PostgreSQL APIs

Postgresqls

PostgresBackup

PostgresRe

PostgreSQLs

`acid.zalan.do` group

Postgresql defines a PostgreSQL cluster managed by the Zalando Postgres Operator. It allows configuring various aspects of PostgreSQL including replication, backups, resource allocation, and more. Postgresql defines the desired state and status of a PostgreSQL cluster managed by the Zalando Postgres Operator.

`v1` version

▼ spec `object` required

Spec defines the desired state of the PostgreSQL cluster including configuration, resources, replication settings, and more.

▼ additionalVolumes `[]object`

AdditionalVolume defines additional volumes that can be mounted to PostgreSQL pods

▼ mountPath `string` required

Path where the volume will be mounted in the container

▼ name `string` required

Name of the volume, must be unique within the pod

▼ subPath `string`

SubPath within the volume to mount (optional)

▼ targetContainers `[]string` required

List of container names that should mount this volume

▼ volumeSource `object` required

Kubernetes VolumeSource defining the volume type and configuration

▼ awsElasticBlockStore `object`

awsElasticBlockStore represents an AWS Disk resource that is attached to a kubelet's host machine and then exposed to the pod. More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore ↗

▼ fsType `string`

fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore ↗ TODO: how do we prevent errors in the filesystem from compromising the machine

▼ partition `integer`

partition is the partition in the volume that you want to mount. If omitted, the default is to mount by volume name. Examples: For volume /dev/sda1, you specify the partition as "1". Similarly, the volume partition for /dev/sda is "0" (or you can leave the property empty).

▼ readOnly `boolean`

readOnly value true will force the readOnly setting in VolumeMounts. More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore ↗

▼ **volumeID** `string` required

volumeID is unique ID of the persistent disk resource in AWS (Amazon EBS volume). More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore ↗

▼ **azureDisk** `object`

azureDisk represents an Azure Data Disk mount on the host and bind mount to the pod.

▼ **cachingMode** `string`

cachingMode is the Host Caching mode: None, Read Only, Read Write.

▼ **diskName** `string` required

diskName is the Name of the data disk in the blob storage

▼ **diskURI** `string` required

diskURI is the URI of data disk in the blob storage

▼ **fsType** `string`

fsType is Filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.

▼ **kind** `string`

kind expected values are Shared: multiple blob disks per storage account Dedicated: single blob disk per storage account Managed: azure managed data disk (only in managed availability set). defaults to shared

▼ **readOnly** `boolean`

readOnly Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **azureFile** `object`

azureFile represents an Azure File Service mount on the host and bind mount to the pod.

▼ **readOnly** `boolean`

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **secretName** `string` required

secretName is the name of secret that contains Azure Storage Account Name and Key

▼ **shareName** `string` required

shareName is the azure share Name

▼ cephfs `object`

cephFS represents a Ceph FS mount on the host that shares a pod's lifetime

▼ monitors `[]string` required

monitors is Required: Monitors is a collection of Ceph monitors More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it↗

▼ path `string`

path is Optional: Used as the mounted root, rather than the full Ceph tree, default is /

▼ readOnly `boolean`

readOnly is Optional: Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts. More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it↗

▼ secretFile `string`

secretFile is Optional: SecretFile is the path to key ring for User, default is /etc/ceph/user.secret More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it↗

▼ **secretRef** `object`

secretRef is Optional: SecretRef is reference to the authentication secret for User, default is empty. More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it↗

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ **user** `string`

user is optional: User is the rados user name, default is admin More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it↗

▼ **cinder** `object`

cinder represents a cinder volume attached and mounted on kubelets host machine. More info: https://examples.k8s.io/mysql-cinder-pd/README.md↗

▼ **fsType** `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://examples.k8s.io/mysql-cinder-pd/README.md↗

▼ **readOnly** `boolean`

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts. More info: https://examples.k8s.io/mysql-cinder-pd/README.md ↗

▼ **secretRef** `object`

secretRef is optional: points to a secret object containing parameters used to connect to OpenStack.

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ **volumeID** `string` required

volumeID used to identify the volume in cinder. More info: https://examples.k8s.io/mysql-cinder-pd/README.md ↗

▼ **configMap** `object`

configMap represents a configMap that should populate this volume

▼ **defaultMode** `integer`

defaultMode is optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode

bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ **items** `[]object`

Maps a string key to a path within a volume.

▼ **key** `string` required

key is the key to project.

▼ **mode** `integer`

mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ **path** `string` required

path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names TODO: Add other useful fields. apiVersion, kind, uid?

▼ **optional** `boolean`

optional specify whether the ConfigMap or its keys must be defined

▼ **csi** `object`

csi (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers (Beta feature).

▼ **driver** `string` required

driver is the name of the CSI driver that handles this volume. Consult with your admin for the correct name as registered in the cluster.

▼ **fsType** `string`

fsType to mount. Ex. "ext4", "xfs", "ntfs". If not provided, the empty value is passed to the associated CSI driver which will determine the default filesystem to apply.

▼ **nodePublishSecretRef** `object`

nodePublishSecretRef is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI NodePublishVolume and NodeUnpublishVolume calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ **readOnly** `boolean`

readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).

▼ **volumeAttributes** `object`

volumeAttributes stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

▼ **downwardAPI** `object`

downwardAPI represents downward API about the pod that should populate this volume

▼ **defaultMode** `integer`

Optional: mode bits to use on created files by default. Must be a Optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ items `[]object`

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ fieldRef `object`

Required: Selects a field of the pod: only annotations, labels, name and namespace are supported.

▼ apiVersion `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath `string` required

Path of the field to select in the specified API version.

▼ mode `integer`

Optional: mode bits used to set permissions on this file, must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ path `string` required

Required: Path is the relative path name of the file to be created. Must not be absolute or contain the '..' path. Must be utf-8 encoded. The first item of the relative path must not start with '..'

▼ **resourceFieldRef** `object`

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **emptyDir** `object`

emptyDir represents a temporary directory that shares a pod's lifetime.
More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir ↗

▼ **medium** `string`

medium represents what type of storage medium should back this directory. The default is "" which means to use the node's default medium. Must be an empty string (default) or Memory. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir ↗

▼ sizeLimit

sizeLimit is the total amount of local storage required for this EmptyDir volume. The size limit is also applicable for memory medium. The maximum usage on memory medium EmptyDir would be the minimum value between the SizeLimit specified here and the sum of memory limits of all containers in a pod. The default is nil which means that the limit is undefined. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir ↗

▼ ephemeral `object`

ephemeral represents a volume that is handled by a cluster storage driver. The volume's lifecycle is tied to the pod that defines it - it will be created before the pod starts, and deleted when the pod is removed. Use this if: a) the volume is only needed while the pod runs, b) features of normal volumes like restoring from snapshot or capacity tracking are needed, c) the storage driver is specified through a storage class, and d) the storage driver supports dynamic volume provisioning through a PersistentVolumeClaim (see EphemeralVolumeSource for more information on the connection between this volume type and PersistentVolumeClaim). Use PersistentVolumeClaim or one of the vendor-specific APIs for volumes that persist for longer than the lifecycle of an individual pod. Use CSI for light-weight local ephemeral volumes if the CSI driver is meant to be used that way - see the documentation of the driver for more information. A pod can use both types of ephemeral volumes and persistent volumes at the same time.

▼ volumeClaimTemplate `object`

Will be used to create a stand-alone PVC to provision the volume. The pod in which this EphemeralVolumeSource is embedded will be the owner of the PVC, i.e. the PVC will be deleted together with the pod. The name of the PVC will be `<pod name>-<volume name>` where `<volume name>` is the name from the `PodSpec.Volumes` array entry. Pod validation will reject the pod if the concatenated name is not valid for a PVC (for example, too long). An existing PVC with that name that is not owned by the pod will *not* be used for the pod to avoid using an unrelated volume by mistake. Starting the pod is then blocked until the unrelated PVC is removed. If such a pre-created PVC is meant to be used by the pod, the PVC has to updated with an owner reference to the pod once the pod exists. Normally this should not be necessary, but it may be useful when manually reconstructing a broken cluster. This field is read-only and no changes will be made by Kubernetes to the PVC after it has been created. Required, must not be nil.

▼ **metadata** `object`

May contain labels and annotations that will be copied into the PVC when creating it. No other fields are allowed and will be rejected during validation.

▼ **spec** `object` required

The specification for the PersistentVolumeClaim. The entire content is copied unchanged into the PVC that gets created from this template. The same fields as in a PersistentVolumeClaim are also valid here.

▼ **accessModes** `[]string`

accessModes contains the desired access modes the volume should have. More info:

https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1↗

▼ dataSource `object`

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

▼ 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` required

Kind is the type of resource being referenced

▼ name `string` required

Name is the name of resource being referenced

▼ dataSourceRef `object`

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the

CrossNamespaceVolumeDataSource feature gate to be enabled.

