipBlock	object	IPBlock describes a particular CIDR (Ex. "192.168.1.0/24", "2001:db8::/64") that is allowed to the pods matched by a NetworkPolicySpec's podSelector. The except entry describes CIDRs that should not be included within this rule.
namespaceSelector	LabelSelector	namespaceSelector selects namespaces using cluster-scoped labels. This field follows standard label selector semantics; if present but empty, it selects all namespaces. If podSelector is also set, then the NetworkPolicyPeer as a whole selects the pods matching podSelector in the namespaces selected by namespaceSelector. Otherwise it selects all pods in the namespaces selected by namespaceSelector.

Property	Type	Description
podSelector	LabelSelector	podSelector is a label selector which selects pods. This field follows standard label selector semantics; if present but empty, it selects all pods. If namespaceSelector is also set, then the NetworkPolicyPeer as a whole selects the pods matching podSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects the pods matching podSelector in the policy's own namespace.

25.1.8. .spec.egress[].to[].ipBlock

Description

IPBlock describes a particular CIDR (Ex. "192.168.1.0/24","2001:db8::/64") that is allowed to the pods matched by a NetworkPolicySpec's podSelector. The except entry describes CIDRs that should not be included within this rule.

Type

object

Required

- **cidr**

Property	Type	Description
cidr	string	cidr is a string representing the IPBlock Valid examples are "192.168.1.0/24" or "2001:db8::/64"
except	array (string)	except is a slice of CIDRs that should not be included within an IPBlock Valid examples are "192.168.1.0/24" or "2001:db8::/64" Except values will be rejected if they are outside the cidr range

25.1.9. .spec.ingress

Description

ingress is a list of ingress rules to be applied to the selected pods. Traffic is allowed to a pod if there are no NetworkPolicies selecting the pod (and cluster policy otherwise allows the traffic), OR if the

traffic source is the pod's local node, OR if the traffic matches at least one ingress rule across all of the NetworkPolicy objects whose podSelector matches the pod. If this field is empty then this NetworkPolicy does not allow any traffic (and serves solely to ensure that the pods it selects are isolated by default)

Type

array

25.1.10. .spec.ingress[]

Description

NetworkPolicyIngressRule describes a particular set of traffic that is allowed to the pods matched by a NetworkPolicySpec's podSelector. The traffic must match both ports and from.

Type

object

Property	Type	Description
from	array	from is a list of sources which should be able to access the pods selected for this rule. Items in this list are combined using a logical OR operation. If this field is empty or missing, this rule matches all sources (traffic not restricted by source). If this field is present and contains at least one item, this rule allows traffic only if the traffic matches at least one item in the from list.
from[]	object	NetworkPolicyPeer describes a peer to allow traffic to/from. Only certain combinations of fields are allowed
ports	array	ports is a list of ports which should be made accessible on the pods selected for this rule. Each item in this list is combined using a logical OR. If this field is empty or missing, this rule matches all ports (traffic not restricted by port). If this field is present and contains at least one item, then this rule allows traffic only if the traffic matches at least one port in the list.
ports[]	object	NetworkPolicyPort describes a port to allow traffic on

25.1.11. .spec.ingress[].from

Description

from is a list of sources which should be able to access the pods selected for this rule. Items in this list are combined using a logical OR operation. If this field is empty or missing, this rule matches all sources (traffic not restricted by source). If this field is present and contains at least one item, this rule allows traffic only if the traffic matches at least one item in the from list.

Type

array

25.1.12. .spec.ingress[].from[]

Description

NetworkPolicyPeer describes a peer to allow traffic to/from. Only certain combinations of fields are allowed

Type

object

Property	Type	Description
ipBlock	object	IPBlock describes a particular CIDR (Ex. "192.168.1.0/24", "2001:db8::/64") that is allowed to the pods matched by a NetworkPolicySpec's podSelector. The except entry describes CIDRs that should not be included within this rule.
namespaceSelector	LabelSelector	namespaceSelector selects namespaces using cluster-scoped labels. This field follows standard label selector semantics; if present but empty, it selects all namespaces. If podSelector is also set, then the NetworkPolicyPeer as a whole selects the pods matching podSelector in the namespaces selected by namespaceSelector. Otherwise it selects all pods in the namespaces selected by namespaceSelector.

Property	Type	Description
podSelector	LabelSelector	podSelector is a label selector which selects pods. This field follows standard label selector semantics; if present but empty, it selects all pods. If namespaceSelector is also set, then the NetworkPolicyPeer as a whole selects the pods matching podSelector in the Namespaces selected by NamespaceSelector. Otherwise it selects the pods matching podSelector in the policy's own namespace.

25.1.13. .spec.ingress[].from[].ipBlock

Description

IPBlock describes a particular CIDR (Ex. "192.168.1.0/24","2001:db8::/64") that is allowed to the pods matched by a NetworkPolicySpec's podSelector. The except entry describes CIDRs that should not be included within this rule.

Type

object

Required

- **cidr**

Property	Type	Description
cidr	string	cidr is a string representing the IPBlock Valid examples are "192.168.1.0/24" or "2001:db8::/64"
except	array (string)	except is a slice of CIDRs that should not be included within an IPBlock Valid examples are "192.168.1.0/24" or "2001:db8::/64" Except values will be rejected if they are outside the cidr range

25.1.14. .spec.ingress[].ports

Description

ports is a list of ports which should be made accessible on the pods selected for this rule. Each item in

this list is combined using a logical OR. If this field is empty or missing, this rule matches all ports (traffic not restricted by port). If this field is present and contains at least one item, then this rule allows traffic only if the traffic matches at least one port in the list.

Type

array

25.1.15. .spec.ingress[].ports[]

Description

NetworkPolicyPort describes a port to allow traffic on

Type

object

Property	Type	Description
endPort	integer	endPort indicates that the range of ports from port to endPort if set, inclusive, should be allowed by the policy. This field cannot be defined if the port field is not defined or if the port field is defined as a named (string) port. The endPort must be equal or greater than port.
port	IntOrString	port represents the port on the given protocol. This can either be a numerical or named port on a pod. If this field is not provided, this matches all port names and numbers. If present, only traffic on the specified protocol AND port will be matched.
protocol	string	protocol represents the protocol (TCP, UDP, or SCTP) which traffic must match. If not specified, this field defaults to TCP. Possible enum values: - "SCTP" is the SCTP protocol. - "TCP" is the TCP protocol. - "UDP" is the UDP protocol.

25.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/networking.k8s.io/v1/networkpolicies**
 - o **GET**: list or watch objects of kind NetworkPolicy

- **GET**: list or watch objects of kind NetworkPolicy
- **/apis/networking.k8s.io/v1/watch/networkpolicies**
 - **GET**: watch individual changes to a list of NetworkPolicy. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies**
 - **DELETE**: delete collection of NetworkPolicy
 - **GET**: list or watch objects of kind NetworkPolicy
 - **POST**: create a NetworkPolicy
- **/apis/networking.k8s.io/v1/watch/namespaces/{namespace}/networkpolicies**
 - **GET**: watch individual changes to a list of NetworkPolicy. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies/{name}**
 - **DELETE**: delete a NetworkPolicy
 - **GET**: read the specified NetworkPolicy
 - **PATCH**: partially update the specified NetworkPolicy
 - **PUT**: replace the specified NetworkPolicy
- **/apis/networking.k8s.io/v1/watch/namespaces/{namespace}/networkpolicies/{name}**
 - **GET**: watch changes to an object of kind NetworkPolicy. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

25.2.1. /apis/networking.k8s.io/v1/networkpolicies

HTTP method

GET

Description

list or watch objects of kind NetworkPolicy

Table 25.1. HTTP responses

HTTP code	Response body
200 - OK	NetworkPolicyList schema
401 - Unauthorized	Empty

25.2.2. /apis/networking.k8s.io/v1/watch/networkpolicies

HTTP method

GET**Description**

watch individual changes to a list of NetworkPolicy. deprecated: use the 'watch' parameter with a list operation instead.

Table 25.2. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

25.2.3. /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies**HTTP method****DELETE****Description**

delete collection of NetworkPolicy

Table 25.3. 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

Table 25.4. HTTP responses

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

HTTP method**GET****Description**

list or watch objects of kind NetworkPolicy

Table 25.5. HTTP responses

HTTP code	Reponse body
200 - OK	NetworkPolicyList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create a NetworkPolicy

Table 25.6. 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 25.7. Body parameters

Parameter	Type	Description
body	NetworkPolicy schema	

Table 25.8. HTTP responses

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

25.2.4. /apis/networking.k8s.io/v1/watch/namespaces/{namespace}/networkpolicies

HTTP method

GET

Description

watch individual changes to a list of NetworkPolicy. deprecated: use the 'watch' parameter with a list operation instead.

Table 25.9. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

25.2.5. /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies/{name}

Table 25.10. Global path parameters

Parameter	Type	Description
name	string	name of the NetworkPolicy

HTTP method

DELETE

Description

delete a NetworkPolicy

Table 25.11. 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

Table 25.12. HTTP responses

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

HTTP method**GET****Description**

read the specified NetworkPolicy

Table 25.13. HTTP responses

HTTP code	Reponse body
200 - OK	NetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified NetworkPolicy

Table 25.14. 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 25.15. HTTP responses

HTTP code	Response body
200 - OK	NetworkPolicy schema
201 - Created	NetworkPolicy schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified NetworkPolicy

Table 25.16. 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 25.17. Body parameters

Parameter	Type	Description
body	NetworkPolicy schema	

Table 25.18. HTTP responses

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

25.2.6. /apis/networking.k8s.io/v1/watch/namespaces/{namespace}/networkpolicies,

Table 25.19. Global path parameters

Parameter	Type	Description
name	string	name of the NetworkPolicy

HTTP method**GET****Description**

watch changes to an object of kind NetworkPolicy. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 25.20. HTTP responses

HTTP code	Response body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

CHAPTER 26. NODESLICEPOOL [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]

Description

NodeSlicePool is the Schema for the nodesliceipools API

Type

object

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	NodeSlicePoolSpec defines the desired state of NodeSlicePool
status	object	NodeSlicePoolStatus defines the desired state of NodeSlicePool

26.1.1. .spec

Description

NodeSlicePoolSpec defines the desired state of NodeSlicePool

Type

object

Required

- **range**
- **sliceSize**

Property	Type	Description
range	string	Range is a RFC 4632/4291-style string that represents an IP address and prefix length in CIDR notation this refers to the entire range where the node is allocated a subset
sliceSize	string	SliceSize is the size of subnets or slices of the range that each node will be assigned

26.1.2. .status

Description

NodeSlicePoolStatus defines the desired state of NodeSlicePool

Type

object

Required

- **allocations**

Property	Type	Description
allocations	array	Allocations holds the allocations of nodes to slices
allocations[]	object	

26.1.3. .status.allocations

Description

Allocations holds the allocations of nodes to slices

Type

array

26.1.4. .status.allocations[]

Description

Type

object

Required

- **nodeName**
- **sliceRange**

Property	Type	Description
nodeName	string	nodeName is the name of the node assigned to this slice, empty node name is an available slice for assignment
sliceRange	string	SliceRange is the subnet of this slice

26.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/whereabouts.cni.cncf.io/v1alpha1/nodeslice pools**
 - **GET**: list objects of kind NodeSlicePool
- **/apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/nodeslice pools**
 - **DELETE**: delete collection of NodeSlicePool
 - **GET**: list objects of kind NodeSlicePool
 - **POST**: create a NodeSlicePool
- **/apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/nodeslice pools/{name}**
 - **DELETE**: delete a NodeSlicePool
 - **GET**: read the specified NodeSlicePool
 - **PATCH**: partially update the specified NodeSlicePool
 - **PUT**: replace the specified NodeSlicePool

