



OpenShift Container Platform 4.9

Migration Toolkit for Containers

OpenShift Container Platform 4로 마이그레이션

OpenShift Container Platform 4.9 Migration Toolkit for Containers

OpenShift Container Platform 4로 마이그레이션

법적 공지

Copyright © 2023 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

초록

이 문서에서는 OpenShift Container Platform 4 클러스터 간에 상태 저장 애플리케이션 워크로드를 마이그레이션하는 방법을 설명합니다.

차례

1장. MIGRATION TOOLKIT FOR CONTAINERS 정보	4
1.1. 용어	4
1.2. MTC 워크플로	5
1.3. 데이터 복사 방법 정보	7
1.4. 직접 볼륨 마이그레이션 및 직접 이미지 마이그레이션	9
2장. MIGRATION TOOLKIT FOR CONTAINERS 릴리스 노트	10
2.1. MIGRATION TOOLKIT FOR CONTAINERS 1.6 릴리스 노트	10
2.2. MIGRATION TOOLKIT FOR CONTAINERS 1.5 릴리스 노트	10
2.3. MIGRATION TOOLKIT FOR CONTAINERS 1.4 릴리스 노트	12
3장. MTC 설치	13
3.1. 호환성 지침	13
3.2. OPENSIFT CONTAINER PLATFORM 4.2에서 4.5로 레거시 MIGRATION TOOLKIT FOR CONTAINERS OPERATOR 설치	14
3.3. OPENSIFT CONTAINER PLATFORM 4.9에 MIGRATION TOOLKIT FOR CONTAINERS OPERATOR 설치	15
3.4. 프록시 설정	16
3.5. 복제 리포지토리 구성	20
3.6. MTC 설치 제거 및 리소스 삭제	27
4장. 제한된 네트워크 환경에서 MTC 설치	29
4.1. 호환성 지침	29
4.2. OPENSIFT CONTAINER PLATFORM 4.9에 MIGRATION TOOLKIT FOR CONTAINERS OPERATOR 설치	30
4.3. OPENSIFT CONTAINER PLATFORM 4.2에서 4.5로 레거시 MIGRATION TOOLKIT FOR CONTAINERS OPERATOR 설치	31
4.4. 프록시 설정	33
4.5. 복제 리포지토리 구성	37
4.6. MTC 설치 제거 및 리소스 삭제	38
5장. MIGRATION TOOLKIT FOR CONTAINERS 업그레이드	40
5.1. OPENSIFT CONTAINER PLATFORM 4.9에서 MIGRATION TOOLKIT FOR CONTAINERS 업그레이드	40
5.2. OPENSIFT CONTAINER PLATFORM 버전 4.2에서 4.5로 MIGRATION TOOLKIT FOR CONTAINERS 업그레이드	40
5.3. MTC 1.3을 1.7로 업그레이드	41
6장. 마이그레이션 전 체크리스트	43
6.1. 클러스터 상태 체크리스트	43
6.2. 소스 클러스터 체크리스트	43
6.3. 대상 클러스터 체크리스트	44
7장. 네트워크 고려 사항	45
7.1. DNS 고려 사항	45
7.2. 네트워크 트래픽 리디렉션 전략	46
8장. 애플리케이션 마이그레이션	48
8.1. 마이그레이션 사전 요구 사항	48
8.2. MTC 웹 콘솔을 사용하여 애플리케이션 마이그레이션	49
9장. 고급 마이그레이션 옵션	57
9.1. 용어	57
9.2. 명령줄을 사용하여 애플리케이션 마이그레이션	58
9.3. 마이그레이션 후크	71

9.4. 마이그레이션 계획 옵션	73
9.5. 마이그레이션 컨트롤러 옵션	80
10장. 문제 해결	84
10.1. MTC 워크플로	84
10.2. MTC 사용자 정의 리소스 매니페스트	87
10.3. 로그 및 디버깅 툴	95
10.4. 일반적인 문제 및 우려 사항	105
10.5. 마이그레이션 롤백	110

1장. MIGRATION TOOLKIT FOR CONTAINERS 정보

MTC(Migration Toolkit for Containers)를 사용하면 네임스페이스 단위로 OpenShift Container Platform 4 클러스터 간에 상태 저장 애플리케이션 워크로드를 마이그레이션할 수 있습니다.



참고

OpenShift Container Platform 3에서 마이그레이션하는 경우 OpenShift Container Platform 3 [에서 4로 마이그레이션하고 OpenShift Container Platform 3 에 레거시 Migration Toolkit for Containers Operator 설치 정보를 참조하십시오.](#)

상태 마이그레이션을 사용하여 동일한 클러스터 또는 클러스터 간에 애플리케이션을 마이그레이션할 수 있습니다.

MTC는 마이그레이션을 제어하고 애플리케이션 다운타임을 최소화할 수 있도록 Kubernetes 사용자 지정 리소스를 기반으로 하는 웹 콘솔 및 API를 제공합니다.

MTC 콘솔은 기본적으로 대상 클러스터에 설치되어 있습니다. [원격 클러스터](#)에 콘솔을 설치하도록 Migration Toolkit for Containers Operator를 구성할 수 있습니다.

다음에 대한 자세한 내용은 [고급 마이그레이션 옵션](#)을 참조하십시오.

- 마이그레이션 후크 및 MTC API를 사용하여 마이그레이션 자동화.
- 리소스를 제외하고 대규모 마이그레이션을 지원하며 직접 볼륨 마이그레이션을 위해 자동 PV 크기 조정을 활성화하도록 마이그레이션 계획을 구성.

1.1. 용어

표 1.1. MTC 용어

용어	정의
소스 클러스터	애플리케이션이 마이그레이션되는 클러스터입니다.
대상 클러스터 ^[1]	애플리케이션이 마이그레이션될 대상 클러스터입니다.
복제 리포지토리	간접 마이그레이션 중 또는 직접 볼륨 마이그레이션 또는 직접 이미지 마이그레이션 중에 Kubernetes 오브젝트에 대한 이미지, 볼륨 및 Kubernetes 오브젝트 복사에 사용되는 오브젝트 스토리지입니다. 복제 리포지토리는 모든 클러스터에서 액세스할 수 있어야 합니다.
호스트 클러스터	migration-controller pod 및 웹 콘솔이 실행 중인 클러스터입니다. host 클러스터는 일반적으로 대상 클러스터이지만 필수는 아닙니다. 호스트 클러스터에 직접 이미지 마이그레이션을 위해 노출된 레지스트리 경로가 필요하지 않습니다.

용어	정의
원격 클러스터	원격 클러스터는 일반적으로 소스 클러스터이지만 필수는 아닙니다. 원격 클러스터에는 migration-controller 서비스 계정 토큰이 포함된 Secret 사용자 정의 리소스가 필요합니다. 원격 클러스터에는 직접 이미지 마이그레이션을 위해 노출된 보안 레지스트리 경로가 필요합니다.
간접 마이그레이션	이미지, 볼륨 및 Kubernetes 오브젝트는 소스 클러스터에서 복제 리포지토리로 복사한 다음 복제 리포지토리에 대상 클러스터로 복사됩니다.
직접 볼륨 마이그레이션	영구 볼륨은 소스 클러스터에서 대상 클러스터로 직접 복사됩니다.
직접 이미지 마이그레이션	이미지가 소스 클러스터에서 대상 클러스터로 직접 복사됩니다.
마이그레이션 단계	애플리케이션을 중지하지 않고 데이터가 대상 클러스터에 복사됩니다. 단계적 마이그레이션을 여러 번 실행하면 컷오버 마이그레이션 기간이 단축됩니다.
컷오버 마이그레이션	소스 클러스터에서 애플리케이션이 중지되고 해당 리소스가 대상 클러스터로 마이그레이션됩니다.
상태 마이그레이션	애플리케이션 상태는 특정 영구 볼륨 클레임을 대상 클러스터에 복사하여 마이그레이션됩니다.
마이그레이션 롤백	마이그레이션 롤백은 완료된 마이그레이션을 롤백합니다.

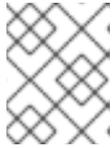
¹ MTC 웹 콘솔에서 *대상* 클러스터를 호출합니다.

1.2. MTC 워크플로

MTC(Migration Toolkit for Containers) 웹 콘솔 또는 Kubernetes API를 사용하여 Kubernetes 리소스, 영구 볼륨 데이터 및 내부 컨테이너 이미지를 OpenShift Container Platform 4.9로 마이그레이션할 수 있습니다.

(MTC)는 다음 리소스를 마이그레이션합니다.

- 마이그레이션 계획에 지정된 네임스페이스입니다.
- 네임스페이스가 지정된 리소스: MTC가 네임스페이스를 마이그레이션하면 서비스 또는 포트와 같은 해당 네임스페이스와 연결된 모든 오브젝트와 리소스가 마이그레이션됩니다. 또한 네임스페이스에 존재하지만 클러스터 수준에 없는 리소스가 클러스터 수준에 존재하는 리소스에 따라 달라지는 경우 MTC가 두 개의 리소스 모두를 마이그레이션합니다.
예를 들어 SCC(보안 컨텍스트 제약 조건)는 클러스터 수준에 존재하는 리소스이며, 서비스 계정(SA)은 네임스페이스 수준에 존재하는 리소스입니다. MTC가 마이그레이션하는 네임스페이스에 SA가 있는 경우 MTC는 SA에 연결된 모든 SCC를 자동으로 찾고 해당 SCC도 마이그레이션합니다. 마찬가지로 MTC는 네임스페이스의 영구 볼륨 클레임에 연결된 영구 볼륨을 마이그레이션합니다.



참고

리소스에 따라 클러스터 범위 리소스를 수동으로 마이그레이션해야 할 수 있습니다.

- CR(사용자 정의 리소스) 및 CRD(사용자 정의 리소스 정의): MTC는 네임스페이스 수준에서 CR 및 CRD를 자동으로 마이그레이션합니다.

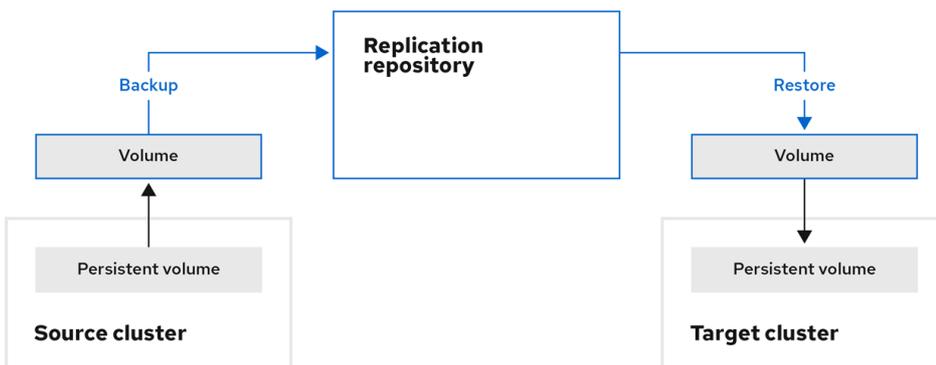
MTC 웹 콘솔을 사용하여 애플리케이션을 마이그레이션하는 데는 다음 단계가 포함됩니다.

1. 모든 클러스터에 Migration Toolkit for Containers Operator를 설치합니다.
인터넷 액세스가 제한되거나 없는 제한된 환경에서 Migration Toolkit for Containers Operator를 설치할 수 있습니다. 소스 및 대상 클러스터는 상호 액세스 권한 및 미리 레지스트리에 대한 네트워크 액세스 권한이 있어야 합니다.
2. MTC가 데이터를 마이그레이션하는 데 사용하는 중간 오브젝트 스토리지인 복제 리포지토리를 구성합니다.
소스 및 대상 클러스터는 마이그레이션 중에 복제 리포지토리에 대한 네트워크 액세스 권한이 있어야 합니다. 프록시 서버를 사용하는 경우 복제 리포지토리와 클러스터 간의 네트워크 트래픽을 허용하도록 해당 서버를 구성해야 합니다.
3. MTC 웹 콘솔에 소스 클러스터를 추가합니다.
4. MTC 웹 콘솔에 복제 리포지토리를 추가합니다.
5. 다음 데이터 마이그레이션 옵션 중 하나를 사용하여 마이그레이션 계획을 생성합니다.
 - **복사:** MTC는 소스 클러스터에서 복제 리포지토리로, 복제 리포지토리에서 대상 클러스터로 데이터를 복사합니다.



참고

직접 이미지 마이그레이션 또는 직접 볼륨 마이그레이션을 사용하는 경우 소스 클러스터에서 대상 클러스터로 이미지 또는 볼륨이 직접 복사됩니다.



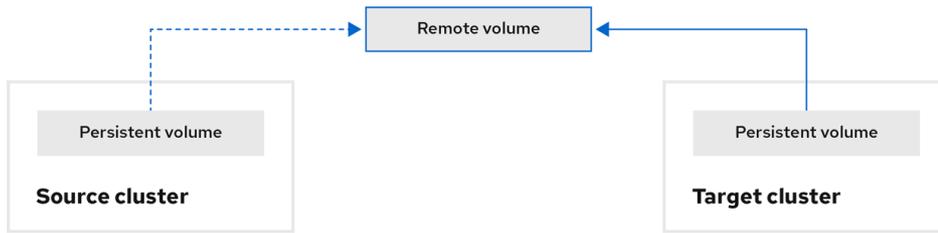
OpenShift_45_1019

- **이동:** MTC는 소스 클러스터에서 원격 볼륨(예: NFS)을 마운트 해제하고 원격 볼륨을 가리키는 대상 클러스터에 PV 리소스를 만든 다음 대상 클러스터에 원격 볼륨을 마운트합니다. 대상 클러스터에서 실행되는 애플리케이션은 소스 클러스터와 동일한 원격 볼륨을 사용합니다. 소스 및 대상 클러스터가 원격 볼륨에 액세스할 수 있어야 합니다.



참고

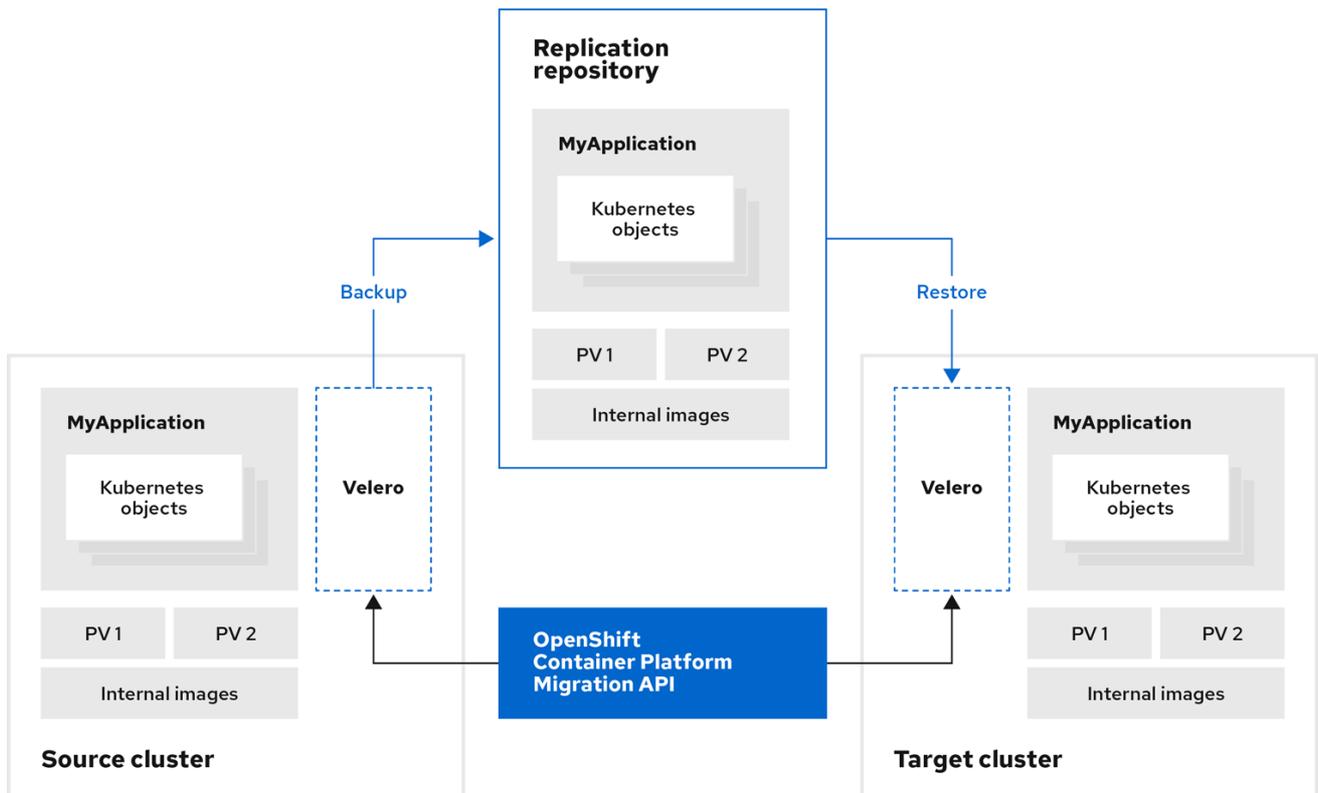
이 다이어그램에는 복제 리포지토리가 나타나지 않지만 마이그레이션에는 필수입니다.



OpenShift_45_1019

6. 다음 옵션 중 하나를 사용하여 마이그레이션 계획을 실행합니다.

