Red Hat OpenShift Container Storage 4.8

Metro-DR 확장 클러스터를 위한 OpenShift Container Storage 구성

클러스터 및 스토리지 관리자를 위한 재해 복구 작업

Last Updated: 2023-02-20

Red Hat OpenShift Container Storage 4.8 Metro-DR 확장 클러스터를 위한 OpenShift Container Storage 구성

클러스터 및 스토리지 관리자를 위한 재해 복구 작업

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

법적 공지

Copyright © 2022 | You need to change the HOLDER entity in the en-US/Configuring_OpenShift_Container_Storage_for_Metro-DR_stretch_cluster.ent file |.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

초록

이 솔루션 가이드의 목적은 OpenShift Container Storage를 Kubernetes 영역 토폴로지 레이블과 함께 배포하여 고가용성 스토리지 인프라를 달성하는 데 필요한 단계와 명령을 자세히 설명하는 것입니다. This is a technology preview feature and is available only for deployment using local storage devices. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

차례

보다 포괄적 수용을 위한 오픈 소스 용어 교체

Red Hat은 코드, 문서, 웹 속성에서 문제가 있는 용어를 교체하기 위해 최선을 다하고 있습니다. 먼저 마스터(master), 슬레이브(slave), 블랙리스트(blacklist), 화이트리스트(whitelist) 등 네 가지 용어를 교체하고 있습니다. 이러한 변경 작업은 작업 범위가 크므로 향후 여러 릴리스에 걸쳐 점차 구현할 예정입니다. 자세한 내용은 CTO Chris Wright의 메시지 를 참조하십시오.

RED HAT 문서에 관한 피드백 제공

문서 개선을 위한 의견을 보내 주십시오. Red Hat이 이를 개선하는 방법을 알려 주십시오. 피드백을 제공하려면 다음을 수행하십시오.

- 특정 문구에 대한 간단한 주석은 다음과 같습니다.
  1. 문서가 *Multi-page HTML* 형식으로 표시되는지 확인합니다. 또한 문서 오른쪽 상단에 **Feedback** (피드백) 버튼이 있는지 확인합니다.
  2. 마우스 커서를 사용하여 주석 처리하려는 텍스트 부분을 강조 표시합니다.
  3. 강조 표시된 텍스트 아래에 표시되는 **피드백 추가** 팝업을 클릭합니다.
  4. 표시된 지침을 따릅니다.
- 보다 상세하게 피드백을 제출하려면 다음과 같이 Bugzilla 티켓을 생성하십시오.
  1. Bugzilla 웹 사이트로 이동하십시오.
  2. Component로 **Documentation**을 선택하십시오.
  3. **Description** 필드에 문서 개선을 위한 제안 사항을 기입하십시오. 관련된 문서의 해당 부분 링크를 알려주십시오.
  4. **Submit Bug**를 클릭하십시오.

1장. 확장 클러스터를 사용한 재해 복구 소개

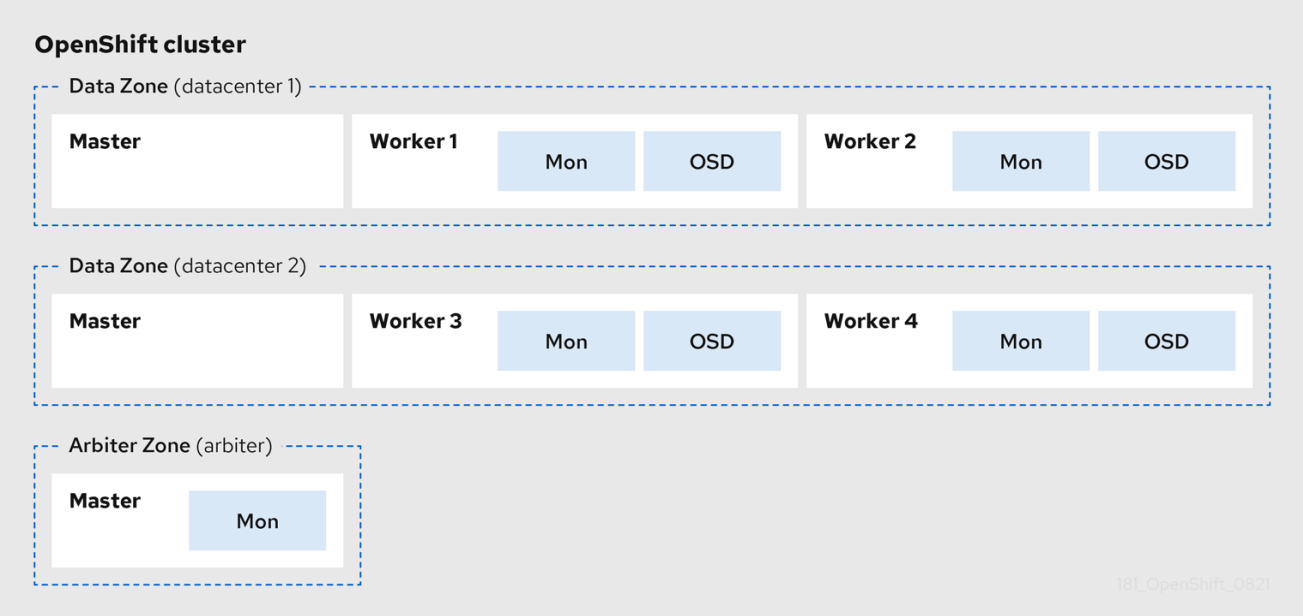
Red Hat OpenShift Container Storage 배포는 스토리지 인프라에 재해 복구 기능을 제공하기 위해 두 곳의 지리적 위치 간에 확장될 수 있습니다. 두 위치 중 하나와 같은 재해가 부분적으로 또는 완전히 사용할 수 없는 경우 OpenShift Container Platform 배포에 배포된 OpenShift Container Storage를 생존할 수 있어야 합니다. 이 솔루션은 인프라 서버 간에 특정 대기 시간 요구 사항이 있는 데이터 센터에서만 사용할 수 있습니다.