26.2.1. /apis/whereabouts.cni.cncf.io/v1alpha1/nodeslice pools

HTTP method

GET

Description

list objects of kind NodeSlicePool

Table 26.1. HTTP responses

HTTP code	Reponse body
200 - OK	NodeSlicePoolList schema
401 - Unauthorized	Empty

26.2.2. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/nodeslice**HTTP method****DELETE****Description**

delete collection of NodeSlicePool

Table 26.2. HTTP responses

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

HTTP method**GET****Description**

list objects of kind NodeSlicePool

Table 26.3. HTTP responses

HTTP code	Reponse body
200 - OK	NodeSlicePoolList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create a NodeSlicePool

Table 26.4. 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 26.5. Body parameters

Parameter	Type	Description
body	NodeSlicePool schema	

Table 26.6. HTTP responses

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

26.2.3. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/nodeslice

Table 26.7. Global path parameters

Parameter	Type	Description
name	string	name of the NodeSlicePool

HTTP method

DELETE

Description

delete a NodeSlicePool

Table 26.8. 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 26.9. HTTP responses

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

HTTP method

GET

Description

read the specified NodeSlicePool

Table 26.10. HTTP responses

HTTP code	Reponse body
200 - OK	NodeSlicePool schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified NodeSlicePool

Table 26.11. 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 26.12. HTTP responses

HTTP code	Response body
200 - OK	NodeSlicePool schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified NodeSlicePool

Table 26.13. 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 26.14. Body parameters

Parameter	Type	Description
body	NodeSlicePool schema	

Table 26.15. HTTP responses

HTTP code	Response body
200 - OK	NodeSlicePool schema
201 - Created	NodeSlicePool schema
401 - Unauthorized	Empty

CHAPTER 27. OVERLAPPINGRANGEIPRESERVATION [WHEREABOUTS.CNI.CNCF.IO/V1ALPHA1]

Description

OverlappingRangeIPReservation is the Schema for the OverlappingRangeIPReservations API

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	OverlappingRangeIPReservationSpec defines the desired state of OverlappingRangeIPReservation

27.1.1. .spec

Description

OverlappingRangeIPReservationSpec defines the desired state of OverlappingRangeIPReservation

Type

object

Required

- **podref**

Property	Type	Description
containerid	string	
ifname	string	
podref	string	

27.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/whereabouts.cni.cncf.io/v1alpha1/overlappingrangeipreservations**
 - **GET**: list objects of kind OverlappingRangeIPReservation
- **/apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/overlappingrangeipreservations**
 - **DELETE**: delete collection of OverlappingRangeIPReservation
 - **GET**: list objects of kind OverlappingRangeIPReservation
 - **POST**: create an OverlappingRangeIPReservation
- **/apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/overlappingrangeipreservations/{name}**
 - **DELETE**: delete an OverlappingRangeIPReservation
 - **GET**: read the specified OverlappingRangeIPReservation
 - **PATCH**: partially update the specified OverlappingRangeIPReservation
 - **PUT**: replace the specified OverlappingRangeIPReservation

27.2.1. /apis/whereabouts.cni.cncf.io/v1alpha1/overlappingrangeipreservations

HTTP method

GET

Description

list objects of kind OverlappingRangeIPReservation

Table 27.1. HTTP responses

HTTP code	Reponse body
200 - OK	OverlappingRangeIPReservationList schema
401 - Unauthorized	Empty

27.2.2. /apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/overlapping**HTTP method****DELETE****Description**

delete collection of OverlappingRangeIPReservation

Table 27.2. HTTP responses

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

HTTP method**GET****Description**

list objects of kind OverlappingRangeIPReservation

Table 27.3. HTTP responses

HTTP code	Reponse body
200 - OK	OverlappingRangeIPReservationList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an OverlappingRangeIPReservation

Table 27.4. 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 27.5. Body parameters

Parameter	Type	Description
body	OverlappingRangeIPReservation schema	

Table 27.6. HTTP responses

HTTP code	Response body
200 - OK	OverlappingRangeIPReservation schema
201 - Created	OverlappingRangeIPReservation schema
202 - Accepted	OverlappingRangeIPReservation schema
401 - Unauthorized	Empty

27.2.3. `/apis/whereabouts.cni.cncf.io/v1alpha1/namespaces/{namespace}/overlapping`

Table 27.7. Global path parameters

Parameter	Type	Description
name	string	name of the OverlappingRangeIPReservation

HTTP method

DELETE

Description

delete an OverlappingRangeIPReservation

Table 27.8. 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 27.9. HTTP responses

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

HTTP method

GET

Description

read the specified OverlappingRangeIPReservation

Table 27.10. HTTP responses

HTTP code	Reponse body
200 - OK	OverlappingRangeIPReservation schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified `OverlappingRangeIPReservation`

Table 27.11. Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> 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	<code>fieldValidation</code> 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 <code>BadRequest</code> 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 27.12. HTTP responses

HTTP code	Response body
200 - OK	OverlappingRangeIPReservation schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified `OverlappingRangeIPReservation`

Table 27.13. 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 27.14. Body parameters

Parameter	Type	Description
body	OverlappingRangeIPReservation schema	

Table 27.15. HTTP responses

HTTP code	Response body
200 - OK	OverlappingRangeIPReservation schema
201 - Created	OverlappingRangeIPReservation schema
401 - Unauthorized	Empty

CHAPTER 28. PODNETWORKCONNECTIVITYCHECK [CONTROLPLANE.OPERATOR.OPENSIFT.IO/V1ALPHA1]

Description

PodNetworkConnectivityCheck

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**

28.1. 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 defines the source and target of the connectivity check

Property	Type	Description
status	object	status contains the observed status of the connectivity check

28.1.1. .spec

Description

spec defines the source and target of the connectivity check

Type

object

Required

- **sourcePod**
- **targetEndpoint**

Property	Type	Description
sourcePod	string	sourcePod names the pod from which the condition will be checked
targetEndpoint	string	EndpointAddress to check. A TCP address of the form host:port. Note that if host is a DNS name, then the check would fail if the DNS name cannot be resolved. Specify an IP address for host to bypass DNS name lookup.
tlsClientCert	object	TLSCert, if specified, references a kubernetes.io/tls type secret with 'tls.crt' and 'tls.key' entries containing an optional TLS client certificate and key to be used when checking endpoints that require a client certificate in order to gracefully perform the scan without causing excessive logging in the endpoint process. The secret must exist in the same namespace as this resource.

28.1.2. .spec.tlsClientCert

Description

TLSCert, if specified, references a kubernetes.io/tls type secret with 'tls.crt' and 'tls.key' entries

containing an optional TLS client certificate and key to be used when checking endpoints that require a client certificate in order to gracefully perform the scan without causing excessive logging in the endpoint process. The secret must exist in the same namespace as this resource.

Type

object

Required

- **name**

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

28.1.3. .status

Description

status contains the observed status of the connectivity check

Type

object

Property	Type	Description
conditions	array	conditions summarize the status of the check
conditions[]	object	PodNetworkConnectivityCheckCondition represents the overall status of the pod network connectivity.
failures	array	failures contains logs of unsuccessful check actions
failures[]	object	LogEntry records events
outages	array	outages contains logs of time periods of outages
outages[]	object	OutageEntry records time period of an outage
successes	array	successes contains logs successful check actions
successes[]	object	LogEntry records events

28.1.4. .status.conditions

Description

conditions summarize the status of the check

Type

array

28.1.5. .status.conditions[]

Description

PodNetworkConnectivityCheckCondition represents the overall status of the pod network connectivity.

Type

object

Required

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

Property	Type	Description
lastTransitionTime	string	Last time the condition transitioned from one status to another.
message	string	message indicating details about last transition in a human readable format.
reason	string	reason for the condition's last status transition in a machine readable format.
status	string	status of the condition
type	string	type of the condition

28.1.6. .status.failures

Description

failures contains logs of unsuccessful check actions

Type

array

28.1.7. .status.failures[]

Description

LogEntry records events

Type

object

Required

- **success**

Property	Type	Description
latency	``	latency records how long the action mentioned in the entry took.
message	string	message explaining status in a human readable format.
reason	string	reason for status in a machine readable format.
success	boolean	success indicates if the log entry indicates a success or failure.
time	``	Start time of check action.

28.1.8. .status.outages**Description**

outages contains logs of time periods of outages

Type

array

28.1.9. .status.outages[]**Description**

OutageEntry records time period of an outage

Type

object

Property	Type	Description
end	``	end of outage detected

Property	Type	Description
endLogs	array	endLogs contains log entries related to the end of this outage. Should contain the success entry that resolved the outage and possibly a few of the failure log entries that preceded it.
endLogs[]	object	LogEntry records events
message	string	message summarizes outage details in a human readable format.
start	^^	start of outage detected
startLogs	array	startLogs contains log entries related to the start of this outage. Should contain the original failure, any entries where the failure mode changed.
startLogs[]	object	LogEntry records events

28.1.10. .status.outages[].endLogs

Description

endLogs contains log entries related to the end of this outage. Should contain the success entry that resolved the outage and possibly a few of the failure log entries that preceded it.

Type

array

28.1.11. .status.outages[].endLogs[]

Description

LogEntry records events

Type

object

Required

- **success**

Property	Type	Description
latency	``	latency records how long the action mentioned in the entry took.
message	string	message explaining status in a human readable format.
reason	string	reason for status in a machine readable format.
success	boolean	success indicates if the log entry indicates a success or failure.
time	``	Start time of check action.

28.1.12. `.status.outages[].startLogs`

Description

`startLogs` contains log entries related to the start of this outage. Should contain the original failure, any entries where the failure mode changed.

Type

array

28.1.13. `.status.outages[].startLogs[]`

Description

`LogEntry` records events

Type

object

Required

- **success**

Property	Type	Description
latency	``	latency records how long the action mentioned in the entry took.
message	string	message explaining status in a human readable format.

Property	Type	Description
reason	string	reason for status in a machine readable format.
success	boolean	success indicates if the log entry indicates a success or failure.
time	``	Start time of check action.

28.1.14. .status.successes

Description

successes contains logs successful check actions

Type

array

28.1.15. .status.successes[]

Description

LogEntry records events

Type

object

Required

- **success**

Property	Type	Description
latency	``	latency records how long the action mentioned in the entry took.
message	string	message explaining status in a human readable format.
reason	string	reason for status in a machine readable format.
success	boolean	success indicates if the log entry indicates a success or failure.
time	``	Start time of check action.