- 스태이지 (Stage)**는 애플리케이션을 중지하지 않고 데이터를 대상 클러스터에 복사합니다. 스테이지 마이그레이션을 마이그레이션 전에 대부분의 데이터가 대상에 복사되도록 여러 번 실행할 수 있습니다. 하나 이상의 단계적 마이그레이션을 실행하면 컷오버 마이그레이션 기간이 단축됩니다.
- 컷오버 (Cutover)**는 소스 클러스터에서 애플리케이션을 중지하고 리소스를 대상 클러스터로 이동합니다.
 선택 사항: 마이그레이션 중에 소스 클러스터에서 트랜잭션 중지 확인란의 선택을 취소할 수 있습니다.



OpenShift_45_1019

1.3. 데이터 복사 방법 정보

Migration Toolkit for Containers(MTC)는 소스 클러스터에서 대상 클러스터로 데이터를 마이그레이션하기 위한 파일 시스템 및 스냅샷 데이터 복사 방법을 지원합니다. 환경에 적합하고 스토리지 공급자가 지원하는 방법을 선택할 수 있습니다.

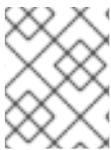
1.3.1. 파일 시스템 복사 방법

MTC는 소스 클러스터에서 복제 리포지토리로 데이터 파일을 복사하고 다시 대상 클러스터로 복사합니다.

파일 시스템 복사 방법은 간접 마이그레이션에 Restic을 사용하거나 직접 볼륨 마이그레이션에 Rsync를 사용합니다.

표 1.2. 파일 시스템 복사 방법 요약

혜택	제한
<ul style="list-style-type: none"> 클러스터는 다른 스토리지 클래스를 보유할 수 있습니다. 모든 S3 스토리지 공급자에 대해 지원됨. 체크섬을 통한 선택적 데이터 확인. 성능이 크게 증가하는 직접 볼륨 마이그레이션을 지원합니다. 	<ul style="list-style-type: none"> 스냅샷 복사 방법보다 느림. 선택적 데이터 확인으로 성능이 크게 저하합니다.



참고

Restic 및 Rsync PV 마이그레이션은 지원되는 PV가 **volumeMode=filesystem** 인 것으로 가정합니다. 파일 시스템 마이그레이션에는 **volumeMode=Block** 을 사용할 수 없습니다.

1.3.2. 스냅샷 복사 방법

MTC는 소스 클러스터 데이터의 스냅샷을 클라우드 공급자의 복제 리포지토리에 복사합니다. 대상 클러스터에서 데이터가 복원됩니다.

스냅샷 복사 방법은 Amazon Web Services, Google Cloud Provider 및 Microsoft Azure와 함께 사용할 수 있습니다.

표 1.3. 스냅샷 복사 방법 요약

혜택	제한
----	----

혜택	제한
<ul style="list-style-type: none"> ● 파일 시스템 복사 방법보다 빠름. 	<ul style="list-style-type: none"> ● 클라우드 공급자는 스냅샷을 지원해야 합니다. ● 클러스터는 동일한 클라우드 공급자에 있어야 합니다. ● 클러스터는 동일한 위치 또는 지역에 있어야 합니다. ● 클러스터는 동일한 스토리지 클래스를 보유해야 합니다. ● 스토리지 클래스는 스냅샷과 호환 가능해야 합니다. ● 직접 볼륨 마이그레이션을 지원하지 않습니다.

1.4. 직접 볼륨 마이그레이션 및 직접 이미지 마이그레이션

직접 이미지 마이그레이션(DIM) 및 직접 볼륨 마이그레이션(DVM)을 사용하여 소스 클러스터에서 대상 클러스터로 직접 이미지 및 데이터를 마이그레이션할 수 있습니다.

다른 가용성 영역에 있는 노드에서 DVM을 실행하는 경우 마이그레이션된 Pod가 영구 볼륨 클레임에 액세스할 수 없기 때문에 마이그레이션이 실패할 수 있습니다.

DIM 및 DVM은 소스 클러스터에서 복제 리포지토리로 파일을 백업하고 복제 리포지토리에서 대상 클러스터로 파일을 복원하는 중간 단계를 건너뛰기 때문에 상당한 성능 이점을 지닙니다. [Rsync](#)를 사용하여 데이터를 전송합니다.

DIM과 DVM에는 다른 사전 요구 사항이 있습니다.

2장. MIGRATION TOOLKIT FOR CONTAINERS 릴리스 노트

MTC(Migration Toolkit for Containers)를 사용하면 네임스페이스 단위로 OpenShift Container Platform 클러스터 간에 애플리케이션 워크로드를 마이그레이션할 수 있습니다.

[OpenShift Container Platform 3에서 4.9](#) 및 OpenShift Container Platform 4 클러스터 간에 마이그레이션할 수 있습니다.

MTC는 마이그레이션을 제어하고 애플리케이션 다운타임을 최소화할 수 있도록 Kubernetes 사용자 지정 리소스를 기반으로 하는 웹 콘솔 및 API를 제공합니다.

2.1. MIGRATION TOOLKIT FOR CONTAINERS 1.6 릴리스 노트

MTC(Migration Toolkit for Containers) 버전 1.6 릴리스 노트에서는 새로운 기능 및 개선 사항, 사용되지 않는 기능 및 알려진 문제에 대해 설명합니다.

2.1.1. 새로운 기능 및 개선 사항

이 릴리스에는 다음과 같은 새로운 기능 및 개선 사항이 있습니다.

- 상태 마이그레이션: 특정 PVC(영구 볼륨 클레임)를 선택하여 반복 가능한 상태 전용 마이그레이션을 수행할 수 있습니다.
- "새 운영자 버전 사용 가능" 알림: MTC 웹 콘솔의 클러스터 페이지에 새 Migration Toolkit for Containers Operator가 사용 가능한 경우 알림이 표시됩니다.

2.1.2. 더 이상 사용되지 않는 기능

더 이상 사용되지 않는 기능은 다음과 같습니다.

- MTC 버전 1.4는 더 이상 지원되지 않습니다.

2.1.3. 확인된 문제

이 릴리스에는 다음과 같은 알려진 문제가 있습니다.

- OpenShift Container Platform 3.10에서 **MigrationController** Pod는 재시작하는 데 시간이 너무 오래 걸립니다. Bugzilla 보고서에는 해결방법이 포함되어 있습니다. ([BZ#1986796](#))
- IBM Cloud의 기존 OpenShift Container Platform 소스 클러스터에서 직접 볼륨을 마이그레이션하는 동안 **스태이지** Pod가 실패합니다. IBM 블록 스토리지 플러그인에서는 동일한 노드의 여러 포트에 동일한 볼륨을 마운트할 수 없습니다. 결과적으로 PVC를 Rsync Pod와 애플리케이션 Pod에 동시에 마운트할 수 없습니다. 이 문제를 해결하려면 마이그레이션 전에 애플리케이션 포드를 중지합니다. ([BZ#1887526](#))
- AWS gp2 PVC에 사용 가능한 공간이 없는 경우 **MigPlan** 사용자 정의 리소스에 경고가 표시되지 않습니다. ([BZ#1963927](#))
- IBM Cloud용 블록 스토리지는 동일한 가용성 영역에 있어야 합니다. [가상 사설 클라우드용 블록 스토리지는 IBM FAQ](#)를 참조하십시오.

2.2. MIGRATION TOOLKIT FOR CONTAINERS 1.5 릴리스 노트

MTC(Migration Toolkit for Containers) 버전 1.5의 릴리스 노트는 새로운 기능, 개선 사항 및 알려진 문제에 대해 설명합니다.

2.2.1. 새로운 기능 및 개선 사항

이 릴리스에는 다음과 같은 새로운 기능 및 개선 사항이 있습니다.

- 웹 콘솔의 **마이그레이션 세부 정보** 페이지의 **마이그레이션 리소스** 트리는 마이그레이션 모니터링 및 디버깅을 위한 추가 리소스, Kubernetes 이벤트 및 실시간 상태 정보로 향상되었습니다.
- 웹 콘솔은 수백 개의 마이그레이션 계획을 지원할 수 있습니다.
- 소스 네임스페이스는 마이그레이션 계획의 다른 대상 네임스페이스에 매핑될 수 있습니다. 이전에는 소스 네임스페이스가 동일한 이름의 대상 네임스페이스에 매핑되었습니다.
- 상태 정보가 있는 후크 단계는 마이그레이션 중에 웹 콘솔에 표시됩니다.
- 직접 볼륨 마이그레이션 중에 Rsync 재시도 시도 횟수가 웹 콘솔에 표시됩니다.
- 직접 볼륨 마이그레이션에 대해 영구 볼륨(PV) 크기 조정을 활성화하여 대상 클러스터에 디스크 공간이 부족하지 않도록 할 수 있습니다.
- PV 크기 조정을 트리거하는 임계값을 구성할 수 있습니다. 이전에는 디스크 사용량이 97%를 초과하면 PV 크기 조정이 발생했습니다.
- Velero가 버전 1.6으로 업데이트되어 다양한 수정 사항 및 개선 사항을 제공합니다.
- 캐시된 Kubernetes 클라이언트를 활성화하여 성능을 향상시킬 수 있습니다.

2.2.2. 더 이상 사용되지 않는 기능

더 이상 사용되지 않는 기능은 다음과 같습니다.

- MTC 버전 1.2 및 1.3은 더 이상 지원되지 않습니다.
- **oc convert** 명령이 더 이상 사용되지 않으므로 사용되지 않는 API를 업데이트하는 절차는 설명서의 문제 해결 섹션에서 제거되었습니다.

2.2.3. 확인된 문제

이 릴리스에는 다음과 같은 알려진 문제가 있습니다.

- 400개 이상의 마이그레이션 계획을 생성하는 경우 Microsoft Azure 스토리지를 사용할 수 없습니다. **MigStorage** 사용자 지정 리소스에는 **The request is being throttled as the limit has been reached for operation type** 메시지가 표시됩니다. ([BZ#1977226](#))
- 마이그레이션에 실패하면 마이그레이션 계획에서 quiesced Pod에 대한 사용자 정의 PV(영구 볼륨) 설정이 유지되지 않습니다. 마이그레이션을 수동으로 롤백하고 마이그레이션 계획을 삭제하고 PV 설정으로 새 마이그레이션 계획을 생성해야 합니다. ([BZ#1784899](#))
- **pv_resizing_threshold**가 42% 이상인 경우가 아니면 PV 크기 조정은 AWS gp2 스토리지에 대해 예상대로 작동하지 않습니다. ([BZ#1973148](#))
- 다음 시나리오의 OpenShift Container Platform 3.7 및 3.9 소스 클러스터에서 PV 크기 조정이 작동하지 않습니다.

- MTC를 설치한 후 애플리케이션이 설치되었습니다.
- MTC를 설치한 후 애플리케이션 pod가 다른 노드에 다시 예약되었습니다.
OpenShift Container Platform 3.7 및 3.9는 Velero가 **Restic** Pod에 자동으로 PV를 마운트할 수 있도록 하는 Mount Propagation 기능을 지원하지 않습니다. **MigAnalytic** CR(사용자 정의 리소스)은 **Restic** Pod에서 PV 데이터를 수집하지 못하고 리소스를 **0**으로 보고합니다. **MigPlan** CR에는 다음과 유사한 상태가 표시됩니다.

출력 예

```
status:
  conditions:
  - category: Warn
    lastTransitionTime: 2021-07-15T04:11:44Z
    message: Failed gathering extended PV usage information for PVs [nginx-logs nginx-
html], please see MigAnalytic openshift-migration/ocp-24706-basicvolmig-migplan-
1626319591-szwd6 for details
    reason: FailedRunningDf
    status: "True"
    type: ExtendedPVAnalysisFailed
```

PV 크기 조정을 활성화하려면 소스 클러스터에서 Restic 데몬 세트를 수동으로 다시 시작하거나 애플리케이션과 동일한 노드에서 **Restic** Pod를 다시 시작할 수 있습니다. Restic을 다시 시작하지 않으면 PV 크기 조정 없이 직접 볼륨 마이그레이션을 실행할 수 있습니다. ([BZ#1982729](#))

2.2.4. 기술 변경 사항

이 릴리스에는 다음과 같은 기술 변경 사항이 있습니다.

- 기존 Migration Toolkit for Containers Operator 버전 1.5.1은 OpenShift Container Platform 버전 3.7에서 4.5로 수동으로 설치됩니다.
- Migration Toolkit for Containers Operator 버전 1.5.1은 Operator Lifecycle Manager를 사용하여 OpenShift Container Platform 버전 4.6 이상에 설치됩니다.

2.3. MIGRATION TOOLKIT FOR CONTAINERS 1.4 릴리스 노트

MTC(Migration Toolkit for Containers) 버전 1.4의 릴리스 노트는 새로운 기능, 개선 사항 및 알려진 문제에 대해 설명합니다.

2.3.1. 확인된 문제

- MTC 1.4.6은 OpenShift Container Platform 3.9 ([BZ#1981794](#)) 또는 3.10([BZ#1981537](#))에 배포할 수 없습니다.

3장. MTC 설치

OpenShift Container Platform 4에 MTC(Migration Toolkit for Containers)를 설치할 수 있습니다.



참고

OpenShift Container Platform 3에 MTC를 설치하려면 [OpenShift Container Platform 3에 레거시 Migration Toolkit for Containers Operator](#) 설치를 참조하십시오.

기본적으로 MTC 웹 콘솔 및 **Migration Controller** Pod는 대상 클러스터에서 실행됩니다. [원격 클러스터](#)에서 MTC 웹 콘솔 및 **Migration Controller** Pod를 실행하도록 **Migration Controller** 사용자 정의 리소스 매니페스트를 구성할 수 있습니다.

MTC를 설치한 후에는 복제 리포지토리로 사용할 오브젝트 스토리지를 구성해야 합니다.

MTC를 [설치 제거하려면 MTC 설치 및 리소스 삭제](#)를 참조하십시오.

3.1. 호환성 지침

OpenShift Container Platform 버전과 호환되는 MTC(Migration Toolkit for Containers) Operator를 설치해야 합니다.

정의

기존 플랫폼

OpenShift Container Platform 4.5 및 이전 버전.

최신 플랫폼

OpenShift Container Platform 4.6 이상

기존 Operator

레거시 플랫폼을 위해 설계된 MTC Operator입니다.

최신 Operator

최신 플랫폼을 위해 설계된 MTC Operator입니다.

클러스터 제어

MTC 컨트롤러 및 GUI를 실행하는 클러스터입니다.

원격 클러스터

Velero를 실행하는 마이그레이션의 소스 또는 대상 클러스터입니다. Control Cluster는 Velero API를 통해 원격 클러스터와 통신하여 마이그레이션을 구동합니다.

표 3.1. MTC 호환성: 레거시 플랫폼에서 마이그레이션

OpenShift Container Platform 4.5 또는 이전

OpenShift Container Platform 4.6 이상

	OpenShift Container Platform 4.5 또는 이전	OpenShift Container Platform 4.6 이상
안정적인 MTC 버전	<p>MTC 1.7.z</p> <p>legacy 1.7 operator: operator.yml 파일을 사용하여 수동으로 설치합니다.</p> <div style="display: flex; align-items: center;">  <p>중요</p> </div> <p>이 클러스터는 제어 클러스터가 될 수 없습니다.</p>	<p>MTC 1.7.z</p> <p>OLM과 함께 설치, 릴리스 채널 release-v1.7</p>



참고

현대 클러스터가 마이그레이션에 관련된 다른 클러스터에 연결하지 못하도록 하는 에지 사례가 있습니다. 예를 들어 온프레미스의 OpenShift Container Platform 3.11 클러스터에서 클라우드의 최신 OpenShift Container Platform 클러스터로 마이그레이션할 때 최신 클러스터가 OpenShift Container Platform 3.11 클러스터에 연결할 수 없습니다.

MTC 1.7을 사용하면 네트워크 제한으로 인해 원격 클러스터 중 하나가 컨트롤 클러스터와 통신할 수 없는 경우 **crane tunnel-api** 명령을 사용합니다.

안정적인 MTC 릴리스에서는 항상 최신 클러스터를 제어 클러스터로 지정해야 하지만 이 특정 경우 레거시 클러스터를 제어 클러스터로 지정하고 워크로드를 원격 클러스터로 푸시할 수 있습니다.

3.2. OPENSIFT CONTAINER PLATFORM 4.2에서 4.5로 레거시 MIGRATION TOOLKIT FOR CONTAINERS OPERATOR 설치

OpenShift Container Platform 버전 4.2에서 4.5로 레거시 Migration Toolkit for Containers Operator를 수동으로 설치할 수 있습니다.

사전 요구 사항

- 모든 클러스터에서 **cluster-admin** 권한이 있는 사용자로 로그인합니다.
- **registry.redhat.io**에 대한 액세스 권한이 있어야 합니다.
- **podman**이 설치되어 있어야 합니다.