참고

현재 확장 클러스터를 사용하는 major-DR 솔루션을 배포할 수 있습니다. 여기서 대기 시간이 다른 위치에 있는 OpenShift Container Platform 노드 간에 4밀리초 왕복 시간(RTT)을 초과하지 않습니다. 대기 시간이 길어질 경우 Red Hat 고객 지원팀에 문의하십시오.

다음 다이어그램은 Metro-DR 확장 클러스터에 대한 가장 간단한 배포를 보여줍니다.

OpenShift 노드 및 OpenShift Container Storage 데몬

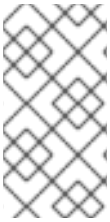


다이어그램에서 OpenShift Container Storage 모니터는 Arbiter 영역에 배포된 Pod에 마스터 노드에 대한 기본 제공 허용 오차가 있습니다. 고가용성 OpenShift Container Platform 컨트롤 플레인에는 마스터 노드가 필요합니다. 또한 한 영역의 OpenShift Container Platform 노드가 다른 두 영역의 OpenShift Container Platform 노드와 연결되어 있어야 합니다.

2장. 재해 복구가 활성화된 스토리지 클러스터 배포 준비

2.1. 메트로 - DR 활성화 요구 사항

- 세 개 이상의 OpenShift Container Platform 마스터 노드가 있는지 확인합니다. 3개의 영역 각각에 있는 하나의 마스터 노드.
- 두 데이터 영역에 걸쳐 4개 이상의 OpenShift Container Platform 작업자 노드가 균등하게 분산되어 있는지 확인합니다.
- 베어 메탈에서 클러스터 확장의 경우 OpenShift Container Platform 마스터 노드의 루트 드라이브로 SSD 드라이브를 사용해야 합니다.
- 각 노드의 레이블이 해당 영역 레이블로 미리 지정되어 있는지 확인합니다. 자세한 내용은 OpenShift Container Platform 노드에 토폴로지 영역 레이블 적용 섹션을 참조하십시오.
- Metro-DR 솔루션은 대기 시간이 영역 간에 2밀리초를 초과하지 않는 경우(4ms RTT 최대값) 배포되도록 설계되었습니다. 대기 시간이 길어질 경우 Red Hat 고객 지원팀에 문의하십시오.



참고

확장 논리가 충돌하는 경우 유연한 확장 및 중재자를 모두 활성화할 수 없습니다. 가변 확장을 사용하면 한 번에 하나의 노드를 OpenShift Container Storage 클러스터에 추가할 수 있습니다. 반면 중재자 클러스터에서는 2개의 데이터 영역 각각에 하나 이상의 노드를 추가해야 합니다.

2.2. OPENSHIFT CONTAINER PLATFORM 노드에 토폴로지 영역 레이블 적용

사이트 중단 중에 중재자 기능이 있는 영역이 중재자 레이블을 사용합니다. 이러한 레이블은 임의의이며 세 위치에 대해 고유해야 합니다.

예를 들어 다음과 같이 노드에 레이블을 지정할 수 있습니다.

```
topology.kubernetes.io/zone=arbiter for Master0

topology.kubernetes.io/zone=datacenter1 for Master1, Worker1, Worker2

topology.kubernetes.io/zone=datacenter2 for Master2, Worker3, Worker4
```

- oc CLI를 사용하여 노드에 라벨을 적용하려면 다음을 수행합니다.

```
$ oc label node <NODENAME> topology.kubernetes.io/zone=<LABEL>
```

- 라벨을 확인하려면 3 영역에 example 레이블을 사용하여 다음 명령을 실행합니다.

```
$ oc get nodes -l topology.kubernetes.io/zone=<LABEL> -o name
```

또는 단일 명령을 실행하여 영역이 있는 모든 노드를 볼 수 있습니다.

```
$ oc get nodes -L topology.kubernetes.io/zone
```

이제 Metro DR 확장 클러스터 토폴로지 영역 레이블이 적절한 OpenShift Container Platform 노드에 적용되어 세 위치를 정의합니다.

다음 단계

- OpenShift Container Platform OperatorHub에서 스토리지 운영자 설치.

2.3. LOCAL STORAGE OPERATOR 설치

Red Hat OpenShift Container Platform Operator Hub를 사용하여 Local Storage Operator를 설치할 수 있습니다.

사전 요구 사항

- cluster-admin 및 Operator 설치 권한이 있는 계정을 사용하여 OpenShift Container Platform 클러스터에 액세스할 수 있습니다.

절차

1. OpenShift 웹 콘솔에 로그인합니다.
2. **Operators → OperatorHub** 를 클릭합니다.
3. **Filter by keyword...** 상자에 **로컬 스토리지를** 입력하여 운영자 목록에서 **Local Storage** Operator를 검색하고 클릭합니다.
4. **설치**를 클릭합니다.
5. **Operator 설치** 페이지에서 다음 옵션을 설정합니다.
   a. channel은 **stable-4.8**.
   b. **클러스터의 특정 네임스페이스로서의 설치 모드입니다**.
   c. **Operator 권장 네임스페이스 openshift-local-storage** 로 설치된 네임스페이스입니다.
   d. **자동** 승인 전략.
6. **설치**를 클릭합니다.

검증 단계

- Local Storage Operator가 **Succeeded** 로 **상태가** 표시되는지 확인합니다.