28.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/controlplane.operator.openshift.io/v1alpha1/podnetworkconnectivitychecks**
 - **GET**: list objects of kind PodNetworkConnectivityCheck
- **/apis/controlplane.operator.openshift.io/v1alpha1/namespaces/{namespace}/podnetworkconnectivitychecks**
 - **DELETE**: delete collection of PodNetworkConnectivityCheck
 - **GET**: list objects of kind PodNetworkConnectivityCheck
 - **POST**: create a PodNetworkConnectivityCheck
- **/apis/controlplane.operator.openshift.io/v1alpha1/namespaces/{namespace}/podnetworkconnectivitychecks/{name}**
 - **DELETE**: delete a PodNetworkConnectivityCheck
 - **GET**: read the specified PodNetworkConnectivityCheck
 - **PATCH**: partially update the specified PodNetworkConnectivityCheck
 - **PUT**: replace the specified PodNetworkConnectivityCheck
- **/apis/controlplane.operator.openshift.io/v1alpha1/namespaces/{namespace}/podnetworkconnectivitychecks/{name}/status**
 - **GET**: read status of the specified PodNetworkConnectivityCheck
 - **PATCH**: partially update status of the specified PodNetworkConnectivityCheck
 - **PUT**: replace status of the specified PodNetworkConnectivityCheck

28.2.1. /apis/controlplane.operator.openshift.io/v1alpha1/podnetworkconnectivityche

HTTP method

GET

Description

list objects of kind PodNetworkConnectivityCheck

Table 28.1. HTTP responses

HTTP code	Reponse body
200 - OK	PodNetworkConnectivityCheckList schema
401 - Unauthorized	Empty

28.2.2. /apis/controlplane.operator.openshift.io/v1alpha1/namespaces/{namespace}/

HTTP method

DELETE**Description**

delete collection of PodNetworkConnectivityCheck

Table 28.2. HTTP responses

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

HTTP method**GET****Description**

list objects of kind PodNetworkConnectivityCheck

Table 28.3. HTTP responses

HTTP code	Reponse body
200 - OK	PodNetworkConnectivityCheckList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create a PodNetworkConnectivityCheck

Table 28.4. 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

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 28.5. Body parameters

Parameter	Type	Description
body	PodNetworkConnectivityCheck schema	

Table 28.6. HTTP responses

HTTP code	Response body
200 - OK	PodNetworkConnectivityCheck schema
201 - Created	PodNetworkConnectivityCheck schema
202 - Accepted	PodNetworkConnectivityCheck schema
401 - Unauthorized	Empty

28.2.3. /apis/controlplane.operator.openshift.io/v1alpha1/namespaces/{namespace}/

Table 28.7. Global path parameters

Parameter	Type	Description
name	string	name of the PodNetworkConnectivityCheck

HTTP method**DELETE****Description**

delete a PodNetworkConnectivityCheck

Table 28.8. 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 28.9. HTTP responses

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

HTTP method**GET****Description**

read the specified PodNetworkConnectivityCheck

Table 28.10. HTTP responses

HTTP code	Reponse body
200 - OK	PodNetworkConnectivityCheck schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified PodNetworkConnectivityCheck

Table 28.11. 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 28.12. HTTP responses

HTTP code	Response body
200 - OK	PodNetworkConnectivityCheck schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified PodNetworkConnectivityCheck

Table 28.13. 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 28.14. Body parameters

Parameter	Type	Description
body	PodNetworkConnectivityCheck schema	

Table 28.15. HTTP responses

HTTP code	Response body
200 - OK	PodNetworkConnectivityCheck schema
201 - Created	PodNetworkConnectivityCheck schema
401 - Unauthorized	Empty

28.2.4. /apis/controlplane.operator.openshift.io/v1alpha1/namespaces/{namespace}/

Table 28.16. Global path parameters

Parameter	Type	Description
name	string	name of the PodNetworkConnectivityCheck

HTTP method

GET**Description**

read status of the specified PodNetworkConnectivityCheck

Table 28.17. HTTP responses

HTTP code	Response body
200 - OK	PodNetworkConnectivityCheck schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified PodNetworkConnectivityCheck

Table 28.18. 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 28.19. HTTP responses

HTTP code	Reponse body
200 - OK	PodNetworkConnectivityCheck schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified PodNetworkConnectivityCheck

Table 28.20. 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 28.21. Body parameters

Parameter	Type	Description
body	PodNetworkConnectivityCheck schema	

Table 28.22. HTTP responses

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

CHAPTER 29. REFERENCEGRANT [GATEWAY.NETWORKING.K8S.IO/V1BETA1]

Description

ReferenceGrant identifies kinds of resources in other namespaces that are trusted to reference the specified kinds of resources in the same namespace as the policy.

Each ReferenceGrant can be used to represent a unique trust relationship. Additional Reference Grants can be used to add to the set of trusted sources of inbound references for the namespace they are defined within.

All cross-namespace references in Gateway API (with the exception of cross-namespace Gateway-route attachment) require a ReferenceGrant.

ReferenceGrant is a form of runtime verification allowing users to assert which cross-namespace object references are permitted. Implementations that support ReferenceGrant MUST NOT permit cross-namespace references which have no grant, and MUST respond to the removal of a grant by revoking the access that the grant allowed.

Type

object

29.1. 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 defines the desired state of ReferenceGrant.

29.1.1. .spec

Description

Spec defines the desired state of ReferenceGrant.

Type

object

Required

- **from**
- **to**

Property	Type	Description
from	array	From describes the trusted namespaces and kinds that can reference the resources described in "To". Each entry in this list MUST be considered to be an additional place that references can be valid from, or to put this another way, entries MUST be combined using OR. Support: Core
from[]	object	ReferenceGrantFrom describes trusted namespaces and kinds.

Property	Type	Description
to	array	To describes the resources that may be referenced by the resources described in "From". Each entry in this list MUST be considered to be an additional place that references can be valid to, or to put this another way, entries MUST be combined using OR. Support: Core
to[]	object	ReferenceGrantTo describes what Kinds are allowed as targets of the references.

29.1.2. .spec.from

Description

From describes the trusted namespaces and kinds that can reference the resources described in "To". Each entry in this list MUST be considered to be an additional place that references can be valid from, or to put this another way, entries MUST be combined using OR.

Support: Core

Type

array

29.1.3. .spec.from[]

Description

ReferenceGrantFrom describes trusted namespaces and kinds.

Type

object

Required

- **group**
- **kind**
- **namespace**

Property	Type	Description

Property	Type	Description
group	string	Group is the group of the referent. When empty, the Kubernetes core API group is inferred. Support: Core
kind	string	Kind is the kind of the referent. Although implementations may support additional resources, the following types are part of the "Core" support level for this field. When used to permit a SecretObjectReference: * Gateway When used to permit a BackendObjectReference: * GRPCRoute * HTTPRoute * TCPRoute * TLSRoute * UDPRoute
namespace	string	Namespace is the namespace of the referent. Support: Core

29.1.4. .spec.to

Description

To describes the resources that may be referenced by the resources described in "From". Each entry in this list **MUST** be considered to be an additional place that references can be valid to, or to put this another way, entries **MUST** be combined using OR.

Support: Core

Type

array

29.1.5. .spec.to[]

Description

ReferenceGrantTo describes what Kinds are allowed as targets of the references.

Type

object

Required

- **group**
- **kind**

Property	Type	Description
group	string	Group is the group of the referent. When empty, the Kubernetes core API group is inferred. Support: Core
kind	string	Kind is the kind of the referent. Although implementations may support additional resources, the following types are part of the "Core" support level for this field: * Secret when used to permit a SecretObjectReference * Service when used to permit a BackendObjectReference
name	string	Name is the name of the referent. When unspecified, this policy refers to all resources of the specified Group and Kind in the local namespace.

29.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/gateway.networking.k8s.io/v1beta1/referencegrants**
 - **GET**: list objects of kind ReferenceGrant
- **/apis/gateway.networking.k8s.io/v1beta1/namespaces/{namespace}/referencegrants**
 - **DELETE**: delete collection of ReferenceGrant
 - **GET**: list objects of kind ReferenceGrant
 - **POST**: create a ReferenceGrant
- **/apis/gateway.networking.k8s.io/v1beta1/namespaces/{namespace}/referencegrants/{name}**
 - **DELETE**: delete a ReferenceGrant
 - **GET**: read the specified ReferenceGrant

- **PATCH**: partially update the specified ReferenceGrant
- **PUT**: replace the specified ReferenceGrant

29.2.1. /apis/gateway.networking.k8s.io/v1beta1/referencegrants

HTTP method

GET

Description

list objects of kind ReferenceGrant

Table 29.1. HTTP responses

HTTP code	Reponse body
200 - OK	ReferenceGrantList schema
401 - Unauthorized	Empty

29.2.2. /apis/gateway.networking.k8s.io/v1beta1/namespaces/{namespace}/referencegrants

HTTP method

DELETE

Description

delete collection of ReferenceGrant

Table 29.2. HTTP responses

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

HTTP method

GET

Description

list objects of kind ReferenceGrant

Table 29.3. HTTP responses

HTTP code	Reponse body
200 - OK	ReferenceGrantList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create a ReferenceGrant

Table 29.4. 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 29.5. Body parameters

Parameter	Type	Description
body	ReferenceGrant schema	

Table 29.6. HTTP responses

HTTP code	Reponse body
200 - OK	ReferenceGrant schema
201 - Created	ReferenceGrant schema

HTTP code	Reponse body
202 - Accepted	ReferenceGrant schema
401 - Unauthorized	Empty

29.2.3. /apis/gateway.networking.k8s.io/v1beta1/namespaces/{namespace}/referenc

Table 29.7. Global path parameters

Parameter	Type	Description
name	string	name of the ReferenceGrant

HTTP method

DELETE

Description

delete a ReferenceGrant

Table 29.8. 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 29.9. HTTP responses

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

HTTP method

GET

Description

read the specified ReferenceGrant

Table 29.10. HTTP responses

HTTP code	Reponse body
200 - OK	ReferenceGrant schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified ReferenceGrant

Table 29.11. 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 29.12. HTTP responses

HTTP code	Reponse body
200 - OK	ReferenceGrant schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified ReferenceGrant

Table 29.13. 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 29.14. Body parameters

Parameter	Type	Description
body	ReferenceGrant schema	

Table 29.15. HTTP responses

HTTP code	Response body
200 - OK	ReferenceGrant schema
201 - Created	ReferenceGrant schema

HTTP code	Reponse body
401 - Unauthorized	Empty

CHAPTER 30. ROUTE [ROUTE.OPENSIFT.IO/V1]

Description

A route allows developers to expose services through an HTTP(S) aware load balancing and proxy layer via a public DNS entry. The route may further specify TLS options and a certificate, or specify a public CNAME that the router should also accept for HTTP and HTTPS traffic. An administrator typically configures their router to be visible outside the cluster firewall, and may also add additional security, caching, or traffic controls on the service content. Routers usually talk directly to the service endpoints.

Once a route is created, the **host** field may not be changed. Generally, routers use the oldest route with a given host when resolving conflicts.

Routers are subject to additional customization and may support additional controls via the annotations field.

Because administrators may configure multiple routers, the route status field is used to return information to clients about the names and states of the route under each router. If a client chooses a duplicate name, for instance, the route status conditions are used to indicate the route cannot be chosen.

To enable HTTP/2 ALPN on a route it requires a custom (non-wildcard) certificate. This prevents connection coalescing by clients, notably web browsers. We do not support HTTP/2 ALPN on routes that use the default certificate because of the risk of connection re-use/coalescing. Routes that do not have their own custom certificate will not be HTTP/2 ALPN-enabled on either the frontend or the backend.

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**

30.1. 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

Property	Type	Description
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_v2	metadata is the standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	object	RouteSpec describes the hostname or path the route exposes, any security information, and one to four backends (services) the route points to. Requests are distributed among the backends depending on the weights assigned to each backend. When using roundrobin scheduling the portion of requests that go to each backend is the backend weight divided by the sum of all of the backend weights. When the backend has more than one endpoint the requests that end up on the backend are roundrobin distributed among the endpoints. Weights are between 0 and 256 with default 100. Weight 0 causes no requests to the backend. If all weights are zero the route will be considered to have no backends and return a standard 503 response. The tls field is optional and allows specific certificates or behavior for the route. Routers typically configure a default certificate on a wildcard domain to terminate routes without explicit certificates, but custom hostnames usually must choose passthrough (send traffic directly to the backend via the TLS Server-Name- Indication field) or provide a certificate.
status	object	RouteStatus provides relevant info about the status of a route, including which routers acknowledge it.

30.1.1. .spec

Description

RouteSpec describes the hostname or path the route exposes, any security information, and one to four backends (services) the route points to. Requests are distributed among the backends depending on the weights assigned to each backend. When using roundrobin scheduling the portion of requests that go to each backend is the backend weight divided by the sum of all of the backend

weights. When the backend has more than one endpoint the requests that end up on the backend are roundrobin distributed among the endpoints. Weights are between 0 and 256 with default 100. Weight 0 causes no requests to the backend. If all weights are zero the route will be considered to have no backends and return a standard 503 response.