▼ **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` required

Kind is the type of resource being referenced

▼ **name** `string` required

Name is the name of resource being referenced

▼ **namespace** `string`

Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

▼ **resources** `object`

resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources ↗

▼ **limits** `object`

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

▼ **requests** `object`

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

▼ **selector** `object`

selector is a label query over volumes to consider for binding.

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ **values** `[]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.

▼ **matchLabels** `object`

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.

▼ storageClassName `string`

storageClassName is the name of the StorageClass required by the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1 ↗

▼ volumeAttributesClassName `string`

volumeAttributesClassName may be used to set the VolumeAttributesClass used by this claim. If specified, the CSI driver will create or update the volume with the attributes defined in the corresponding VolumeAttributesClass. This has a different purpose than storageClassName, it can be changed after the claim is created. An empty string value means that no VolumeAttributesClass will be applied to the claim but it's not allowed to reset this field to empty string once it is set. If unspecified and the PersistentVolumeClaim is unbound, the default VolumeAttributesClass will be set by the persistentvolume controller if it exists. If the resource referred to by volumeAttributesClass does not exist, this PersistentVolumeClaim will be set to a Pending state, as reflected by the modifyVolumeStatus field, until such as a resource exists. More info:

https://kubernetes.io/docs/concepts/storage/persistent-volumes#volumeattributesclass ↗ (Alpha)
Using this field requires the VolumeAttributesClass feature gate to be enabled.

▼ **volumeMode** `string`

volumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.

▼ **volumeName** `string`

volumeName is the binding reference to the PersistentVolume backing this claim.

▼ **fc** `object`

fc represents a Fibre Channel resource that is attached to a kubelet's host machine and then exposed to the pod.

▼ **fsType** `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. TODO: how do we prevent errors in the filesystem from compromising the machine

▼ **lun** `integer`

lun is Optional: FC target lun number

▼ **readOnly** `boolean`

readOnly is Optional: Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **targetWWNs** `[]string`

targetWWNs is Optional: FC target worldwide names (WWNs)

▼ **wwids** `[]string`

wwids Optional: FC volume world wide identifiers (wwids) Either wwids or combination of targetWWNs and lun must be set, but not both simultaneously.

▼ **flexVolume** `object`

flexVolume represents a generic volume resource that is provisioned/attached using an exec based plugin.

▼ **driver** `string` required

driver is the name of the driver to use for this volume.

▼ **fsType** `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". The default filesystem depends on FlexVolume script.

▼ **options** `object`

options is Optional: this field holds extra command options if any.

▼ readOnly `boolean`

readOnly is Optional: defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ secretRef `object`

secretRef is Optional: secretRef is reference to the secret object containing sensitive information to pass to the plugin scripts. This may be empty if no secret object is specified. If the secret object contains more than one secret, all secrets are passed to the plugin scripts.

▼ name `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ flocker `object`

flocker represents a Flocker volume attached to a kubelet's host machine. This depends on the Flocker control service being running

▼ datasetName `string`

datasetName is Name of the dataset stored as metadata -> name on the dataset for Flocker should be considered as deprecated

▼ datasetUUID `string`

datasetUUID is the UUID of the dataset. This is unique identifier of a Flocker dataset

▼ gcePersistentDisk `object`

gcePersistentDisk represents a GCE Disk resource that is attached to a kubelet's host machine and then exposed to the pod. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk ↗

▼ fsType `string`

fsType is filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk ↗ TODO: how do we prevent errors in the filesystem from compromising the machine

▼ partition `integer`

partition is the partition in the volume that you want to mount. If omitted, the default is to mount by volume name. Examples: For volume /dev/sda1, you specify the partition as "1". Similarly, the volume partition for /dev/sda is "0" (or you can leave the property empty). More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk ↗

▼ pdName `string` required

pdName is unique name of the PD resource in GCE. Used to identify the disk in GCE. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk↗

▼ **readOnly** `boolean`

readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk↗

▼ **gitRepo** `object`

gitRepo represents a git repository at a particular revision. DEPRECATED: GitRepo is deprecated. To provision a container with a git repo, mount an EmptyDir into an InitContainer that clones the repo using git, then mount the EmptyDir into the Pod's container.

▼ **directory** `string`

directory is the target directory name. Must not contain or start with '..'. If '.' is supplied, the volume directory will be the git repository. Otherwise, if specified, the volume will contain the git repository in the subdirectory with the given name.

▼ **repository** `string` required

repository is the URL

▼ **revision** `string`

revision is the commit hash for the specified revision.

▼ glusterfs `object`

glusterfs represents a Glusterfs mount on the host that shares a pod's lifetime. More info: https://examples.k8s.io/volumes/glusterfs/README.md ↗

▼ endpoints `string` required

endpoints is the endpoint name that details Glusterfs topology. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod ↗

▼ path `string` required

path is the Glusterfs volume path. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod ↗

▼ readOnly `boolean`

readOnly here will force the Glusterfs volume to be mounted with read-only permissions. Defaults to false. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod ↗

▼ hostPath `object`

hostPath represents a pre-existing file or directory on the host machine that is directly exposed to the container. This is generally used for system

agents or other privileged things that are allowed to see the host machine. Most containers will NOT need this. More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath --- TODO(jonesdl) We need to restrict who can use host directory mounts and who can/can not mount host directories as read/write.

▼ **path** `string` required

path of the directory on the host. If the path is a symlink, it will follow the link to the real path. More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath

▼ **type** `string`

type for HostPath Volume Defaults to "" More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath

▼ **iscsi** `object`

iscsi represents an ISCSI Disk resource that is attached to a kubelet's host machine and then exposed to the pod. More info: https://examples.k8s.io/volumes/iscsi/README.md

▼ **chapAuthDiscovery** `boolean`

chapAuthDiscovery defines whether support iSCSI Discovery CHAP authentication

▼ **chapAuthSession** `boolean`

chapAuthSession defines whether support iSCSI Session CHAP authentication

▼ fsType `string`

fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#iscsi ↗ TODO: how do we prevent errors in the filesystem from compromising the machine

▼ initiatorName `string`

initiatorName is the custom iSCSI Initiator Name. If initiatorName is specified with iscsiInterface simultaneously, new iSCSI interface : will be created for the connection.

▼ iqn `string` required

iqn is the target iSCSI Qualified Name.

▼ iscsiInterface `string`

iscsiInterface is the interface Name that uses an iSCSI transport. Defaults to 'default' (tcp).

▼ lun `integer` required

lun represents iSCSI Target Lun number.

▼ portals `[]string`

portals is the iSCSI Target Portal List. The portal is either an IP or ip_addr:port if the port is other than default (typically TCP ports 860 and 3260).

▼ **readOnly** `boolean`

readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false.

▼ **secretRef** `object`

secretRef is the CHAP Secret for iSCSI target and initiator authentication

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ **targetPortal** `string` required

targetPortal is iSCSI Target Portal. The Portal is either an IP or ip_addr:port if the port is other than default (typically TCP ports 860 and 3260).

▼ **nfs** `object`

nfs represents an NFS mount on the host that shares a pod's lifetime More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs ↗

▼ **path** `string` required

path that is exported by the NFS server. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs ↗

▼ **readOnly** `boolean`

readOnly here will force the NFS export to be mounted with read-only permissions. Defaults to false. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs ↗

▼ **server** `string` required

server is the hostname or IP address of the NFS server. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs ↗

▼ **persistentVolumeClaim** `object`

persistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

▼ **claimName** `string` required

claimName is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗

▼ **readOnly** `boolean`

readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

▼ photonPersistentDisk `object`

photonPersistentDisk represents a PhotonController persistent disk attached and mounted on kubelets host machine

▼ fsType `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.

▼ pdID `string` required

pdID is the ID that identifies Photon Controller persistent disk

▼ portworxVolume `object`

portworxVolume represents a portworx volume attached and mounted on kubelets host machine

▼ fsType `string`

fSType represents the filesystem type to mount Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs". Implicitly inferred to be "ext4" if unspecified.

▼ readOnly `boolean`

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ volumeID `string` required

volumeID uniquely identifies a Portworx volume

▼ projected `object`

projected items for all in one resources secrets, configmaps, and downward API

▼ defaultMode `integer`

defaultMode are the mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ sources `[]object`

Projection that may be projected along with other supported volume types

▼ clusterTrustBundle `object`

ClusterTrustBundle allows a pod to access the `.spec.trustBundle` field of ClusterTrustBundle objects in an auto-updating file. Alpha, gated by the ClusterTrustBundleProjection feature gate. ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label

selector. Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem. Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.

▼ **labelSelector** `object`

Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ **values** `[]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.

▼ **matchLabels** `object`

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.

▼ **name** `string`

Select a single ClusterTrustBundle by object name. Mutually-exclusive with signerName and labelSelector.