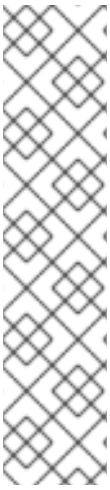
2.4. RED HAT OPENSHIFT CONTAINER STORAGE OPERATOR 설치

Red Hat OpenShift Container Platform Operator Hub를 사용하여 Red Hat OpenShift Container Storage Operator를 설치할 수 있습니다.

사전 요구 사항

- cluster-admin 및 Operator 설치 권한이 있는 계정을 사용하여 OpenShift Container Platform 클러스터에 액세스할 수 있습니다.

- Red Hat OpenShift Container Platform 클러스터의 데이터 센터에 4개 이상의 작업자 노드가 균등하게 배포되어 있습니다.
- 추가 리소스 요구 사항은 배포 계획을 참조하십시오.



참고

- OpenShift Container Storage의 클러스터 전체 기본 노드 선택기를 재정의해야 하는 경우 명령줄 인터페이스에서 다음 명령을 사용하여 **openshift-storage 네임스페이스(이 경우 openshift-storage** 네임스페이스 생성)에 빈 노드 선택기를 지정할 수 있습니다.

```
$ oc annotate namespace openshift-storage openshift.io/node-selector=
```

- 노드를 인프라로 테인트 **하여** Red Hat OpenShift Container Storage 리소스만 해당 노드에 예약되도록 합니다. 이를 통해 서브스크립션 비용을 절감할 수 있습니다. 자세한 내용은 스토리지 리소스 관리 및 할당 가이드 의 Red Hat OpenShift Container Storage 전용 작업자 노드를 사용하는 방법을 참조하십시오.

절차

1. 웹 콘솔에서 **Operator → OperatorHub** 를 클릭합니다.
2. OpenShift Container Storage Operator를 검색하려면 키워드를 Filter by keyword 상자에 입력하거나 스크롤합니다.
3. OpenShift Container Storage 운영자 페이지에서 **Install** (설치)을 클릭합니다.
4. **Operator 설치** 페이지에서 기본적으로 다음과 같은 필수 옵션이 선택됩니다.
   a. 채널을 **stable-4.8** 로 업데이트.
   b. **클러스터의 특정 네임스페이스로서의 설치 모드입니다**.
   c. **Operator 권장 네임스페이스 openshift-storage로 설치된 네임스페이스** 입니다. 네임스페이스 **openshift-storage** 가 없으면 운영자 설치 중에 생성됩니다.
   d. **Approval Strategy** (승인 전략)를 **Automatic** (자동) 또는 **Manual** (수동)으로 선택합니다.
   e. **설치**를 클릭합니다.
   자동 업데이트를 선택한 경우 OLM(Operator Lifecycle Manager)은 개입 없이 Operator의 실행 중인 인스턴스를 자동으로 업그레이드합니다.

   수동 업데이트를 선택한 경우 OLM에서 업데이트 요청을 생성합니다. 클러스터 관리자는 Operator를 새 버전으로 업데이트하려면 OLM 업데이트 요청을 수동으로 승인해야 합니다.

검증 단계

OpenShift Container Storage Operator에 설치에 성공한 녹색 눈금이 표시되는지 확인합니다.

다음 단계

- OpenShift Container Storage 클러스터를 생성합니다.
  자세한 내용은 OpenShift Container Storage 클러스터 생성을 참조하십시오.

3장. OPENSHIFT CONTAINER STORAGE 클러스터 생성

OpenShift Container Storage Operator를 설치한 후 OpenShift Container Storage 클러스터를 생성하려면 다음 절차를 사용하십시오.

사전 요구 사항

- 재해 복구가 활성화된 스토리지 클러스터 배포 준비 의 모든 요구 사항이 충족되었는지 확인합니다.

절차

1. OpenShift 웹 콘솔에 로그인합니다.
2. **Operators → 설치된 Operator**를 클릭하여 설치된 모든 Operator를 확인합니다.
   선택한 프로젝트가 **openshift-storage** 인지 확인합니다.
3. 스토리지 클러스터의 **OpenShift Container Storage → 인스턴스 생성** 링크를 클릭합니다.
4. 모드를 **내부 연결 장치로** 선택합니다.
   Local Storage Operator가 아직 설치되지 않은 경우 설치하라는 메시지가 표시됩니다. **설치를** 클릭하고 Installing Local Storage Operator 에 설명된 대로 절차를 따릅니다.

   스토리지 볼륨 세트를 필터링하여 스토리지를 사용할 전용 스토리지 클래스를 생성할 수 있습니다.
5. 디스크 검색
   a. 데이터 센터 영역에서 연결된 스토리지 장치가 있는 레이블이 지정된 노드를 선택하려면 **Select nodes** (노드 선택) 옵션을 선택합니다.
   선택할 노드가 테인트되어 마법사에서 검색되지 않은 경우 Red Hat Knowledgebase Solution 에서 제공하는 단계를 수행하여 Local Storage Operator 리소스에 대한 허용 오차를 추가합니다.

   선택한 노드가 집계된 30 개의 CPU 및 72GiB RAM의 OpenShift Container Storage 클러스터 요구 사항과 일치하지 않으면 최소 클러스터가 배포됩니다. 최소 노드 요구 사항은 계획 가이드의 리소스 요구 사항 섹션을 참조하십시오.

   b. **다음**을 클릭합니다.
6. 스토리지 클래스 만들기
   a. **로컬 볼륨 세트 이름을 입력합니다.**

   b. **Storage Class Name(스토리지 클래스 이름)**을 입력합니다. 기본적으로 볼륨 세트 이름은 스토리지 클래스 이름에 표시됩니다. 이름을 변경할 수도 있습니다.

   c. 이전 단계에서 디스크 검색에 대해 선택한 노드가 **디스크별 필터** 섹션에 표시됩니다.
   다음 중 하나를 선택합니다.

