QSAN Official Document

CSI Driver and Kubernetes Integration

Application Note

January 2024

ANNOUNCEMENT

Copyright

© Copyright 2024 QSAN Technology, Inc. All rights reserved. No part of this document may be reproduced or transmitted without written permission from QSAN Technology, Inc.

QSAN believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

Trademarks

- QSAN, the QSAN logo, QSAN.com, XCubeFAS, XCubeSAN, XCubeNXT, XCubeNAS, XCubeDAS, XEVO, and QSM are trademarks or registered trademarks of QSAN Technology, Inc.
- Microsoft, Windows, Windows Server, and Hyper-V are trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries.
- Linux is a trademark of Linus Torvalds in the United States and/or other countries.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Mac and OS X are trademarks of Apple Inc., registered in the U.S. and other countries.
- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
- VMware, ESXi, and vSphere are registered trademarks or trademarks of VMware, Inc. in the United States and/or other countries.
- Citrix and Xen are registered trademarks or trademarks of Citrix Systems, Inc. in the United States and/or other countries.
- Other trademarks and trade names used in this document to refer to either the entities claiming the marks and names or their products are the property of their respective owners.

TABLE OF CONTENTS

Announcement i
Notices iv
Preface v
Technical Support v
Information, Tip, and Caution v
1. Overview 1
1.1. CSI Driver 1
1.2. When to Use? 1
1.3. How to Use? 1
1.4. Challenges 2
1.5. Benefits 2
2. Implementation 4
2.1. Topology and Configuration 4
2.2. Install CSI Driver 5
2.3. Setup Multipath Policy 9
2.4. Fibre Channel Usage 10
2.5. Supported and Unsupported Types 11
2.6. Constraints 12
3. Conclusion 13
4. Appendix 14
4.1. Apply To 14
4.2. Reference 14

©2024 QSAN Technology, Inc. All rights reserved.

FIGURES

Figure 2-1 Topology 4
Figure 2-2 Create a Pool 6
Figure 2-3 Create a Host Group 7
Figure 2-4 Set the IP Addresses of iSCSI Data Ports 7
Figure 2-5 Supported Scenario 1 11
Figure 2-6 Supported Scenario 2 11
Figure 2-7 Supported Scenario 3 11
Figure 2-8 Supported Scenario 4 12
Figure 2-9 Unsupported Scenario 1 12

Announcement
©2024 QSAN Technology, Inc. All rights reserved.
www.qsan.com
Official Document

NOTICES

Information contained in this document has been reviewed for accuracy. But it could include typographical errors or technical inaccuracies. Changes are made to the document periodically. These changes will be incorporated in new editions of the publication. QSAN may make improvements or changes in the products. All features, functionality, and product specifications are subject to change without prior notice or obligation. All statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

©2024 QSAN Technology, Inc. All rights reserved.

PREFACE

Technical Support

Do you have any questions or need help trouble-shooting a problem? Please contact QSAN Support, we will reply to you as soon as possible.

- Via the Web: https://www.qsan.com/technical_support
- Via Telephone: +886-2-77206355
- (Service hours: 09:30 - 18:00, Monday - Friday, UTC+8)
- Via Skype Chat, Skype ID: qsan.support
- (Service hours: 09:30 - 02:00, Monday - Friday, UTC+8, Summer time: 09:30 - 01:00)
- Via Email: support@qsan.com

Information, Tip, and Caution

This document uses the following symbols to draw attention to important safety and operational information.



INFORMATION

INFORMATION provides useful knowledge, definition, or terminology for reference.



TIP

TIP provides helpful suggestions for performing tasks more effectively.

©2024 QSAN Technology, Inc. All rights reserved.



CAUTION

CAUTION indicates that failure to take a specified action could result in damage to the system.

©2024 QSAN Technology, Inc. All rights reserved.

1. OVERVIEW

1.1. CSI Driver

Kubernetes (also known as K8s) CSI (Container Storage Interface) driver is a plugin that enables dynamic provisioning and management of storage resources for Kubernetes clusters. It allows you to connect external storage systems to your Kubernetes cluster and use them as persistent volumes for your containerized applications. CSI drivers can be used in a variety of scenarios, such as running stateful applications that require data persistence, performing backups and disaster recovery, or integrating with specialized storage hardware.

