Operations

RWX logging

6/15/2026

Contents

- 1 RWX logging
- 2 Storage Prerequisites
  - 2.1 Direct NFS Persistent Storage
  - 2.2 Azure-Files Persistent Storage for ARO (NFS Backed)

Learn about the legacy logging method of writing logs to an RWX storage such as NFS or NAS server.

Related documentation:

-
-

RSS:

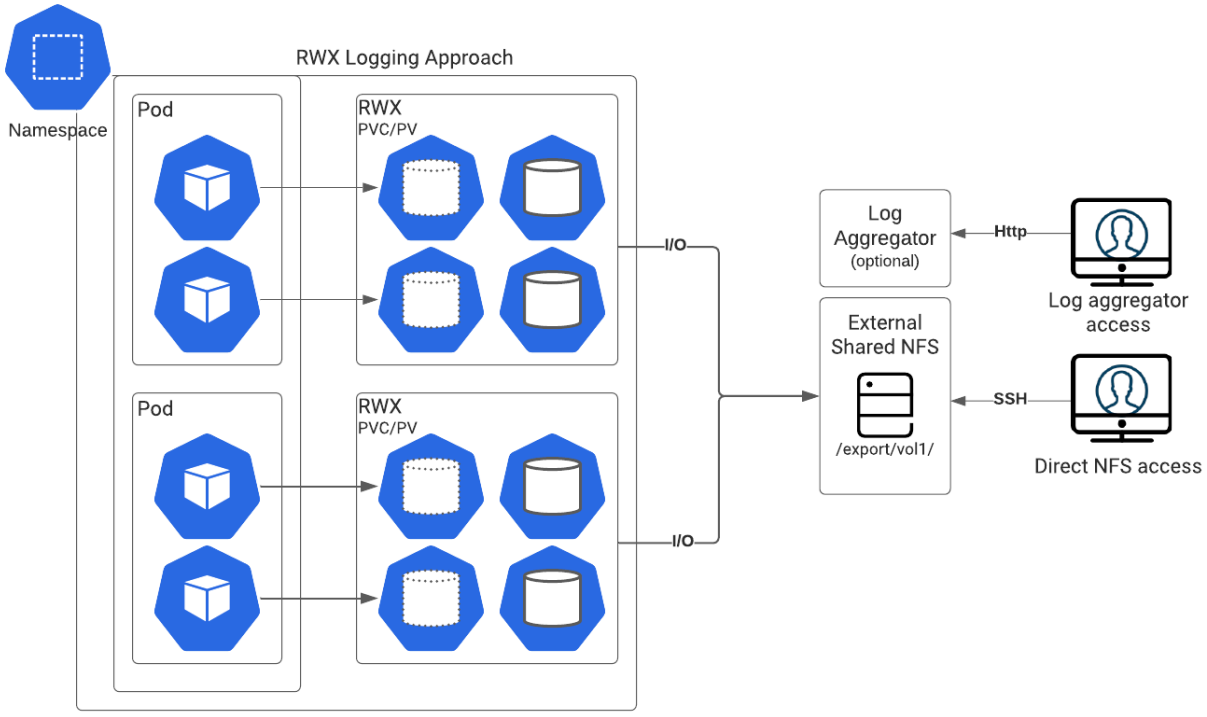
- For private edition

RWX logging

> **Important**
>
> RWX logging is deprecated. It will be phased out with the use of sidecars to facilitate legacy logging behavior.

Some Genesys Multicloud CX services neither write structured logs in Kubernetes format nor do they write to the stdout/stderr console. These services use RWX logging, which is the legacy logging method of writing logs to an RWX storage such as NFS or NAS server.

Legacy Genesys Multicloud CX applications are not structured to be supported by logging capabilities offered in Kubernetes, nor do they write to sufficient detail in stdout/stderr. To accommodate this type of logging behavior, deployments must be provisioned to support mounting PVC/PV to NFS storage for the application to write its logs. Each Service mounts to its own PV which is backed by an external NFS share. After the logs are written to NFS share, the application controls the size and retention of the file and files can be accessed externally from NFS share directly to package and provide to care.



The method of logging unstructured logs is not suitable for kubernetes-supported logging aggregators such as Elasticsearch.

The sample procedures provided in the following section, help in setting up the RWX storage of your choice.

Services that use the RWX logging approach:

- WebRTC
- GVP
- GCXI
- Voice Microservices
- Genesys Pulse
- Interaction Server
- Tenant Services

Storage Prerequisites

Direct NFS Persistent Storage

With Direct NFS approach, shares are mounted using NFS IP/FQDN and share path is mounted using NFS-subdir-external-provisioner.

For more details about this provisioner, refer to NFS Subdir External Provisioner.

Prerequisite: You must have a dedicated NFS server to create NFS persistent storage.

Create StorageClass for NFS Retained Storage

Here is a sample configuration to create StorageClass for NFS persistent storage. The following configuration is suitable for a bare metal server.

bare-metal-sc.yaml

```
provisioner: cluster.local/nfs-vce-c00ds-vol1-nfs-subdir-external-provisioner
mountOptions:
- nfsvers=3
- uid=500
- gid=500
parameters:
archiveOnDelete: 'false'
volumeBindingMode: Immediate metadata
name:
kind: StorageClass
reclaimPolicy: Retain
allowVolumeExpansion: true
apiVersion: storage.k8s.io/v1

oc apply -f bare-metal-sc.yaml
```

Create PVC to dynamically create and bind to PV

create-pvc.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
 name: -pvc
 namespace:
spec:
 accessModes:
  - ReadWriteOnce
resources:
requests:
storage: 10Gi
storageClassName:
volumeMode: Filesystem
```

Azure-Files Persistent Storage for ARO (NFS Backed)

For ARO type deployments you can map NFS directly. Therefore, you can create NFS share within

Azure using Azure-files. You need to create storage class of type Azure-Files:

- "recalimPolicy" set to "Retain"
- "parameters" set based on your specific Azure deployment

For more details, refer to:

- How to create an NFS share
- Dynamically create and use a persistent volume with Azure Files in AKS

Create Storage Class for retained Azure-File NFS storage

azure-file-retain-sc.yaml

```
allowVolumeExpansion: true
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
 name: azure-files-retain
annotations:
 description: azure-files-retain
provisioner: kubernetes.io/azure-file
parameters:
 location: westus2
 skuName: Standard_LRS
reclaimPolicy: Retain
volumeBindingMode: Immediate

oc apply -f azure-file-retain-sc.yaml
```

Create PVC to dynamically create and bind to PV

create-pvc.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
name: -pvc
namespace:
spec:
accessModes:
- ReadWriteOnce
resources:
requests:
storage: 10Gi
storageClassName:
volumeMode: Filesystem

oc apply -f create-pvc.yaml
```