The **tls** field is optional and allows specific certificates or behavior for the route. Routers typically configure a default certificate on a wildcard domain to terminate routes without explicit certificates, but custom hostnames usually must choose passthrough (send traffic directly to the backend via the TLS Server-Name- Indication field) or provide a certificate.

Type

object

Required

- to

Property	Type	Description
alternateBackends	array	alternateBackends allows up to 3 additional backends to be assigned to the route. Only the Service kind is allowed, and it will be defaulted to Service. Use the weight field in RouteTargetReference object to specify relative preference.
alternateBackends[]	object	RouteTargetReference specifies the target that resolve into endpoints. Only the 'Service' kind is allowed. Use 'weight' field to emphasize one over others.
host	string	host is an alias/DNS that points to the service. Optional. If not specified a route name will typically be automatically chosen. Must follow DNS952 subdomain conventions.
httpHeaders	object	RouteHTTPHeaders defines policy for HTTP headers.
path	string	path that the router watches for, to route traffic for to the service. Optional
port	object	RoutePort defines a port mapping from a router to an endpoint in the service endpoints.

Property	Type	Description
subdomain	string	subdomain is a DNS subdomain that is requested within the ingress controller's domain (as a subdomain). If host is set this field is ignored. An ingress controller may choose to ignore this suggested name, in which case the controller will report the assigned name in the status.ingress array or refuse to admit the route. If this value is set and the server does not support this field host will be populated automatically. Otherwise host is left empty. The field may have multiple parts separated by a dot, but not all ingress controllers may honor the request. This field may not be changed after creation except by a user with the update routes/custom-host permission. Example: subdomain frontend automatically receives the router subdomain apps.mycluster.com to have a full hostname frontend.apps.mycluster.com.
tls	object	TLSConfig defines config used to secure a route and provide termination
to	object	RouteTargetReference specifies the target that resolve into endpoints. Only the 'Service' kind is allowed. Use 'weight' field to emphasize one over others.
wildcardPolicy	string	Wildcard policy if any for the route. Currently only 'Subdomain' or 'None' is allowed.

30.1.2. .spec.alternateBackends

Description

alternateBackends allows up to 3 additional backends to be assigned to the route. Only the Service kind is allowed, and it will be defaulted to Service. Use the weight field in RouteTargetReference object to specify relative preference.

Type

array

30.1.3. .spec.alternateBackends[]

Description

RouteTargetReference specifies the target that resolve into endpoints. Only the 'Service' kind is allowed. Use 'weight' field to emphasize one over others.

Type

object

Required

- **kind**
- **name**

Property	Type	Description
kind	string	The kind of target that the route is referring to. Currently, only 'Service' is allowed
name	string	name of the service/target that is being referred to. e.g. name of the service
weight	integer	weight as an integer between 0 and 256, default 100, that specifies the target's relative weight against other target reference objects. 0 suppresses requests to this backend.

30.1.4. .spec.httpHeaders

Description

RouteHTTPHeaders defines policy for HTTP headers.

Type

object

Property	Type	Description

Property	Type	Description
actions	object	RouteHTTPHeaderActions defines configuration for actions on HTTP request and response headers.

30.1.5. .spec.httpHeaders.actions

Description

RouteHTTPHeaderActions defines configuration for actions on HTTP request and response headers.

Type

object

Property	Type	Description

Property	Type	Description
request	array	request is a list of HTTP request headers to modify. Currently, actions may define to either Set or Delete headers values. Actions defined here will modify the request headers of all requests made through a route. These actions are applied to a specific Route defined within a cluster i.e. connections made through a route. Currently, actions may define to either Set or Delete headers values. Route actions will be executed after IngressController actions for request headers. Actions are applied in sequence as defined in this list. A maximum of 20 request header actions may be configured. You can use this field to specify HTTP request headers that should be set or deleted when forwarding connections from the client to your application. 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]". Any request header configuration applied directly via a Route resource using this API will override header configuration for a header of the same name applied via spec.httpHeaders.actions on the IngressController or route annotation. Note: This field cannot be used if your route uses TLS passthrough.
request[]	object	RouteHTTPHeader specifies configuration for setting or deleting an HTTP header.

Property	Type	Description
response	array	response is a list of HTTP response headers to modify. Currently, actions may define to either Set or Delete headers values. Actions defined here will modify the response headers of all requests made through a route. These actions are applied to a specific Route defined within a cluster i.e. connections made through a route. Route actions will be executed before IngressController actions for response headers. Actions are applied in sequence as defined in this list. A maximum of 20 response header actions may be configured. You can use this field to specify HTTP response headers that should be set or deleted when forwarding responses from your application to the client. 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]". Note: This field cannot be used if your route uses TLS passthrough.
response[]	object	RouteHTTPHeader specifies configuration for setting or deleting an HTTP header.

30.1.6. .spec.httpHeaders.actions.request

Description

request is a list of HTTP request headers to modify. Currently, actions may define to either **Set** or **Delete** headers values. Actions defined here will modify the request headers of all requests made through a route. These actions are applied to a specific Route defined within a cluster i.e. connections made through a route. Currently, actions may define to either **Set** or **Delete** headers values. Route actions will be executed after IngressController actions for request headers. Actions are applied in sequence as defined in this list. A maximum of 20 request header actions may be configured. You can use this field to specify HTTP request headers that should be set or deleted when forwarding connections from the client to your application. 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]". Any request header configuration applied directly via a

Route resource using this API will override header configuration for a header of the same name applied via `spec.httpHeaders.actions` on the IngressController or route annotation. Note: This field cannot be used if your route uses TLS passthrough.

Type

array

30.1.7. `.spec.httpHeaders.actions.request[]`

Description

RouteHTTPHeader specifies configuration for setting or deleting an HTTP header.

Type

object

Required

- **name**
- **action**

Property	Type	Description
action	object	RouteHTTPHeaderActionUnion specifies an action to take on an HTTP header.
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, Cookie, Set-Cookie. It must be no more than 255 characters in length. Header name must be unique.

30.1.8. `.spec.httpHeaders.actions.request[].action`

Description

RouteHTTPHeaderActionUnion specifies an action to take on an HTTP header.

Type

object

Required

- **type**

Property	Type	Description
set	object	RouteSetHTTPHeader specifies what value needs to be set on an HTTP header.
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.

30.1.9. .spec.httpHeaders.actions.request[].action.set

Description

RouteSetHTTPHeader specifies what value needs to be set on an HTTP header.

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.

30.1.10. .spec.httpHeaders.actions.response

Description

response is a list of HTTP response headers to modify. Currently, actions may define to either **Set** or **Delete** headers values. Actions defined here will modify the response headers of all requests made through a route. These actions are applied to a specific Route defined within a cluster i.e. connections made through a route. Route actions will be executed before IngressController actions for response headers. Actions are applied in sequence as defined in this list. A maximum of 20 response header actions may be configured. You can use this field to specify HTTP response headers that should be set or deleted when forwarding responses from your application to the client. 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]". Note: This field cannot be used if your route uses TLS passthrough.

Type

array

30.1.11. .spec.httpHeaders.actions.response[]

Description

RouteHTTPHeader specifies configuration for setting or deleting an HTTP header.

Type

object

Required

- **name**
- **action**

Property	Type	Description
action	object	RouteHTTPHeaderActionUnion specifies an action to take on an HTTP header.
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, Cookie, Set-Cookie. It must be no more than 255 characters in length. Header name must be unique.

30.1.12. .spec.httpHeaders.actions.response[].action

Description

RouteHTTPHeaderActionUnion specifies an action to take on an HTTP header.

Type

object

Required

- **type**

Property	Type	Description
set	object	RouteSetHTTPHeader specifies what value needs to be set on an HTTP header.
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.

30.1.13. .spec.httpHeaders.actions.response[].action.set

Description

RouteSetHTTPHeader specifies what value needs to be set on an HTTP header.

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.

30.1.14. .spec.port

Description

RoutePort defines a port mapping from a router to an endpoint in the service endpoints.

Type

object

Required

- **targetPort**

Property	Type	Description
targetPort	IntOrString	The target port on pods selected by the service this route points to. If this is a string, it will be looked up as a named port in the target endpoints port list. Required

30.1.15. .spec.tls

Description

TLSConfig defines config used to secure a route and provide termination

Type

object**Required**

- **termination**

Property	Type	Description
caCertificate	string	caCertificate provides the cert authority certificate contents
certificate	string	certificate provides certificate contents. This should be a single serving certificate, not a certificate chain. Do not include a CA certificate.
destinationCACertificate	string	destinationCACertificate provides the contents of the ca certificate of the final destination. When using reencrypt termination this file should be provided in order to have routers use it for health checks on the secure connection. If this field is not specified, the router may provide its own destination CA and perform hostname validation using the short service name (service.namespace.svc), which allows infrastructure generated certificates to automatically verify.
externalCertificate	object	LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Property	Type	Description
insecureEdgeTerminationPolicy	string	insecureEdgeTerminationPolicy indicates the desired behavior for insecure connections to a route. While each router may make its own decisions on which ports to expose, this is normally port 80. If a route does not specify insecureEdgeTerminationPolicy, then the default behavior is "None". * Allow - traffic is sent to the server on the insecure port (edge/reencrypt terminations only). * None - no traffic is allowed on the insecure port (default). * Redirect - clients are redirected to the secure port.
key	string	key provides key file contents
termination	string	termination indicates termination type. * edge - TLS termination is done by the router and http is used to communicate with the backend (default) * passthrough - Traffic is sent straight to the destination without the router providing TLS termination * reencrypt - TLS termination is done by the router and https is used to communicate with the backend Note: passthrough termination is incompatible with httpHeader actions

30.1.16. .spec.tls.externalCertificate

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

object

Property	Type	Description
name	string	name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names

30.1.17. .spec.to

Description

RouteTargetReference specifies the target that resolve into endpoints. Only the 'Service' kind is allowed. Use 'weight' field to emphasize one over others.

Type

object

Required

- **kind**
- **name**

Property	Type	Description
kind	string	The kind of target that the route is referring to. Currently, only 'Service' is allowed
name	string	name of the service/target that is being referred to. e.g. name of the service
weight	integer	weight as an integer between 0 and 256, default 100, that specifies the target's relative weight against other target reference objects. 0 suppresses requests to this backend.

30.1.18. .status

Description

RouteStatus provides relevant info about the status of a route, including which routers acknowledge it.

Type

object

Property	Type	Description
ingress	array	ingress describes the places where the route may be exposed. The list of ingress points may contain duplicate Host or RouterName values. Routes are considered live once they are Ready
ingress[]	object	RouteIngress holds information about the places where a route is exposed.

30.1.19. .status.ingress

Description

ingress describes the places where the route may be exposed. The list of ingress points may contain duplicate Host or RouterName values. Routes are considered live once they are **Ready**

Type

array

30.1.20. .status.ingress[]

Description

RouteIngress holds information about the places where a route is exposed.

Type

object

Property	Type	Description
conditions	array	Conditions is the state of the route, may be empty.
conditions[]	object	RouteIngressCondition contains details for the current condition of this route on a particular router.
host	string	Host is the host string under which the route is exposed; this value is required
routerCanonicalHostname	string	CanonicalHostname is the external host name for the router that can be used as a CNAME for the host requested for this route. This value is optional and may not be set in all cases.

Property	Type	Description
routerName	string	Name is a name chosen by the router to identify itself; this value is required
wildcardPolicy	string	Wildcard policy is the wildcard policy that was allowed where this route is exposed.

30.1.21. .status.ingress[].conditions

Description

Conditions is the state of the route, may be empty.