1.2. When to Use?

Kubernetes provides a number of built-in volume plugins that can be used to provision and manage storage resources for containerized applications. However, when provisioning and managing storage resources for containerized applications in Kubernetes, the built-in volume plugins may not always meet specific storage requirements or be compatible with certain storage systems. This can result in insufficient local disk performance.

To overcome this, Kubernetes provides a CSI (Container Storage Interface) driver that allows dynamic provisioning and management of storage resources from an external storage system. With a CSI driver, any storage system that supports the CSI specification can be used with Kubernetes, regardless of the underlying technology. This enables improved storage performance, scalability, flexibility, simplified management, and future-proofing of storage needs.

1.3. How to Use?

To use the QSAN CSI driver with Kubernetes, you will need to deploy a QSAN XCubeSAN series storage using the tutorial provided. You will also need to deploy the CSI driver via the how-to video clip, which involves installing the driver's software components and configuring it to communicate with the XCubeSAN storage system.

©2024 QSAN Technology, Inc. All rights reserved.

Create a new storage class that uses the CSI driver. The storage class defines the storage properties, such as the type of storage, the capacity, and the access mode. You can then create a new PVC (Persistent Volume Claim) for your application that uses the new storage class. The PVC defines the storage requirements for your application. Deploy your application to the Kubernetes cluster using the new PVC.

1.4. Challenges

Storage Space Is Running Out

In the situation which the local drive is running out of the storage space, it is hard to make an online expansion to meeting the non-stoppable Micro Services running on the K8s nodes. You may however use a CSI driver to dynamically provision and manage storage resources from an external storage system, and the CSI driver can be added at the same time while you are using local drive as the original storage space.

Local Storage Performance is Limited

Using a CSI driver to address insufficient local disk performance in Kubernetes has several benefits, one of which is the ability to dynamically add external storage resources to an existing Kubernetes environment without affecting the microservices running on the local drive. This means that you can add additional storage resources on-demand to meet the needs of your application and workload, without worrying about the impact on other microservices or the need to migrate data between different storage systems.

1.5. Benefits

Using an External Storage System with a CSI Driver

Using an external storage system with a CSI driver in a Kubernetes cluster can provide a more flexible, scalable, and reliable storage solution for your containerized applications. It can also offer advanced storage features and security capabilities that may not be available with local storage on Kubernetes worker nodes.

©2024 QSAN Technology, Inc. All rights reserved.

1. **Scalability:** External storage systems can offer much larger storage capacity and more advanced features than local drives on Kubernetes worker nodes. This allows you to scale your storage capacity and performance as your application requirements grow.
2. **Data Persistence:** By using an external storage system with data replication and backup features, you can ensure that your application data is stored persistently and can be easily recovered in the event of a failure.
3. **High Availability:** External storage systems can provide highly available storage volumes with automatic failover and redundancy. This helps to ensure that your application can continue to operate even if one or more storage devices fail.
4. **Storage Efficiency:** Many external storage systems offer advanced storage features such as thin provisioning, data compression, and deduplication, which can help to reduce your storage costs and improve your storage efficiency.
5. **Security:** External storage systems can offer advanced security features such as encryption, access controls, and auditing. This can help to protect your application data and meet regulatory compliance requirements.

Kubernetes Storage Optimized Performance and Compatibility

1. **Performance:** Monitor the performance of your application and the CSI driver to ensure that it meets your expectations. You can use Kubernetes monitoring tools or the Performance Monitor Analysis provided by the XCubeSAN.
2. **Capacity Planning:** Ensure that you have sufficient capacity available in your external storage system to meet the needs of your application and workload.
3. **Scalability:** Consider the scalability of your storage resources and ensure that you can add additional storage resources on-demand to meet the needs of your application and workload. And each Host Group is capable of hosting 32 volumes for now.
4. **Compatibility:** Ensure that the QSAN CSI driver is compatible with the version of Kubernetes that you are using. The compatible K8s version is from 1.21 to 1.25. Please download CSI driver from GitHub and download the image from Docker Hub. Please create Thick Provisioning RAID pool from QSAN XCubeSAN series product when trying to use QSAN CSI driver.
5. **Volume Timestamp:** This function is set as disabled by default, please visit **System** -> **Settings** -> **General Settings** to enable it if needed. Performance might be slightly impacted once it is enabled.