▼ **optional** `boolean`

If true, don't block pod startup if the referenced ClusterTrustBundle(s) aren't available. If using name, then the named ClusterTrustBundle is allowed not to exist. If using signerName, then the combination of signerName and labelSelector is allowed to match zero ClusterTrustBundles.

▼ **path** `string` required

Relative path from the volume root to write the bundle.

▼ **signerName** `string`

Select all ClusterTrustBundles that match this signer name. Mutually-exclusive with name. The contents of all selected ClusterTrustBundles will be unified and deduplicated.

▼ **configMap** `object`

configMap information about the configMap data to project

▼ **items** `[]object`

Maps a string key to a path within a volume.

▼ **key** `string` required

key is the key to project.

▼ **mode** `integer`

mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the

file mode, like fsGroup, and the result can be other mode bits set.

▼ **path** `string` required

path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names TODO: Add other useful fields. apiVersion, kind, uid?

▼ **optional** `boolean`

optional specify whether the ConfigMap or its keys must be defined

▼ **downwardAPI** `object`

downwardAPI information about the downwardAPI data to project

▼ **items** `[]object`

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ **fieldRef** `object`

Required: Selects a field of the pod: only annotations, labels, name and namespace are supported.

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string`

required

Path of the field to select in the specified API version.

▼ **mode** `integer`

Optional: mode bits used to set permissions on this file, must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ **path** `string` required

Required: Path is the relative path name of the file to be created. Must not be absolute

or contain the '..' path. Must be utf-8 encoded. The first item of the relative path must not start with '..'

▼ **resourceFieldRef** `object`

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string`

required

Required: resource to select

▼ **secret** `object`

secret information about the secret data to project

▼ **items** `[]object`

Maps a string key to a path within a volume.

▼ **key** `string` required

key is the key to project.

▼ **mode** `integer`

mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ **path** `string` required

path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ **optional** `boolean`

optional field specify whether the Secret or its key must be defined

▼ serviceAccountToken `object`

serviceAccountToken is information about the serviceAccountToken data to project

▼ audience `string`

audience is the intended audience of the token. A recipient of a token must identify itself with an identifier specified in the audience of the token, and otherwise should reject the token. The audience defaults to the identifier of the apiserver.

▼ expirationSeconds `integer`

expirationSeconds is the requested duration of validity of the service account token. As the token approaches expiration, the kubelet volume plugin will proactively rotate the service account token. The kubelet will start trying to rotate the token if the token is older than 80 percent of its time to live or if the token is older than 24 hours.Defaults to 1 hour and must be at least 10 minutes.

▼ path `string` required

path is the path relative to the mount point of the file to project the token into.

▼ **quobyte** `object`

quobyte represents a Quobyte mount on the host that shares a pod's lifetime

▼ **group** `string`

group to map volume access to Default is no group

▼ **readOnly** `boolean`

readOnly here will force the Quobyte volume to be mounted with read-only permissions. Defaults to false.

▼ **registry** `string` required

registry represents a single or multiple Quobyte Registry services specified as a string as host:port pair (multiple entries are separated with commas) which acts as the central registry for volumes

▼ **tenant** `string`

tenant owning the given Quobyte volume in the Backend Used with dynamically provisioned Quobyte volumes, value is set by the plugin

▼ **user** `string`

user to map volume access to Defaults to serivceaccount user

▼ **volume** `string` required

volume is a string that references an already created Quobyte volume by name.

▼ rbd `object`

rbd represents a Rados Block Device mount on the host that shares a pod's lifetime. More info: https://examples.k8s.io/volumes/rbd/README.md ↗

▼ fsType `string`

fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#rbd ↗ TODO: how do we prevent errors in the filesystem from compromising the machine

▼ image `string` required

image is the rados image name. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it ↗

▼ keyring `string`

keyring is the path to key ring for RBDUser. Default is /etc/ceph/keyring. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it ↗

▼ monitors `[]string` required

monitors is a collection of Ceph monitors. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it ↗

▼ **pool** `string`

pool is the rados pool name. Default is rbd. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it ↗

▼ **readOnly** `boolean`

readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it ↗

▼ **secretRef** `object`

secretRef is name of the authentication secret for RBDUser. If provided overrides keyring. Default is nil. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it ↗

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ **user** `string`

user is the rados user name. Default is admin. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it ↗

▼ scaleIO `object`

scaleIO represents a ScaleIO persistent volume attached and mounted on Kubernetes nodes.

▼ fsType `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Default is "xfs".

▼ gateway `string` required

gateway is the host address of the ScaleIO API Gateway.

▼ protectionDomain `string`

protectionDomain is the name of the ScaleIO Protection Domain for the configured storage.

▼ readOnly `boolean`

readOnly Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ secretRef `object` required

secretRef references to the secret for ScaleIO user and other sensitive information. If this is not provided, Login operation will fail.

▼ name `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names TODO: Add other useful fields. apiVersion, kind, uid?

▼ sslEnabled `boolean`

sslEnabled Flag enable/disable SSL communication with Gateway, default false

▼ storageMode `string`

storageMode indicates whether the storage for a volume should be ThickProvisioned or ThinProvisioned. Default is ThinProvisioned.

▼ storagePool `string`

storagePool is the ScaleIO Storage Pool associated with the protection domain.

▼ system `string` required

system is the name of the storage system as configured in ScaleIO.

▼ volumeName `string`

volumeName is the name of a volume already created in the ScaleIO system that is associated with this volume source.

▼ secret `object`

secret represents a secret that should populate this volume. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret ↗

▼ **defaultMode** `integer`

defaultMode is Optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ **items** `[]object`

Maps a string key to a path within a volume.

▼ **key** `string` required

key is the key to project.

▼ **mode** `integer`

mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

▼ **path** `string` required

path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

▼ **optional** `boolean`

optional field specify whether the Secret or its keys must be defined

▼ **secretName** `string`

secretName is the name of the secret in the pod's namespace to use. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret ↗

▼ **storageos** `object`

storageOS represents a StorageOS volume attached and mounted on Kubernetes nodes.

▼ **fsType** `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.

▼ **readOnly** `boolean`

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **secretRef** `object`

secretRef specifies the secret to use for obtaining the StorageOS API credentials. If not specified, default values will be attempted.

▼ name `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ volumeName `string`

volumeName is the human-readable name of the StorageOS volume. Volume names are only unique within a namespace.

▼ volumeNamespace `string`

volumeNamespace specifies the scope of the volume within StorageOS. If no namespace is specified then the Pod's namespace will be used. This allows the Kubernetes name scoping to be mirrored within StorageOS for tighter integration. Set VolumeName to any name to override the default behaviour. Set to "default" if you are not using namespaces within StorageOS. Namespaces that do not pre-exist within StorageOS will be created.

▼ vsphereVolume `object`

vsphereVolume represents a vSphere volume attached and mounted on kubelets host machine

▼ fsType `string`

fsType is filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.

▼ **storagePolicyID** `string`

storagePolicyID is the storage Policy Based Management (SPBM) profile ID associated with the StoragePolicyName.

▼ **storagePolicyName** `string`

storagePolicyName is the storage Policy Based Management (SPBM) profile name.

▼ **volumePath** `string` required

volumePath is the path that identifies vSphere volume vmdk

▼ **allowedSourceRanges** `[]string`

AllowedSourceRanges specifies IP ranges allowed to access load balancers

▼ **backup** `object`

Backup defines backup configuration

▼ **resources** `object`

Resources describes requests and limits for the Backup sidecar resouces.

▼ **limits** `object`

ResourceLimits defines the maximum resources allowed for containers

▼ **cpu** `string`

CPU requirement (e.g., "500m" or "1")

▼ **memory** `string`

Memory requirement (e.g., "512Mi" or "2Gi")

▼ **requests** `object`

ResourceRequests defines the minimum resources required for containers

▼ **cpu** `string`

CPU requirement (e.g., "500m" or "1")

▼ **memory** `string`

Memory requirement (e.g., "512Mi" or "2Gi")

▼ **retainDay** `integer`

RetainDay specifies how many days to retain backup files before deletion

▼ **schedule** `string`

Schedule defines the cron schedule for automated backups (e.g. "0 0 * * *" for daily at midnight)

▼ **storage** `object` required

storage defines the configuration for backup storage including the storage name, namespace, and bucket location.