   - 장치를 검색한 **모든 노드를 선택하는 모든 노드의 디스크**입니다.
   - **선택한 노드** 의 디스크로 장치를 검색한 노드의 하위 집합을 선택합니다. 고가용성을 위해 3개의 다른 물리적 노드, 랙 또는 장애 도메인에 작업자 노드를 분산합니다.

d. 지원되는 구성을 빌드하려면 **SSD 또는 NVMe** 를 선택합니다. 지원되지 않는 테스트 설치에 대해 **HDD를** 선택할 수 있습니다.

e. **Advanced(고급)** 섹션을 확장하고 다음 옵션을 설정합니다.

볼륨 모드	블록은 기본적으로 선택됩니다.
장치 유형	디스크 유형 선택. 기본적으로 Disk(디스크) 및 Part(파트)가 선택됩니다.
디스크 크기	장치를 포함해야 하는 최소 및 최대 사용 가능한 크기입니다.  **참고** 장치에 대해 최소 크기를 100GB로 설정해야 합니다.
최대 디스크 제한	이는 노드에서 생성할 수 있는 최대 PV 수를 나타냅니다. 이 필드를 비워 두면 일치하는 노드에서 사용 가능한 모든 디스크에 PV가 생성됩니다.

f. **다음**을 클릭합니다. 새 스토리지 클래스 생성을 확인하는 팝업이 표시됩니다.

g. **Yes** (예)를 클릭하여 계속합니다.

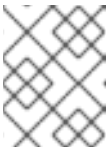
7. **용량 및 노드**설정

a. 확장 클러스터를 사용하려면 **Enable arbiter** 확인란을 선택합니다.

- 사용 가능한 드롭다운 목록에서 the **arbiter 영역을** 선택합니다.

b. **스토리지 클래스** 선택. 기본적으로 이전 단계에서 만든 새 스토리지 클래스가 선택됩니다.

c. **선택한 노드** 는 이전 단계에서 선택한 노드를 표시합니다. 이 목록은 이전 단계에서 검색한 디스크를 표시하는 데 몇 분이 걸립니다.



참고

선택한 노드 섹션에 표시된 각 노드의 영역 레이블을 확인하여 레이블이 올바르게 지정되었는지 확인합니다.

d. **다음**을 클릭합니다.

8. (선택 사항) **보안 및 네트워크** 설정 설정

a. **Enable encryption(암호화 사용)** 확인란을 선택하여 블록 및 파일 스토리지를 암호화합니다.

b. 하나 또는 둘 다 **암호화 수준을** 선택하십시오 :

- **전체 클러스터(블록 및 파일)를 암호화** 하기 위한 클러스터 전체 암호화.
- **암호화가 활성화된 스토리지 클래스를** 사용하여 암호화된 영구 볼륨을 만드는 스토리지 클래스 암호화(블록만 해당).

c. **Connect to an external key management service**확인란을 선택합니다. 이는 클러스터 전체 암호화에 대해 선택 사항입니다.

i. **Key Management Service Provider**는 기본적으로 **Vault** 로 설정됩니다.

ii. Vault **Service Name**, host **Address** of Vault server (https://<hostname 또는 ip>'), **Port number** and **Token** 을 입력합니다.

iii. **고급 설정을 확장하여 Vault 구성에 따라 추가 설정 및 인증서 세부 정보를 입력합니다**

A. OpenShift Container Storage 전용으로 **고유한 백엔드** 경로에 Key Value 보안 경로를 입력합니다.

B. (선택 사항) **TLS 서버 이름과 Vault Enterprise 네임 스페이스를** 입력합니다.

C. 각 PEM 인코딩 **인증서 파일을 업로드하여 CA 인증서, 클라이언트 인증서 및 클라이언트 개인** 키를 제공합니다.

D. **저장**을 클릭합니다.

d. 여러 네트워크 인터페이스를 사용하려는 경우 단일 네트워크 또는 **사용자 지정** (Multus) 네트워크를 사용하는 경우 **Default** (SDN)를 선택합니다.

i. 드롭다운에서 **Public Network Interface** (공용 네트워크 인터페이스)를 선택합니다.

ii. 드롭다운에서 **Cluster Network Interface** (클러스터 네트워크 인터페이스)를 선택합니다.

e. **다음**을 클릭합니다.

9. 구성 세부 정보를 검토합니다. 구성 설정을 수정하려면 **Back(뒤로)**을 클릭하여 이전 구성 페이지로 돌아갑니다.

10. **생성**을 클릭합니다.

검증 단계

1. 설치된 스토리지 클러스터의 최종 **상태가 단계로 표시되는지 확인합니다. 녹색 눈금 표시가 있는 ready** 입니다.

a. **Operators → 설치된 Operator → 스토리지 클러스터** 링크를 클릭하여 스토리지 클러스터 설치 상태를 확인합니다.

b. 또는 Operator **Details(운영자 세부 정보)** 탭에 있는 경우 **Storage Cluster(스토리지 클러스터)** 탭을 클릭하여 상태를 확인할 수 있습니다.

2. **Storage Cluster** (스토리지 클러스터) 탭에서 **ocs-storagecluster** 를 클릭합니다.

a. YAML 탭에서 spec 섹션에서 the **arbiter** 키를 검색하고 다음을 확인합니다.

- 'enable'이 **true** 로 설정되어 있습니다.
- 'arbiterLocation'은 set **arbiter** 입니다.
- 'replica'가 **4**로 설정되어 있습니다.
- 'failureDomain'이 **zone** 으로 설정됩니다.

```
spec:
  arbiter:
    enable: true
```

```
    [..]
    nodeTopologies:
      arbiterLocation: arbiter
        [..]
      replica: 4
status:
    conditions:
    [..]
    failureDomain: zone
    [..]
```

3. OpenShift Container Storage의 모든 구성 요소가 성공적으로 설치되었는지 확인하려면 OpenShift Container Storage 설치 확인을 참조하십시오.

4장. OPENSHIFT CONTAINER STORAGE 배포 확인

이 섹션을 사용하여 OpenShift Container Storage가 올바르게 배포되었는지 확인합니다.