©2024 QSAN Technology, Inc. All rights reserved.

2. IMPLEMENTATION

2.1. Topology and Configuration

When using an external iSCSI storage system with a CSI driver in a Kubernetes cluster, the typical topology involves a set of Kubernetes worker nodes that are connected to the external storage system via an iSCSI network.

In this topology, the iSCSI network provides a dedicated network connection between the Kubernetes worker nodes and the external storage system. This helps to ensure that the storage traffic does not interfere with the regular network traffic in the Kubernetes cluster, and provides a high level of performance and reliability for storage operations.

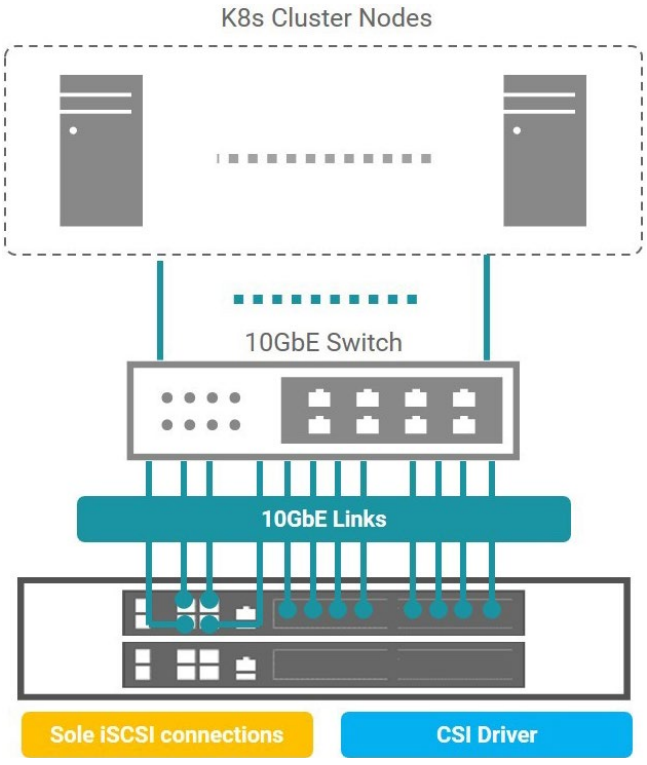


Figure 2-1 Topology

©2024 QSAN Technology, Inc. All rights reserved.

2.2. Install CSI Driver

This section will describe how to install the CSI driver on a Kubernetes environment, which is a critical step in enabling dynamic storage provisioning. To install the CSI driver, please follow the steps below:

1. Download the CSI driver package.

```
root@qaubuntufunction:/home/qa# git clone https://github.com/QsanJohnson/csi-qsan
Cloning into 'csi-qsan'...
remote: Enumerating objects: 129, done.
remote: Counting objects: 100% (129/129), done.
remote: Compressing objects: 100% (87/87), done.
remote: Total 129 (delta 61), reused 105 (delta 40), pack-reused 0
Receiving objects: 100% (129/129), 5.09 MiB | 332.00 KiB/s, done.
Resolving deltas: 100% (61/61), done.
```

2. Go to the path as follows.

```
root@qaubuntufunction:/home/qa# cd csi-qsan/deploy/
```

3. Edit the **qsan-auth** file as follows.

```
storageArrays:
- server: 192.168.202.221
  port: 80
  https: false
  username: admin
  password: 1234
```

4. Enter **./install.sh** to install this package.

```
root@qaubuntufunction:/home/qa/csi-qsan/deploy# ./install.sh
Error from server (AlreadyExists): namespaces "qsan" already exists
error: failed to create secret secrets "qsan-auth-secret" already exists
serviceaccount/xevo-csi-controller-sa unchanged
serviceaccount/xevo-csi-node-sa unchanged
clusterrole.rbac.authorization.k8s.io/xevo-csi-controller-role unchanged
clusterrole.rbac.authorization.k8s.io/xevo-csi-node-role unchanged
clusterrolebinding.rbac.authorization.k8s.io/xevo-csi-controller-binding unchanged
clusterrolebinding.rbac.authorization.k8s.io/xevo-csi-node-binding unchanged
csidriver.storage.k8s.io/xevo.csi.qsan.com unchanged
deployment.apps/xevo-csi-controller configured
daemonset.apps/xevo-csi-node configured
configmap/configmap-csi-iscsiadm unchanged
root@qaubuntufunction:/home/qa/csi-qsan/deploy#
```


©2024 QSAN Technology, Inc. All rights reserved.