▼ **bucket** `string` required

S3 bucket name

▼ **name** `string` required

Name of the storage configuration to use for backups

▼ **namespace** `string` required

Namespace where the storage configuration is defined

▼ **clone** `object`

Clone defines configuration for cloning from another cluster

▼ **cluster** `string`

ClusterName specifies the name of the source cluster to clone from

▼ **s3_access_key_id** `string`

S3AccessKeyId specifies the access key ID for S3 authentication

▼ **s3_endpoint** `string`

S3Endpoint specifies the S3-compatible storage endpoint

▼ s3_force_path_style `boolean`

S3ForcePathStyle enables path-style S3 URLs (bucket.s3.amazonaws.com)

▼ s3_secret_access_key `string`

S3SecretAccessKey specifies the secret access key for S3 authentication

▼ s3_wal_path `string`

S3WalPath specifies the S3 path containing WAL files for PITR cloning

▼ timestamp `string`

EndTimestamp specifies the point-in-time to clone up to (RFC3339 format)

▼ uid `string`

UID specifies the unique identifier of the source cluster

▼ clusterReplication `object`

ClusterReplication defines cross-cluster replication

▼ bootstrapSecret `string`

BootstrapSecret contains credentials for initial replication setup

▼ enabled `boolean`

Enabled controls whether cross-cluster replication is active

▼ isReplica `boolean`

IsReplica indicates if this cluster should act as a replica

▼ peerHost `string`

PeerHost specifies the hostname or IP of the peer cluster

▼ peerPort `integer`

PeerPort specifies the port number for connecting to the peer cluster

▼ replSvcType `string`

ReplSvcType defines the Kubernetes Service type for replication traffic

▼ syncMode `boolean`

SyncMode enables synchronous replication between clusters

▼ connectionPooler `object`

ConnectionPooler configuration for connection pooling

▼ dockerImage `string`

DockerImage specifies the container image to use for the connection pooler. If not specified, uses the default image configured in the operator.

▼ maxDBConnections `integer`

MaxDBConnections defines the maximum number of database connections that the pooler will maintain to PostgreSQL. Defaults to 60 if not specified.

▼ mode `string`

Mode defines the connection pooling mode. Can be "session" or "transaction". In session mode, connections are held for the duration of a client session. In transaction mode, connections are returned to the pool after each transaction. Defaults to "transaction" if not specified.

▼ numberOfInstances `integer`

NumberOfInstances specifies how many connection pooler instances to run. If not set, defaults to 1 for PgBouncer or 2 for Pgpool-II.

▼ resources `object`

Resources defines CPU and memory requirements for the connection pooler containers. If not specified, uses operator defaults.

▼ limits `object`

ResourceLimits defines the maximum resources allowed for containers

▼ cpu `string`

CPU requirement (e.g., "500m" or "1")

▼ memory `string`

Memory requirement (e.g., "512Mi" or "2Gi")

▼ **requests** `object`

ResourceRequests defines the minimum resources required for containers

▼ **cpu** `string`

CPU requirement (e.g., "500m" or "1")

▼ **memory** `string`

Memory requirement (e.g., "512Mi" or "2Gi")

▼ **schema** `string`

Schema specifies the database schema where connection pooler will operate. Defaults to "pooler" if not specified.

▼ **user** `string`

User specifies the database user that the connection pooler will use. Defaults to "pooler" if not specified.

▼ **databases** `object`

Databases defines databases to be created in the cluster

▼ **dockerImage** `string`

DockerImage specifies the container image to use for PostgreSQL instances

▼ enableConnectionPooler `boolean`

EnableConnectionPooler enables connection pooling for the primary instance

▼ enableExporter `boolean`

EnableExporter enables Prometheus exporter

▼ enableLogicalBackup `boolean`

EnableLogicalBackup enables logical backups for the cluster

▼ enableMasterLoadBalancer `boolean`

EnableMasterLoadBalancer enables load balancer for master instance

▼ enableMasterPoolerLoadBalancer `boolean`

EnableMasterPoolerLoadBalancer enables load balancer for master pooler

▼ enablePgpool2 `boolean`

EnablePgpool2 enables Pgpool-II connection pooling

▼ enableReadinessProbe `boolean`

EnableReadinessProbe enables readiness probes for containers

▼ enableReplicaConnectionPooler `boolean`

EnableReplicaConnectionPooler enables connection pooling for replica instances

▼ enableReplicaLoadBalancer `boolean`

EnableReplicaLoadBalancer enables load balancer for replica instances

▼ enableReplicaPoolerLoadBalancer `boolean`

EnableReplicaPoolerLoadBalancer enables load balancer for replica pooler

▼ enableShmVolume `boolean`

ShmVolume enables shared memory volume for the pod

▼ env `[]object`

EnvVar represents an environment variable present in a Container.

▼ name `string` required

Name of the environment variable. Must be a C_IDENTIFIER.

▼ value `string`

Variable references $(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double $$ are reduced to a single $, which allows for escaping the $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will produce the string literal "$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to "".

▼ valueFrom `object`

Source for the environment variable's value. Cannot be used if value is not empty.

▼ configMapKeyRef `object`

Selects a key of a ConfigMap.

▼ key `string` required

The key to select.

▼ name `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names TODO: Add other useful fields. apiVersion, kind, uid?

▼ optional `boolean`

Specify whether the ConfigMap or its key must be defined

▼ fieldRef `object`

Selects a field of the pod: supports metadata.name, metadata.namespace, `metadata.labels['<KEY>']`, `metadata.annotations['<KEY>']`, spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

▼ apiVersion `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `object`

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

Selects a key of a secret in the pod's namespace

▼ **key** `string` required

The key of the secret to select from. Must be a valid secret key.

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names TODO: Add other useful fields. apiVersion, kind, uid?

▼ **optional** `boolean`

Specify whether the Secret or its key must be defined

▼ **exporter** `object`

Exporter defines Prometheus exporter configuration

▼ **env** `[]object`

EnvVar represents an environment variable present in a Container.

▼ **name** `string` required

Name of the environment variable. Must be a C_IDENTIFIER.

▼ **value** `string`

Variable references $(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double $$ are reduced to a single $,

which allows for escaping the $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will produce the string literal "$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to "".

▼ valueFrom `object`

Source for the environment variable's value. Cannot be used if value is not empty.

▼ configMapKeyRef `object`

Selects a key of a ConfigMap.

▼ key `string` required

The key to select.

▼ name `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ optional `boolean`

Specify whether the ConfigMap or its key must be defined

▼ fieldRef `object`

Selects a field of the pod: supports metadata.name, metadata.namespace, `metadata.labels['<KEY>']`,

`metadata.annotations['<KEY>']`, spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `object`

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

Selects a key of a secret in the pod's namespace

▼ **key** `string` required

The key of the secret to select from. Must be a valid secret key.

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ **optional** `boolean`

Specify whether the Secret or its key must be defined

▼ **init_containers** `[]object`

A single application container that you want to run within a pod.

▼ **args** `[]string`

Arguments to the entrypoint. The container image's CMD is used if this is not provided. Variable references $(VAR_NAME) are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will

be unchanged. Double $$ are reduced to a single $, which allows for escaping the $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will produce the string literal "$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell ↗

▼ command `[]string`

Entrypoint array. Not executed within a shell. The container image's ENTRYPOINT is used if this is not provided. Variable references $(VAR_NAME) are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double $$ are reduced to a single $, which allows for escaping the $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will produce the string literal "$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell ↗

▼ env `[]object`

EnvVar represents an environment variable present in a Container.

▼ name `string` required

Name of the environment variable. Must be a C_IDENTIFIER.

▼ value `string`

Variable references $(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double $$ are reduced to a single $, which allows for escaping the $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will produce the string literal "$(VAR_NAME)". Escaped references will

never be expanded, regardless of whether the variable exists or not. Defaults to "".

▼ valueFrom `object`

Source for the environment variable's value. Cannot be used if value is not empty.

▼ configMapKeyRef `object`

Selects a key of a ConfigMap.

▼ key `string` required

The key to select.

▼ name `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names TODO: Add other useful fields. apiVersion, kind, uid?

▼ optional `boolean`

Specify whether the ConfigMap or its key must be defined

▼ fieldRef `object`

Selects a field of the pod: supports metadata.name, metadata.namespace, `metadata.labels['<KEY>']`, `metadata.annotations['<KEY>']`, spec.nodeName,

spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `object`

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

Selects a key of a secret in the pod's namespace

▼ **key** `string` required

The key of the secret to select from. Must be a valid secret key.

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ **optional** `boolean`

Specify whether the Secret or its key must be defined

▼ **envFrom** `[]object`

EnvFromSource represents the source of a set of ConfigMaps

▼ **configMapRef** `object`

The ConfigMap to select from

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-

objects/names/#names↗ TODO: Add other useful fields.

apiVersion, kind, uid?

▼ **optional** `boolean`

Specify whether the ConfigMap must be defined

▼ **prefix** `string`

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ **secretRef** `object`

The Secret to select from

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names↗ TODO: Add other useful fields.

apiVersion, kind, uid?

▼ **optional** `boolean`

Specify whether the Secret must be defined

▼ **image** `string`

Container image name. More info: https://kubernetes.io/docs/concepts/containers/images↗ This field is optional to

allow higher level config management to default or override container images in workload controllers like Deployments and StatefulSets.

▼ imagePullPolicy `string`

Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images#updating-images ↗

▼ lifecycle `object`

Actions that the management system should take in response to container lifecycle events. Cannot be updated.