Type

array

30.1.22. .status.ingress[].conditions[]

Description

RouteIngressCondition contains details for the current condition of this route on a particular router.

Type

object

Required

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

Property	Type	Description
lastTransitionTime	Time	RFC 3339 date and time when this condition last transitioned
message	string	Human readable message indicating details about last transition.
reason	string	(brief) reason for the condition's last transition, and is usually a machine and human readable constant
status	string	Status is the status of the condition. Can be True, False, Unknown.

Property	Type	Description
type	string	Type is the type of the condition. Currently only Admitted or UnservedInFutureVersions.

30.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/route.openshift.io/v1/routes**
 - **GET**: list or watch objects of kind Route
- **/apis/route.openshift.io/v1/watch/routes**
 - **GET**: watch individual changes to a list of Route. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/route.openshift.io/v1/namespaces/{namespace}/routes**
 - **DELETE**: delete collection of Route
 - **GET**: list or watch objects of kind Route
 - **POST**: create a Route
- **/apis/route.openshift.io/v1/watch/namespaces/{namespace}/routes**
 - **GET**: watch individual changes to a list of Route. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/route.openshift.io/v1/namespaces/{namespace}/routes/{name}**
 - **DELETE**: delete a Route
 - **GET**: read the specified Route
 - **PATCH**: partially update the specified Route
 - **PUT**: replace the specified Route
- **/apis/route.openshift.io/v1/watch/namespaces/{namespace}/routes/{name}**
 - **GET**: watch changes to an object of kind Route. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.
- **/apis/route.openshift.io/v1/namespaces/{namespace}/routes/{name}/status**
 - **GET**: read status of the specified Route
 - **PATCH**: partially update status of the specified Route

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

30.2.1. /apis/route.openshift.io/v1/routes

HTTP method

GET

Description

list or watch objects of kind Route

Table 30.1. HTTP responses

HTTP code	Reponse body
200 - OK	RouteList schema
401 - Unauthorized	Empty

30.2.2. /apis/route.openshift.io/v1/watch/routes

HTTP method

GET

Description

watch individual changes to a list of Route. deprecated: use the 'watch' parameter with a list operation instead.

Table 30.2. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

30.2.3. /apis/route.openshift.io/v1/namespaces/{namespace}/routes

HTTP method

DELETE

Description

delete collection of Route

Table 30.3. 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

Table 30.4. HTTP responses

HTTP code	Reponse body
200 - OK	Status_v7 schema
401 - Unauthorized	Empty

HTTP method**GET****Description**

list or watch objects of kind Route

Table 30.5. HTTP responses

HTTP code	Reponse body
200 - OK	RouteList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create a Route

Table 30.6. 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 30.7. Body parameters

Parameter	Type	Description
body	Route schema	

Table 30.8. HTTP responses

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

30.2.4. /apis/route.openshift.io/v1/watch/namespaces/{namespace}/routes

HTTP method

GET

Description

watch individual changes to a list of Route. deprecated: use the 'watch' parameter with a list operation instead.

Table 30.9. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

30.2.5. /apis/route.openshift.io/v1/namespaces/{namespace}/routes/{name}

Table 30.10. Global path parameters

Parameter	Type	Description
name	string	name of the Route

HTTP method

DELETE

Description

delete a Route

Table 30.11. 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

Table 30.12. HTTP responses

HTTP code	Reponse body
200 - OK	Status_v7 schema
202 - Accepted	Status_v7 schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Route

Table 30.13. HTTP responses

HTTP code	Response body
200 - OK	Route schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified Route

Table 30.14. 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 30.15. HTTP responses

HTTP code	Response body
200 - OK	Route schema
201 - Created	Route schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified Route

Table 30.16. 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 30.17. Body parameters

Parameter	Type	Description
body	Route schema	

Table 30.18. HTTP responses

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

30.2.6. /apis/route.openshift.io/v1/watch/namespaces/{namespace}/routes/{name}

Table 30.19. Global path parameters

Parameter	Type	Description
name	string	name of the Route

HTTP method

GET

Description

watch changes to an object of kind Route. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 30.20. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

30.2.7. /apis/route.openshift.io/v1/namespaces/{namespace}/routes/{name}/status

Table 30.21. Global path parameters

Parameter	Type	Description
name	string	name of the Route

HTTP method

GET

Description

read status of the specified Route

Table 30.22. HTTP responses

HTTP code	Reponse body
200 - OK	Route schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified Route

Table 30.23. 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 30.24. HTTP responses

HTTP code	Reponse body
200 - OK	Route schema
201 - Created	Route schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified Route

Table 30.25. 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 30.26. Body parameters

Parameter	Type	Description
body	Route schema	

Table 30.27. HTTP responses

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

CHAPTER 31. SERVICE [V1]

Description

Service is a named abstraction of software service (for example, mysql) consisting of local port (for example 3306) that the proxy listens on, and the selector that determines which pods will answer requests sent through the proxy.

Type

object

31.1. 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	ServiceSpec describes the attributes that a user creates on a service.
status	object	ServiceStatus represents the current status of a service.

31.1.1. .spec

Description

ServiceSpec describes the attributes that a user creates on a service.

Type

object

Property	Type	Description
allocateLoadBalancerNodePorts	boolean	allocateLoadBalancerNodePorts defines if NodePorts will be automatically allocated for services with type LoadBalancer. Default is "true". It may be set to "false" if the cluster load-balancer does not rely on NodePorts. If the caller requests specific NodePorts (by specifying a value), those requests will be respected, regardless of this field. This field may only be set for services with type LoadBalancer and will be cleared if the type is changed to any other type.

Property	Type	Description
clusterIP	string	clusterIP is the IP address of the service and is usually assigned randomly. If an address is specified manually, is in-range (as per system configuration), and is not in use, it will be allocated to the service; otherwise creation of the service will fail. This field may not be changed through updates unless the type field is also being changed to ExternalName (which requires this field to be blank) or the type field is being changed from ExternalName (in which case this field may optionally be specified, as describe above). Valid values are "None", empty string (""), or a valid IP address. Setting this to "None" makes a "headless service" (no virtual IP), which is useful when direct endpoint connections are preferred and proxying is not required. Only applies to types ClusterIP, NodePort, and LoadBalancer. If this field is specified when creating a Service of type ExternalName, creation will fail. This field will be wiped when updating a Service to type ExternalName. More info: https://kubernetes.io/docs/concepts/services-networking/service/#virtual-ips-and-service-proxies

Property	Type	Description
clusterIPs	array (string)	ClusterIPs is a list of IP addresses assigned to this service, and are usually assigned randomly. If an address is specified manually, is in-range (as per system configuration), and is not in use, it will be allocated to the service; otherwise creation of the service will fail. This field may not be changed through updates unless the type field is also being changed to ExternalName (which requires this field to be empty) or the type field is being changed from ExternalName (in which case this field may optionally be specified, as describe above). Valid values are "None", empty string (""), or a valid IP address. Setting this to "None" makes a "headless service" (no virtual IP), which is useful when direct endpoint connections are preferred and proxying is not required. Only applies to types ClusterIP, NodePort, and LoadBalancer. If this field is specified when creating a Service of type ExternalName, creation will fail. This field will be wiped when updating a Service to type ExternalName. If this field is not specified, it will be initialized from the clusterIP field. If this field is specified, clients must ensure that clusterIPs[0] and clusterIP have the same value. This field may hold a maximum of two entries (dual-stack IPs, in either order). These IPs must correspond to the values of the ipFamilies field. Both clusterIPs and ipFamilies are governed by the ipFamilyPolicy field. More info: https://kubernetes.io/docs/concepts/services-networking/service/#virtual-ips-and-service-proxies

Property	Type	Description
externalIPs	array (string)	externalIPs is a list of IP addresses for which nodes in the cluster will also accept traffic for this service. These IPs are not managed by Kubernetes. The user is responsible for ensuring that traffic arrives at a node with this IP. A common example is external load-balancers that are not part of the Kubernetes system.
externalName	string	externalName is the external reference that discovery mechanisms will return as an alias for this service (e.g. a DNS CNAME record). No proxying will be involved. Must be a lowercase RFC-1123 hostname (https://tools.ietf.org/html/rfc1123) and requires type to be "ExternalName".

Property	Type	Description
externalTrafficPolicy	string	externalTrafficPolicy describes how nodes distribute service traffic they receive on one of the Service's "externally-facing" addresses (NodePorts, ExternalIPs, and LoadBalancer IPs). If set to "Local", the proxy will configure the service in a way that assumes that external load balancers will take care of balancing the service traffic between nodes, and so each node will deliver traffic only to the node-local endpoints of the service, without masquerading the client source IP. (Traffic mistakenly sent to a node with no endpoints will be dropped.) The default value, "Cluster", uses the standard behavior of routing to all endpoints evenly (possibly modified by topology and other features). Note that traffic sent to an External IP or LoadBalancer IP from within the cluster will always get "Cluster" semantics, but clients sending to a NodePort from within the cluster may need to take traffic policy into account when picking a node. Possible enum values: - "Cluster" routes traffic to all endpoints. - "Local" preserves the source IP of the traffic by routing only to endpoints on the same node as the traffic was received on (dropping the traffic if there are no local endpoints).

Property	Type	Description
healthCheckNodePort	integer	healthCheckNodePort specifies the healthcheck nodePort for the service. This only applies when type is set to LoadBalancer and externalTrafficPolicy is set to Local. If a value is specified, is in-range, and is not in use, it will be used. If not specified, a value will be automatically allocated. External systems (e.g. load-balancers) can use this port to determine if a given node holds endpoints for this service or not. If this field is specified when creating a Service which does not need it, creation will fail. This field will be wiped when updating a Service to no longer need it (e.g. changing type). This field cannot be updated once set.
internalTrafficPolicy	string	InternalTrafficPolicy describes how nodes distribute service traffic they receive on the ClusterIP. If set to "Local", the proxy will assume that pods only want to talk to endpoints of the service on the same node as the pod, dropping the traffic if there are no local endpoints. The default value, "Cluster", uses the standard behavior of routing to all endpoints evenly (possibly modified by topology and other features). Possible enum values: - "Cluster" routes traffic to all endpoints. - "Local" routes traffic only to endpoints on the same node as the client pod (dropping the traffic if there are no local endpoints).

Property	Type	Description
ipFamilies	array (string)	IPFamilies is a list of IP families (e.g. IPv4, IPv6) assigned to this service. This field is usually assigned automatically based on cluster configuration and the ipFamilyPolicy field. If this field is specified manually, the requested family is available in the cluster, and ipFamilyPolicy allows it, it will be used; otherwise creation of the service will fail. This field is conditionally mutable: it allows for adding or removing a secondary IP family, but it does not allow changing the primary IP family of the Service. Valid values are "IPv4" and "IPv6". This field only applies to Services of types ClusterIP, NodePort, and LoadBalancer, and does apply to "headless" services. This field will be wiped when updating a Service to type ExternalName. This field may hold a maximum of two entries (dual-stack families, in either order). These families must correspond to the values of the clusterIPs field, if specified. Both clusterIPs and ipFamilies are governed by the ipFamilyPolicy field.
ipFamilyPolicy	string	IPFamilyPolicy represents the dual-stack-ness requested or required by this Service. If there is no value provided, then this field will be set to SingleStack. Services can be "SingleStack" (a single IP family), "PreferDualStack" (two IP families on dual-stack configured clusters or a single IP family on single-stack clusters), or "RequireDualStack" (two IP families on dual-stack configured clusters, otherwise fail). The ipFamilies and clusterIPs fields depend on the value of this field. This field will be wiped when updating a service to type ExternalName.