4.1. POD 상태 확인

OpenShift Container Storage의 Pod가 실행 중인지 확인하려면 다음 절차를 따르십시오.

절차

1. OpenShift 웹 콘솔에 로그인합니다.
2. OpenShift 웹 콘솔의 왼쪽 창에서 **워크로드 → 포드**를 클릭합니다.
3. **프로젝트** 드롭다운 목록에서 **openshift-storage** 를 선택합니다.
   각 구성 요소에 대해 예상되는 Pod 수와 노드 수에 따라 달라지는 방법에 대한 자세한 내용은 표 4.1. "OpenShift Container Storage 클러스터에 해당하는 Pod" 을 참조하십시오.
4. **Running** 및 **Completed** 탭을 클릭하여 Pod가 실행 중이고 완료된 상태인지 확인합니다.

표 4.1. OpenShift Container Storage 클러스터에 해당하는 Pod

구성 요소	해당 Pod
OpenShift Container Storage Operator	• **OCS-operator-*** (모든 작업자 노드에 1 Pod) • **ocs-metrics-exporter-***
Rook-ceph Operator	**rook-ceph-operator-*** (모든 작업자 노드에 1 Pod)
Multicloud Object Gateway	• **NooBaa-operator-*** (모든 작업자 노드에 1 Pod) • **NooBaa-core-*** (모든 스토리지 노드에 1 Pod) • **NooBaa-db-*** (1 스토리지 노드의 Pod) • **NooBaa-endpoint-*** (모든 스토리지 노드에 1 Pod)
MON	**rook-ceph-mon-*** (5 Pod는 데이터 센터 영역당 2개, 중재자 영역에서 1개)에 분산됩니다.
MGR	**rook-ceph-mgr-*** (모든 스토리지 노드의 Pod)

구성 요소	해당 Pod
MDS	**rook-ceph-mds-ocs-storagecluster-cephfilesystem-*** (2개의 Pod는 두 개의 데이터 센터 영역에 배포됩니다.
RGW	**rook-ceph-rgw-ocs-storagecluster-cephobjectstore-*** (2개의 Pod는 두 개의 데이터 센터 영역에 배포됩니다.

CSI	• **cephfs**
	  ◦ **CSI-cephfsplugin-*** (모든 작업자 노드에 1 Pod)
	  ◦ **CSI-cephfsplugin-provisioner-*** (작업자 노드에 분산된 2 Pod)
	• **rbd**
	  ◦ **CSI-rbdplugin-*** (각 작업자 노드에서 1 Pod)
	  ◦ **CSI-rbdplugin-provisioner-*** (2 작업자 노드에 분산된 2 Pod)

rook-ceph-crashcollector	**rook-ceph-crashcollector-*** (1개의 스토리지 노드 및 중재 영역의 Pod 1개)
OSD	• **rook-ceph-osd-*** (각 장치의 1 Pod) • **rook-ceph-osd-prepare-ocs-deviceset-*** (각 장치의 1 Pod)

4.2. OPENSHIFT CONTAINER STORAGE 클러스터가 정상인지 확인

OpenShift Container Storage 클러스터가 정상인지 확인하려면 절차의 단계를 따르십시오.

절차

1. **스토리지 → 개요**를 클릭하고 **블록 및 파일** 탭을 클릭합니다.
2. **상태 카드에서** *Storage Cluster* (스토리지 클러스터) 및 *Data Resiliency* (데이터 복원력)에 녹색 눈금이 있는지 확인합니다.
3. **세부 정보 카드에서** 클러스터 정보가 표시되는지 확인합니다.

블록 및 파일 대시보드를 사용하는 OpenShift Container Storage 클러스터의 상태에 대한 자세한 내용은 Monitoring OpenShift Container Storage 를 참조하십시오.

4.3. MULTICLOUD OBJECT GATEWAY가 정상인지 확인

OpenShift Container Storage Multicloud Object Gateway가 정상인지 확인하려면 절차의 단계를 따르십시오.

절차

1. OpenShift 웹 콘솔에서 **스토리지 → 개요**를 클릭하고 **오브젝트** 탭을 클릭합니다.
2. **상태 카드에서** *오브젝트 서비스 및 데이터 복원성 이 모두Ready* **상태(녹색 틱)인지 확인합니다.**
3. **세부 정보 카드에서 Multicloud Object Gateway 정보가 표시되는지 확인합니다.**

오브젝트 서비스 대시보드를 사용하는 OpenShift Container Storage 클러스터의 상태에 대한 자세한 내용은 OpenShift Container Storage 모니터링을 참조하십시오.

4.4. OPENSHIFT CONTAINER STORAGE 특정 스토리지 클래스가 있는지 확인

클러스터에 스토리지 클래스가 있는지 확인하려면 절차의 단계를 따르십시오.

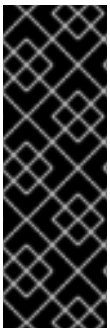
절차

1. **OpenShift 웹 콘솔에서 스토리지 → 스토리지 클래스를 클릭합니다.**
2. **OpenShift Container Storage 클러스터 생성을 사용하여 다음 스토리지 클래스가 생성되었는지 확인합니다.**
   - **ocs-storagecluster-ceph-rbd**
   - **ocs-storagecluster-cephfs**
   - **openshift-storage.noobaa.io**

- **ocs-storagecluster-ceph-rgw**

5장. 영역 인식 샘플 애플리케이션 설치

이 섹션을 사용하여 **OpenShift Container Storage Metro Disaster** 복구 설정을 검증하기 위해 영역 인식 샘플 애플리케이션을 배포합니다.