5. Go to the path as follows and edit the **sc.yaml** file and set the **StorageClass** name.

```
root@qaubuntufunction:/home/qa/csi-qsan/deploy# cd example/
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example# vi sc.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: qsan-xevo-storage1
provisioner: xevo.csi.qsan.com
parameters:
  protocol: iscsi
  server: 192.168.202.221
  pool: Pool_02
  iscsiPortals: 192.168.222.65:3260,192.168.222.91:3260
  target: HostGroup_001
allowVolumeExpansion: true
~
```

Now that you have completed the initial setup of your environment, continue with the following steps.

6. Create a pool and a host group, then modify the management IP address and IP addresses of iSCSI data ports in the storage system.
   - Create a pool.

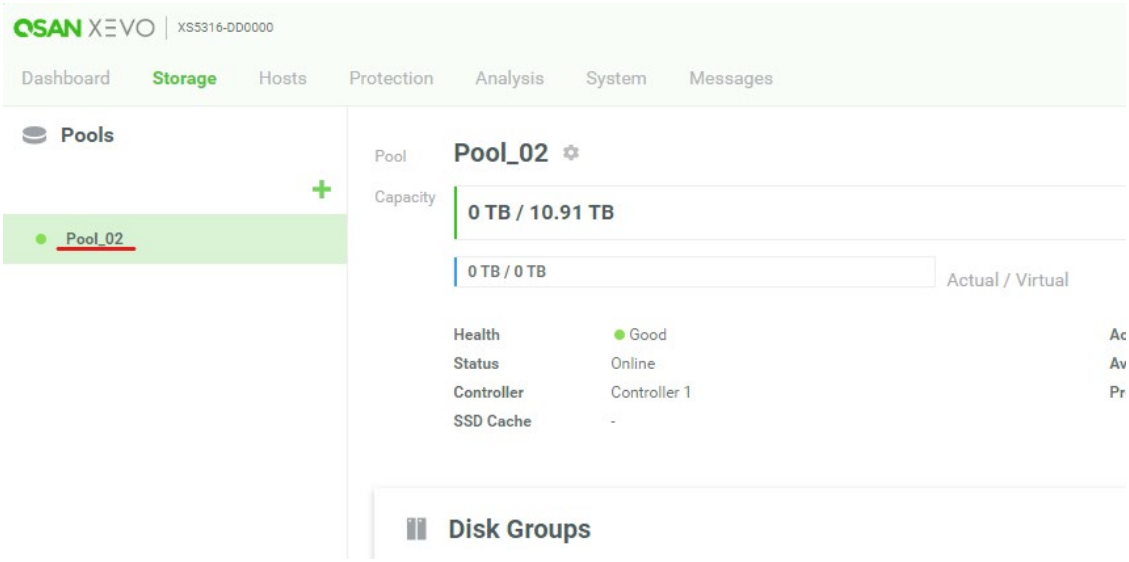


Figure 2-2 Create a Pool

©2024 QSAN Technology, Inc. All rights reserved.

- Create a host group.