Property	Type	Possible enum values: - Description
		that this service prefers dual-stack when the cluster is configured for dual-stack. If the cluster is not configured for dual-stack the service will be assigned a single IPFamily. If the IPFamily is not set in service.spec.ipFamilies then the service will be assigned the default IPFamily configured on the cluster - "RequireDualStack" indicates that this service requires dual-stack. Using IPFamilyPolicyRequireDualStack on a single stack cluster will result in validation errors. The IPFamilies (and their order) assigned to this service is based on service.spec.ipFamilies. If service.spec.ipFamilies was not provided then it will be assigned according to how they are configured on the cluster. If service.spec.ipFamilies has only one entry then the alternative IPFamily will be added by apiserver - "SingleStack" indicates that this service is required to have a single IPFamily. The IPFamily assigned is based on the default IPFamily used by the cluster or as identified by service.spec.ipFamilies field

Property	Type	Description
loadBalancerClass	string	loadBalancerClass is the class of the load balancer implementation this Service belongs to. If specified, the value of this field must be a label-style identifier, with an optional prefix, e.g. "internal-vip" or "example.com/internal-vip". Unprefixed names are reserved for end-users. This field can only be set when the Service type is 'LoadBalancer'. If not set, the default load balancer implementation is used, today this is typically done through the cloud provider integration, but should apply for any default implementation. If set, it is assumed that a load balancer implementation is watching for Services with a matching class. Any default load balancer implementation (e.g. cloud providers) should ignore Services that set this field. This field can only be set when creating or updating a Service to type 'LoadBalancer'. Once set, it can not be changed. This field will be wiped when a service is updated to a non 'LoadBalancer' type.
loadBalancerIP	string	Only applies to Service Type: LoadBalancer. This feature depends on whether the underlying cloud-provider supports specifying the loadBalancerIP when a load balancer is created. This field will be ignored if the cloud-provider does not support the feature. Deprecated: This field was under-specified and its meaning varies across implementations. Using it is non-portable and it may not support dual-stack. Users are encouraged to use implementation-specific annotations when available.

Property	Type	Description
loadBalancerSourceRanges	array (string)	If specified and supported by the platform, this will restrict traffic through the cloud-provider load-balancer will be restricted to the specified client IPs. This field will be ignored if the cloud-provider does not support the feature." More info: https://kubernetes.io/docs/tasks/access-application-cluster/create-external-load-balancer/
ports	array	The list of ports that are exposed by this service. More info: https://kubernetes.io/docs/concepts/services-networking/service/#virtual-ips-and-service-proxies
ports[]	object	ServicePort contains information on service's port.
publishNotReadyAddresses	boolean	publishNotReadyAddresses indicates that any agent which deals with endpoints for this Service should disregard any indications of ready/not-ready. The primary use case for setting this field is for a StatefulSet's Headless Service to propagate SRV DNS records for its Pods for the purpose of peer discovery. The Kubernetes controllers that generate Endpoints and EndpointSlice resources for Services interpret this to mean that all endpoints are considered "ready" even if the Pods themselves are not. Agents which consume only Kubernetes generated endpoints through the Endpoints or EndpointSlice resources can safely assume this behavior.

Property	Type	Description
selector	object (string)	Route service traffic to pods with label keys and values matching this selector. If empty or not present, the service is assumed to have an external process managing its endpoints, which Kubernetes will not modify. Only applies to types ClusterIP, NodePort, and LoadBalancer. Ignored if type is ExternalName. More info: https://kubernetes.io/docs/concepts/services-networking/service/
sessionAffinity	string	Supports "ClientIP" and "None". Used to maintain session affinity. Enable client IP based session affinity. Must be ClientIP or None. Defaults to None. More info: https://kubernetes.io/docs/concepts/services-networking/service/#virtual-ips-and-service-proxies Possible enum values: - - "ClientIP" is the Client IP based. - "None" - no session affinity.
sessionAffinityConfig	object	SessionAffinityConfig represents the configurations of session affinity.
trafficDistribution	string	TrafficDistribution offers a way to express preferences for how traffic is distributed to Service endpoints. Implementations can use this field as a hint, but are not required to guarantee strict adherence. If the field is not set, the implementation will apply its default routing strategy. If set to "PreferClose", implementations should prioritize endpoints that are in the same zone.
type	string	type determines how the Service is exposed. Defaults to ClusterIP. Valid options are ExternalName, ClusterIP, NodePort, and LoadBalancer. "ClusterIP"

Property	Type	Description
		allocates a cluster-internal IP address for load-balancing to endpoints. Endpoints are determined by the selector or if that is not specified, by manual construction of an Endpoints object or EndpointSlice objects. If clusterIP is "None", no virtual IP is allocated and the endpoints are published as a set of endpoints rather than a virtual IP. "NodePort" builds on ClusterIP and allocates a port on every node which routes to the same endpoints as the clusterIP. "LoadBalancer" builds on NodePort and creates an external load-balancer (if supported in the current cloud) which routes to the same endpoints as the clusterIP. "ExternalName" aliases this service to the specified externalName. Several other fields do not apply to ExternalName services. More info: https://kubernetes.io/docs/concepts/services-networking/service/#publishing-services-service-types Possible enum values: - "ClusterIP" means a service will only be accessible inside the cluster, via the cluster IP. - "ExternalName" means a service consists of only a reference to an external name that kubernetes or equivalent will return as a CNAME record, with no exposing or proxying of any pods involved. - "LoadBalancer" means a service will be exposed via an external load balancer (if the cloud provider supports it), in addition to 'NodePort' type. - "NodePort" means a service will be exposed on one port of every node, in addition to 'ClusterIP' type.

31.1.2. .spec.ports

Description

The list of ports that are exposed by this service. More info:

<https://kubernetes.io/docs/concepts/services-networking/service/#virtual-ips-and-service-proxies>

Type

array

31.1.3. .spec.ports[]**Description**

ServicePort contains information on service's port.

Type

object

Required

- **port**

Property	Type	Description

Property	Type	Description
appProtocol	string	The application protocol for this port. This is used as a hint for implementations to offer richer behavior for protocols that they understand. This field follows standard Kubernetes label syntax. Valid values are either: * Un-prefixed protocol names - reserved for IANA standard service names (as per RFC-6335 and https://www.iana.org/assignments/service-names/). * Kubernetes-defined prefixed names: * 'kubernetes.io/h2c' - HTTP/2 prior knowledge over cleartext as described in https://www.rfc-editor.org/rfc/rfc9113.html#name-starting-http-2-with-prior- * 'kubernetes.io/ws' - WebSocket over cleartext as described in https://www.rfc-editor.org/rfc/rfc6455 * 'kubernetes.io/wss' - WebSocket over TLS as described in https://www.rfc-editor.org/rfc/rfc6455 * Other protocols should use implementation-defined prefixed names such as mycompany.com/my-custom-protocol.
name	string	The name of this port within the service. This must be a DNS_LABEL. All ports within a ServiceSpec must have unique names. When considering the endpoints for a Service, this must match the 'name' field in the EndpointPort. Optional if only one ServicePort is defined on this service.

Property	Type	Description
nodePort	integer	The port on each node on which this service is exposed when type is NodePort or LoadBalancer. Usually assigned by the system. If a value is specified, in-range, and not in use it will be used, otherwise the operation will fail. If not specified, a port will be allocated if this Service requires one. If this field is specified when creating a Service which does not need it, creation will fail. This field will be wiped when updating a Service to no longer need it (e.g. changing type from NodePort to ClusterIP). More info: https://kubernetes.io/docs/concepts/services-networking/service/#type-nodeport
port	integer	The port that will be exposed by this service.
protocol	string	The IP protocol for this port. Supports "TCP", "UDP", and "SCTP". Default is TCP. Possible enum values: - "SCTP" is the SCTP protocol. - "TCP" is the TCP protocol. - "UDP" is the UDP protocol.
targetPort	IntOrString	Number or name of the port to access on the pods targeted by the service. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME. If this is a string, it will be looked up as a named port in the target Pod's container ports. If this is not specified, the value of the 'port' field is used (an identity map). This field is ignored for services with clusterIP=None, and should be omitted or set equal to the 'port' field. More info: https://kubernetes.io/docs/concepts/services-networking/service/#defining-a-service

31.1.4. .spec.sessionAffinityConfig

Description

SessionAffinityConfig represents the configurations of session affinity.

Type

object

Property	Type	Description
clientIP	object	ClientIPConfig represents the configurations of Client IP based session affinity.

31.1.5. .spec.sessionAffinityConfig.clientIP

Description

ClientIPConfig represents the configurations of Client IP based session affinity.

Type

object

Property	Type	Description
timeoutSeconds	integer	timeoutSeconds specifies the seconds of ClientIP type session sticky time. The value must be >0 && ≤86400(for 1 day) if ServiceAffinity == "ClientIP". Default value is 10800(for 3 hours).

31.1.6. .status

Description

ServiceStatus represents the current status of a service.

Type

object

Property	Type	Description
conditions	array (Condition)	Current service state
loadBalancer	object	LoadBalancerStatus represents the status of a load-balancer.

31.1.7. .status.loadBalancer

Description

LoadBalancerStatus represents the status of a load-balancer.

Type

object

Property	Type	Description
ingress	array	Ingress is a list containing ingress points for the load-balancer. Traffic intended for the service should be sent to these ingress points.
ingress[]	object	LoadBalancerIngress represents the status of a load-balancer ingress point: traffic intended for the service should be sent to an ingress point.

31.1.8. .status.loadBalancer.ingress**Description**

Ingress is a list containing ingress points for the load-balancer. Traffic intended for the service should be sent to these ingress points.

Type

array

31.1.9. .status.loadBalancer.ingress[]**Description**

LoadBalancerIngress represents the status of a load-balancer ingress point: traffic intended for the service should be sent to an ingress point.

Type

object

Property	Type	Description
hostname	string	Hostname is set for load-balancer ingress points that are DNS based (typically AWS load-balancers)
ip	string	IP is set for load-balancer ingress points that are IP based (typically GCE or OpenStack load-balancers)

Property	Type	Description
ipMode	string	IPMode specifies how the load-balancer IP behaves, and may only be specified when the ip field is specified. Setting this to "VIP" indicates that traffic is delivered to the node with the destination set to the load-balancer's IP and port. Setting this to "Proxy" indicates that traffic is delivered to the node or pod with the destination set to the node's IP and node port or the pod's IP and port. Service implementations may use this information to adjust traffic routing.
ports	array	Ports is a list of records of service ports If used, every port defined in the service should have an entry in it
ports[]	object	PortStatus represents the error condition of a service port

31.1.10. .status.loadBalancer.ingress[].ports

Description

Ports is a list of records of service ports If used, every port defined in the service should have an entry in it

Type

array

31.1.11. .status.loadBalancer.ingress[].ports[]

Description

PortStatus represents the error condition of a service port

Type

object

Required

- **port**
- **protocol**

Property	Type	Description
error	string	Error is to record the problem with the service port The format of the error shall comply with the following rules: - built-in error values shall be specified in this file and those shall use CamelCase names - cloud provider specific error values must have names that comply with the format foo.example.com/CamelCase.
port	integer	Port is the port number of the service port of which status is recorded here
protocol	string	Protocol is the protocol of the service port of which status is recorded here The supported values are: "TCP", "UDP", "SCTP" Possible enum values: - "SCTP" is the SCTP protocol. - "TCP" is the TCP protocol. - "UDP" is the UDP protocol.

31.2. API ENDPOINTS

The following API endpoints are available:

- **/api/v1/services**
 - **GET**: list or watch objects of kind Service
- **/api/v1/watch/services**
 - **GET**: watch individual changes to a list of Service. deprecated: use the 'watch' parameter with a list operation instead.
- **/api/v1/namespaces/{namespace}/services**
 - **DELETE**: delete collection of Service
 - **GET**: list or watch objects of kind Service
 - **POST**: create a Service
- **/api/v1/watch/namespaces/{namespace}/services**
 - **GET**: watch individual changes to a list of Service. deprecated: use the 'watch' parameter with a list operation instead.