절차

1. Red Hat Customer Portal 자격 증명을 사용하여 **registry.redhat.io**에 로그인합니다.

```
$ sudo podman login registry.redhat.io
```

2. 다음 명령을 입력하여 **operator.yml** 파일을 다운로드합니다.

```
$ sudo podman cp $(sudo podman create \
registry.redhat.io/rhmtc/openshift-migration-legacy-rhel8-operator:v1.7):/operator.yml ./
```

3. 다음 명령을 입력하여 **controller.yml** 파일을 다운로드합니다.

```
$ sudo podman cp $(sudo podman create \
registry.redhat.io/rhmtc/openshift-migration-legacy-rhel8-operator:v1.7):/controller.yml ./
```

4. 소스 클러스터에 로그인합니다.
5. 클러스터가 **registry.redhat.io**로 인증할 수 있는지 확인합니다.

```
$ oc run test --image registry.redhat.io/ubi8 --command sleep infinity
```

6. Migration Toolkit for Containers Operator 오브젝트 생성:

```
$ oc create -f operator.yml
```

출력 예

```
namespace/openshift-migration created
rolebinding.rbac.authorization.k8s.io/system:deployers created
serviceaccount/migration-operator created
customresourcedefinition.apiextensions.k8s.io/migrationcontrollers.migration.openshift.io
created
role.rbac.authorization.k8s.io/migration-operator created
rolebinding.rbac.authorization.k8s.io/migration-operator created
clusterrolebinding.rbac.authorization.k8s.io/migration-operator created
deployment.apps/migration-operator created
Error from server (AlreadyExists): error when creating "./operator.yml":
rolebindings.rbac.authorization.k8s.io "system:image-builders" already exists 1
Error from server (AlreadyExists): error when creating "./operator.yml":
rolebindings.rbac.authorization.k8s.io "system:image-pullers" already exists
```

- 1** **Error from server (AlreadyExists)** 메시지를 무시할 수 있습니다. 이는 Migration Toolkit for Containers Operator가 이후 릴리스에서 제공되는 OpenShift Container Platform 4 이전 버전에 대한 리소스를 생성하기 때문에 발생합니다.

7. **MigrationController** 오브젝트를 만듭니다.

```
$ oc create -f controller.yml
```

8. MTC pod가 실행 중인지 확인합니다.

```
$ oc get pods -n openshift-migration
```

3.3. OPENSIFT CONTAINER PLATFORM 4.9에 MIGRATION TOOLKIT FOR CONTAINERS OPERATOR 설치

Operator Lifecycle Manager를 사용하여 OpenShift Container Platform 4.9에 Migration Toolkit for Containers Operator를 설치합니다.

사전 요구 사항

- 모든 클러스터에서 **cluster-admin** 권한이 있는 사용자로 로그인합니다.

절차

- OpenShift Container Platform 웹 콘솔에서 **Operator** → **OperatorHub**를 클릭합니다.
- 키워드로 필터링 필드를 사용하여 **Migration Toolkit for Containers Operator**를 찾습니다.
- Migration Toolkit for Containers Operator**를 선택하고 **설치**를 클릭합니다.
- 설치**를 클릭합니다.
설치된 **Operators** 페이지에서 **Migration Toolkit for Containers Operator**는 **openshift-migration** 프로젝트에 **Succeeded** 상태로 나타납니다.
- Migration Toolkit for Containers Operator**를 클릭합니다.
- 제공된 **API** 아래에서 **마이그레이션 컨트롤러** 타일을 찾고 **인스턴스 작성**을 클릭합니다.
- 생성**을 클릭합니다.
- 워크로드** → **Pod**를 클릭하여 MTC pod가 실행 중인지 확인합니다.

3.4. 프록시 설정

OpenShift Container Platform 4.1 및 이전 버전의 경우 이러한 버전은 클러스터 전체 프록시 오브젝트를 지원하지 않기 때문에 Migration Toolkit for Containers Operator를 설치한 후 **MigrationController** CR(사용자 정의 리소스) 매니페스트에서 **proxy**를 구성해야 합니다.

OpenShift Container Platform 4.2 ~ 4.9의 경우 MTC(Migration Toolkit for Containers)는 클러스터 전체 프록시 설정을 상속합니다. 클러스터 전체 프록시 설정을 재정의하려면 프록시 매개변수를 변경할 수 있습니다.

3.4.1. 직접 볼륨 마이그레이션

MTC 1.4.2에서 직접 볼륨 마이그레이션(DVM)이 도입되었습니다. DVM은 하나의 프록시만 지원합니다. 대상 클러스터가 프록시 뒤에 있는 경우 소스 클러스터는 대상 클러스터의 경로에 액세스할 수 없습니다.

프록시 뒤에서 소스 클러스터에서 DVM을 수행하려면 전송 계층에서 작동하는 TCP 프록시를 구성하고 자체 SSL 인증서로 암호를 해독하고 재암호화하지 않고도 SSL 연결을 투명하게 전달해야 합니다. Stunnel 프록시는 이러한 프록시의 예입니다.

3.4.1.1. DVM용 TCP 프록시 설정

TCP 프록시를 통해 소스와 대상 클러스터 간에 직접 연결하고 프록시를 사용하도록 **MigrationController** CR에서 **stunnel_tcp_proxy** 변수를 구성할 수 있습니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigrationController
metadata:
  name: migration-controller
  namespace: openshift-migration
```

```
spec:
  [...]
  stunnel_tcp_proxy: http://username:password@ip:port
```

직접 볼륨 마이그레이션(DVM)은 프록시에 대한 기본 인증만 지원합니다. 또한 DVM은 TCP 연결을 투명하게 터널링할 수 있는 프록시 뒤에서만 작동합니다. 중간자 모드에서 HTTP/HTTPS 프록시가 작동하지 않습니다. 기존 클러스터 전체 프록시에서 이 동작을 지원하지 않을 수 있습니다. 결과적으로 DVM의 프록시 설정은 의도적으로 MTC의 일반 프록시 구성과 다르게 유지됩니다.

3.4.1.2. HTTP/HTTPS 프록시 대신 TCP 프록시를 사용하는 이유는 무엇입니까?

OpenShift 경로를 통해 소스와 대상 클러스터 간에 Rsync를 실행하여 DVM을 활성화할 수 있습니다. 트래픽은 TCP 프록시인 Stunnel을 사용하여 암호화됩니다. 소스 클러스터에서 실행되는 Stunnel은 대상 Stunnel을 사용한 TLS 연결을 시작하고 암호화된 채널을 통해 데이터를 전송합니다.

OpenShift의 클러스터 전체 HTTP/HTTPS 프록시는 일반적으로 자체 TLS 세션을 외부 서버와 협상하는 중간자 모드로 구성됩니다. 그러나 이 작업은 Stunnel에서는 작동하지 않습니다. Stunnel은 프록시에서 TLS 세션을 그대로 전환해야 하므로 기본적으로 프록시를 통해 TCP 연결을 그대로 전달하는 투명한 터널로 프록시를 설정해야 합니다. 따라서 TCP 프록시를 사용해야 합니다.

3.4.1.3. 알려진 문제

마이그레이션 실패 오류 업그레이드 요청 필요

마이그레이션 컨트롤러는 SPDY 프로토콜을 사용하여 원격 Pod 내에서 명령을 실행합니다. 원격 클러스터가 프록시 또는 SPDY 프로토콜을 지원하지 않는 방화벽 뒤에 있는 경우 마이그레이션 컨트롤러가 원격 명령을 실행하지 못합니다. 오류 메시지 **업그레이드 요청** 과 함께 마이그레이션이 실패합니다. 해결방법: SPDY 프로토콜을 지원하는 프록시를 사용합니다.

SPDY 프로토콜 지원 외에도 프록시 또는 방화벽은 **Upgrade** HTTP 헤더를 API 서버에 전달해야 합니다. 클라이언트는 이 헤더를 사용하여 API 서버와의 websocket 연결을 엽니다. **업그레이드** 헤더가 프록시 또는 방화벽에 의해 차단된 경우 마이그레이션은 오류 메시지 **업그레이드 요청 필요** 와 함께 실패합니다. 해결방법: 프록시가 **Upgrade** 헤더를 전달하도록 합니다.

3.4.2. 마이그레이션을 위한 네트워크 정책 튜닝

OpenShift는 클러스터에서 사용하는 네트워크 플러그인을 기반으로 *NetworkPolicy* 또는 *EgressFirewall* 을 사용하여 Pod로 트래픽을 제한할 수 있습니다. 마이그레이션과 관련된 소스 네임스페이스가 이러한 메커니즘을 사용하여 네트워크 트래픽을 포드로 제한하는 경우 제한이 마이그레이션 중에 Rsync pod로의 트래픽을 실수로 중지할 수 있습니다.

소스 및 대상 클러스터에서 둘 다 실행되는 rsync Pod는 OpenShift 경로를 통해 서로 연결되어야 합니다. 기존 *NetworkPolicy* 또는 *EgressNetworkPolicy* 오브젝트는 이러한 트래픽 제한에서 Rsync Pod를 자동으로 제외하도록 구성할 수 있습니다.

3.4.2.1. NetworkPolicy 구성

3.4.2.1.1. Rsync Pod의 송신 트래픽

소스 또는 대상 네임스페이스의 **NetworkPolicy** 구성이 이러한 유형의 트래픽을 차단하는 경우 Rsync Pod의 고유한 레이블을 사용하여 송신 트래픽이 해당 트래픽에서 전달되도록 허용할 수 있습니다. 다음 정책은 네임스페이스의 Rsync Pod에서 모든 송신 트래픽을 허용합니다.

```
apiVersion: networking.k8s.io/v1
```

```

kind: NetworkPolicy
metadata:
  name: allow-all-egress-from-rsync-pods
spec:
  podSelector:
    matchLabels:
      owner: directvolumemigration
      app: directvolumemigration-rsync-transfer
  egress:
  - {}
  policyTypes:
  - Egress

```

3.4.2.1.2. Rsync pod로의 수신 트래픽

```

apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-all-egress-from-rsync-pods
spec:
  podSelector:
    matchLabels:
      owner: directvolumemigration
      app: directvolumemigration-rsync-transfer
  ingress:
  - {}
  policyTypes:
  - Ingress

```

3.4.2.2. EgressNetworkPolicy 구성

EgressNetworkPolicy 오브젝트 또는 *Egress Firewalls* 는 클러스터를 나가는 송신 트래픽을 차단하도록 설계된 OpenShift 구조입니다.

NetworkPolicy 오브젝트와 달리 Egress Firewall은 네임스페이스의 모든 포트에 적용되므로 프로젝트 수준에서 작동합니다. 따라서 Rsync Pod의 고유 레이블은 제한 사항에서 Rsync Pod만 제외하지 않습니다. 그러나 두 클러스터 간에 직접 연결을 설정할 수 있도록 소스 또는 대상 클러스터의 CIDR 범위를 정책의 허용 규칙에 추가할 수 있습니다.

Egress Firewall이 있는 클러스터를 기반으로 다른 클러스터의 CIDR 범위를 추가하여 둘 사이의 송신 트래픽을 허용할 수 있습니다.

```

apiVersion: network.openshift.io/v1
kind: EgressNetworkPolicy
metadata:
  name: test-egress-policy
  namespace: <namespace>
spec:
  egress:
  - to:
      cidrSelector: <cidr_of_source_or_target_cluster>
    type: Deny

```

3.4.2.3. 데이터 전송을 위한 대체 끝점 선택

기본적으로 DVM은 OpenShift Container Platform 경로를 끝점으로 사용하여 PV 데이터를 대상 클러스터로 전송합니다. 클러스터 토폴로지가 허용하는 경우 다른 유형의 지원되는 끝점을 선택할 수 있습니다.

각 클러스터에 대해 **MigrationController** CR의 적절한 대상 클러스터에서 **rsync_endpoint_type** 변수를 설정하여 끝점을 구성할 수 있습니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigrationController
metadata:
  name: migration-controller
  namespace: openshift-migration
spec:
  [...]
  rsync_endpoint_type: [NodePort|ClusterIP|Route]
```

3.4.2.4. Rsync Pod에 대한 추가 그룹 구성

PVC에서 공유 스토리지를 사용하는 경우 Pod에서 액세스를 허용하기 위해 Rsync Pod 정의에 추가 그룹을 추가하여 해당 스토리지에 대한 액세스를 구성할 수 있습니다.

표 3.2. Rsync Pod의 보조 그룹

변수	유형	기본값	설명
src_supplemental_groups	string	설정되지 않음	소스 Rsync Pod에 대한 콤마로 구분된 추가 그룹 목록
target_supplemental_groups	string	설정되지 않음	대상 Rsync Pod에 대한 콤마로 구분된 추가 그룹 목록

사용 예

MigrationController CR을 업데이트하여 이러한 추가 그룹에 대한 값을 설정할 수 있습니다.

```
spec:
  src_supplemental_groups: "1000,2000"
  target_supplemental_groups: "2000,3000"
```

3.4.3. 프록시 구성

사전 요구 사항

- 모든 클러스터에서 **cluster-admin** 권한이 있는 사용자로 로그인합니다.

절차

- MigrationController** CR 매니페스트를 가져옵니다.

```
$ oc get migrationcontroller <migration_controller> -n openshift-migration
```

2. 프록시 매개변수를 업데이트합니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: MigrationController
metadata:
  name: <migration_controller>
  namespace: openshift-migration
...
spec:
  stunnel_tcp_proxy: http://<username>:<password>@<ip>:<port> 1
  noProxy: example.com 2

```

- 1 직접 볼륨 마이그레이션을 위한 Stunnel 프록시 URL입니다.
- 2 프록시를 제외할 대상 도메인 이름, 도메인, IP 주소 또는 기타 네트워크 CIDR의 쉼표로 구분된 목록입니다.

하위 도메인과 일치하려면 도메인 앞에 `.`을 입력합니다. 예를 들어, `.y.com`은 `x.y.com`과 일치하지만 `y.com`은 일치하지 않습니다. `*`를 사용하여 모든 대상에 대해 프록시를 바이패스합니다. `networking.machineNetwork[].cidr` 필드에 의해 정의된 네트워크에 포함되어 있지 않은 작업자를 설치 구성에서 확장하려면 연결 문제를 방지하기 위해 이 목록에 해당 작업자를 추가해야 합니다.

`httpProxy` 또는 `httpsProxy` 필드가 설정되지 않은 경우 이 필드는 무시됩니다.

3. 매니페스트를 `migration-controller.yaml`로 저장합니다.
4. 업데이트된 매니페스트를 적용합니다.

```
$ oc replace -f migration-controller.yaml -n openshift-migration
```

자세한 내용은 [클러스터 전체 프록시 구성](#) 을 참조하십시오.

3.5. 복제 리포지토리 구성

복제 리포지토리로 사용할 오브젝트 스토리지를 구성해야 합니다. MTC(Migration Toolkit for Containers)는 소스 클러스터에서 복제 리포지토리로 데이터를 복사한 다음 복제 리포지토리에서 대상 클러스터로 데이터를 복사합니다.

MTC는 소스 클러스터에서 대상 클러스터로 데이터를 마이그레이션하기 위한 [파일 시스템 및 스냅샷 데이터 복사 방법](#)을 지원합니다. 환경에 적합하며 스토리지 공급자가 지원하는 방법을 선택합니다.

MTC는 다음과 같은 스토리지 제공자를 지원합니다.

- [Multicloud Object Gateway](#)
- [Amazon Web Services S3](#)
- [Google Cloud Platform](#)
- [Microsoft Azure Blob](#)
- 일반 S3 오브젝트 스토리지(예: Minio 또는 Ceph S3)

3.5.1. 사전 요구 사항

- 모든 클러스터에는 복제 리포지토리에 대한 중단없는 네트워크 액세스 권한이 있어야 합니다.
- 내부 호스팅 복제 리포지토리와 함께 프록시 서버를 사용하는 경우 프록시가 복제 리포지토리에 액세스할 수 있는지 확인해야 합니다.

3.5.2. Multicloud Object Gateway 자격 증명 검색

MCG를 MTC(Migration Toolkit for Containers)의 복제 리포지토리로 구성하려면 MCG(Multicloud Object Gateway) 인증 정보 및 S3 끝점을 검색해야 합니다. OADP(Data Protection) API에 대한 **Secret** 사용자 정의 리소스(CR)를 생성하려면 MCG(Multicloud Object Gateway) 자격 증명을 검색해야 합니다.

MCG는 OpenShift Container Storage의 구성 요소입니다.

사전 요구 사항

- 적절한 OpenShift Container Storage 배포 [가이드](#)를 사용하여 [OpenShift Container Storage](#)를 배포해야 합니다.

절차

1. **NooBaa** 사용자 지정 리소스에서 **describe** 명령을 실행하여 S3 엔드포인트, **AWS_ACCESS_KEY** 및 **AWS_SECRET_ACCESS_KEY**를 가져옵니다.
이러한 인증 정보를 사용하여 MCG를 복제 리포지토리로 추가합니다.

3.5.3. Amazon Web Services 구성

AWS(Amazon Web Services) S3 오브젝트 스토리를 MTC(Migration Toolkit for Containers)의 복제 리포지토리로 구성합니다.

사전 요구 사항

- **AWS CLI**가 설치되어 있어야 합니다.
- 소스 및 대상 클러스터에서 AWS S3 스토리지 버킷에 액세스할 수 있어야 합니다.
- 스냅샷 복사 방법을 사용하는 경우:
 - EC2 EBS(Elastic Block Storage)에 액세스할 수 있어야 합니다.
 - 소스 및 대상 클러스터는 동일한 지역에 있어야 합니다.
 - 소스 및 대상 클러스터는 동일한 스토리지 클래스를 보유해야 합니다.
 - 스토리지 클래스는 스냅샷과 호환 가능해야 합니다.

프로세스

1. **BUCKET** 변수를 설정합니다.

```
$ BUCKET=<your_bucket>
```

2. **REGION** 변수를 설정합니다.

```
$ REGION=<your_region>
```

3. AWS S3 버킷을 생성합니다.

```
$ aws s3api create-bucket \
  --bucket $BUCKET \
  --region $REGION \
  --create-bucket-configuration LocationConstraint=$REGION 1
```

- 1 **us-east-1** 은 **LocationConstraint**를 지원하지 않습니다. 해당 리전이 **us-east-1** 인 경우 **--create-bucket-configuration LocationConstraint=\$REGION** 을 생략합니다.

4. IAM 사용자를 생성합니다.

```
$ aws iam create-user --user-name velero 1
```

- 1 Velero를 사용하여 S3 버킷이 여러 개 있는 여러 클러스터를 백업하려면 각 클러스터에 고유한 사용자 이름을 생성합니다.

5. a **velero-policy.json** 파일을 생성합니다.