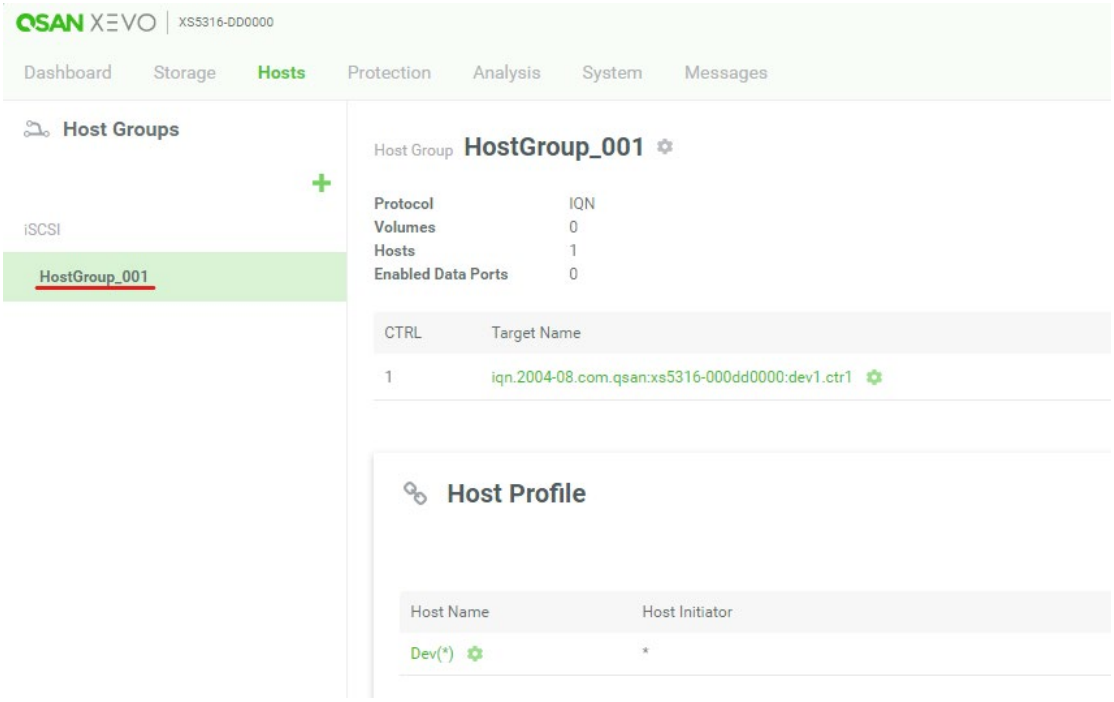


Figure 2-3 Create a Host Group

- Set the IP addresses of iSCSI data ports.

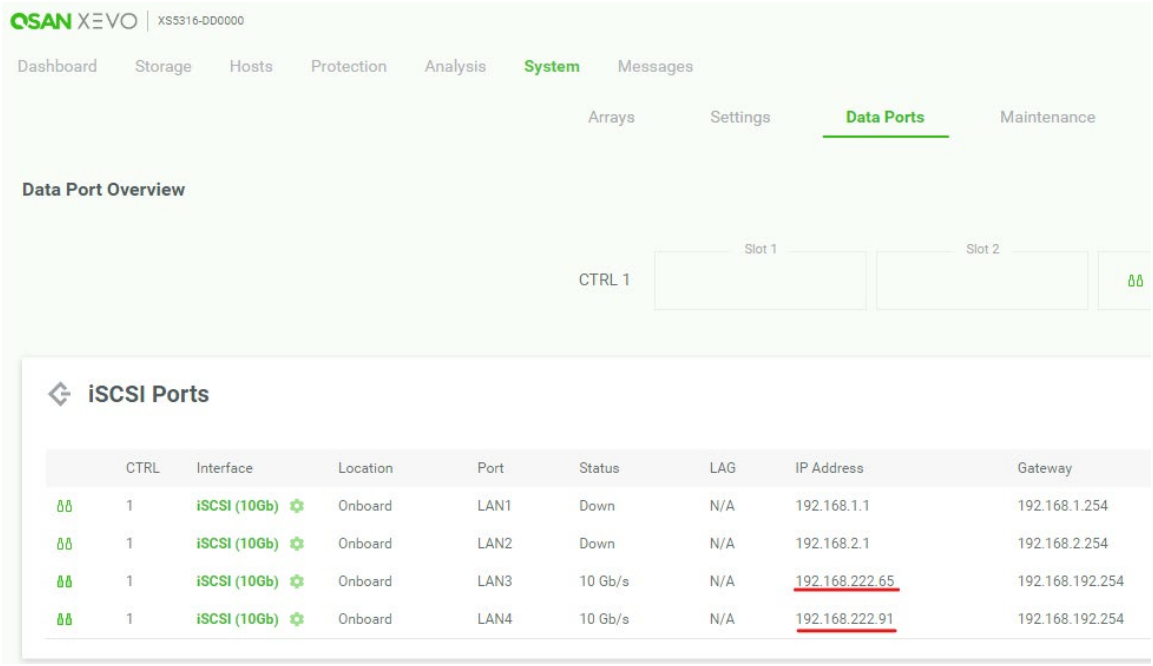


Figure 2-4 Set the IP Addresses of iSCSI Data Ports

©2024 QSAN Technology, Inc. All rights reserved.