▼ postStart `object`

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to

explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ scheme `string`

Scheme to use for connecting to the host. Defaults to HTTP.

▼ sleep `object`

Sleep represents the duration that the container should sleep before being terminated.

▼ seconds `integer` required

Seconds is the number of seconds to sleep.

▼ tcpSocket `object`

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

▼ host `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be

an IANA_SVC_NAME.

▼ preStop `object`

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ scheme `string`

Scheme to use for connecting to the host. Defaults to HTTP.

▼ **sleep** `object`

Sleep represents the duration that the container should sleep before being terminated.

▼ **seconds** `integer` required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** `object`

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **livenessProbe** `object`

Periodic probe of container liveness. Container will be restarted if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ **exec** `object`

Exec specifies the action to take.

▼ **command** `[]string`

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

▼ **failureThreshold** `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** `object`

GRPC specifies an action involving a GRPC port.

▼ **port** `integer` required

Port number of the gRPC service. Number must be in the range 1 to 65535.

▼ **service** `string`

Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ **port** required

Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **scheme** `string`

Scheme to use for connecting to the host. Defaults to HTTP.

▼ **initialDelaySeconds** `integer`

Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. Set this value longer than the expected cleanup time for your process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **timeoutSeconds** `integer`

Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ **name** `string` required

Name of the container specified as a DNS_LABEL. Each container in a pod must have a unique name (DNS_LABEL). Cannot be updated.

▼ ports `[]object`

ContainerPort represents a network port in a single container.

▼ containerPort `integer` required

Number of port to expose on the pod's IP address. This must be a valid port number, 0 < x < 65536.

▼ hostIP `string`

What host IP to bind the external port to.

▼ hostPort `integer`

Number of port to expose on the host. If specified, this must be a valid port number, 0 < x < 65536. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

▼ name `string`

If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.

▼ protocol `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ readinessProbe `object`

Periodic probe of container service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

▼ failureThreshold `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc `object`

GRPC specifies an action involving a GRPC port.

▼ port `integer` required

Port number of the gRPC service. Number must be in the range 1 to 65535.

▼ **service** `string`

Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

▼ **httpGet** `object`

HTTPGet specifies the http request to perform.

▼ **host** `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ **value** `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** required

Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **scheme** `string`

Scheme to use for connecting to the host. Defaults to HTTP.

▼ **initialDelaySeconds** `integer`

Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. Set this value longer than the expected cleanup time for your process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **timeoutSeconds** `integer`

Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ resizePolicy `[]object`

ContainerResizePolicy represents resource resize policy for the container.

▼ resourceName `string` required

Name of the resource to which this resource resize policy applies.

Supported values: cpu, memory.

▼ restartPolicy `string` required

Restart policy to apply when specified resource is resized. If not specified, it defaults to NotRequired.

▼ resources `object`

Compute Resources required by this container. Cannot be updated. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

▼ claims `[]object`

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name `string` required

Name must match the name of one entry in pod.spec.resourceClaims of the Pod where this field is used. It makes that resource available inside a container.

▼ limits `object`

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/↗

▼ requests `object`

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/↗

▼ restartPolicy `string`

RestartPolicy defines the restart behavior of individual containers in a pod. This field may only be set for init containers, and the only allowed value is "Always". For non-init containers or when this field is not specified, the restart behavior is defined by the Pod's restart policy and the container type. Setting the RestartPolicy as "Always" for the init container will have the following effect: this init container will be continually restarted on exit until all regular containers have terminated. Once all regular containers have completed, all init containers with restartPolicy "Always" will be shut down. This lifecycle differs from normal init containers and is often referred to as a "sidecar" container. Although this init container still starts in the init container sequence, it does not wait for the container to complete before proceeding to the next init container. Instead, the next init container starts immediately after this init container is started, or after any startupProbe has successfully completed.

▼ securityContext `object`

SecurityContext defines the security options the container should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext.

More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/↗

▼ **allowPrivilegeEscalation** `boolean`

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the no_new_privs flag will be set on the container process. AllowPrivilegeEscalation is true always when the container is: 1) run as Privileged 2) has CAP_SYS_ADMIN Note that this field cannot be set when spec.os.name is windows.

▼ **capabilities** `object`

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when spec.os.name is windows.

▼ **add** `[]string`

Added capabilities

▼ **drop** `[]string`

Removed capabilities

▼ **privileged** `boolean`

Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when spec.os.name is windows.

▼ procMount `string`

procMount denotes the type of proc mount to use for the containers. The default is DefaultProcMount which uses the container runtime defaults for readonly paths and masked paths. This requires the ProcMountType feature flag to be enabled. Note that this field cannot be set when spec.os.name is windows.

▼ readOnlyRootFilesystem `boolean`

Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when spec.os.name is windows.

▼ runAsGroup `integer`

The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

▼ runAsNonRoot `boolean`

Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

▼ runAsUser `integer`

The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in

PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

▼ **seLinuxOptions** `object`

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

▼ **level** `string`

Level is SELinux level label that applies to the container.

▼ **role** `string`

Role is a SELinux role label that applies to the container.

▼ **type** `string`

Type is a SELinux type label that applies to the container.

▼ **user** `string`

User is a SELinux user label that applies to the container.

▼ **seccompProfile** `object`

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.

▼ **localhostProfile** `string`

localhostProfile indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.

▼ **type** `string` required

type indicates which kind of seccomp profile will be applied. Valid options are: Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

▼ **windowsOptions** `object`

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

▼ **gmsaCredentialSpec** `string`

GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa↗) inlines the contents of the GMSA credential spec named by the GMSACredentialSpecName field.

▼ **gmsaCredentialSpecName** `string`

GMSACredentialSpecName is the name of the GMSA credential spec to use.

▼ **hostProcess** `boolean`

HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.

▼ **runAsUserName** `string`

The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

▼ **startupProbe** `object`

StartupProbe indicates that the Pod has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state operation. This cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ **exec** `object`

Exec specifies the action to take.

▼ **command** `[]string`

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

▼ **failureThreshold** `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** `object`

GRPC specifies an action involving a GRPC port.

▼ **port** `integer` required

Port number of the gRPC service. Number must be in the range 1 to 65535.

▼ **service** `string`

Service is the name of the service to place in the gRPC HealthCheckRequest (see

https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗

). If this is not specified, the default behavior is defined by gRPC.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **scheme** `string`

Scheme to use for connecting to the host. Defaults to HTTP.

▼ **initialDelaySeconds** `integer`

Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. Set this value longer than the expected cleanup time for your process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **timeoutSeconds** `integer`

Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ **stdin** `boolean`

Whether this container should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the container will always result in EOF. Default is

false.

▼ stdinOnce `boolean`

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on container start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the container is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ terminationMessagePath `string`

Optional: Path at which the file to which the container's termination message will be written is mounted into the container's filesystem. Message written is intended to be brief final status, such as an assertion failure message. Will be truncated by the node if greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to /dev/termination-log. Cannot be updated.

▼ terminationMessagePolicy `string`

Indicate how the termination message should be populated. File will use the contents of terminationMessagePath to populate the container status message on both success and failure. FallbackToLogsOnError will use the last chunk of container log output if the termination message file is empty and the container exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to File. Cannot be updated.

▼ tty `boolean`

Whether this container should allocate a TTY for itself, also requires 'stdin' to be true. Default is false.

▼ volumeDevices `[]object`

volumeDevice describes a mapping of a raw block device within a container.

▼ devicePath `string` required

devicePath is the path inside of the container that the device will be mapped to.

▼ name `string` required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath `string` required

Path within the container at which the volume should be mounted. Must not contain ':'.

▼ mountPropagation `string`

mountPropagation determines how mounts are propagated from the host to container and the other way around. When not set, MountPropagationNone is used. This field is beta in 1.10.

▼ name `string` required

This must match the Name of a Volume.

▼ readOnly `boolean`

Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.

▼ subPath `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ subPathExpr `string`

Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to SubPath but environment variable references $(VAR_NAME) are expanded using the container's environment. Defaults to "" (volume's root). SubPathExpr and SubPath are mutually exclusive.

▼ workingDir `string`

Container's working directory. If not specified, the container runtime's default will be used, which might be configured in the container image. Cannot be updated.

▼ initContainers `[]object`

A single application container that you want to run within a pod.

▼ args `[]string`

Arguments to the entrypoint. The container image's CMD is used if this is not provided. Variable references $(VAR_NAME) are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double $$ are reduced to a single $, which allows for escaping the

$(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will produce the string literal "$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell ↗

▼ command `[]string`

Entrypoint array. Not executed within a shell. The container image's ENTRYPOINT is used if this is not provided. Variable references $(VAR_NAME) are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double $$ are reduced to a single $, which allows for escaping the $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will produce the string literal "$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell ↗

▼ env `[]object`

EnvVar represents an environment variable present in a Container.