```
$ cat > velero-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeVolumes",
        "ec2:DescribeSnapshots",
        "ec2:CreateTags",
        "ec2:CreateVolume",
        "ec2:CreateSnapshot",
        "ec2>DeleteSnapshot"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3>DeleteObject",
        "s3:PutObject",
        "s3:AbortMultipartUpload",
        "s3:ListMultipartUploadParts"
      ],
      "Resource": [
        "arn:aws:s3:::${BUCKET}/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
```

```

        "s3:ListBucket",
        "s3:GetBucketLocation",
        "s3:ListBucketMultipartUploads"
    ],
    "Resource": [
        "arn:aws:s3:::${BUCKET}"
    ]
}
]
}
EOF

```

6. 정책을 연결하여 **velero** 사용자에게 필요한 최소 권한을 부여합니다.

```

$ aws iam put-user-policy \
  --user-name velero \
  --policy-name velero \
  --policy-document file://velero-policy.json

```

7. **velero** 사용자에게 대한 액세스 키를 생성합니다.

```

$ aws iam create-access-key --user-name velero

```

출력 예

```

{
  "AccessKey": {
    "UserName": "velero",
    "Status": "Active",
    "CreateDate": "2017-07-31T22:24:41.576Z",
    "SecretAccessKey": <AWS_SECRET_ACCESS_KEY>,
    "AccessKeyId": <AWS_ACCESS_KEY_ID>
  }
}

```

AWS_SECRET_ACCESS_KEY 및 **AWS_ACCESS_KEY_ID** 를 기록합니다. 인증 정보를 사용하여 AWS를 복제 리포지토리로 추가합니다.

3.5.4. GCP(Google Cloud Platform) 구성

GCP(Google Cloud Platform) 스토리지 버킷을 MTC(Migration Toolkit for Containers)의 복제 리포지토리로 구성합니다.

사전 요구 사항

- **gcloud** 및 **gsutil** CLI 툴이 설치되어 있어야 합니다. 자세한 내용은 [Google 클라우드 설명서를 참조하십시오.](#)
- 소스 및 대상 클러스터에서 GCP 스토리지 버킷에 액세스할 수 있어야 합니다.
- 스냅샷 복사 방법을 사용하는 경우:
 - 소스 및 대상 클러스터는 동일한 지역에 있어야 합니다.

- 소스 및 대상 클러스터는 동일한 스토리지 클래스를 보유해야 합니다.
- 스토리지 클래스는 스냅샷과 호환 가능해야 합니다.

프로세스

1. GCP에 로그인합니다.

```
$ gcloud auth login
```

2. **BUCKET** 변수를 설정합니다.

```
$ BUCKET=<bucket> 1
```

- 1 버킷 이름을 지정합니다.

3. 스토리지 버킷을 생성합니다.

```
$ gsutil mb gs://$BUCKET/
```

4. **PROJECT_ID** 변수를 활성 프로젝트로 설정합니다.

```
$ PROJECT_ID=$(gcloud config get-value project)
```

5. 서비스 계정을 생성합니다.

```
$ gcloud iam service-accounts create velero \
  --display-name "Velero service account"
```

6. 서비스 계정을 나열합니다.

```
$ gcloud iam service-accounts list
```

7. 이메일 값과 일치하도록 **SERVICE_ACCOUNT_EMAIL** 변수를 설정합니다.

```
$ SERVICE_ACCOUNT_EMAIL=$(gcloud iam service-accounts list \
  --filter="displayName:Velero service account" \
  --format 'value(email)')
```

8. 정책을 연결하여 **velero** 사용자에게 필요한 최소 권한을 부여합니다.

```
$ ROLE_PERMISSIONS=(
  compute.disks.get
  compute.disks.create
  compute.disks.createSnapshot
  compute.snapshots.get
  compute.snapshots.create
  compute.snapshots.useReadOnly
  compute.snapshots.delete
  compute.zones.get
)
```

9. **velero.server** 사용자 정의 역할을 생성합니다.

```
$ gcloud iam roles create velero.server \
  --project $PROJECT_ID \
  --title "Velero Server" \
  --permissions "$(IFS=","; echo "${ROLE_PERMISSIONS[*]}")"
```

10. 프로젝트에 IAM 정책 바인딩을 추가합니다.

```
$ gcloud projects add-iam-policy-binding $PROJECT_ID \
  --member serviceAccount:$SERVICE_ACCOUNT_EMAIL \
  --role projects/$PROJECT_ID/roles/velero.server
```

11. IAM 서비스 계정을 업데이트합니다.

```
$ gsutil iam ch serviceAccount:$SERVICE_ACCOUNT_EMAIL:objectAdmin gs://${BUCKET}
```

12. IAM 서비스 계정 키를 현재 디렉터리의 **credentials-velero** 파일에 저장합니다.

```
$ gcloud iam service-accounts keys create credentials-velero \
  --iam-account $SERVICE_ACCOUNT_EMAIL
```

credentials-velero 파일을 사용하여 GCP를 복제 리포지토리로 추가합니다.

3.5.5. Microsoft Azure 구성

Microsoft Azure Blob 스토리지 컨테이너를 MTC(Migration Toolkit for Containers)의 복제 리포지토리로 구성합니다.

사전 요구 사항

- [Azure CLI](#) 가 설치되어 있어야 합니다.
- Azure Blob 스토리지 컨테이너는 소스 및 대상 클러스터에 액세스할 수 있어야 합니다.
- 스냅샷 복사 방법을 사용하는 경우:
 - 소스 및 대상 클러스터는 동일한 지역에 있어야 합니다.
 - 소스 및 대상 클러스터는 동일한 스토리지 클래스를 보유해야 합니다.
 - 스토리지 클래스는 스냅샷과 호환 가능해야 합니다.

프로세스

1. Azure에 로그인합니다.

```
$ az login
```

2. **AZURE_RESOURCE_GROUP** 변수를 설정합니다.

```
$ AZURE_RESOURCE_GROUP=Velero_Backups
```

- Azure 리소스 그룹을 생성합니다.

```
$ az group create -n $AZURE_RESOURCE_GROUP --location CentralUS 1
```

- 1 위치를 지정합니다.

- AZURE_STORAGE_ACCOUNT_ID** 변수를 설정합니다.

```
$ AZURE_STORAGE_ACCOUNT_ID="velero$(uuidgen | cut -d '-' -f5 | tr '[A-Z]' '[a-z]')
```

- Azure 스토리지 계정을 생성합니다.

```
$ az storage account create \
  --name $AZURE_STORAGE_ACCOUNT_ID \
  --resource-group $AZURE_RESOURCE_GROUP \
  --sku Standard_GRS \
  --encryption-services blob \
  --https-only true \
  --kind BlobStorage \
  --access-tier Hot
```

- BLOB_CONTAINER** 변수를 설정합니다.

```
$ BLOB_CONTAINER=velero
```

- Azure Blob 스토리지 컨테이너를 생성합니다.

```
$ az storage container create \
  -n $BLOB_CONTAINER \
  --public-access off \
  --account-name $AZURE_STORAGE_ACCOUNT_ID
```

- velero**에 대한 서비스 주체 및 자격 증명을 생성합니다.

```
$ AZURE_SUBSCRIPTION_ID=`az account list --query '[?isDefault].id' -o tsv` \
  AZURE_TENANT_ID=`az account list --query '[?isDefault].tenantId' -o tsv` \
  AZURE_CLIENT_SECRET=`az ad sp create-for-rbac --name "velero" \
  --role "Contributor" --query 'password' -o tsv` \
  AZURE_CLIENT_ID=`az ad sp list --display-name "velero" \
  --query '[0].appId' -o tsv`
```

- credentials-velero** 파일에 서비스 주체 자격 증명을 저장합니다.

```
$ cat << EOF > ./credentials-velero
AZURE_SUBSCRIPTION_ID=${AZURE_SUBSCRIPTION_ID}
AZURE_TENANT_ID=${AZURE_TENANT_ID}
AZURE_CLIENT_ID=${AZURE_CLIENT_ID}
AZURE_CLIENT_SECRET=${AZURE_CLIENT_SECRET}
AZURE_RESOURCE_GROUP=${AZURE_RESOURCE_GROUP}
AZURE_CLOUD_NAME=AzurePublicCloud
EOF
```

credentials-velero 파일을 사용하여 Azure를 복제 리포지토리로 추가합니다.

3.5.6. 추가 리소스

- [MTC 워크플로](#)
- [데이터 복사 방법 정보](#)
- [MTC 웹 콘솔에 복제 리포지토리 추가](#)

3.6. MTC 설치 제거 및 리소스 삭제

MTC(Migration Toolkit for Containers)를 설치 제거하고 해당 리소스를 삭제하여 클러스터를 정리할 수 있습니다.



참고

velero CRD를 삭제하면 클러스터에서 Velero가 제거됩니다.

사전 요구 사항

- **cluster-admin** 권한이 있는 사용자로 로그인해야 합니다.

절차

1. 모든 클러스터에서 **MigrationController** CR(사용자 정의 리소스)을 삭제합니다.

```
$ oc delete migrationcontroller <migration_controller>
```

2. Operator Lifecycle Manager를 사용하여 OpenShift Container Platform 4에서 Migration Toolkit for Containers Operator를 설치 제거합니다.
3. 다음 명령을 실행하여 모든 클러스터에서 클러스터 범위 리소스를 삭제합니다.

- **migration** CRD(사용자 정의 리소스 정의):

```
$ oc delete $(oc get crds -o name | grep 'migration.openshift.io')
```

- **Velero** CRD:

```
$ oc delete $(oc get crds -o name | grep 'velero')
```

- **migration** 클러스터 역할:

```
$ oc delete $(oc get clusterroles -o name | grep 'migration.openshift.io')
```

- **migration-operator** 클러스터 역할:

```
$ oc delete clusterrole migration-operator
```

- **Velero** 클러스터 역할:

```
$ oc delete $(oc get clusterroles -o name | grep 'velero')
```

-
- **migration** 클러스터 역할 바인딩:

```
$ oc delete $(oc get clusterrolebindings -o name | grep 'migration.openshift.io')
```
- **migration-operator** 클러스터 역할 바인딩:

```
$ oc delete clusterrolebindings migration-operator
```
- **Velero** 클러스터 역할 바인딩:

```
$ oc delete $(oc get clusterrolebindings -o name | grep 'velero')
```

4장. 제한된 네트워크 환경에서 MTC 설치

다음 절차를 수행하여 제한된 네트워크 환경에서 OpenShift Container Platform 4에 MTC(Migration Toolkit for Containers)를 설치할 수 있습니다.

1. [미러링된 Operator 카탈로그](#)를 생성합니다.
이 프로세스에서는 [registry.redhat.io](#) 이미지와 미리 레지스트리 이미지 간의 매핑을 포함하는 **mapping.txt** 파일을 생성합니다. OpenShift Container Platform 4.2에서 4.5 소스 클러스터에 레거시 Migration Toolkit for Containers Operator를 설치하는 데 **mapping.txt** 파일이 필요합니다.
2. Operator Lifecycle Manager를 사용하여 OpenShift Container Platform 4.9 대상 클러스터에 Migration Toolkit for Containers Operator를 설치합니다.
기본적으로 MTC 웹 콘솔 및 **Migration Controller** Pod는 대상 클러스터에서 실행됩니다. [원격 클러스터에서](#) MTC 웹 콘솔 및 **Migration Controller** Pod를 실행하도록 **Migration Controller** 사용자 정의 리소스 매니페스트를 구성할 수 있습니다.
3. 소스 클러스터에 Migration Toolkit for Containers Operator를 설치합니다.
 - OpenShift Container Platform 4.6 이상: Operator Lifecycle Manager를 사용하여 Migration Toolkit for Containers Operator를 설치합니다.
 - OpenShift Container Platform 4.2에서 4.5로: 명령줄 인터페이스에서 레거시 Migration Toolkit for Containers Operator를 설치합니다.
4. 복제 리포지토리로 사용할 오브젝트 스토리지를 구성합니다.



참고

OpenShift Container Platform 3에 MTC를 설치하려면 [OpenShift Container Platform 3에 레거시 Migration Toolkit for Containers Operator](#) 설치를 참조하십시오.

MTC를 [설치 제거](#)하려면 [MTC 설치 및 리소스 삭제](#)를 참조하십시오.

4.1. 호환성 지침

OpenShift Container Platform 버전과 호환되는 MTC(Migration Toolkit for Containers) Operator를 설치해야 합니다.

정의

기존 플랫폼

OpenShift Container Platform 4.5 및 이전 버전.

최신 플랫폼

OpenShift Container Platform 4.6 이상

기존 Operator

레거시 플랫폼을 위해 설계된 MTC Operator입니다.

최신 Operator

최신 플랫폼을 위해 설계된 MTC Operator입니다.

클러스터 제어

MTC 컨트롤러 및 GUI를 실행하는 클러스터입니다.

원격 클러스터

Velero를 실행하는 마이그레이션의 소스 또는 대상 클러스터입니다. Control Cluster는 Velero API를 통해 원격 클러스터와 통신하여 마이그레이션을 구동합니다.

표 4.1. MTC 호환성: 레거시 플랫폼에서 마이그레이션

	OpenShift Container Platform 4.5 또는 이전	OpenShift Container Platform 4.6 이상
안정적인 MTC 버전	MTC 1.7.z legacy 1.7 operator: operator.yml 파일을 사용하여 수동으로 설치합니다.	MTC 1.7.z OLM과 함께 설치, 릴리스 채널 release-v1.7
	 <p>중요</p> <p>이 클러스터는 제어 클러스터가 될 수 없습니다.</p>	



참고

현대 클러스터가 마이그레이션에 관련된 다른 클러스터에 연결하지 못하도록 하는 에지 사례가 있습니다. 예를 들어 온프레미스의 OpenShift Container Platform 3.11 클러스터에서 클라우드의 최신 OpenShift Container Platform 클러스터로 마이그레이션할 때 최신 클러스터가 OpenShift Container Platform 3.11 클러스터에 연결할 수 없습니다.

MTC 1.7을 사용하면 네트워크 제한으로 인해 원격 클러스터 중 하나가 컨트롤 클러스터와 통신할 수 없는 경우 **crane tunnel-api** 명령을 사용합니다.

안정적인 MTC 릴리스에서는 항상 최신 클러스터를 제어 클러스터로 지정해야 하지만 이 특정 경우 레거시 클러스터를 제어 클러스터로 지정하고 워크로드를 원격 클러스터로 푸시할 수 있습니다.

4.2. OPENSIFT CONTAINER PLATFORM 4.9에 MIGRATION TOOLKIT FOR CONTAINERS OPERATOR 설치

Operator Lifecycle Manager를 사용하여 OpenShift Container Platform 4.9에 Migration Toolkit for Containers Operator를 설치합니다.

사전 요구 사항

- 모든 클러스터에서 **cluster-admin** 권한이 있는 사용자로 로그인합니다.
- 로컬 레지스트리의 미러 이미지에서 Operator 카탈로그를 생성해야 합니다.

절차

1. OpenShift Container Platform 웹 콘솔에서 **Operator** → **OperatorHub**를 클릭합니다.
2. 키워드로 필터링 필드를 사용하여 **Migration Toolkit for Containers Operator**를 찾습니다.
3. **Migration Toolkit for Containers Operator**를 선택하고 **설치**를 클릭합니다.
4. **설치**를 클릭합니다.

설치된 Operators 페이지에서 **Migration Toolkit for Containers Operator**는 **openshift-migration** 프로젝트에 **Succeeded** 상태로 나타납니다.

5. **Migration Toolkit for Containers Operator**를 클릭합니다.
6. 제공된 API 아래에서 **마이그레이션 컨트롤러** 타일을 찾고 **인스턴스 작성**을 클릭합니다.
7. **생성**을 클릭합니다.
8. **워크로드** → **Pod**를 클릭하여 MTC pod가 실행 중인지 확인합니다.

4.3. OPENSIFT CONTAINER PLATFORM 4.2에서 4.5로 레거시 MIGRATION TOOLKIT FOR CONTAINERS OPERATOR 설치

OpenShift Container Platform 버전 4.2에서 4.5로 레거시 Migration Toolkit for Containers Operator를 수동으로 설치할 수 있습니다.

사전 요구 사항

- 모든 클러스터에서 **cluster-admin** 권한이 있는 사용자로 로그인합니다.
- **registry.redhat.io**에 대한 액세스 권한이 있어야 합니다.
- **podman**이 설치되어 있어야 합니다.
- **registry.redhat.io**에서 파일을 다운로드하려면 네트워크 액세스 권한이 있는 Linux 워크스테이션이 있어야 합니다.
- Operator 카탈로그의 미러 이미지를 생성해야 합니다.
- OpenShift Container Platform 4.9의 미러링된 Operator 카탈로그에서 Migration Toolkit for Containers Operator를 설치해야 합니다.

절차

1. Red Hat Customer Portal 자격 증명을 사용하여 **registry.redhat.io**에 로그인합니다.

```
$ sudo podman login registry.redhat.io
```

2. 다음 명령을 입력하여 **operator.yml** 파일을 다운로드합니다.

```
$ sudo podman cp $(sudo podman create \
registry.redhat.io/rhmtc/openshift-migration-legacy-rhel8-operator:v1.7):/operator.yml ./
```

3. 다음 명령을 입력하여 **controller.yml** 파일을 다운로드합니다.

```
$ sudo podman cp $(sudo podman create \
registry.redhat.io/rhmtc/openshift-migration-legacy-rhel8-operator:v1.7):/controller.yml ./
```

4. 다음 명령을 실행하여 Operator 이미지 매핑을 가져옵니다.