- **/api/v1/namespaces/{namespace}/services/{name}**
 - **DELETE:** delete a Service
 - **GET:** read the specified Service
 - **PATCH:** partially update the specified Service
 - **PUT:** replace the specified Service
- **/api/v1/watch/namespaces/{namespace}/services/{name}**
 - **GET:** watch changes to an object of kind Service. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.
- **/api/v1/namespaces/{namespace}/services/{name}/status**
 - **GET:** read status of the specified Service
 - **PATCH:** partially update status of the specified Service
 - **PUT:** replace status of the specified Service

31.2.1. /api/v1/services

HTTP method

GET

Description

list or watch objects of kind Service

Table 31.1. HTTP responses

HTTP code	Reponse body
200 - OK	ServiceList schema
401 - Unauthorized	Empty

31.2.2. /api/v1/watch/services

HTTP method

GET

Description

watch individual changes to a list of Service. deprecated: use the 'watch' parameter with a list operation instead.

Table 31.2. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema

HTTP code	Reponse body
401 - Unauthorized	Empty

31.2.3. /api/v1/namespaces/{namespace}/services

HTTP method

DELETE

Description

delete collection of Service

Table 31.3. 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

Table 31.4. HTTP responses

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

HTTP method

GET

Description

list or watch objects of kind Service

Table 31.5. HTTP responses

HTTP code	Reponse body
200 - OK	ServiceList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a Service

Table 31.6. 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 31.7. Body parameters

Parameter	Type	Description
body	Service schema	

Table 31.8. HTTP responses

HTTP code	Response body
200 - OK	Service schema
201 - Created	Service schema
202 - Accepted	Service schema
401 - Unauthorized	Empty

31.2.4. /api/v1/watch/namespaces/{namespace}/services

HTTP method

GET

Description

watch individual changes to a list of Service. deprecated: use the 'watch' parameter with a list operation instead.

Table 31.9. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

31.2.5. /api/v1/namespaces/{namespace}/services/{name}

Table 31.10. Global path parameters

Parameter	Type	Description
name	string	name of the Service

HTTP method

DELETE

Description

delete a Service

Table 31.11. 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

Table 31.12. HTTP responses

HTTP code	Reponse body
200 - OK	Service schema
202 - Accepted	Service schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method**GET****Description**

read the specified Service

Table 31.13. HTTP responses

HTTP code	Reponse body
200 - OK	Service schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified Service

Table 31.14. 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 31.15. HTTP responses

HTTP code	Response body
200 - OK	Service schema
201 - Created	Service schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified Service

Table 31.16. 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 31.17. Body parameters

Parameter	Type	Description
body	Service schema	

Table 31.18. HTTP responses

HTTP code	Response body
200 - OK	Service schema
201 - Created	Service schema
401 - Unauthorized	Empty

31.2.6. /api/v1/watch/namespaces/{namespace}/services/{name}

Table 31.19. Global path parameters

Parameter	Type	Description
name	string	name of the Service

HTTP method

GET**Description**

watch changes to an object of kind Service. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 31.20. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

31.2.7. /api/v1/namespaces/{namespace}/services/{name}/status**Table 31.21. Global path parameters**

Parameter	Type	Description
name	string	name of the Service

HTTP method**GET****Description**

read status of the specified Service

Table 31.22. HTTP responses

HTTP code	Reponse body
200 - OK	Service schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified Service

Table 31.23. 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 31.24. HTTP responses

HTTP code	Response body
200 - OK	Service schema
201 - Created	Service schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified Service

Table 31.25. 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 31.26. Body parameters

Parameter	Type	Description
body	Service schema	

Table 31.27. HTTP responses

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

CHAPTER 32. SERVICECIDR [NETWORKING.K8S.IO/V1]

Description

ServiceCIDR defines a range of IP addresses using CIDR format (e.g. 192.168.0.0/24 or 2001:db2::/64). This range is used to allocate ClusterIPs to Service objects.

Type

object

32.1. 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	ServiceCIDRSpec define the CIDRs the user wants to use for allocating ClusterIPs for Services.
status	object	ServiceCIDRStatus describes the current state of the ServiceCIDR.

32.1.1. .spec

Description

ServiceCIDRSpec define the CIDRs the user wants to use for allocating ClusterIPs for Services.

Type

object

Property	Type	Description
cidrs	array (string)	CIDRs defines the IP blocks in CIDR notation (e.g. "192.168.0.0/24" or "2001:db8::/64") from which to assign service cluster IPs. Max of two CIDRs is allowed, one of each IP family. This field is immutable.

32.1.2. .status

Description

ServiceCIDRStatus describes the current state of the ServiceCIDR.

Type

object

Property	Type	Description
conditions	array (Condition)	conditions holds an array of metav1.Condition that describe the state of the ServiceCIDR. Current service state

32.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/networking.k8s.io/v1/servicecidrs**
 - **DELETE**: delete collection of ServiceCIDR
 - **GET**: list or watch objects of kind ServiceCIDR
 - **POST**: create a ServiceCIDR
- **/apis/networking.k8s.io/v1/watch/servicecidrs**
 - **GET**: watch individual changes to a list of ServiceCIDR. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/networking.k8s.io/v1/servicecidrs/{name}**
 - **DELETE**: delete a ServiceCIDR

- **GET**: read the specified ServiceCIDR
- **PATCH**: partially update the specified ServiceCIDR
- **PUT**: replace the specified ServiceCIDR
- **/apis/networking.k8s.io/v1/watch/servicecidrs/{name}**
 - **GET**: watch changes to an object of kind ServiceCIDR. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.
- **/apis/networking.k8s.io/v1/servicecidrs/{name}/status**
 - **GET**: read status of the specified ServiceCIDR
 - **PATCH**: partially update status of the specified ServiceCIDR
 - **PUT**: replace status of the specified ServiceCIDR

32.2.1. /apis/networking.k8s.io/v1/servicecidrs

HTTP method

DELETE

Description

delete collection of ServiceCIDR

Table 32.1. 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 32.2. HTTP responses

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

HTTP method

GET

Description

list or watch objects of kind ServiceCIDR

Table 32.3. HTTP responses

HTTP code	Response body
200 - OK	ServiceCIDRList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create a ServiceCIDR

Table 32.4. 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 32.5. Body parameters

Parameter	Type	Description
body	ServiceCIDR schema	

Table 32.6. HTTP responses

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

32.2.2. /apis/networking.k8s.io/v1/watch/servicecidrs

HTTP method

GET

Description

watch individual changes to a list of ServiceCIDR. deprecated: use the 'watch' parameter with a list operation instead.

Table 32.7. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

32.2.3. /apis/networking.k8s.io/v1/servicecidrs/{name}

Table 32.8. Global path parameters

Parameter	Type	Description
name	string	name of the ServiceCIDR

HTTP method

DELETE

Description

delete a ServiceCIDR

Table 32.9. 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

Table 32.10. HTTP responses

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

HTTP method**GET****Description**

read the specified ServiceCIDR

Table 32.11. HTTP responses

HTTP code	Reponse body
200 - OK	ServiceCIDR schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified ServiceCIDR

Table 32.12. 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 32.13. HTTP responses

HTTP code	Response body
200 - OK	ServiceCIDR schema
201 - Created	ServiceCIDR schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified ServiceCIDR

Table 32.14. 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 32.15. Body parameters

Parameter	Type	Description
body	ServiceCIDR schema	

Table 32.16. HTTP responses

HTTP code	Response body
200 - OK	ServiceCIDR schema
201 - Created	ServiceCIDR schema
401 - Unauthorized	Empty

32.2.4. /apis/networking.k8s.io/v1/watch/servicecidrs/{name}

Table 32.17. Global path parameters

Parameter	Type	Description
name	string	name of the ServiceCIDR

HTTP method

GET

Description

watch changes to an object of kind ServiceCIDR. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 32.18. HTTP responses

HTTP code	Reponse body
200 - OK	WatchEvent schema
401 - Unauthorized	Empty

32.2.5. /apis/networking.k8s.io/v1/servicecidrs/{name}/status**Table 32.19. Global path parameters**

Parameter	Type	Description
name	string	name of the ServiceCIDR

HTTP method**GET****Description**

read status of the specified ServiceCIDR

Table 32.20. HTTP responses

HTTP code	Reponse body
200 - OK	ServiceCIDR schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified ServiceCIDR

Table 32.21. 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 32.22. HTTP responses

HTTP code	Response body
200 - OK	ServiceCIDR schema
201 - Created	ServiceCIDR schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified ServiceCIDR

Table 32.23. 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 32.24. Body parameters

Parameter	Type	Description
body	ServiceCIDR schema	

Table 32.25. HTTP responses

HTTP code	Response body
200 - OK	ServiceCIDR schema
201 - Created	ServiceCIDR schema
401 - Unauthorized	Empty

CHAPTER 33. IPADDRESSCLAIM [IPAM.CLUSTER.X-K8S.IO/V1BETA1]

Description

IPAddressClaim is the Schema for the ipaddressclaim API.

Type

object

33.1. 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 desired state of IPAddressClaim.
status	object	status is the observed state of IPAddressClaim.

33.1.1. .spec

Description

spec is the desired state of IPAddressClaim.

Type

object

Required

- **poolRef**

Property	Type	Description
clusterName	string	clusterName is the name of the Cluster this object belongs to.
poolRef	object	poolRef is a reference to the pool from which an IP address should be created.

33.1.2. .spec.poolRef**Description**

poolRef is a reference to the pool from which an IP address should be created.

Type

object

Required

- **kind**
- **name**

Property	Type	Description
apiGroup	string	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
kind	string	Kind is the type of resource being referenced
name	string	Name is the name of resource being referenced

33.1.3. .status

Description

status is the observed state of IPAddressClaim.

Type

object

Property	Type	Description
addressRef	object	addressRef is a reference to the address that was created for this claim.
conditions	array	conditions summarises the current state of the IPAddressClaim
conditions[]	object	Condition defines an observation of a Cluster API resource operational state.
v1beta2	object	v1beta2 groups all the fields that will be added or modified in IPAddressClaim's status with the V1Beta2 version.

33.1.4. .status.addressRef**Description**

addressRef is a reference to the address that was created for this claim.

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

33.1.5. .status.conditions**Description**

conditions summarises the current state of the IPAddressClaim

Type

array

33.1.6. .status.conditions[]

Description

Condition defines an observation of a Cluster API resource operational state.

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	message is a human readable message indicating details about the transition. This field may be empty.
reason	string	reason is the reason for the condition's last transition in CamelCase. The specific API may choose whether or not this field is considered a guaranteed API. This field may be empty.
severity	string	severity provides an explicit classification of Reason code, so the users or machines can immediately understand the current situation and act accordingly. The Severity field MUST be set only when Status=False.
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.

33.1.7. .status.v1beta2

Description

v1beta2 groups all the fields that will be added or modified in IPAddressClaim's status with the V1Beta2 version.

Type

object

Property	Type	Description
conditions	array	conditions represents the observations of a IPAddressClaim's current state.
conditions[]	object	Condition contains details for one aspect of the current state of this API Resource.

33.1.8. .status.v1beta2.conditions

Description

conditions represents the observations of a IPAddressClaim's current state.

Type

array

33.1.9. .status.v1beta2.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.