7. Apply the **sc.yaml** file and check the status.

```
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example# kubectl apply -f sc.yaml
storageclass.storage.k8s.io/qsan-xevo-storage created
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example# kubectl get sc
NAME                PROVISIONER          RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION
AGE
qsan-xevo-storage   xevo.csi.qsan.com    Delete          Immediate           true
lls
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example#
```

8. Edit the **pvc.yaml** file, first check the **pvc** name, then set the CSI driver size and enter the **storageClassName**.

```
---
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example# vi pvc.yaml
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name:qsan-csi-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
  storageClassName: qsan-xevo-straogel
```

9. Apply the **pvc.yaml** file and check the status.

```
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example# kubectl apply -f pvc.yaml
persistentvolumeclaim/qsan-csi-pvc created
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example# kubectl get pvc
NAME           STATUS    VOLUME    CAPACITY    ACCESS MODES    STORAGECLASS         AGE
qsan-csi-pvc   Pending                                         xevo.csi.qsan.com    6s
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example#
```

10. Edit the **pod.yaml** file, then enter the **pvc** name.

```
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example# vi pod.yaml
```


©2024 QSAN Technology, Inc. All rights reserved.

```
kind: Pod
apiVersion: v1
metadata:
  name: test-pod
spec:
  containers:
  - image: busybox
    command: ["sleep", "99999999"]
    name: test-container
    volumeMounts:
    - mountPath: /test-pv
      name: test-pv-storage
  volumes:
    - name: test-pv-storage
      persistentVolumeClaim:
        claimName: qsan-csi-pvc
~
```

11. Apply the **pod.yaml** file and check the status.

```
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example# kubectl apply -f pod.yaml
pod/test-pod created
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example# kubectl get pod
NAME       READY   STATUS    RESTARTS   AGE
test-pod   0/1     Pending   0          10s
root@qaubuntufunction:/home/qa/csi-qsan/deploy/example#
```

2.3. Setup Multipath Policy

This section describes how to configure multipathing policies in a K8s environment to maximize the performance and reliability of the CSI driver. MPIO (Multi-Path I/O) plays a vital role in enhancing storage performance and redundancy. To set up a multipath policy, follow the steps below.

1. Edit the **multipath.conf** file as follows.

```
root@qak:~/csi-qsan-release/deploy/example# vi /etc/multipath.conf
```


©2024 QSAN Technology, Inc. All rights reserved.

```
blacklist {
    devnode "^sda[0-9]$"
}


defaults {
    path_grouping_policy        multibus
    user_friendly_names no
    find_multipaths no
    getuid_callout      "/lib/udev/scsi_id --replace-whitespace --whitelisted --dev
}
```

2. Run the **multipath** command and then check the device status. Here you can see that the multipath device has been used the CSI driver.

```
root@qak:~/csi-qsan-release/deploy/example# multipath
95615.216515 | sda: using deprecated getuid callout
95615.218200 | /lib/udev/scsi_id exited with 1
95615.218231 | sda: failed to get callout uid: Input/output error
95615.221279 | sdb: using deprecated getuid callout
95615.226831 | sdc: using deprecated getuid callout
root@qak:~/csi-qsan-release/deploy/example# multipath -l
3201100137811d300 dm-1 QSAN,XS5316
size=100G features='0' hwhandler='1 alua' wp=rw
`-+- policy='service-time 0' prio=0 status=active
  |- 4:0:0:0 sdc 8:32 active undef running
  `- 3:0:0:0 sdb 8:16 active undef running
root@qak:~/csi-qsan-release/deploy/example#
```

2.4. Fibre Channel Usage

The setup process used by the CSI driver for FC (Fibre Channel) closely follows the steps outlined for CSI iSCSI. The establishment of FC protocols follows a similar sequence to iSCSI setup, ensuring seamless and efficient integration of FC protocols. Being familiar with iSCSI setup will help you understand and perform the FC setup steps.