```
$ grep openshift-migration-legacy-rhel8-operator ./mapping.txt | grep rhmtc
```

Operator 카탈로그를 미리링할 때 **mapping.txt** 파일이 생성되었습니다. 출력은 **registry.redhat.io** 이미지와 미리 레지스트리 이미지 간의 매핑을 보여줍니다.

출력 예

```
registry.redhat.io/rhmtc/openshift-migration-legacy-rhel8-
operator@sha256:468a6126f73b1ee12085ca53a312d1f96ef5a2ca03442bcb63724af5e2614e8
a=<registry.apps.example.com>/rhmtc/openshift-migration-legacy-rhel8-operator
```

5. **operator.yml** 파일에서 **ansible** 및 **operator** 컨테이너의 **image** 값과 **REGISTRY** 값을 업데이트 합니다.

```
containers:
  - name: ansible
    image: <registry.apps.example.com>/rhmtc/openshift-migration-legacy-rhel8-
operator@sha256:
<468a6126f73b1ee12085ca53a312d1f96ef5a2ca03442bcb63724af5e2614e8a> ❶
  ...
  - name: operator
    image: <registry.apps.example.com>/rhmtc/openshift-migration-legacy-rhel8-
operator@sha256:
<468a6126f73b1ee12085ca53a312d1f96ef5a2ca03442bcb63724af5e2614e8a> ❷
  ...
  env:
  - name: REGISTRY
    value: <registry.apps.example.com> ❸
```

❶ ❷ 미리 레지스트리와 Operator 이미지의 **sha256** 값을 지정합니다.

❸ 미리 레지스트리를 지정합니다.

6. 소스 클러스터에 로그인합니다.
7. Migration Toolkit for Containers Operator 오브젝트 생성:

```
$ oc create -f operator.yml
```

출력 예

```
namespace/openshift-migration created
rolebinding.rbac.authorization.k8s.io/system:deployers created
serviceaccount/migration-operator created
customresourcedefinition.apiextensions.k8s.io/migrationcontrollers.migration.openshift.io
created
role.rbac.authorization.k8s.io/migration-operator created
rolebinding.rbac.authorization.k8s.io/migration-operator created
clusterrolebinding.rbac.authorization.k8s.io/migration-operator created
deployment.apps/migration-operator created
Error from server (AlreadyExists): error when creating "./operator.yml":
rolebindings.rbac.authorization.k8s.io "system:image-builders" already exists ❶
Error from server (AlreadyExists): error when creating "./operator.yml":
rolebindings.rbac.authorization.k8s.io "system:image-pullers" already exists
```

- 1 **Error from server (AlreadyExists)** 메시지를 무시할 수 있습니다. 이는 Migration Toolkit for Containers Operator가 이후 릴리스에서 제공되는 OpenShift Container Platform 4 이전

8. **MigrationController** 오브젝트를 만듭니다.

```
$ oc create -f controller.yml
```

9. MTC pod가 실행 중인지 확인합니다.

```
$ oc get pods -n openshift-migration
```

4.4. 프록시 설정

OpenShift Container Platform 4.1 및 이전 버전의 경우 이러한 버전은 클러스터 전체 프록시 오브젝트를 지원하지 않기 때문에 Migration Toolkit for Containers Operator를 설치한 후 **MigrationController** CR(사용자 정의 리소스) 매니페스트에서 **proxy**를 구성해야 합니다.

OpenShift Container Platform 4.2 ~ 4.9의 경우 MTC(Migration Toolkit for Containers)는 클러스터 전체 프록시 설정을 상속합니다. 클러스터 전체 프록시 설정을 재정의하려면 프록시 매개변수를 변경할 수 있습니다.

4.4.1. 직접 볼륨 마이그레이션

MTC 1.4.2에서 직접 볼륨 마이그레이션(DVM)이 도입되었습니다. DVM은 하나의 프록시만 지원합니다. 대상 클러스터가 프록시 뒤에 있는 경우 소스 클러스터는 대상 클러스터의 경로에 액세스할 수 없습니다.

프록시 뒤에서 소스 클러스터에서 DVM을 수행하려면 전송 계층에서 작동하는 TCP 프록시를 구성하고 자체 SSL 인증서로 암호를 해독하고 재암호화하지 않고도 SSL 연결을 투명하게 전달해야 합니다. Stunnel 프록시는 이러한 프록시의 예입니다.

4.4.1.1. DVM용 TCP 프록시 설정

TCP 프록시를 통해 소스와 대상 클러스터 간에 직접 연결하고 프록시를 사용하도록 **MigrationController** CR에서 **stunnel_tcp_proxy** 변수를 구성할 수 있습니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigrationController
metadata:
  name: migration-controller
  namespace: openshift-migration
spec:
  [...]
  stunnel_tcp_proxy: http://username:password@ip:port
```

직접 볼륨 마이그레이션(DVM)은 프록시에 대한 기본 인증만 지원합니다. 또한 DVM은 TCP 연결을 투명하게 터널링할 수 있는 프록시 뒤에서만 작동합니다. 중간자 모드에서 HTTP/HTTPS 프록시가 작동하지 않습니다. 기존 클러스터 전체 프록시에서 이 동작을 지원하지 않을 수 있습니다. 결과적으로 DVM의 프록시 설정은 의도적으로 MTC의 일반 프록시 구성과 다르게 유지됩니다.

4.4.1.2. HTTP/HTTPS 프록시 대신 TCP 프록시를 사용하는 이유는 무엇입니까?

OpenShift 경로를 통해 소스와 대상 클러스터 간에 Rsync를 실행하여 DVM을 활성화할 수 있습니다. 트래픽은 TCP 프록시인 Stunnel을 사용하여 암호화됩니다. 소스 클러스터에서 실행되는 Stunnel은 대상 Stunnel을 사용한 TLS 연결을 시작하고 암호화된 채널을 통해 데이터를 전송합니다.

OpenShift의 클러스터 전체 HTTP/HTTPS 프록시는 일반적으로 자체 TLS 세션을 외부 서버와 협상하는 중간자 모드로 구성됩니다. 그러나 이 작업은 Stunnel에서는 작동하지 않습니다. Stunnel은 프록시에서 TLS 세션을 그대로 전환해야 하므로 기본적으로 프록시를 통해 TCP 연결을 그대로 전달하는 투명한 터널로 프록시를 설정해야 합니다. 따라서 TCP 프록시를 사용해야 합니다.

4.4.1.3. 알려진 문제

마이그레이션 실패 오류 업그레이드 요청 필요

마이그레이션 컨트롤러는 SPDY 프로토콜을 사용하여 원격 Pod 내에서 명령을 실행합니다. 원격 클러스터가 프록시 또는 SPDY 프로토콜을 지원하지 않는 방화벽 뒤에 있는 경우 마이그레이션 컨트롤러가 원격 명령을 실행하지 못합니다. 오류 메시지 **업그레이드 요청** 과 함께 마이그레이션이 실패합니다. 해결방법: SPDY 프로토콜을 지원하는 프록시를 사용합니다.

SPDY 프로토콜 지원 외에도 프록시 또는 방화벽은 **Upgrade** HTTP 헤더를 API 서버에 전달해야 합니다. 클라이언트는 이 헤더를 사용하여 API 서버와의 websocket 연결을 엽니다. **업그레이드** 헤더가 프록시 또는 방화벽에 의해 차단된 경우 마이그레이션은 오류 메시지 **업그레이드 요청 필요** 와 함께 실패합니다. 해결방법: 프록시가 **Upgrade** 헤더를 전달하도록 합니다.

4.4.2. 마이그레이션을 위한 네트워크 정책 튜닝

OpenShift는 클러스터에서 사용하는 네트워크 플러그인을 기반으로 *NetworkPolicy* 또는 *EgressFirewall* 을 사용하여 Pod로 트래픽을 제한할 수 있습니다. 마이그레이션과 관련된 소스 네임스페이스가 이러한 메커니즘을 사용하여 네트워크 트래픽을 포드로 제한하는 경우 제한이 마이그레이션 중에 Rsync pod로의 트래픽을 실수로 중지할 수 있습니다.

소스 및 대상 클러스터에서 둘 다 실행되는 rsync Pod는 OpenShift 경로를 통해 서로 연결되어야 합니다. 기존 *NetworkPolicy* 또는 *EgressNetworkPolicy* 오브젝트는 이러한 트래픽 제한에서 Rsync Pod를 자동으로 제외하도록 구성할 수 있습니다.

4.4.2.1. NetworkPolicy 구성

4.4.2.1.1. Rsync Pod의 송신 트래픽

소스 또는 대상 네임스페이스의 **NetworkPolicy** 구성이 이러한 유형의 트래픽을 차단하는 경우 Rsync Pod의 고유한 레이블을 사용하여 송신 트래픽이 해당 트래픽에서 전달되도록 허용할 수 있습니다. 다음 정책은 네임스페이스의 Rsync Pod에서 모든 송신 트래픽을 허용합니다.

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-all-egress-from-rsync-pods
spec:
  podSelector:
    matchLabels:
      owner: directvolumemigration
      app: directvolumemigration-rsync-transfer
  egress:
  - {}
  policyTypes:
  - Egress
```

4.4.2.1.2. Rsync pod로의 수신 트래픽

```

apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-all-egress-from-rsync-pods
spec:
  podSelector:
    matchLabels:
      owner: directvolumemigration
      app: directvolumemigration-rsync-transfer
  ingress:
  - {}
  policyTypes:
  - Ingress

```

4.4.2.2. EgressNetworkPolicy 구성

EgressNetworkPolicy 오브젝트 또는 *Egress Firewalls* 는 클러스터를 나가는 송신 트래픽을 차단하도록 설계된 OpenShift 구조입니다.

NetworkPolicy 오브젝트와 달리 Egress Firewall은 네임스페이스의 모든 포트에 적용되므로 프로젝트 수준에서 작동합니다. 따라서 Rsync Pod의 고유 레이블은 제한 사항에서 Rsync Pod만 제외하지 않습니다. 그러나 두 클러스터 간에 직접 연결을 설정할 수 있도록 소스 또는 대상 클러스터의 CIDR 범위를 정책의 허용 규칙에 추가할 수 있습니다.

Egress Firewall이 있는 클러스터를 기반으로 다른 클러스터의 CIDR 범위를 추가하여 둘 사이의 송신 트래픽을 허용할 수 있습니다.

```

apiVersion: network.openshift.io/v1
kind: EgressNetworkPolicy
metadata:
  name: test-egress-policy
  namespace: <namespace>
spec:
  egress:
  - to:
      cidrSelector: <cidr_of_source_or_target_cluster>
    type: Deny

```

4.4.2.3. 데이터 전송을 위한 대체 끝점 선택

기본적으로 DVM은 OpenShift Container Platform 경로를 끝점으로 사용하여 PV 데이터를 대상 클러스터로 전송합니다. 클러스터 토폴로지가 허용하는 경우 다른 유형의 지원되는 끝점을 선택할 수 있습니다.

각 클러스터에 대해 **MigrationController** CR의 적절한 대상 클러스터에서 **rsync_endpoint_type** 변수를 설정하여 끝점을 구성할 수 있습니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: MigrationController
metadata:
  name: migration-controller
  namespace: openshift-migration

```

```
spec:
  [...]
  rsync_endpoint_type: [NodePort|ClusterIP|Route]
```

4.4.2.4. Rsync Pod에 대한 추가 그룹 구성

PVC에서 공유 스토리지를 사용하는 경우 Pod에서 액세스를 허용하기 위해 Rsync Pod 정의에 추가 그룹을 추가하여 해당 스토리지에 대한 액세스를 구성할 수 있습니다.

표 4.2. Rsync Pod의 보조 그룹

변수	유형	기본값	설명
src_supplemental_groups	string	설정되지 않음	소스 Rsync Pod에 대한 콤마로 구분된 추가 그룹 목록
target_supplemental_groups	string	설정되지 않음	대상 Rsync Pod에 대한 콤마로 구분된 추가 그룹 목록

사용 예

MigrationController CR을 업데이트하여 이러한 추가 그룹에 대한 값을 설정할 수 있습니다.

```
spec:
  src_supplemental_groups: "1000,2000"
  target_supplemental_groups: "2000,3000"
```

4.4.3. 프록시 구성

사전 요구 사항

- 모든 클러스터에서 **cluster-admin** 권한이 있는 사용자로 로그인합니다.

절차

- MigrationController** CR 매니페스트를 가져옵니다.

```
$ oc get migrationcontroller <migration_controller> -n openshift-migration
```

- 프록시 매개변수를 업데이트합니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigrationController
metadata:
  name: <migration_controller>
  namespace: openshift-migration
...
spec:
  stunnel_tcp_proxy: http://<username>:<password>@<ip>:<port> 1
  noProxy: example.com 2
```

■

- 1 직접 볼륨 마이그레이션을 위한 Stunnel 프록시 URL입니다.
- 2 프록시를 제외할 대상 도메인 이름, 도메인, IP 주소 또는 기타 네트워크 CIDR의 쉼표로 구분된 목록입니다.

하위 도메인과 일치하려면 도메인 앞에 `.`을 입력합니다. 예를 들어, `.y.com`은 `x.y.com`과 일치하지만 `y.com`은 일치하지 않습니다. `*`를 사용하여 모든 대상에 대해 프록시를 바이패스합니다.

`networking.machineNetwork[].cidr` 필드에 의해 정의된 네트워크에 포함되어 있지 않은 작업자를 설치 구성에서 확장하려면 연결 문제를 방지하기 위해 이 목록에 해당 작업자를 추가해야 합니다.

`httpProxy` 또는 `httpsProxy` 필드가 설정되지 않은 경우 이 필드는 무시됩니다.

3. 매니페스트를 `migration-controller.yaml`로 저장합니다.
4. 업데이트된 매니페스트를 적용합니다.

```
$ oc replace -f migration-controller.yaml -n openshift-migration
```

자세한 내용은 [클러스터 전체 프록시 구성](#) 을 참조하십시오.

4.5. 복제 리포지토리 구성

Multicloud Object Gateway는 제한된 네트워크 환경에 지원되는 유일한 옵션입니다.

MTC는 소스 클러스터에서 대상 클러스터로 데이터를 마이그레이션하기 위한 [파일 시스템 및 스냅샷 데이터 복사 방법](#)을 지원합니다. 환경에 적합하고 스토리지 공급자가 지원하는 방법을 선택할 수 있습니다.

4.5.1. 사전 요구 사항

- 모든 클러스터에는 복제 리포지토리에 대한 중단없는 네트워크 액세스 권한이 있어야 합니다.
- 내부 호스팅 복제 리포지토리와 함께 프록시 서버를 사용하는 경우 프록시가 복제 리포지토리에 액세스할 수 있는지 확인해야 합니다.

4.5.2. Multicloud Object Gateway 자격 증명 검색

OADP(Data Protection) API에 대한 **Secret** 사용자 정의 리소스(CR)를 생성하려면 MCG(Multicloud Object Gateway) 자격 증명을 검색해야 합니다.

MCG는 OpenShift Container Storage의 구성 요소입니다.

사전 요구 사항

- 적절한 OpenShift Container Storage 배포 [가이드](#)를 사용하여 OpenShift Container Storage를 배포해야 합니다.

절차

1. **NooBaa** 사용자 지정 리소스에서 `describe` 명령을 실행하여 S3 엔드포인트, `AWS_ACCESS_KEY` 및 `AWS_SECRET_ACCESS_KEY`를 가져옵니다.

4.5.3. 추가 리소스

- Red Hat OpenShift Container Storage 설명서의 [연결이 끊긴 환경](#).
- [MTC 워크플로](#)
- [데이터 복사 방법 정보](#)
- [MTC 웹 콘솔에 복제 리포지토리 추가](#)

4.6. MTC 설치 제거 및 리소스 삭제

MTC(Migration Toolkit for Containers)를 설치 제거하고 해당 리소스를 삭제하여 클러스터를 정리할 수 있습니다.



참고

velero CRD를 삭제하면 클러스터에서 Velero가 제거됩니다.

사전 요구 사항

- **cluster-admin** 권한이 있는 사용자로 로그인해야 합니다.

절차

1. 모든 클러스터에서 **MigrationController** CR(사용자 정의 리소스)을 삭제합니다.