중요

데이터 영역 간 대기 시간을 사용하면 노드와 영역(예: 동일한 위치에 있는 모든 노드) 간 대기 시간이 짧은 **OpenShift** 클러스터에 비해 성능 저하가 발생할 수 있습니다. 성능이 저하되는 정도는 스토리지(예: 쓰기 트래픽)를 사용하는 영역과 애플리케이션 동작 사이의 대기 시간에 따라 달라집니다. 필요한 서비스 수준에 대한 애플리케이션 성능을 충분히 보장하기 위해 **Metro DR** 클러스터 구성으로 중요한 애플리케이션을 테스트하십시오.

5.1. 영역 인식 샘플 애플리케이션 설치

이 섹션에서는 **ocs-storagecluster-cephfs** 스토리지 클래스를 사용하여 동시에 여러 포드에서 사용할 수 있는 **Read-Write-Many(RWX) PVC**를 생성합니다. 우리가 사용할 애플리케이션을 **File Uploader**라고 합니다.

이 애플리케이션은 파일을 저장하기 위해 동일한 **RWX** 볼륨을 공유하므로 사이트 중단 시 계속 사용할 수 있도록 애플리케이션을 토폴로지 영역에 분산하는 방법을 보여줄 수 있습니다. **OpenShift Container Storage**는 영역 인식과 고가용성을 갖춘 **Metro DR** 확장 클러스터로 구성되어 있기 때문에 영구 데이터 액세스에도 사용됩니다.

1. 새 프로젝트를 생성합니다.

```
oc new-project my-shared-storage
```

2. **file-uploader**라는 예제 **PHP** 애플리케이션을 배포합니다.

```
oc new-app openshift/php:7.2-ubi8~https://github.com/christianh814/openshift-php-upload-demo --name=file-uploader
```

샘플 출력:

```
Found image 4f2dcc0 (9 days old) in image stream "openshift/php" under tag "7.2-ubi8" for "openshift/php:7.2-
ubi8"

Apache 2.4 with PHP 7.2
```

```
----------------------
PHP 7.2 available as container is a base platform for building and running various PHP 7.2
applications and frameworks. PHP is an HTML-embedded scripting language. PHP attempts
to make it easy for developers to write dynamically generated web pages. PHP also offers
built-in database integration for several commercial and non-commercial database
management systems, so writing a database-enabled webpage with PHP is fairly simple.
The most common
use of PHP coding is probably as a replacement for CGI scripts.

Tags: builder, php, php72, php-72

* A source build using source code from https://github.com/christianh814/openshift-php-
upload-demo will be cr
eated
* The resulting image will be pushed to image stream tag "file-uploader:latest"
* Use 'oc start-build' to trigger a new build

--> Creating resources ...
    imagestream.image.openshift.io "file-uploader" created
    buildconfig.build.openshift.io "file-uploader" created
    deployment.apps "file-uploader" created
    service "file-uploader" created
--> Success
    Build scheduled, use 'oc logs -f buildconfig/file-uploader' to track its progress.

    Application is not exposed. You can expose services to the outside world by executing one
or more of the commands below:
     'oc expose service/file-uploader'

    Run 'oc status' to view your app.
```

3. **빌드 로그를 보고 애플리케이션이 배포될 때까지 기다립니다.**

```
oc logs -f bc/file-uploader -n my-shared-storage
```

출력 예:

```
Cloning "https://github.com/christianh814/openshift-php-upload-demo" ...

    [...]
    Generating dockerfile with builder image image-registry.openshift-image-regis
try.svc:5000/openshift/php@sha256:d97466f33999951739a76bce922ab17088885db610c
0e05b593844b41d5494ea
STEP 1: FROM image-registry.openshift-image-registry.svc:5000/openshift/php@s
ha256:d97466f33999951739a76bce922ab17088885db610c0e05b593844b41d5494ea
STEP 2: LABEL "io.openshift.build.commit.author"="Christian Hernandez <christ
ian.hernandez@yahoo.com>"       "io.openshift.build.commit.date"="Sun Oct 1 1
7:15:09 2017 -0700"       "io.openshift.build.commit.id"="288eda3dff43b02f7f7
b6b6b6f93396ffdf34cb2"       "io.openshift.build.commit.ref"="master"       "
io.openshift.build.commit.message"="trying to modularize"       "io.openshift
.build.source-location"="https://github.com/christianh814/openshift-php-uploa
```

```
d-demo"       "io.openshift.build.image"="image-registry.openshift-image-regi
stry.svc:5000/openshift/php@sha256:d97466f33999951739a76bce922ab17088885db610
c0e05b593844b41d5494ea"
STEP 3: ENV OPENSHIFT_BUILD_NAME="file-uploader-1"
OPENSHIFT_BUILD_NAMESP
ACE="my-shared-storage"     OPENSHIFT_BUILD_SOURCE="https://github.com/christ
ianh814/openshift-php-upload-demo"     OPENSHIFT_BUILD_COMMIT="288eda3dff43b0
2f7f7b6b6b6f93396ffdf34cb2"
STEP 4: USER root
STEP 5: COPY upload/src /tmp/src
STEP 6: RUN chown -R 1001:0 /tmp/src
STEP 7: USER 1001
STEP 8: RUN /usr/libexec/s2i/assemble
---> Installing application source...
=> sourcing 20-copy-config.sh ...
---> 17:24:39     Processing additional arbitrary httpd configuration provide
d by s2i ...
=> sourcing 00-documentroot.conf ...
=> sourcing 50-mpm-tuning.conf ...
=> sourcing 40-ssl-certs.sh ...
STEP 9: CMD /usr/libexec/s2i/run
STEP 10: COMMIT temp.builder.openshift.io/my-shared-storage/file-uploader-1:3
b83e447
Getting image source signatures

[...]
```

Push successful(내보내기 성공)이 표시되면 명령 프롬프트에서 tail 모드로 돌아갑니다.