The integration of the CSI driver in this article with the FC protocol enables higher IOPS (Input / Output Per Second) and meets stringent requirements for low latency. This setting is customized to meet the requirements of enterprise users operating in virtual machine environments. Additionally, it integrates seamlessly with KubeVirt (a virtual machine management add-on for Kubernetes), simplifying the deployment of virtual machines. The addition of FC provides a fast and effective storage solution that accurately meets your needs for deploying virtual machines in a Kubernetes environment. This enhancement will increase

©2024 QSAN Technology, Inc. All rights reserved.

the performance and flexibility of enterprise users, allowing them to effectively manage and operate their virtualized workloads.

2.5. Supported and Unsupported Types

The figures below illustrate the supported scenarios. "HG" is the abbreviation of Host Group.

- Supported scenario 1

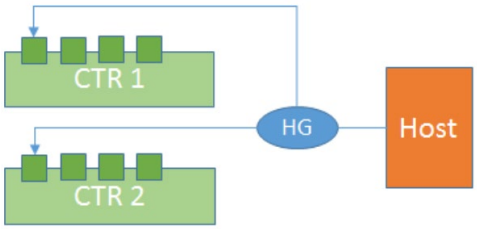


Figure 2-5 Supported Scenario 1

- Supported scenario 2

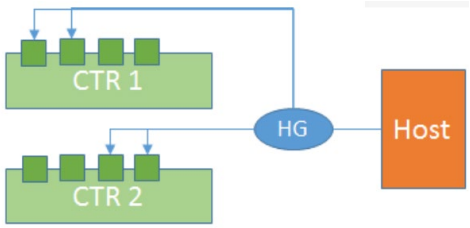


Figure 2-6 Supported Scenario 2

- Supported scenario 3

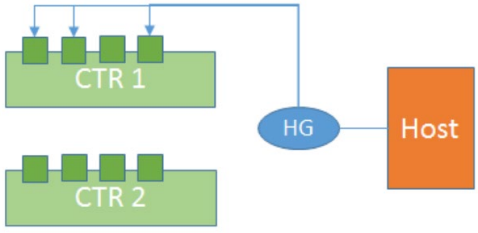


Figure 2-7 Supported Scenario 3

©2024 QSAN Technology, Inc. All rights reserved.

- Supported scenario 4

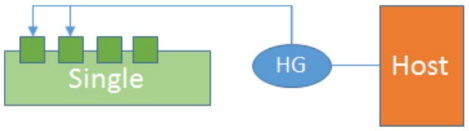


Figure 2-8 Supported Scenario 4

The figure below illustrates the unsupported scenarios.

- Unsupported scenario 1, ports on different host groups.

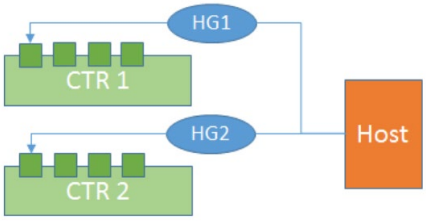


Figure 2-9 Unsupported Scenario 1

2.6. Constraints

1. Please note that thin pool and MPIO is not supported yet while being used with CSI, please using a thick pool and connect the iSCSI connection only to one of the controllers now.
2. Maximum quantity of CSI volume in a Host Group is 32, 'exceed max volume count' will be shown if the error code were checked by 'kubectl describe pod' command.

©2024 QSAN Technology, Inc. All rights reserved.

3. CONCLUSION

Using the QSAN CSI driver with Kubernetes can provide significant benefits such as improved performance, scalability, flexibility, simplified management, and future-proofing of storage needs. However, it is important to consider performance, capacity planning, scalability, and compatibility when using this driver.

In addition, this article also extensively discusses how to install the CSI driver in a K8s environment and leverage MPIO technology to further optimize its performance and reliability. This process not only helps improve storage resource management, but also meets the growing storage needs of applications. Hopefully it will provide you with the knowledge and tools to successfully install CSI drivers for a great storage solution.

©2024 QSAN Technology, Inc. All rights reserved.

4. APPENDIX

4.1. Apply To

- XEVO firmware 2.3.0 and later

4.2. Reference

Web Page

- XS5300 Series Web Page
- XS3300 Series Web Page
- Cloud Native Web Page

Document

- XEVO Software Manual

Tutorial Video

- How to use Wizard to create a Thin pool
- Experience K8s CSI Driver

Software

- Docker Hub
- QSAN CSI

©2024 QSAN Technology, Inc. All rights reserved.