```
$ oc delete migrationcontroller <migration_controller>
```

2. Operator Lifecycle Manager를 사용하여 OpenShift Container Platform 4에서 Migration Toolkit for Containers Operator를 설치 제거합니다.
3. 다음 명령을 실행하여 모든 클러스터에서 클러스터 범위 리소스를 삭제합니다.

- **migration** CRD(사용자 정의 리소스 정의):

```
$ oc delete $(oc get crds -o name | grep 'migration.openshift.io')
```

- **Velero** CRD:

```
$ oc delete $(oc get crds -o name | grep 'velero')
```

- **migration** 클러스터 역할:

```
$ oc delete $(oc get clusterroles -o name | grep 'migration.openshift.io')
```

- **migration-operator** 클러스터 역할:

```
$ oc delete clusterrole migration-operator
```

- **Velero** 클러스터 역할:

```
$ oc delete $(oc get clusterroles -o name | grep 'velero')
```

- **migration** 클러스터 역할 바인딩:

```
$ oc delete $(oc get clusterrolebindings -o name | grep 'migration.openshift.io')
```

- **migration-operator** 클러스터 역할 바인딩:

```
$ oc delete clusterrolebindings migration-operator
```

- **Velero** 클러스터 역할 바인딩:

```
$ oc delete $(oc get clusterrolebindings -o name | grep 'velero')
```

5장. MIGRATION TOOLKIT FOR CONTAINERS 업그레이드

Operator Lifecycle Manager를 사용하여 OpenShift Container Platform 4.9에서 MTC(Migration Toolkit for Containers)를 업그레이드할 수 있습니다.

레거시 Migration Toolkit for Containers Operator를 다시 설치하여 OpenShift Container Platform 4.5 및 이전 버전에서 MTC를 업그레이드할 수 있습니다.



중요

MTC 버전 1.3에서 업그레이드하는 경우 **MigPlan** 사용자 정의 리소스(CR)를 업데이트하려면 추가 절차를 수행해야 합니다.

5.1. OPENSIFT CONTAINER PLATFORM 4.9에서 MIGRATION TOOLKIT FOR CONTAINERS 업그레이드

Operator Lifecycle Manager를 사용하여 OpenShift Container Platform 4.9에서 MTC(Migration Toolkit for Containers)를 업그레이드할 수 있습니다.

사전 요구 사항

- **cluster-admin** 권한이 있는 사용자로 로그인해야 합니다.

절차

1. OpenShift Container Platform 콘솔에서 **Operators > 설치된 Operators**로 이동합니다. 보류 중인 업그레이드가 있는 Operator에 **업그레이드 사용 가능** 상태가 표시됩니다.
2. **Migration Toolkit for Containers Operator**를 클릭합니다.
3. **서브스크립션** 탭을 클릭합니다. 승인이 필요한 업그레이드는 **업그레이드 상태** 옆에 표시됩니다. 예를 들어 **1승인 필요**가 표시될 수 있습니다.
4. **1승인 필요**를 클릭한 다음 **설치 계획 프리뷰**를 클릭합니다.
5. 업그레이드에 사용할 수 있는 리소스를 보고 **승인**을 클릭합니다.
6. **Operator → 설치된 Operator** 페이지로 이동하여 업그레이드 진행 상황을 모니터링합니다. 완료 되면 상태가 **성공** 및 **최신**으로 변경됩니다.
7. **워크로드 → Pod**를 클릭하여 MTC pod가 실행 중인지 확인합니다.

5.2. OPENSIFT CONTAINER PLATFORM 버전 4.2에서 4.5로 MIGRATION TOOLKIT FOR CONTAINERS 업그레이드

레거시 MTC(Migration Toolkit for Containers Operator)를 수동으로 설치하여 OpenShift Container Platform 버전 4.2에서 MTC(Migration Toolkit for Containers)를 업그레이드할 수 있습니다.

사전 요구 사항

- **cluster-admin** 권한이 있는 사용자로 로그인해야 합니다.
- **registry.redhat.io**에 대한 액세스 권한이 있어야 합니다.

- **podman**이 설치되어 있어야 합니다.

절차

1. 다음 명령을 입력하여 Red Hat Customer Portal 자격 증명을 사용하여 **registry.redhat.io** 에 로그인합니다.

```
$ sudo podman login registry.redhat.io
```

2. 다음 명령을 입력하여 **operator.yml** 파일을 다운로드합니다.

```
$ sudo podman cp $(sudo podman create \
registry.redhat.io/rhmtc/openshift-migration-legacy-rhel8-operator:v1.7):operator.yml ./
```

3. 다음 명령을 입력하여 Migration Toolkit for Containers Operator를 교체합니다.

```
$ oc replace --force -f operator.yml
```

4. 다음 명령을 입력하여 **migration-operator** 배포를 **0** 으로 확장하여 배포를 중지합니다.

```
$ oc scale -n openshift-migration --replicas=0 deployment/migration-operator
```

5. **migration-operator** 배포를 **1** 로 확장하여 배포를 시작하고 다음 명령을 입력하여 변경 사항을 적용합니다.

```
$ oc scale -n openshift-migration --replicas=1 deployment/migration-operator
```

6. 다음 명령을 입력하여 **migration-operator** 가 업그레이드되었는지 확인합니다.

```
$ oc -o yaml -n openshift-migration get deployment/migration-operator | grep image: | awk -F
":" '{ print $NF }'
```

7. 다음 명령을 입력하여 **controller.yml** 파일을 다운로드합니다.

```
$ sudo podman cp $(sudo podman create \
registry.redhat.io/rhmtc/openshift-migration-legacy-rhel8-operator:v1.7):controller.yml ./
```

8. 다음 명령을 입력하여 **migration-controller** 오브젝트를 생성합니다.

```
$ oc create -f controller.yml
```

9. 다음 명령을 입력하여 MTC Pod가 실행 중인지 확인합니다.

```
$ oc get pods -n openshift-migration
```

5.3. MTC 1.3을 1.7로 업그레이드

MTC(Migration Toolkit for Containers) 버전 1.3.x를 1.7로 업그레이드하는 경우 **MigrationController** Pod가 실행 중인 클러스터에서 **MigPlan** 사용자 정의 리소스(CR) 매니페스트를 업데이트해야 합니다.

indirectImageMigration 및 **indirectVolumeMigration** 매개변수는 MTC 1.3에 존재하지 않기 때문에 버

전 1.4의 기본값은 **false**이며 직접 이미지 마이그레이션 및 직접 볼륨 마이그레이션이 활성화됩니다. 직접 마이그레이션 요구 사항이 충족되지 않기 때문에 이러한 매개변수 값이 **true**로 변경되지 않는 한 마이그레이션 계획에서 **Ready** 상태에 도달할 수 없습니다.

사전 요구 사항

- **cluster-admin** 권한이 있는 사용자로 로그인해야 합니다.

절차

1. **MigrationController** Pod를 실행하는 클러스터에 로그인합니다.
2. **MigPlan** CR 매니페스트를 가져옵니다.

```
$ oc get migplan <migplan> -o yaml -n openshift-migration
```

3. 다음 매개변수 값을 업데이트하고 파일을 **migplan.yaml**로 저장합니다.

```
...
spec:
  indirectImageMigration: true
  indirectVolumeMigration: true
```

4. **MigPlan** CR 매니페스트를 교체하여 변경 사항을 적용합니다.

```
$ oc replace -f migplan.yaml -n openshift-migration
```

5. 업데이트된 **MigPlan** CR 매니페스트를 가져와 변경 사항을 확인합니다.

```
$ oc get migplan <migplan> -o yaml -n openshift-migration
```

6장. 마이그레이션 전 체크리스트

MTC(Migration Toolkit for Containers)를 사용하여 애플리케이션 워크로드를 마이그레이션하기 전에 다음 체크리스트를 검토하십시오.

6.1. 클러스터 상태 체크리스트

- 클러스터는 특정 플랫폼 및 설치 방법(예: [베어 메탈](#))에 대한 최소 하드웨어 요구 사항을 충족합니다.
- 모든 [MTC 사전 요구 사항](#)을 충족합니다.
- 모든 노드에는 유효한 OpenShift Container Platform 서브스크립션이 있습니다.
- [노드 상태를 확인](#)했습니다.
- [ID 공급자가 작동](#) 중입니다.
- 마이그레이션 네트워크의 최소 처리량은 10Gbps입니다.
- 클러스터에 마이그레이션에 필요한 충분한 리소스가 있습니다.



참고

일반 워크로드에서 마이그레이션을 실행하려면 클러스터에 추가 메모리, CPU 및 스토리지가 필요합니다. 실제 리소스 요구 사항은 단일 마이그레이션 계획에서 마이그레이션되는 Kubernetes 리소스 수에 따라 다릅니다. 리소스 요구 사항을 추정하려면 비 프로덕션 환경에서 마이그레이션을 테스트해야 합니다.

- [fio](#)는 클러스터의 [etcd 디스크 성능](#)을 확인하는 데 사용되었습니다.

6.2. 소스 클러스터 체크리스트

- 다음 명령을 실행하여 [종료 중](#) 상태에서 비정상적으로 설정된 PV(영구 볼륨)가 있는지 확인했습니다.

```
$ oc get pv
```

- 다음 명령을 실행하여 [실행 중](#) 또는 [완료](#) 이외의 상태인 Pod를 확인했습니다.

```
$ oc get pods --all-namespaces | egrep -v 'Running | Completed'
```

- 다음 명령을 실행하여 재시작 횟수가 높은 Pod를 확인했습니다.

```
$ oc get pods --all-namespaces --field-selector=status.phase=Running \
-o json | jq '.items[]|select(any( .status.containerStatuses[]; \
.restartCount > 3))|.metadata.name'
```

Pod가 [실행 중](#) 상태인 경우에도 재시작 횟수가 많으면 기본적인 문제가 될 수 있습니다.

- 클러스터 인증서는 마이그레이션 프로세스 기간 동안 유효합니다.
- 다음 명령을 실행하여 보류 중인 인증서 서명 요청이 있는지 확인했습니다.

```
$ oc get csr -A | grep pending -i
```

- 레지스트리는 **권장 스토리지 유형**을 사용합니다.
- 레지스트리에 이미지를 읽고 쓸 수 있습니다.
- etcd 클러스터**는 정상입니다.
- 소스 클러스터의 **평균 API 서버 응답 시간**은 50ms 미만입니다.

6.3. 대상 클러스터 체크리스트

- 클러스터에 올바른 네트워크 구성과 외부 서비스(예: 데이터베이스, 소스 코드 저장소, 컨테이너 미러 레지스트리, CI/CD 툴)에 액세스할 수 있는 권한이 있습니다.
- 클러스터에서 제공하는 서비스를 사용하는 외부 애플리케이션과 서비스에는 클러스터에 액세스할 수 있는 올바른 네트워크 구성 및 권한이 있습니다.
- 내부 컨테이너 이미지에 필요한 종속성을 충족합니다.
- 대상 클러스터와 복제 리포지토리에 충분한 스토리지 공간이 있습니다.

7장. 네트워크 고려 사항

마이그레이션 후 애플리케이션 네트워크 트래픽을 리디렉션하는 전략을 검토합니다.

7.1. DNS 고려 사항

대상 클러스터의 DNS 도메인은 소스 클러스터의 도메인과 다릅니다. 기본적으로 애플리케이션은 마이그레이션 후 대상 클러스터의 FQDN을 가져옵니다.

마이그레이션된 애플리케이션의 소스 DNS 도메인을 보존하려면 아래에 설명된 두 옵션 중 하나를 선택합니다.

7.1.1. 클라이언트에서 대상 클러스터의 DNS 도메인 격리

소스 클러스터의 DNS 도메인으로 전송된 클라이언트의 요청이 대상 클러스터를 클라이언트에 노출하지 않고 대상 클러스터의 DNS 도메인에 도달할 수 있도록 허용할 수 있습니다.

절차

1. 클라이언트와 대상 클러스터 간에 애플리케이션 로드 밸런서 또는 역방향 프록시와 같은 기타 네트워크 구성 요소를 배치합니다.
2. DNS 서버의 소스 클러스터에서 애플리케이션 FQDN을 업데이트하여 가상 네트워크 구성 요소의 IP 주소를 반환합니다.
3. 소스 도메인의 애플리케이션에 대해 수신된 요청을 대상 클러스터 도메인의 로드 밸런서에 보내도록 네트워크 구성 요소를 구성합니다.
4. 소스 클러스터의 로드 밸런서의 IP 주소를 가리키는 ***.apps.source.example.com** 도메인에 대한 와일드카드 DNS 레코드를 만듭니다.
5. 대상 클러스터 앞의 브릿지 네트워크 구성 요소의 IP 주소를 가리키는 각 애플리케이션에 대한 DNS 레코드를 만듭니다. 특정 DNS 레코드는 와일드카드 레코드보다 우선 순위가 높으므로 애플리케이션 FQDN이 확인되면 충돌이 발생하지 않습니다.

참고

- 기존의 네트워크 구성 요소는 모든 보안 TLS 연결을 종료해야 합니다. 연결이 대상 클러스터 로드 밸런서 장치로 전달되면 대상 애플리케이션의 FQDN이 클라이언트에 노출되고 인증서 오류가 발생합니다.
- 애플리케이션은 대상 클러스터 도메인을 클라이언트에 참조하는 링크를 반환해서는 안 됩니다. 그렇지 않으면 애플리케이션 일부가 제대로 로드되거나 작동하지 않을 수 있습니다.

7.1.2. 소스 DNS 도메인을 허용하도록 대상 클러스터 설정

소스 클러스터의 DNS 도메인에서 마이그레이션된 애플리케이션에 대한 요청을 수락하도록 대상 클러스터를 설정할 수 있습니다.

절차

비보안 HTTP 액세스 및 보안 HTTPS 액세스 모두에서 다음 단계를 수행합니다.

1. 소스 클러스터에서 애플리케이션의 FQDN으로 주소가 지정된 요청을 수락하도록 구성된 대상 클러스터의 프로젝트에서 경로를 생성합니다.

```
$ oc expose svc <app1-svc> --hostname <app1.apps.source.example.com> \
-n <app1-namespace>
```

이 새 경로를 적용하면 서버는 해당 FQDN에 대한 모든 요청을 수락하고 해당 애플리케이션 포드로 보냅니다. 또한 애플리케이션을 마이그레이션하면 대상 클러스터 도메인에 또 다른 경로가 생성됩니다. 요청은 이러한 호스트 이름 중 하나를 사용하여 마이그레이션된 애플리케이션에 연결합니다.

2. 소스 클러스터의 애플리케이션의 FQDN을 가리키고 대상 클러스터의 기본 로드 밸런서의 IP 주소를 가리키는 DNS 공급자를 사용하여 DNS 레코드를 생성합니다. 그러면 소스 클러스터에서 대상 클러스터로 트래픽이 리디렉션됩니다.

애플리케이션의 FQDN은 대상 클러스터의 로드 밸런서로 확인됩니다. 기본 수신 컨트롤러 라우터는 해당 호스트 이름의 경로가 노출되므로 해당 FQDN에 대한 요청을 수락합니다.

보안 HTTPS 액세스를 위해 다음 추가 단계를 수행합니다.

1. 설치 프로세스 중에 생성된 기본 수신 컨트롤러의 x509 인증서를 사용자 지정 인증서로 교체합니다.
2. **subjectAltName** 필드에 소스 및 대상 클러스터의 와일드카드 DNS 도메인을 포함하도록 이 인증서를 구성합니다.
새 인증서는 DNS 도메인을 사용하여 만든 연결 보안에 유효합니다.

추가 리소스

- 자세한 내용은 [기본 수신 인증서 교체](#) 를 참조하십시오.

7.2. 네트워크 트래픽 리디렉션 전략

마이그레이션에 성공한 후 소스 클러스터에서 대상 클러스터로 상태 비저장 애플리케이션의 네트워크 트래픽을 리디렉션해야 합니다.

네트워크 트래픽을 리디렉션하는 전략은 다음과 같은 가정을 기반으로 합니다.

- 애플리케이션 포드는 소스 클러스터와 대상 클러스터에서 모두 실행됩니다.
- 각 애플리케이션에는 소스 클러스터 호스트 이름이 포함된 경로가 있습니다.
- 소스 클러스터 호스트 이름이 있는 경로에는 CA 인증서가 포함되어 있습니다.
- HTTPS의 경우 대상 라우터 CA 인증서에는 소스 클러스터의 와일드카드 DNS 레코드에 대한 주체 대체 이름이 포함되어 있습니다.

다음 전략을 고려하여 목표를 충족하는 전략을 선택합니다.

- 모든 애플리케이션에 대한 모든 네트워크 트래픽 리디렉션 동시에 대상 클러스터의 가상 IP 주소(VIP)를 가리키도록 소스 클러스터의 와일드카드 DNS 레코드를 변경합니다.

이 전략은 간단한 애플리케이션 또는 소규모 마이그레이션에 적합합니다.

- 개별 애플리케이션에 대한 네트워크 트래픽 리디렉션

대상 클러스터 라우터의 VIP를 가리키는 소스 클러스터 호스트 이름을 사용하여 각 애플리케이션의 DNS 레코드를 만듭니다. 이 DNS 레코드는 소스 클러스터 와일드카드 DNS 레코드보다 우선합니다.

- 개별 애플리케이션에 대해 네트워크 트래픽 점진적 리디렉션
 1. 각 애플리케이션에 대해 소스 클러스터 라우터의 VIP와 대상 클러스터 라우터의 VIP로 트래픽을 보낼 수 있는 프록시를 만듭니다.
 2. 프록시를 가리키는 소스 클러스터 호스트 이름을 사용하여 각 애플리케이션에 대한 DNS 레코드를 만듭니다.
 3. 트래픽의 백분율을 대상 클러스터 라우터의 VIP로 라우팅하고 나머지 트래픽을 소스 클러스터 라우터의 VIP로 라우팅하도록 애플리케이션의 프록시 항목을 구성합니다.
 4. 모든 네트워크 트래픽이 리디렉션될 때까지 대상 클러스터 라우터의 VIP로 라우팅하는 트래픽의 백분율을 점차 늘립니다.
- 개별 애플리케이션에 대한 사용자 기반 트래픽 리디렉션

이 전략을 사용하면 사용자 요청의 TCP/IP 헤더를 필터링하여 사전 정의된 사용자 그룹의 네트워크 트래픽을 리디렉션할 수 있습니다. 이를 통해 전체 네트워크 트래픽을 리디렉션하기 전에 특정 사용자 수에 대한 리디렉션 프로세스를 테스트할 수 있습니다.

 1. 각 애플리케이션에 대해 소스 클러스터 라우터의 VIP와 대상 클러스터 라우터의 VIP로 트래픽을 보낼 수 있는 프록시를 만듭니다.
 2. 프록시를 가리키는 소스 클러스터 호스트 이름을 사용하여 각 애플리케이션에 대한 DNS 레코드를 만듭니다.
 3. **테스트 고객과** 같이 지정된 헤더 패턴과 일치하는 트래픽을 대상 클러스터 라우터의 VIP로 라우팅하고 나머지 트래픽을 소스 클러스터 라우터의 VIP로 라우팅하도록 애플리케이션의 프록시 항목을 구성합니다.
 4. 모든 트래픽이 대상 클러스터 라우터의 VIP에 있을 때까지 트래픽을 단계별로 대상 클러스터 라우터의 VIP로 리디렉션합니다.

8장. 애플리케이션 마이그레이션

MTC(Migration Toolkit for Containers) 웹 콘솔 또는 [명령줄](#) 을 사용하여 애플리케이션을 마이그레이션할 수 있습니다.

대부분의 클러스터 범위 리소스는 아직 MTC에서 처리되지 않습니다. 애플리케이션에 클러스터 범위의 리소스가 필요한 경우 대상 클러스터에서 수동으로 리소스를 생성해야 할 수 있습니다.

스테이지 마이그레이션과 컷오버 마이그레이션을 사용하여 클러스터 간에 애플리케이션을 마이그레이션할 수 있습니다.

- 스테이지 마이그레이션은 애플리케이션을 중지하지 않고 소스 클러스터에서 대상 클러스터로 데이터를 복사합니다. 스테이지 마이그레이션을 여러 번 실행하여 할당 마이그레이션 기간을 줄일 수 있습니다.
- 컷오버 마이그레이션은 소스 클러스터에서 트랜잭션을 중지하고 리소스를 대상 클러스터로 이동합니다.

상태 마이그레이션을 사용하여 애플리케이션 상태를 마이그레이션할 수 있습니다.

- 상태 마이그레이션은 선택한 PVC(영구 볼륨 클레임)를 복사합니다.
- 상태 마이그레이션을 사용하여 동일한 클러스터 내에서 네임스페이스를 마이그레이션할 수 있습니다.

마이그레이션 중에 MTC(Migration Toolkit for Containers)는 다음 네임스페이스 주석을 유지합니다.

- **openshift.io/sa.scc.mcs**
- **openshift.io/sa.scc.supplemental-groups**
- **openshift.io/sa.scc.uid-range**
이러한 주석은 UID 범위를 유지하여 컨테이너가 대상 클러스터에 대한 파일 시스템 권한을 유지하도록 합니다. 마이그레이션된 UID가 대상 클러스터의 기존 또는 향후 네임스페이스 내에서 UID를 복제할 위험이 있습니다.

8.1. 마이그레이션 사전 요구 사항

- 모든 클러스터에서 **cluster-admin** 권한이 있는 사용자로 로그인합니다.

직접 이미지 마이그레이션

- 소스 클러스터의 보안 내부 레지스트리가 노출되었는지 확인해야 합니다.
- 노출된 레지스트리에 대한 경로를 생성해야 합니다.

직접 볼륨 마이그레이션

- 클러스터에서 프록시를 사용하는 경우 Stunnel TCP 프록시를 구성해야 합니다.

클러스터

- 소스 클러스터를 최신 MTC z-stream 릴리스로 업그레이드해야 합니다.
- MTC 버전은 모든 클러스터에서 동일해야 합니다.

네트워크

- 클러스터는 서로 및 복제 리포지토리에 제한 없이 네트워크 액세스할 수 있습니다.
- **move**를 사용하여 영구 볼륨을 복사하는 경우 클러스터에 원격 볼륨에 대한 무제한 네트워크 액세스 권한이 있어야 합니다.
- OpenShift Container Platform 4 클러스터에서 다음 포트를 활성화해야 합니다.
 - **6443** (API 서버)
 - **443** (라우트)
 - **53** (DNS)
- TLS를 사용하는 경우 복제 리포지토리에서 포트 **443**을 활성화해야 합니다.

영구 볼륨 (PV)

- PV가 유효해야 합니다.
- PV를 영구 볼륨 클레임에 바인딩해야 합니다.
- 스냅샷을 사용하여 PV를 복사하는 경우 다음과 같은 추가 사전 요구 사항이 적용됩니다.
 - 클라우드 공급자는 스냅샷을 지원해야 합니다.
 - PV는 동일한 클라우드 공급자에 있어야 합니다.
 - PV는 동일한 지역 리전에 있어야 합니다.
 - PV는 동일한 스토리지 클래스를 보유해야 합니다.

8.2. MTC 웹 콘솔을 사용하여 애플리케이션 마이그레이션

MTC 웹 콘솔을 사용하여 클러스터와 복제 리포지토리를 구성할 수 있습니다. 그러면 마이그레이션 계획을 생성하고 실행할 수 있습니다.

8.2.1. MTC 웹 콘솔 시작

브라우저에서 MTC(Migration Toolkit for Containers) 웹 콘솔을 시작할 수 있습니다.

사전 요구 사항

- MTC 웹 콘솔에는 OpenShift Container Platform 웹 콘솔에 대한 네트워크 액세스 권한이 있어야 합니다.
- MTC 웹 콘솔에는 OAuth 인증 서버에 대한 네트워크 액세스 권한이 있어야 합니다.

절차

1. MTC를 설치한 OpenShift Container Platform 클러스터에 로그인합니다.
2. 다음 명령을 입력하여 MTC 웹 콘솔 URL을 확보합니다.