참고

new-app 명령은 git 리포지토리에서 직접 애플리케이션을 배포하고 OpenShift 템플릿을 사용하지 않으므로 OpenShift 경로 리소스는 기본적으로 생성되지 않습니다. 경로를 수동으로 생성해야 합니다.

4개의 복제본으로 확장하고 서비스를 노출하여 애플리케이션 영역을 인식하고 사용할 수 있도록 합니다.

```
oc expose svc/file-uploader -n my-shared-storage
oc scale --replicas=4 deploy/file-uploader -n my-shared-storage
oc get pods -o wide -n my-shared-storage
```

몇 분 내에 4개의 파일 업로더 Pod가 있어야 합니다. Running STATUS에 파일 업로더 Pod 4개가 있을 때까지 위의 명령을 반복합니다.

PersistentVolumeClaim을 생성하여 oc set volume 명령을 사용하여 애플리케이션에 연결할 수 있습

니다. 다음을 실행합니다

```
oc set volume deploy/file-uploader --add --name=my-shared-storage \
-t pvc --claim-mode=ReadWriteMany --claim-size=10Gi \
--claim-name=my-shared-storage --claim-class=ocs-storagecluster-cephfs \
--mount-path=/opt/app-root/src/uploaded \
-n my-shared-storage
```

이 명령은 다음을 수행합니다.

- 영구 볼륨 클레임 만들기
- 볼륨 정의를 포함하도록 애플리케이션 배포를 업데이트합니다.
- 지정된 **mount-path**에 볼륨을 연결하도록 애플리케이션 배포를 업데이트합니다.
- **4** 애플리케이션 포드의 새 배포 발생

이제 볼륨을 추가한 결과를 살펴보겠습니다.

```
oc get pvc -n my-shared-storage
```

출력 예:

```
NAME                STATUS   VOLUME                                     CAPACITY   ACCESS MODES
STORAGECLASS                AGE
my-shared-storage   Bound    pvc-5402cc8a-e874-4d7e-af76-1eb05bd2e7c7   10Gi       RWX
ocs-storagecluster-cephfs   52s
```

ACCESSMODE가 **RWX(ReadWriteMany)**로 설정되어 있는지 확인합니다.

4개의 **file-uploaderPods**가 모두 동일한 **RWX** 볼륨을 사용하고 있습니다. 이 **ACCESSMODE**가 없으면 **OpenShift**는 동일한 영구 볼륨에 여러 포드를 안정적으로 연결하지 않습니다.
RWO(ReadWriteOnce) 영구 볼륨을 사용하는 배포를 확장하려고 하면 **Pod**가 동일한 노드에 공동 배치됩니다.

5.2. 영역 인식으로 배포 수정

현재 **file-uploader** 배포는 영역을 인식하지 않으며 동일한 영역에 있는 모든 **Pod**를 예약할 수 있습니다. 이 경우 사이트 중단이 발생하면 애플리케이션을 사용할 수 없게 됩니다. 자세한 내용은 **Pod** 토폴로지 분배 제약 조건사용을 참조하십시오.

1. 앱 영역을 인식하려면 애플리케이션 배포 구성에 포드 배치 규칙을 추가해야 합니다. 다음 명령을 실행하고 아래 표시된 대로 출력을 검토합니다. 다음 단계에서는 아래 출력의 시작 및 끝 섹션에 표시된 대로 토폴로지 영역 레이블을 사용하도록 배포를 수정합니다.

```
$ oc get deployment file-uploader -o yaml -n my-shared-storage | less
```

출력 예:

```
[...]
spec:
  progressDeadlineSeconds: 600
  replicas: 4
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      deployment: file-uploader
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      annotations:
        openshift.io/generated-by: OpenShiftNewApp
      creationTimestamp: null
      labels:
        deployment: file-uploader
      spec:  # <-- Start inserted lines after here
        containers:  # <-- End inserted lines before here
        - image: image-registry.openshift-image-registry.svc:5000/my-shared-storage/file-
uploader@sha256:a458ea62f990e431ad7d5f84c89e2fa27bdebdd5e29c5418c70c56eb81f0a26
b
          imagePullPolicy: IfNotPresent
          name: file-uploader
[...]
```

2. 배포를 편집하고 위와 같이 시작과 끝 사이에 다음과 같은 새 행을 추가합니다.

```
$ oc edit deployment file-uploader -n my-shared-storage
```

```
[...]
    spec:
      topologySpreadConstraints:
       - labelSelector:
           matchLabels:
             deployment: file-uploader
         maxSkew: 1
         topologyKey: topology.kubernetes.io/zone
         whenUnsatisfiable: DoNotSchedule
       - labelSelector:
           matchLabels:
             deployment: file-uploader
         maxSkew: 1
         topologyKey: kubernetes.io/hostname
         whenUnsatisfiable: ScheduleAnyway
      nodeSelector:
        node-role.kubernetes.io/worker: ""
      containers:
[...]
```

출력 예:

```
deployment.apps/file-uploader edited
```

3. **배포를 0개의 Pod 로 확장한 다음 4개의 Pod로 돌아갑니다. 이는 Pod 배치 측면에서 배포가 변경되었기 때문에 필요합니다.**

```
oc scale deployment file-uploader --replicas=0 -n my-shared-storage
```

출력 예:

```
deployment.apps/file-uploader scaled
```

4개의 Pod로 되돌아갑니다.

```
$ oc scale deployment file-uploader --replicas=4 -n my-shared-storage
```