▼ name `string` required

Name of the environment variable. Must be a C_IDENTIFIER.

▼ value `string`

Variable references $(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double $$ are reduced to a single $, which allows for escaping the $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will produce the string literal "$(VAR_NAME)". Escaped references will

never be expanded, regardless of whether the variable exists or not. Defaults to "".

▼ **valueFrom** `object`

Source for the environment variable's value. Cannot be used if value is not empty.

▼ **configMapKeyRef** `object`

Selects a key of a ConfigMap.

▼ **key** `string` required

The key to select.

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ **optional** `boolean`

Specify whether the ConfigMap or its key must be defined

▼ **fieldRef** `object`

Selects a field of the pod: supports metadata.name, metadata.namespace, `metadata.labels['<KEY>']`, `metadata.annotations['<KEY>']`, spec.nodeName,

spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `object`

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

Selects a key of a secret in the pod's namespace

▼ **key** `string` required

The key of the secret to select from. Must be a valid secret key.

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ **optional** `boolean`

Specify whether the Secret or its key must be defined

▼ **envFrom** `[]object`

EnvFromSource represents the source of a set of ConfigMaps

▼ **configMapRef** `object`

The ConfigMap to select from

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-

objects/names/#names↗ TODO: Add other useful fields.

apiVersion, kind, uid?

▼ **optional** `boolean`

Specify whether the ConfigMap must be defined

▼ **prefix** `string`

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ **secretRef** `object`

The Secret to select from

▼ **name** `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names↗ TODO: Add other useful fields.

apiVersion, kind, uid?

▼ **optional** `boolean`

Specify whether the Secret must be defined

▼ **image** `string`

Container image name. More info: https://kubernetes.io/docs/concepts/containers/images↗ This field is optional to

allow higher level config management to default or override container images in workload controllers like Deployments and StatefulSets.

▼ imagePullPolicy `string`

Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images#updating-images ↗

▼ lifecycle `object`

Actions that the management system should take in response to container lifecycle events. Cannot be updated.

▼ postStart `object`

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to

explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ scheme `string`

Scheme to use for connecting to the host. Defaults to HTTP.

▼ sleep `object`

Sleep represents the duration that the container should sleep before being terminated.

▼ seconds `integer` required

Seconds is the number of seconds to sleep.

▼ tcpSocket `object`

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

▼ host `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be

an IANA_SVC_NAME.

▼ preStop `object`

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ scheme `string`

Scheme to use for connecting to the host. Defaults to HTTP.

▼ **sleep** `object`

Sleep represents the duration that the container should sleep before being terminated.

▼ **seconds** `integer` required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** `object`

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **livenessProbe** `object`

Periodic probe of container liveness. Container will be restarted if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ **exec** `object`

Exec specifies the action to take.

▼ **command** `[]string`

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

▼ **failureThreshold** `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** `object`

GRPC specifies an action involving a GRPC port.

▼ **port** `integer` required

Port number of the gRPC service. Number must be in the range 1 to 65535.

▼ **service** `string`

Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ **port** required

Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **scheme** `string`

Scheme to use for connecting to the host. Defaults to HTTP.

▼ **initialDelaySeconds** `integer`

Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. Set this value longer than the expected cleanup time for your process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **timeoutSeconds** `integer`

Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ **name** `string` required

Name of the container specified as a DNS_LABEL. Each container in a pod must have a unique name (DNS_LABEL). Cannot be updated.

▼ ports `[]object`

ContainerPort represents a network port in a single container.

▼ containerPort `integer` required

Number of port to expose on the pod's IP address. This must be a valid port number, 0 < x < 65536.

▼ hostIP `string`

What host IP to bind the external port to.

▼ hostPort `integer`

Number of port to expose on the host. If specified, this must be a valid port number, 0 < x < 65536. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

▼ name `string`

If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.

▼ protocol `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ readinessProbe `object`

Periodic probe of container service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

▼ failureThreshold `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc `object`

GRPC specifies an action involving a GRPC port.

▼ port `integer` required

Port number of the gRPC service. Number must be in the range 1 to 65535.

▼ **service** `string`

Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

▼ **httpGet** `object`

HTTPGet specifies the http request to perform.

▼ **host** `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ **value** `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** required

Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **scheme** `string`

Scheme to use for connecting to the host. Defaults to HTTP.

▼ **initialDelaySeconds** `integer`

Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. Set this value longer than the expected cleanup time for your process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **timeoutSeconds** `integer`

Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ resizePolicy `[]object`

ContainerResizePolicy represents resource resize policy for the container.

▼ resourceName `string` required

Name of the resource to which this resource resize policy applies.

Supported values: cpu, memory.

▼ restartPolicy `string` required

Restart policy to apply when specified resource is resized. If not specified, it defaults to NotRequired.

▼ resources `object`

Compute Resources required by this container. Cannot be updated. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

▼ claims `[]object`

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name `string` required

Name must match the name of one entry in pod.spec.resourceClaims of the Pod where this field is used. It makes that resource available inside a container.

▼ limits `object`

Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/↗

▼ requests `object`

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/↗

▼ restartPolicy `string`

RestartPolicy defines the restart behavior of individual containers in a pod. This field may only be set for init containers, and the only allowed value is "Always". For non-init containers or when this field is not specified, the restart behavior is defined by the Pod's restart policy and the container type. Setting the RestartPolicy as "Always" for the init container will have the following effect: this init container will be continually restarted on exit until all regular containers have terminated. Once all regular containers have completed, all init containers with restartPolicy "Always" will be shut down. This lifecycle differs from normal init containers and is often referred to as a "sidecar" container. Although this init container still starts in the init container sequence, it does not wait for the container to complete before proceeding to the next init container. Instead, the next init container starts immediately after this init container is started, or after any startupProbe has successfully completed.

▼ securityContext `object`

SecurityContext defines the security options the container should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext.

More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/↗

▼ **allowPrivilegeEscalation** `boolean`

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the no_new_privs flag will be set on the container process. AllowPrivilegeEscalation is true always when the container is: 1) run as Privileged 2) has CAP_SYS_ADMIN Note that this field cannot be set when spec.os.name is windows.

▼ **capabilities** `object`

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when spec.os.name is windows.

▼ **add** `[]string`

Added capabilities

▼ **drop** `[]string`

Removed capabilities

▼ **privileged** `boolean`

Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when spec.os.name is windows.

▼ procMount `string`

procMount denotes the type of proc mount to use for the containers. The default is DefaultProcMount which uses the container runtime defaults for readonly paths and masked paths. This requires the ProcMountType feature flag to be enabled. Note that this field cannot be set when spec.os.name is windows.

▼ readOnlyRootFilesystem `boolean`

Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when spec.os.name is windows.

▼ runAsGroup `integer`

The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

▼ runAsNonRoot `boolean`

Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

▼ runAsUser `integer`

The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in

PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

▼ **seLinuxOptions** `object`

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

▼ **level** `string`

Level is SELinux level label that applies to the container.

▼ **role** `string`

Role is a SELinux role label that applies to the container.

▼ **type** `string`

Type is a SELinux type label that applies to the container.

▼ **user** `string`

User is a SELinux user label that applies to the container.

▼ **seccompProfile** `object`

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.

▼ **localhostProfile** `string`

localhostProfile indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.

▼ **type** `string` required

type indicates which kind of seccomp profile will be applied. Valid options are: Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

▼ **windowsOptions** `object`

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

▼ **gmsaCredentialSpec** `string`

GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa↗) inlines the contents of the GMSA credential spec named by the GMSACredentialSpecName field.

▼ **gmsaCredentialSpecName** `string`

GMSACredentialSpecName is the name of the GMSA credential spec to use.

▼ **hostProcess** `boolean`

HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.

▼ **runAsUserName** `string`

The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

▼ **startupProbe** `object`

StartupProbe indicates that the Pod has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state operation. This cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

▼ failureThreshold `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc `object`

GRPC specifies an action involving a GRPC port.

▼ port `integer` required

Port number of the gRPC service. Number must be in the range 1 to 65535.

▼ service `string`

Service is the name of the service to place in the gRPC HealthCheckRequest (see

https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗

). If this is not specified, the default behavior is defined by gRPC.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ **scheme** `string`

Scheme to use for connecting to the host. Defaults to HTTP.

▼ **initialDelaySeconds** `integer`

Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

▼ terminationGracePeriodSeconds `integer`

Optional duration in seconds the pod needs to terminate gracefully upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. Set this value longer than the expected cleanup time for your process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ timeoutSeconds `integer`

Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes ↗

▼ stdin `boolean`

Whether this container should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the container will always result in EOF. Default is

false.

▼ stdinOnce `boolean`

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on container start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the container is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ terminationMessagePath `string`

Optional: Path at which the file to which the container's termination message will be written is mounted into the container's filesystem. Message written is intended to be brief final status, such as an assertion failure message. Will be truncated by the node if greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to /dev/termination-log. Cannot be updated.