```
$ oc get -n openshift-migration route/migration -o go-template='https://{ .spec.host }'
```


3. MTC 웹 콘솔에서 **클러스터**를 클릭합니다.
4. **클러스터 추가**를 클릭합니다.
5. 다음 필드를 작성합니다.
 - **클러스터 이름**: 클러스터 이름은 소문자(**a-z**)와 숫자(**0-9**)를 포함할 수 있습니다. 공백이나 국제 문자를 포함해서는 안 됩니다.
 - **URL**: API 서버 URL을 지정합니다(예: **https://<www.example.com>:8443**).
 - **서비스 계정 토큰**: **migration-controller** 서비스 계정 토큰을 붙여넣습니다.
 - **이미지 레지스트리로 노출된 경로 호스트**: 직접 이미지 마이그레이션을 사용하는 경우 소스 클러스터의 이미지 레지스트리에 노출된 경로를 지정합니다.
다음 명령을 실행하여 라우트를 생성합니다.
 - OpenShift Container Platform 3의 경우:


```
$ oc create route passthrough --service=docker-registry --port=5000 -n default
```
 - OpenShift Container Platform 4의 경우:


```
$ oc create route passthrough --service=image-registry --port=5000 -n openshift-image-registry
```
 - **Azure 클러스터**: Azure 스냅샷을 사용하여 데이터를 복사하는 경우 이 옵션을 선택해야 합니다.
 - **Azure 리소스 그룹**: 이 필드는 **Azure 클러스터**가 선택된 경우에 표시됩니다. Azure 리소스 그룹을 지정합니다.
 - **SSL 확인 필요**: 선택 사항: 이 옵션을 선택하여 클러스터에 대한 SSL 연결을 확인합니다.
 - **CA 번들 파일**: **SSL 확인 필요**가 선택되어 있으면 이 필드가 표시됩니다. 자체 서명된 인증서에 대한 사용자 정의 CA 인증서 번들 파일을 생성한 경우 **찾아보기**를 클릭하고 CA 번들 파일을 선택하여 업로드합니다.
6. **클러스터 추가**를 클릭합니다.
클러스터가 **클러스터 목록**에 나타납니다.

8.2.3. MTC 웹 콘솔에 복제 리포지토리 추가

MTC(Migration Toolkit for Containers) 웹 콘솔에 복제 리포지토리로 오브젝트 스토리지를 추가할 수 있습니다.

MTC는 다음과 같은 스토리지 제공자를 지원합니다.

- AWS(Amazon Web Services) S3
- MCG(Multi-Cloud Object Gateway)
- 일반 S3 오브젝트 스토리지(예: Minio 또는 Ceph S3)
- GCP(Google Cloud Provider)

- Microsoft Azure Blob

사전 요구 사항

- 복제 리포지토리로 오브젝트 스토리지를 구성해야 합니다.

절차

1. MTC 웹 콘솔에서 **복제 리포지토리**를 클릭합니다.
2. 리포지토리 추가를 클릭합니다.
3. 스토리지 공급자 유형을 선택하고 다음 필드를 작성합니다.
 - **AWS 및 MCG를 포함한 S3 공급자용 AWS:**
 - **복제 리포지토리 이름:** MTC 웹 콘솔에서 복제 리포지토리 이름을 지정합니다.
 - **S3 버킷 이름:** S3 버킷의 이름을 지정합니다.
 - **S3 버킷 영역:** S3 버킷 영역을 지정합니다. AWS S3의 경우 필수입니다. 일부 S3 공급자의 경우 선택 사항입니다. S3 공급자의 제품 문서에서 예상되는 값을 확인합니다.
 - **S3 끝점:** 버킷이 아닌 S3 서비스의 URL을 지정합니다(예: **https://<s3-storage.apps.cluster.com>**). 일반 S3 공급자의 경우 필수입니다. **https://** 접두사를 사용해야 합니다.
 - **S3 공급자 액세스 키:** AWS의 경우 **<AWS_SECRET_ACCESS_KEY>** 또는 MCG 및 기타 S3 공급자의 경우 S3 공급자 액세스 키를 지정합니다.
 - **S3 공급자 보안 액세스 키:** AWS의 경우 **<AWS_ACCESS_KEY_ID>** 또는 MCG 및 기타 S3 공급자의 경우 S3 공급자 보안 액세스 키를 지정합니다.
 - **SSL 확인 필요:** 일반 S3 공급자를 사용하는 경우 이 확인란을 지웁니다.
 - 자체 서명된 인증서에 대한 사용자 정의 CA 인증서 번들 파일을 생성한 경우 **검색**을 클릭하고 Base64로 인코딩된 파일을 검색합니다.
 - **GCP:**
 - **복제 리포지토리 이름:** MTC 웹 콘솔에서 복제 리포지토리 이름을 지정합니다.
 - **GCP 버킷 이름:** GCP 버킷의 이름을 지정합니다.
 - **GCP 자격 증명 JSON blob credentials-velero** 파일에서 문자열을 지정합니다.
 - **Azure:**
 - **복제 리포지토리 이름:** MTC 웹 콘솔에서 복제 리포지토리 이름을 지정합니다.
 - **Azure 리소스 그룹:** Azure Blob 스토리지의 리소스 그룹을 지정합니다.
 - **Azure 스토리지 계정 이름:** Azure Blob 스토리지 계정 이름을 지정합니다.
 - **Azure 자격 증명 - INI 파일 콘텐츠 credentials=velero** 파일에서 문자열을 지정합니다.
4. 리포지토리 추가를 클릭하고 연결 유효성 검사를 기다립니다.

5. 닫기를 클릭합니다.
새 리포지토리가 복제 리포지토리 목록에 나타납니다.

8.2.4. MTC 웹 콘솔에서 마이그레이션 계획 생성

MTC(Migration Toolkit for Containers) 웹 콘솔에서 마이그레이션 계획을 생성할 수 있습니다.

사전 요구 사항

- 모든 클러스터에서 **cluster-admin** 권한이 있는 사용자로 로그인합니다.
- 동일한 MTC 버전이 모든 클러스터에 설치되어 있는지 확인해야 합니다.
- MTC 웹 콘솔에 클러스터와 복제 리포지토리를 추가해야 합니다.
- 이동 데이터 복사 방법을 사용하여 PV(영구 볼륨)를 마이그레이션하려면 소스 및 대상 클러스터에 원격 볼륨에 대한 중단되지 않은 네트워크 액세스가 있어야 합니다.
- 직접 이미지 마이그레이션을 사용하려면 소스 클러스터의 이미지 레지스트리에 노출된 경로를 지정해야 합니다. MTC 웹 콘솔을 사용하거나 **MigCluster** 사용자 정의 리소스 매니페스트를 업데이트하여 수행할 수 있습니다.

절차

1. MTC 웹 콘솔에서 **마이그레이션 계획**을 클릭합니다.
2. **마이그레이션 계획 추가**를 클릭합니다.
3. **계획 이름**을 입력합니다.
마이그레이션 계획 이름에서 253자의 소문자 영숫자(**a-z, 0-9**)를 초과해서는 안 되며 공백이나 밑줄(_)을 포함해서는 안 됩니다.
4. **소스 클러스터, 대상 클러스터, 리포지토리**를 선택합니다.
5. **다음**을 클릭합니다.
6. 마이그레이션할 **프로젝트**를 선택합니다.
7. 선택 사항: 프로젝트 옆에 있는 편집 아이콘을 클릭하여 대상 네임스페이스를 변경합니다.
8. **다음**을 클릭합니다.
9. 각 PV의 **마이그레이션 유형**을 선택합니다.
 - **복사** 옵션은 소스 클러스터의 PV에 있는 데이터를 복제 리포지토리에 복사한 다음 대상 클러스터에서 비슷한 특성을 가진 새로 생성된 PV에 데이터를 복원합니다.
 - **이동** 옵션은 소스 클러스터에서 원격 볼륨(예: NFS)을 마운트 해제하고 원격 볼륨을 가리키는 대상 클러스터에 PV 리소스를 생성한 다음 대상 클러스터에 원격 볼륨을 마운트합니다. 대상 클러스터에서 실행되는 애플리케이션은 소스 클러스터와 동일한 원격 볼륨을 사용합니다.
10. **다음**을 클릭합니다.
11. 각 PV의 **복사 방법**을 선택합니다.
 - **스냅샷 복사**는 클라우드 공급자의 스냅샷 기능을 사용하여 데이터를 백업 및 복원합니다. **파일 시스템 복사**보다 훨씬 빠릅니다.

- **파일 시스템 복사**는 소스 클러스터에서 파일을 백업하고 대상 클러스터에서 해당 파일을 복원합니다.
직접 볼륨 마이그레이션에는 파일 시스템 복사 방법이 필요합니다.
12. **복사 확인**을 선택하여 **파일 시스템 복사**로 마이그레이션된 데이터를 확인할 수 있습니다. 데이터는 각 소스 파일에 대한 체크섬을 생성하고 복원 후 체크섬을 확인합니다. 데이터 확인으로 성능이 크게 저하합니다.
 13. **대상 스토리지 클래스**를 선택합니다.
파일 시스템 복사를 선택한 경우 대상 스토리지 클래스를 변경할 수 있습니다.
 14. **다음**을 클릭합니다.
 15. **마이그레이션 옵션** 페이지에서 소스 클러스터에 대해 노출된 이미지 레지스트리 경로를 지정한 경우 **직접 이미지 마이그레이션** 옵션이 선택됩니다. **파일 시스템 복사**로 데이터를 마이그레이션하는 경우 **직접 PV 마이그레이션** 옵션이 선택됩니다.
직접 마이그레이션 옵션은 소스 클러스터에서 대상 클러스터로 직접 이미지 및 파일을 복사합니다. 이 옵션은 소스 클러스터에서 복제 리포지토리로 이미지 및 파일을 복사한 다음 복제 리포지토리에서 대상 클러스터로 복사합니다.
 16. **다음**을 클릭합니다.
 17. 선택 사항: **후크 추가**를 클릭하여 마이그레이션 계획에 후크를 추가합니다.
후크는 사용자 지정 코드를 실행합니다. 단일 마이그레이션 계획에 최대 4개의 후크를 추가할 수 있습니다. 각 후크는 다른 마이그레이션 단계에서 실행됩니다.
 - a. 웹 콘솔에 표시할 후크 이름을 입력합니다.
 - b. 후크가 Ansible 플레이북인 경우 **Ansible 플레이북**을 선택하고 **찾아보기**를 클릭하여 플레이북을 업로드하거나 필드에 플레이북 콘텐츠를 붙여넣습니다.
 - c. 선택 사항: 기본 후크 이미지를 사용하지 않는 경우 Ansible 런타임 이미지를 지정합니다.
 - d. 후크가 Ansible 플레이북이 아닌 경우 **사용자 정의 컨테이너 이미지**를 선택하고 이미지 이름과 경로를 지정합니다.
사용자 정의 컨테이너 이미지에는 Ansible 플레이북이 포함될 수 있습니다.
 - e. **소스 클러스터** 또는 **대상 클러스터**를 선택합니다.
 - f. **서비스 계정 이름**과 **서비스 계정 네임스페이스**를 입력합니다.
 - g. 후크의 마이그레이션 단계를 선택합니다.
 - **사전 백업**: 소스 클러스터에서 애플리케이션 워크로드를 백업하기 전에
 - **백업 후**: 소스 클러스터에서 애플리케이션 워크로드를 백업한 후
 - **사전 복원**: 대상 클러스터에서 애플리케이션 워크로드를 복원하기 전에
 - **복원 후**: 대상 클러스터에서 애플리케이션 워크로드를 복원한 후
 - h. **추가**를 클릭합니다.
 18. **완료**를 클릭합니다.
마이그레이션 계획이 **마이그레이션 계획** 목록에 표시됩니다.

영구 볼륨 복사 방법에 대한 추가 리소스

- MTC 파일 시스템 복사 방법
- MTC 스냅샷 복사 방법

8.2.5. MTC 웹 콘솔에서 마이그레이션 계획 실행

MTC(Migration Toolkit for Containers) 웹 콘솔에서 생성한 마이그레이션 계획을 사용하여 애플리케이션 및 데이터를 마이그레이션할 수 있습니다.



참고

마이그레이션 프로세스 중에 MTC는 마이그레이션된 PV(영구 볼륨)의 회수 정책을 대상 클러스터에서 **Retain**으로 설정합니다.

Backup 사용자 정의 리소스에는 원래 회수 정책을 나타내는 **PVOriginalReclaimPolicy** 주석이 포함되어 있습니다. 마이그레이션된 PV의 회수 정책을 수동으로 복원할 수 있습니다.

사전 요구 사항

MTC 웹 콘솔에는 다음이 포함되어야 합니다.

- **Ready** 상태의 소스 클러스터
- **Ready** 상태의 대상 클러스터
- 복제 리포지토리
- 유효한 마이그레이션 계획

절차

1. MTC 웹 콘솔에 로그인하고 **마이그레이션 계획**을 클릭합니다.

2.  마이그레이션 계획 옆에 있는 옵션 메뉴를 클릭하고 **마이그레이션**에서 다음 옵션 중 하나를 선택합니다.

- **스태이지**에서는 애플리케이션을 중지하지 않고 소스 클러스터에서 대상 클러스터로 데이터를 복사합니다.
- **컷오버**는 소스 클러스터에서 트랜잭션을 중지하고 리소스를 대상 클러스터로 이동합니다.
선택 사항: **컷오버 마이그레이션** 대화 상자에서 **마이그레이션 중 소스 클러스터에서 트랜잭션 중지** 확인란의 선택을 취소할 수 있습니다.
- **상태 복사**는 선택한 PVC(영구 볼륨 클레임)를 복사합니다.



중요

클러스터 간에 네임스페이스를 마이그레이션하는 데 상태 마이그레이션을 사용하지 마십시오. 대신 스테이지 또는 컷오버 마이그레이션을 사용합니다.

- **상태 마이그레이션** 대화 상자에서 하나 이상의 PVC를 선택하고 **마이그레이션**을 클릭합니다.

3. 마이그레이션이 완료되면 OpenShift Container Platform 웹 콘솔에서 애플리케이션이 성공적으로 마이그레이션되었는지 확인합니다.
 - a. **홈** → **프로젝트**를 클릭합니다.
 - b. 마이그레이션된 프로젝트를 클릭하여 상태를 봅니다.
 - c. **경로** 섹션에서 **위치**를 클릭하여 해당되는 경우 애플리케이션이 작동하는지 확인합니다.
 - d. **워크로드** → **포드**를 클릭하여 포드가 마이그레이션된 네임스페이스에서 실행 중인지 확인합니다.
 - e. **스토리지** → **영구 볼륨**을 클릭하여 마이그레이션된 영구 볼륨이 올바르게 프로비저닝되었는지 확인합니다.

9장. 고급 마이그레이션 옵션

대규모 마이그레이션을 수행하고 성능을 개선하기 위해 마이그레이션을 자동화하고 **MigPlan** 및 **MigrationController** 사용자 정의 리소스를 수정할 수 있습니다.

9.1. 용어

표 9.1. MTC 용어

용어	정의
소스 클러스터	애플리케이션이 마이그레이션되는 클러스터입니다.
대상 클러스터 ^[1]	애플리케이션이 마이그레이션될 대상 클러스터입니다.
복제 리포지토리	간접 마이그레이션 중 또는 직접 볼륨 마이그레이션 또는 직접 이미지 마이그레이션 중에 Kubernetes 오브젝트에 대한 이미지, 볼륨 및 Kubernetes 오브젝트 복사에 사용되는 오브젝트 스토리지입니다. 복제 리포지토리는 모든 클러스터에서 액세스할 수 있어야 합니다.
호스트 클러스터	migration-controller pod 및 웹 콘솔이 실행 중인 클러스터입니다. host 클러스터는 일반적으로 대상 클러스터이지만 필수는 아닙니다. 호스트 클러스터에 직접 이미지 마이그레이션을 위해 노출된 레지스트리 경로가 필요하지 않습니다.
원격 클러스터	원격 클러스터는 일반적으로 소스 클러스터이지만 필수는 아닙니다. 원격 클러스터에는 migration-controller 서비스 계정 토큰이 포함된 Secret 사용자 정의 리소스가 필요합니다. 원격 클러스터에는 직접 이미지 마이그레이션을 위해 노출된 보안 레지스트리 경로가 필요합니다.
간접 마이그레이션	이미지, 볼륨 및 Kubernetes 오브젝트는 소스 클러스터에서 복제 리포지토리로 복사한 다음 복제 리포지토리에서 대상 클러스터로 복사됩니다.
직접 볼륨 마이그레이션	영구 볼륨은 소스 클러스터에서 대상 클러스터로 직접 복사됩니다.
직접 이미지 마이그레이션	이미지가 소스 클러스터에서 대상 클러스터로 직접 복사됩니다.
마이그레이션 단계	애플리케이션을 중지하지 않고 데이터가 대상 클러스터에 복사됩니다. 단계적 마이그레이션을 여러 번 실행하면 컷오버 마이그레이션 기간이 단축됩니다.
컷오버 마이그레이션	소스 클러스터에서 애플리케이션이 중지되고 해당 리소스가 대상 클러스터로 마이그레이션됩니다.

용어	정의
상태 마이그레이션	애플리케이션 상태는 특정 영구 볼륨 클레임을 대상 클러스터에 복사하여 마이그레이션됩니다.
마이그레이션 롤백	마이그레이션 롤백은 완료된 마이그레이션을 롤백합니다.

¹ MTC 웹 콘솔에서 대상 클러스터를 호출합니다.

9.2. 명령줄을 사용하여 애플리케이션 마이그레이션

마이그레이션을 자동화하기 위해 CLI(명령줄 인터페이스)를 사용하여 MTC API로 애플리케이션을 마이그레이션할 수 있습니다.

9.2.1. 마이그레이션 사전 요구 사항

- 모든 클러스터에서 **cluster-admin** 권한이 있는 사용자로 로그인합니다.

직접 이미지 마이그레이션

- 소스 클러스터의 보안 내부 레지스트리가 노출되었는지 확인해야 합니다.
- 노출된 레지스트리에 대한 경로를 생성해야 합니다.

직접 볼륨 마이그레이션

- 클러스터에서 프록시를 사용하는 경우 Stunnel TCP 프록시를 구성해야 합니다.

클러스터

- 소스 클러스터를 최신 MTC z-stream 릴리스로 업그레이드해야 합니다.
- MTC 버전은 모든 클러스터에서 동일해야 합니다.

네트워크

- 클러스터는 서로 및 복제 리포지토리에 제한 없이 네트워크 액세스할 수 있습니다.
- move**를 사용하여 영구 볼륨을 복사하는 경우 클러스터에 원격 볼륨에 대한 무제한 네트워크 액세스 권한이 있어야 합니다.
- OpenShift Container Platform 4 클러스터에서 다음 포트를 활성화해야 합니다.
 - 6443** (API 서버)
 - 443** (라우트)
 - 53** (DNS)
- TLS를 사용하는 경우 복제 리포지토리에서 포트 **443**을 활성화해야 합니다.

영구 볼륨 (PV)

- PV가 유효해야 합니다.
- PV를 영구 볼륨 클레임에 바인딩해야 합니다.
- 스냅샷을 사용하여 PV를 복사하는 경우 다음과 같은 추가 사전 요구 사항이 적용됩니다.
 - 클라우드 공급자는 스냅샷을 지원해야 합니다.
 - PV는 동일한 클라우드 공급자에 있어야 합니다.
 - PV는 동일한 지역 리전에 있어야 합니다.
 - PV는 동일한 스토리지 클래스를 보유해야 합니다.

9.2.2. 직접 이미지 마이그레이션을 위한 레지스트리 경로 생성

직접 이미지 마이그레이션의 경우 모든 원격 클러스터에서 공용 내부 레지스트리에 대한 경로를 생성해야 합니다.

사전 요구 사항

- 내부 레지스트리는 모든 원격 클러스터의 외부 트래픽에 노출되어야 합니다. OpenShift Container Platform 4 레지스트리는 기본적으로 공개됩니다.

절차

- OpenShift Container Platform 4 레지스트리에 대한 경로를 생성하려면 다음 명령을 실행합니다.