33.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/ipam.cluster.x-k8s.io/v1beta1/ipaddressclaims**
 - **GET**: list objects of kind IPAddressClaim
- **/apis/ipam.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/ipaddressclaims**
 - **DELETE**: delete collection of IPAddressClaim
 - **GET**: list objects of kind IPAddressClaim
 - **POST**: create an IPAddressClaim
- **/apis/ipam.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/ipaddressclaims/{name}**
 - **DELETE**: delete an IPAddressClaim
 - **GET**: read the specified IPAddressClaim
 - **PATCH**: partially update the specified IPAddressClaim
 - **PUT**: replace the specified IPAddressClaim
- **/apis/ipam.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/ipaddressclaims/{name}/status**
 - **GET**: read status of the specified IPAddressClaim
 - **PATCH**: partially update status of the specified IPAddressClaim
 - **PUT**: replace status of the specified IPAddressClaim

33.2.1. /apis/ipam.cluster.x-k8s.io/v1beta1/ipaddressclaims

HTTP method

GET

Description

list objects of kind IPAddressClaim

Table 33.1. HTTP responses

HTTP code	Reponse body
200 - OK	IPAddressClaimList schema
401 - Unauthorized	Empty

33.2.2. /apis/ipam.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/ipaddressclaims

HTTP method

DELETE

Description

delete collection of IPAddressClaim

Table 33.2. HTTP responses

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

HTTP method**GET****Description**

list objects of kind IPAddressClaim

Table 33.3. HTTP responses

HTTP code	Reponse body
200 - OK	IPAddressClaimList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an IPAddressClaim

Table 33.4. 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

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 33.5. Body parameters

Parameter	Type	Description
body	IPAddressClaim schema	

Table 33.6. HTTP responses

HTTP code	Response body
200 - OK	IPAddressClaim schema
201 - Created	IPAddressClaim schema
202 - Accepted	IPAddressClaim schema
401 - Unauthorized	Empty

33.2.3. /apis/ipam.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/ipaddressclaims/{name}

Table 33.7. Global path parameters

Parameter	Type	Description
name	string	name of the IPAddressClaim

HTTP method**DELETE****Description**

delete an IPAddressClaim

Table 33.8. 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 33.9. HTTP responses

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

HTTP method**GET****Description**

read the specified IPAddressClaim

Table 33.10. HTTP responses

HTTP code	Reponse body
200 - OK	IPAddressClaim schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified IPAddressClaim

Table 33.11. 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 33.12. HTTP responses

HTTP code	Response body
200 - OK	IPAddressClaim schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified IPAddressClaim

Table 33.13. 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 33.14. Body parameters

Parameter	Type	Description
body	IPAddressClaim schema	

Table 33.15. HTTP responses

HTTP code	Response body
200 - OK	IPAddressClaim schema
201 - Created	IPAddressClaim schema
401 - Unauthorized	Empty

33.2.4. `/apis/ipam.cluster.x-k8s.io/v1beta1/namespaces/{namespace}/ipaddressclaims/{name}/status`

Table 33.16. Global path parameters

Parameter	Type	Description
name	string	name of the IPAddressClaim

HTTP method**GET****Description**

read status of the specified IPAddressClaim

Table 33.17. HTTP responses

HTTP code	Reponse body
200 - OK	IPAddressClaim schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update status of the specified IPAddressClaim

Table 33.18. 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 33.19. HTTP responses

HTTP code	Response body
200 - OK	IPAddressClaim schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified IPAddressClaim

Table 33.20. 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 33.21. Body parameters

Parameter	Type	Description
body	IPAddressClaim schema	

Table 33.22. HTTP responses

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

CHAPTER 34. USERDEFINEDNETWORK [K8S.OVN.ORG/V1]

Description

UserDefinedNetwork describe network request for a Namespace.

Type

object

Required

- **spec**

34.1. 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	UserDefinedNetworkSpec defines the desired state of UserDefinedNetworkSpec.

Property	Type	Description
status	object	UserDefinedNetworkStatus contains the observed status of the UserDefinedNetwork.

34.1.1. .spec

Description

UserDefinedNetworkSpec defines the desired state of UserDefinedNetworkSpec.

Type

object

Required

- **topology**

Property	Type	Description
layer2	object	Layer2 is the Layer2 topology configuration.
layer3	object	Layer3 is the Layer3 topology configuration.
topology	string	Topology describes network configuration. Allowed values are "Layer3", "Layer2". Layer3 topology creates a layer 2 segment per node, each with a different subnet. Layer 3 routing is used to interconnect node subnets. Layer2 topology creates one logical switch shared by all nodes.

34.1.2. .spec.layer2

Description

Layer2 is the Layer2 topology configuration.

Type

object

Required

- **role**

Property	Type	Description
ipam	object	IPAM section contains IPAM-related configuration for the network.
joinSubnets	array (string)	JoinSubnets are used inside the OVN network topology. Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed. This field is only allowed for "Primary" network. It is not recommended to set this field without explicit need and understanding of the OVN network topology. When omitted, the platform will choose a reasonable default which is subject to change over time.
mtu	integer	MTU is the maximum transmission unit for a network. MTU is optional, if not provided, the globally configured value in OVN-Kubernetes (defaults to 1400) is used for the network.
role	string	Role describes the network role in the pod. Allowed value is "Secondary". Secondary network is only assigned to pods that use k8s.v1.cni.cncf.io/networks annotation to select given network.
subnets	array (string)	Subnets are used for the pod network across the cluster. Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed. The format should match standard CIDR notation (for example, "10.128.0.0/16"). This field must be omitted if ipam.mode is Disabled.

34.1.3. .spec.layer2.ipam

Description

IPAM section contains IPAM-related configuration for the network.

Type

object

Property	Type	Description
lifecycle	string	Lifecycle controls IP addresses management lifecycle. The only allowed value is Persistent. When set, OVN Kubernetes assigned IP addresses will be persisted in an ipamclaims.k8s.cni.cncf.io object. These IP addresses will be reused by other pods if requested. Only supported when mode is Enabled.
mode	string	Mode controls how much of the IP configuration will be managed by OVN. Enabled means OVN-Kubernetes will apply IP configuration to the SDN infrastructure and it will also assign IPs from the selected subnet to the individual pods. Disabled means OVN-Kubernetes will only assign MAC addresses and provide layer 2 communication, letting users configure IP addresses for the pods. Disabled is only available for Secondary networks. By disabling IPAM, any Kubernetes features that rely on selecting pods by IP will no longer function (such as network policy, services, etc). Additionally, IP port security will also be disabled for interfaces attached to this network. Defaults to Enabled.

34.1.4. .spec.layer3**Description**

Layer3 is the Layer3 topology configuration.

Type

object

Required

- **role**
- **subnets**

Property	Type	Description
joinSubnets	array (string)	JoinSubnets are used inside the OVN network topology. Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed. This field is only allowed for "Primary" network. It is not recommended to set this field without explicit need and understanding of the OVN network topology. When omitted, the platform will choose a reasonable default which is subject to change over time.
mtu	integer	MTU is the maximum transmission unit for a network. MTU is optional, if not provided, the globally configured value in OVN-Kubernetes (defaults to 1400) is used for the network.
role	string	Role describes the network role in the pod. Allowed values are "Primary" and "Secondary". Primary network is automatically assigned to every pod created in the same namespace. Secondary network is only assigned to pods that use k8s.v1.cni.cncf.io/networks annotation to select given network.
subnets	array	Subnets are used for the pod network across the cluster. Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed. Given subnet is split into smaller subnets for every node.
subnets[]	object	

Property	Type	Description

34.1.5. .spec.layer3.subnets

Description

Subnets are used for the pod network across the cluster.

Dual-stack clusters may set 2 subnets (one for each IP family), otherwise only 1 subnet is allowed.

Given subnet is split into smaller subnets for every node.

Type

array

34.1.6. .spec.layer3.subnets[]

Description

Type

object

Required

- **cidr**

Property	Type	Description
cidr	string	CIDR specifies L3Subnet, which is split into smaller subnets for every node.
hostSubnet	integer	HostSubnet specifies the subnet size for every node. When not set, it will be assigned automatically.

34.1.7. .status

Description

UserDefinedNetworkStatus contains the observed status of the UserDefinedNetwork.

Type

object

Property	Type	Description
conditions	array	

Property	Type	Description
conditions[]	object	Condition contains details for one aspect of the current state of this API Resource.

34.1.8. .status.conditions

Description

Type

array

34.1.9. .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.

Property	Type	Description
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.

34.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/k8s.ovn.org/v1/userdefinednetworks**
 - **GET**: list objects of kind UserDefinedNetwork
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/userdefinednetworks**
 - **DELETE**: delete collection of UserDefinedNetwork
 - **GET**: list objects of kind UserDefinedNetwork
 - **POST**: create an UserDefinedNetwork
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/userdefinednetworks/{name}**
 - **DELETE**: delete an UserDefinedNetwork
 - **GET**: read the specified UserDefinedNetwork
 - **PATCH**: partially update the specified UserDefinedNetwork

- **PUT**: replace the specified UserDefinedNetwork
- **/apis/k8s.ovn.org/v1/namespaces/{namespace}/userdefinednetworks/{name}/status**
 - **GET**: read status of the specified UserDefinedNetwork
 - **PATCH**: partially update status of the specified UserDefinedNetwork
 - **PUT**: replace status of the specified UserDefinedNetwork

34.2.1. /apis/k8s.ovn.org/v1/userdefinednetworks

HTTP method

GET

Description

list objects of kind UserDefinedNetwork

Table 34.1. HTTP responses

HTTP code	Reponse body
200 - OK	UserDefinedNetworkList schema
401 - Unauthorized	Empty

34.2.2. /apis/k8s.ovn.org/v1/namespaces/{namespace}/userdefinednetworks

HTTP method

DELETE

Description

delete collection of UserDefinedNetwork

Table 34.2. HTTP responses

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

HTTP method

GET

Description

list objects of kind UserDefinedNetwork

Table 34.3. HTTP responses

HTTP code	Reponse body
200 - OK	UserDefinedNetworkList schema
401 - Unauthorized	Empty

HTTP method**POST****Description**

create an UserDefinedNetwork

Table 34.4. 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 34.5. Body parameters

Parameter	Type	Description
body	UserDefinedNetwork schema	

Table 34.6. HTTP responses

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

34.2.3. /apis/k8s.ovn.org/v1/namespaces/{namespace}/userdefinednetworks/{name}

Table 34.7. Global path parameters

Parameter	Type	Description
name	string	name of the UserDefinedNetwork

HTTP method

DELETE

Description

delete an UserDefinedNetwork

Table 34.8. 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 34.9. HTTP responses

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

HTTP method

GET

Description

read the specified UserDefinedNetwork

Table 34.10. HTTP responses

HTTP code	Response body
200 - OK	UserDefinedNetwork schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified UserDefinedNetwork

Table 34.11. 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 34.12. HTTP responses

HTTP code	Response body
200 - OK	UserDefinedNetwork schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace the specified UserDefinedNetwork

Table 34.13. 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 34.14. Body parameters

Parameter	Type	Description
body	UserDefinedNetwork schema	

Table 34.15. HTTP responses

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

34.2.4. /apis/k8s.ovn.org/v1/namespaces/{namespace}/userdefinednetworks/{name}

Table 34.16. Global path parameters

Parameter	Type	Description
name	string	name of the UserDefinedNetwork

HTTP method

GET

Description

read status of the specified UserDefinedNetwork

Table 34.17. HTTP responses

HTTP code	Reponse body
200 - OK	UserDefinedNetwork schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified UserDefinedNetwork

Table 34.18. 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 34.19. HTTP responses

HTTP code	Response body
200 - OK	UserDefinedNetwork schema
401 - Unauthorized	Empty

HTTP method**PUT****Description**

replace status of the specified UserDefinedNetwork

Table 34.20. 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 34.21. Body parameters

Parameter	Type	Description
body	UserDefinedNetwork schema	

Table 34.22. HTTP responses

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