▼ terminationMessagePolicy `string`

Indicate how the termination message should be populated. File will use the contents of terminationMessagePath to populate the container status message on both success and failure. FallbackToLogsOnError will use the last chunk of container log output if the termination message file is empty and the container exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to File. Cannot be updated.

▼ tty `boolean`

Whether this container should allocate a TTY for itself, also requires 'stdin' to be true. Default is false.

▼ volumeDevices `[]object`

volumeDevice describes a mapping of a raw block device within a container.

▼ devicePath `string` required

devicePath is the path inside of the container that the device will be mapped to.

▼ name `string` required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath `string` required

Path within the container at which the volume should be mounted. Must not contain ':'.

▼ mountPropagation `string`

mountPropagation determines how mounts are propagated from the host to container and the other way around. When not set, MountPropagationNone is used. This field is beta in 1.10.

▼ name `string` required

This must match the Name of a Volume.

▼ readOnly `boolean`

Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.

▼ subPath `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ subPathExpr `string`

Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to SubPath but environment variable references $(VAR_NAME) are expanded using the container's environment. Defaults to "" (volume's root). SubPathExpr and SubPath are mutually exclusive.

▼ workingDir `string`

Container's working directory. If not specified, the container runtime's default will be used, which might be configured in the container image. Cannot be updated.

▼ injectPass `boolean`

InjectPass enables password injection for users

▼ ipFamilyPrefer `string`

IPFamilyPrefer specifies preferred IP family for services

▼ logicalBackupSchedule `string`

LogicalBackupSchedule defines the cron schedule for logical backups

▼ maintenanceWindows `[]object`

MaintenanceWindow describes the time window when the operator is allowed to do maintenance on a cluster.

▼ endTime `string`

EndTime defines the end of the maintenance window in HH:MM format

▼ everyday `boolean`

Everyday enables maintenance every day of the week

▼ startTime `string`

StartTime defines the beginning of the maintenance window in HH:MM format

▼ masterServiceAnnotations `object`

MasterServiceAnnotations defines annotations for master service

▼ nodeAffinity `object`

NodeAffinity defines node affinity rules for pod scheduling

▼ preferredDuringSchedulingIgnoredDuringExecution `[]object`

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

▼ **preference** `object` required

A node selector term, associated with the corresponding weight.

▼ **matchExpressions** `[]object`

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

▼ **key** `string` required

The label key that the selector applies to.

▼ **operator** `string` required

Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

▼ **values** `[]string`

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

▼ **matchFields** `[]object`

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

▼ **key** `string` required

The label key that the selector applies to.

▼ **operator** `string` required

Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

▼ **values** `[]string`

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

▼ **weight** `integer` required

Weight associated with matching the corresponding nodeSelectorTerm, in the range 1-100.

▼ **requiredDuringSchedulingIgnoredDuringExecution** `object`

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

▼ nodeSelectorTerms `[]object` required

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

▼ matchExpressions `[]object`

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

▼ key `string` required

The label key that the selector applies to.

▼ operator `string` required

Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

▼ values `[]string`

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

▼ matchFields `[]object`

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

▼ **key** `string` required

The label key that the selector applies to.

▼ **operator** `string` required

Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

▼ **values** `[]string`

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

▼ **nodeSelector** `object`

NodeSelector defines node labels for pod scheduling

▼ **numberOfInstances** `integer` required

NumberOfInstances specifies the number of PostgreSQL replicas

▼ **patroni** `object`

Patroni configuration for cluster management

▼ **failsafe_mode** `boolean`

Enable failsafe mode for cluster

▼ initdb `object`

InitDB parameters for database initialization

▼ loop_wait `integer`

Loop wait time in seconds for leader election

▼ maximum_lag_on_failover `number`

Maximum lag in bytes allowed for failover (float32 because https://github.com/kubernetes/kubernetes/issues/30213 ↗)

▼ pg_hba `[]string`

Custom pg_hba.conf entries

▼ retry_timeout `integer`

Retry timeout in seconds for operations

▼ slots `object`

Replication slots configuration

▼ synchronous_mode `boolean`

Enable synchronous replication

▼ **synchronous_mode_strict** `boolean`

Enable strict synchronous replication

▼ **synchronous_node_count** `integer`

Number of synchronous replicas

▼ **ttl** `integer`

Leader key TTL in seconds

▼ **pgpool2Settings** `object`

Pgpool2Settings defines Pgpool-II configuration

▼ **dockerImage** `string`

DockerImage specifies the container image to use for Pgpool-II

▼ **maxPool** `integer`

MaxPool defines the maximum number of cached connections per child process

▼ **numInitChildren** `integer`

NumInitChildren sets the number of preforked Pgpool-II server processes

▼ **numberOfInstances** `integer`

NumberOfInstances specifies how many Pgpool-II instances to run

▼ resources `object`

Resources specifies CPU and memory requirements for Pgpool-II containers

▼ limits `object`

ResourceLimits defines the maximum resources allowed for containers

▼ cpu `string`

CPU requirement (e.g., "500m" or "1")

▼ memory `string`

Memory requirement (e.g., "512Mi" or "2Gi")

▼ requests `object`

ResourceRequests defines the minimum resources required for containers

▼ cpu `string`

CPU requirement (e.g., "500m" or "1")

▼ memory `string`

Memory requirement (e.g., "512Mi" or "2Gi")

▼ user `string`

User specifies the database user for Pgpool-II connections

▼ pod_priority_class_name `string`

PodPriorityClassNameOld is a deprecated field for priority class

▼ podAnnotations `object`

PodAnnotations defines annotations to add to pods

▼ podPriorityClassName `string`

PodPriorityClassName specifies the priority class for pods

▼ postgresql `object` required

PostgreSQL version and configuration parameters including: - PostgreSQL major version (e.g. "14") - Configuration parameters (postgresql.conf) - Configuration parameters (postgresql.conf) PostgreSQL version and configuration parameters

▼ parameters `object`

PostgreSQL configuration parameters (postgresql.conf)

▼ version `string` required

PostgreSQL major version (e.g. "14")

▼ preparedDatabases `object`

PreparedDatabases defines databases with pre-configured schemas and roles

▼ repairOption `object`

RepairOption defines options for cluster repair

▼ autoRecovery `boolean`

AutoRecovery enables automatic recovery attempts for failed clusters

▼ startTimeout `integer`

StartTimeout defines the maximum time (in seconds) to wait for cluster startup before recovery

▼ replicaLoadBalancer `boolean`

ReplicaLoadBalancer is a deprecated field for replica load balancer

▼ replicaServiceAnnotations `object`

ReplicaServiceAnnotations defines annotations for replica service

▼ resources `object`

Resource requests and limits for PostgreSQL containers

▼ limits `object`

ResourceLimits defines the maximum resources allowed for containers

▼ cpu `string`

CPU requirement (e.g., "500m" or "1")

▼ **memory** `string`

Memory requirement (e.g., "512Mi" or "2Gi")

▼ **requests** `object`

ResourceRequests defines the minimum resources required for containers

▼ **cpu** `string`

CPU requirement (e.g., "500m" or "1")

▼ **memory** `string`

Memory requirement (e.g., "512Mi" or "2Gi")

▼ **schedulerName** `string`

SchedulerName specifies the Kubernetes scheduler to use

▼ **serviceAnnotations** `object`

ServiceAnnotations defines annotations to add to services

▼ **serviceTemplates** `object`

ServiceTemplates defines custom service templates

▼ **sidecars** `[]object`

Sidecar defines a container to be run in the same pod as the Postgres container.

▼ env `[]object`

EnvVar represents an environment variable present in a Container.

▼ name `string` required

Name of the environment variable. Must be a C_IDENTIFIER.

▼ value `string`

Variable references $(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double $$ are reduced to a single $, which allows for escaping the $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will produce the string literal "$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to "".

▼ valueFrom `object`

Source for the environment variable's value. Cannot be used if value is not empty.

▼ configMapKeyRef `object`

Selects a key of a ConfigMap.

▼ key `string` required

The key to select.

▼ name `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names TODO: Add other useful fields. apiVersion, kind, uid?

▼ **optional** `boolean`

Specify whether the ConfigMap or its key must be defined

▼ **fieldRef** `object`

Selects a field of the pod: supports metadata.name, metadata.namespace, `metadata.labels['<KEY>']`, `metadata.annotations['<KEY>']`, spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `object`

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

▼ containerName `string`

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource `string` required

Required: resource to select

▼ secretKeyRef `object`

Selects a key of a secret in the pod's namespace

▼ key `string` required

The key of the secret to select from. Must be a valid secret key.

▼ name `string`

Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?

▼ optional `boolean`

Specify whether the Secret or its key must be defined

▼ image `string`

DockerImage specifies the container image to use for the sidecar

▼ name `string`

Name specifies the unique name of the sidecar container within the pod

▼ ports `[]object`

ContainerPort represents a network port in a single container.