출력 예:

```
deployment.apps/file-uploader scaled
```

4. **4개의 포드가 datacenter1 및 datacenter2 영역의 4개 노드에 분산되어 있는지 확인합니다.**

```
$ oc get pods -o wide -n my-shared-storage | egrep '^file-uploader'| grep -v build | awk '{print $7}' | sort | uniq -c
```

출력 예:

```
  1 perf1-mz8bt-worker-d2hdm
  1 perf1-mz8bt-worker-k68rv
  1 perf1-mz8bt-worker-ntkp8
  1 perf1-mz8bt-worker-qpwsr
```

```
$ oc get nodes -L topology.kubernetes.io/zone | grep datacenter | grep -v master
```

출력 예:

```
perf1-mz8bt-worker-d2hdm   Ready    worker   35d   v1.20.0+5fbfd19   datacenter1
perf1-mz8bt-worker-k68rv   Ready    worker   35d   v1.20.0+5fbfd19   datacenter1
perf1-mz8bt-worker-ntkp8   Ready    worker   35d   v1.20.0+5fbfd19   datacenter2
perf1-mz8bt-worker-qpwsr   Ready    worker   35d   v1.20.0+5fbfd19   datacenter2
```

5. 브라우저에서 **file-uploader** 웹 애플리케이션을 사용하여 새 파일을 업로드합니다.

   a. 생성된 경로를 찾습니다.

   ```
   $ oc get route file-uploader -n my-shared-storage -o jsonpath --template="http://{.spec.host}{"\n"}"
   ```

   그러면 이 경로와 유사한 경로가 반환됩니다.

   샘플 출력:

   ```
   http://file-uploader-my-shared-storage.apps.cluster-ocs4-abdf.ocs4-abdf.sandbox744.opentlc.com
   ```

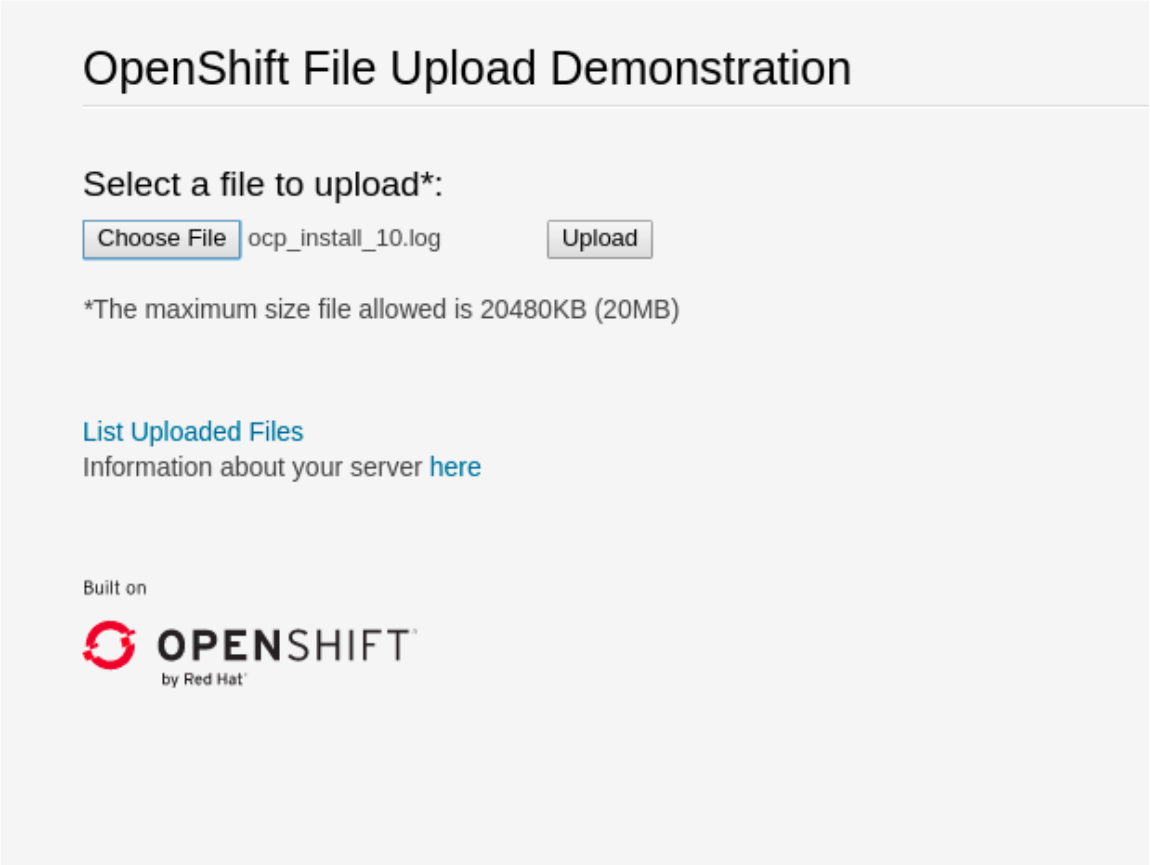
   b. 위의 경로를 사용하여 브라우저에서 웹 애플리케이션을 가리킵니다. 경로는 다릅니다.

   웹 애플리케이션은 업로드한 모든 파일을 나열하고 새 파일을 업로드하고 기존 데이터를 다운로드할 수 있는 기능을 제공합니다. 현재 아무 것도 없습니다.

c.

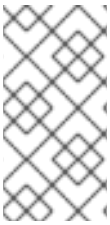
로컬 시스템에서 임의의 파일을 선택하고 앱에 업로드합니다.

그림 5.1. 간단한 PHP 기반 파일 업로드 도구



d.

업로드된 파일 목록을 클릭하여 현재 업로드된 모든 파일 목록을 확인합니다.



참고

OpenShift Container Platform 이미지 레지스트리, 수신 라우팅 및 모니터링 서비스는 영역을 인식하지 않습니다.