▼ containerPort `integer` required

Number of port to expose on the pod's IP address. This must be a valid port number, 0 < x < 65536.

▼ hostIP `string`

What host IP to bind the external port to.

▼ hostPort `integer`

Number of port to expose on the host. If specified, this must be a valid port number, 0 < x < 65536. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

▼ name `string`

If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.

▼ **protocol** `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ **resources** `object`

Resources defines CPU and memory requirements for the sidecar container

▼ **limits** `object`

ResourceLimits defines the maximum resources allowed for containers

▼ **cpu** `string`

CPU requirement (e.g., "500m" or "1")

▼ **memory** `string`

Memory requirement (e.g., "512Mi" or "2Gi")

▼ **requests** `object`

ResourceRequests defines the minimum resources required for containers

▼ **cpu** `string`

CPU requirement (e.g., "500m" or "1")

▼ **memory** `string`

Memory requirement (e.g., "512Mi" or "2Gi")

▼ **spiloAllowPrivilegeEscalation** `boolean`

SpiloAllowPrivilegeEscalation controls privilege escalation for Spilo

▼ **spiloFSGroup** `integer`

SpiloFSGroup specifies the filesystem group ID for Spilo container

▼ **spiloPrivileged** `boolean`

SpiloPrivileged enables privileged mode for Spilo container

▼ **spiloReadOnlyRootFilesystem** `boolean`

SpiloReadOnlyRootFilesystem enables read-only root filesystem for Spilo

▼ **spiloRunAsGroup** `integer`

SpiloRunAsGroup specifies the group ID to run Spilo container as

▼ **spiloRunAsUser** `integer`

SpiloRunAsUser specifies the user ID to run Spilo container as

▼ **standby** `object`

StandbyCluster defines configuration for standby clusters

▼ **gs_wal_path** `string`

GSWalPath specifies the Google Cloud Storage path for WAL archiving (e.g. "gs://bucket/path")

▼ **s3_wal_path** `string`

S3WalPath specifies the S3 path for WAL archiving (e.g. "s3://bucket/path")

▼ **standby_host** `string`

StandbyHost specifies the hostname or IP address of the primary cluster

▼ **standby_port** `string`

StandbyPort specifies the port number of the primary cluster

▼ **streams** `[]object`

Stream defines properties for creating FabricEventStream resources

▼ **applicationId** `string` required

ApplicationId is the unique identifier for the application producing the event stream

▼ **batchSize** `integer`

BatchSize controls how many events are batched together before being processed

▼ **database** `string` required

Database specifies the PostgreSQL database where the event tables are located

▼ filter `object`

Filter specifies optional filtering conditions for the event stream

▼ tables `object` required

Tables defines the mapping of table names to their stream configurations

▼ teamId `string` required

TeamID is the identifier of the team owning the cluster

▼ tls `object`

TLS defines TLS configuration for the cluster

▼ caFile `string`

CAFile specifies the path to the CA certificate file for client verification

▼ caSecretName `string`

CASecretName specifies the Kubernetes secret containing the CA certificate

▼ certificateFile `string`

CertificateFile specifies the path to the server certificate file

▼ privateKeyFile `string`

PrivateKeyFile specifies the path to the server private key file

▼ secretName `string`

SecretName specifies the Kubernetes secret containing TLS certificates

▼ tolerations `[]object`

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

▼ effect `string`

Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.

▼ key `string`

Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.

▼ operator `string`

Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.

▼ tolerationSeconds `integer`

TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.

▼ value `string`

Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

▼ upgradeOption `object`

UpgradeOption defines upgrade options

▼ crVersion `string`

CRVersion specifies the target custom resource version for the upgrade

▼ useLoadBalancer `boolean`

UseLoadBalancer is a deprecated field for enabling load balancer

▼ users `object`

Users defines additional PostgreSQL users and their roles

▼ usersCustomizedPasswd `object`

UsersCustomizedPasswd stores custom passwords for users

▼ usersWithInPlaceSecretRotation `[]string`

UsersWithInPlaceSecretRotation defines users requiring in-place secret rotation

▼ usersWithSecretRotation `[]string`

UsersWithSecretRotation defines users requiring secret rotation

▼ volume `object`

Volume configuration for PostgreSQL data storage

▼ iops `integer`

IOPS for provisioned IOPS SSD volumes

▼ selector `object`

Selector for matching existing PersistentVolumeClaims

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ operator `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]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.

▼ **matchLabels** `object`

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.

▼ **size** `string` required

Size of the volume (e.g., "100Gi")

▼ **storageClass** `string`

StorageClass name for dynamic provisioning

▼ **subPath** `string`

SubPath within the volume to mount

▼ **throughput** `integer`

Throughput for gp3 volumes (MiB/s)

▼ **type** `string`

VolumeType for EBS volumes (gp2, gp3, io1, etc.)

▼ status `object`

Status defines the observed state of the PostgreSQL cluster including current status, patroni state, and any error messages.

▼ PostgresClusterStatus `string`

PostgresClusterStatus indicates the current operational state of the PostgreSQL cluster

▼ UsersCustomizedPasswdStatus `string`

UsersCustomizedPasswdStatus tracks the status of custom password management for database users

▼ clusterReplicationRole `string`

ClusterReplicationRole indicates the replication role (primary/replica) of this cluster

▼ message `string`

Message provides additional details about the cluster status or error conditions

▼ patroniStatus `object`

PatroniStatus contains status information for each Patroni-managed PostgreSQL instance

▼ upgradeStatus `object`

UpgradeStatus tracks version information for operator and custom resource upgrades

▼ crVersion `string`

CRVersion tracks the version of the custom resource specification

▼ operatorVersion `string`

OperatorVersion tracks the version of the operator that last processed this cluster

PostgresBackup

`middleware.alauda.io` group

PostgresBackup is the Schema for the postgresbackups API

`v1` version

▼ spec `object`

PostgresBackupSpec defines the desired state of PostgresBackup

▼ cluster `string` required

Cluster is the name of the PostgreSQL cluster to backup

▼ executeNode `string`

ExecuteNode specifies the node where the backup job should run If empty, the backup controller will automatically select a node

▼ status `object`

PostgresBackupStatus defines the observed state of PostgresBackup

▼ backupName `string`

BackupName is the unique name of this backup instance

▼ clusterUid `string`

ClusterUid is the unique identifier of the PostgreSQL cluster

▼ configBackupStorage `object`

ConfigBackupStorage contains the configuration for the backup storage

▼ bucket `string` required

▼ name `string` required

▼ namespace `string` required

▼ error `string`

Error contains any error messages from the backup operation

▼ executeNode `string`

ExecuteNode is the node where the backup job is running

▼ finishLsn `integer`

FinishLsn is the log sequence number when the backup finished

▼ finishTime `string`

FinishTime is when the backup operation completed

▼ lastModified `string`

LastModified is the timestamp when the backup was last modified

▼ pgVersion `string`

PGVersion is the PostgreSQL version of the cluster being backed up

▼ startLsn `integer`

StartLsn is the log sequence number when the backup started

▼ startTime `string`

StartTime is when the backup operation began

▼ state `string`

State represents the current state of the backup operation Possible values: "", "running", "failed", "succeeded", "deleteFailed"

PostgresRestore

`middleware.alauda.io` group

PostgresRestore is the Schema for the postgresrestores API

`v1` version

▼ spec `object`

PostgresRestoreSpec defines the desired state of PostgresRestore

▼ backupCluster `object` required

BackupCluster contains details about the source backup cluster

▼ name `string`

Name of the source PostgreSQL cluster

▼ storage `object`

Storage configuration for accessing the backup

▼ bucket `string` required

Bucket name where backups are stored

▼ name `string` required

Name of the storage configuration

▼ namespace `string` required

Namespace where the storage configuration is located

▼ s3Option `object`

S3Option contains S3-specific storage options

▼ awsS3ForcePathStyle `string`

AwsS3ForcePathStyle enables path-style S3 URLs

(s3.amazonaws.com/BUCKET/KEY)

▼ s3ForcePathStyle `boolean`

S3ForcePathStyle forces path-style S3 URLs

▼ useWalg `boolean`

UseWalg enables the WAL-G backup tool

▼ useWalgBackup `boolean`

UseWalgBackup enables WAL-G for backup operations

▼ walgDisableS3Sse `boolean`

WalgDisableS3Sse disables S3 server-side encryption for WAL-G

▼ uid `string`

Uid is the unique identifier of the source cluster

▼ **targetCluster** `string` required

TargetCluster is the name of the PostgreSQL cluster to restore into

▼ **timestamp** `string` required

Timestamp specifies the point-in-time to restore to in RFC3339 format

▼ **status** `object`

PostgresRestoreStatus defines the observed state of PostgresRestore

▼ **error** `string`

Error contains any error messages from the restore operation

▼ **state** `string`

State represents the current state of the restore operation