```
$ oc create route passthrough --service=image-registry -n openshift-image-registry
```

9.2.3. 프록시 설정

OpenShift Container Platform 4.1 및 이전 버전의 경우 이러한 버전은 클러스터 전체 프록시 오브젝트를 지원하지 않기 때문에 Migration Toolkit for Containers Operator를 설치한 후 **MigrationController** CR(사용자 정의 리소스) 매니페스트에서 **proxy**를 구성해야 합니다.

OpenShift Container Platform 4.2 ~ 4.9의 경우 MTC(Migration Toolkit for Containers)는 클러스터 전체 프록시 설정을 상속합니다. 클러스터 전체 프록시 설정을 재정의하려면 프록시 매개변수를 변경할 수 있습니다.

9.2.3.1. 직접 볼륨 마이그레이션

MTC 1.4.2에서 직접 볼륨 마이그레이션(DVM)이 도입되었습니다. DVM은 하나의 프록시만 지원합니다. 대상 클러스터가 프록시 뒤에 있는 경우 소스 클러스터는 대상 클러스터의 경로에 액세스할 수 없습니다.

프록시 뒤에서 소스 클러스터에서 DVM을 수행하려면 전송 계층에서 작동하는 TCP 프록시를 구성하고 자체 SSL 인증서로 암호를 해독하고 재암호화하지 않고도 SSL 연결을 투명하게 전달해야 합니다. Stunnel 프록시는 이러한 프록시의 예입니다.

9.2.3.1.1. DVM용 TCP 프록시 설정

TCP 프록시를 통해 소스와 대상 클러스터 간에 직접 연결하고 프록시를 사용하도록 **MigrationController** CR에서 **stunnel_tcp_proxy** 변수를 구성할 수 있습니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: MigrationController
metadata:
  name: migration-controller
  namespace: openshift-migration
spec:
  [...]
  stunnel_tcp_proxy: http://username:password@ip:port

```

직접 볼륨 마이그레이션(DVM)은 프록시에 대한 기본 인증만 지원합니다. 또한 DVM은 TCP 연결을 투명하게 터널링할 수 있는 프록시 뒤에서만 작동합니다. 중간자 모드에서 HTTP/HTTPS 프록시가 작동하지 않습니다. 기존 클러스터 전체 프록시에서 이 동작을 지원하지 않을 수 있습니다. 결과적으로 DVM의 프록시 설정은 의도적으로 MTC의 일반 프록시 구성과 다르게 유지됩니다.

9.2.3.1.2. HTTP/HTTPS 프록시 대신 TCP 프록시를 사용하는 이유는 무엇입니까?

OpenShift 경로를 통해 소스와 대상 클러스터 간에 Rsync를 실행하여 DVM을 활성화할 수 있습니다. 트래픽은 TCP 프록시인 Stunnel을 사용하여 암호화됩니다. 소스 클러스터에서 실행되는 Stunnel은 대상 Stunnel을 사용한 TLS 연결을 시작하고 암호화된 채널을 통해 데이터를 전송합니다.

OpenShift의 클러스터 전체 HTTP/HTTPS 프록시는 일반적으로 자체 TLS 세션을 외부 서버와 협상하는 중간자 모드로 구성됩니다. 그러나 이 작업은 Stunnel에서는 작동하지 않습니다. Stunnel은 프록시에서 TLS 세션을 그대로 전환해야 하므로 기본적으로 프록시를 통해 TCP 연결을 그대로 전달하는 투명한 터널로 프록시를 설정해야 합니다. 따라서 TCP 프록시를 사용해야 합니다.

9.2.3.1.3. 알려진 문제

마이그레이션 실패 오류 업그레이드 요청 필요

마이그레이션 컨트롤러는 SPDY 프로토콜을 사용하여 원격 Pod 내에서 명령을 실행합니다. 원격 클러스터가 프록시 또는 SPDY 프로토콜을 지원하지 않는 방화벽 뒤에 있는 경우 마이그레이션 컨트롤러가 원격 명령을 실행하지 못합니다. 오류 메시지 **업그레이드 요청** 과 함께 마이그레이션이 실패합니다. 해결방법: SPDY 프로토콜을 지원하는 프록시를 사용합니다.

SPDY 프로토콜 지원 외에도 프록시 또는 방화벽은 **Upgrade** HTTP 헤더를 API 서버에 전달해야 합니다. 클라이언트는 이 헤더를 사용하여 API 서버와의 websocket 연결을 엽니다. **업그레이드** 헤더가 프록시 또는 방화벽에 의해 차단된 경우 마이그레이션은 오류 메시지 **업그레이드 요청 필요** 와 함께 실패합니다. 해결방법: 프록시가 **Upgrade** 헤더를 전달하도록 합니다.

9.2.3.2. 마이그레이션을 위한 네트워크 정책 튜닝

OpenShift는 클러스터에서 사용하는 네트워크 플러그인을 기반으로 *NetworkPolicy* 또는 *EgressFirewall* 을 사용하여 Pod로 트래픽을 제한할 수 있습니다. 마이그레이션과 관련된 소스 네임스페이스가 이러한 메커니즘을 사용하여 네트워크 트래픽을 포드로 제한하는 경우 제한이 마이그레이션 중에 Rsync pod로의 트래픽을 실수로 중지할 수 있습니다.

소스 및 대상 클러스터에서 둘 다 실행되는 rsync Pod는 OpenShift 경로를 통해 서로 연결되어야 합니다. 기존 *NetworkPolicy* 또는 *EgressNetworkPolicy* 오브젝트는 이러한 트래픽 제한에서 Rsync Pod를 자동으로 제외하도록 구성할 수 있습니다.

9.2.3.2.1. NetworkPolicy 구성

9.2.3.2.1.1. Rsync Pod의 송신 트래픽

소스 또는 대상 네임스페이스의 **NetworkPolicy** 구성이 이러한 유형의 트래픽을 차단하는 경우 Rsync Pod의 고유한 레이블을 사용하여 송신 트래픽이 해당 트래픽에서 전달되도록 허용할 수 있습니다. 다음 정책은 네임스페이스의 Rsync Pod에서 모든 송신 트래픽을 허용합니다.

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-all-egress-from-rsync-pods
spec:
  podSelector:
    matchLabels:
      owner: directvolumemigration
      app: directvolumemigration-rsync-transfer
  egress:
  - {}
  policyTypes:
  - Egress
```

9.2.3.2.1.2. Rsync pod로의 수신 트래픽

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-all-egress-from-rsync-pods
spec:
  podSelector:
    matchLabels:
      owner: directvolumemigration
      app: directvolumemigration-rsync-transfer
  ingress:
  - {}
  policyTypes:
  - Ingress
```

9.2.3.2.2. EgressNetworkPolicy 구성

EgressNetworkPolicy 오브젝트 또는 *Egress Firewalls* 는 클러스터를 나가는 송신 트래픽을 차단하도록 설계된 OpenShift 구조입니다.

NetworkPolicy 오브젝트와 달리 Egress Firewall은 네임스페이스의 모든 포트에 적용되므로 프로젝트 수준에서 작동합니다. 따라서 Rsync Pod의 고유 레이블은 제한 사항에서 Rsync Pod만 제외하지 않습니다. 그러나 두 클러스터 간에 직접 연결을 설정할 수 있도록 소스 또는 대상 클러스터의 CIDR 범위를 정책의 허용 규칙에 추가할 수 있습니다.

Egress Firewall이 있는 클러스터를 기반으로 다른 클러스터의 CIDR 범위를 추가하여 둘 사이의 송신 트래픽을 허용할 수 있습니다.

```
apiVersion: network.openshift.io/v1
kind: EgressNetworkPolicy
metadata:
  name: test-egress-policy
  namespace: <namespace>
spec:
  egress:
```

```
- to:
  cidrSelector: <cidr_of_source_or_target_cluster>
  type: Deny
```

9.2.3.2.3. 데이터 전송을 위한 대체 끝점 선택

기본적으로 DVM은 OpenShift Container Platform 경로를 끝점으로 사용하여 PV 데이터를 대상 클러스터로 전송합니다. 클러스터 토폴로지가 허용하는 경우 다른 유형의 지원되는 끝점을 선택할 수 있습니다.

각 클러스터에 대해 **MigrationController** CR의 적절한 대상 클러스터에서 **rsync_endpoint_type** 변수를 설정하여 끝점을 구성할 수 있습니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigrationController
metadata:
  name: migration-controller
  namespace: openshift-migration
spec:
  [...]
  rsync_endpoint_type: [NodePort|ClusterIP|Route]
```

9.2.3.2.4. Rsync Pod에 대한 추가 그룹 구성

PVC에서 공유 스토리지를 사용하는 경우 Pod에서 액세스를 허용하기 위해 Rsync Pod 정의에 추가 그룹을 추가하여 해당 스토리지에 대한 액세스를 구성할 수 있습니다.

표 9.2. Rsync Pod의 보조 그룹

변수	유형	기본값	설명
src_supplemental_groups	string	설정되지 않음	소스 Rsync Pod에 대한 콤마로 구분된 추가 그룹 목록
target_supplemental_groups	string	설정되지 않음	대상 Rsync Pod에 대한 콤마로 구분된 추가 그룹 목록

사용 예

MigrationController CR을 업데이트하여 이러한 추가 그룹에 대한 값을 설정할 수 있습니다.

```
spec:
  src_supplemental_groups: "1000,2000"
  target_supplemental_groups: "2000,3000"
```

9.2.3.3. 프록시 구성

사전 요구 사항

- 모든 클러스터에서 **cluster-admin** 권한이 있는 사용자로 로그인합니다.

절차

1. **MigrationController** CR 매니페스트를 가져옵니다.

```
$ oc get migrationcontroller <migration_controller> -n openshift-migration
```

2. 프록시 매개변수를 업데이트합니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigrationController
metadata:
  name: <migration_controller>
  namespace: openshift-migration
...
spec:
  stunnel_tcp_proxy: http://<username>:<password>@<ip>:<port> 1
  noProxy: example.com 2
```

1 직접 볼륨 마이그레이션을 위한 Stunnel 프록시 URL입니다.

2 프록시를 제외할 대상 도메인 이름, 도메인, IP 주소 또는 기타 네트워크 CIDR의 쉼표로 구분된 목록입니다.

하위 도메인과 일치하려면 도메인 앞에 `.`을 입력합니다. 예를 들어, `.y.com`은 `x.y.com`과 일치하지만 `y.com`은 일치하지 않습니다. `*`를 사용하여 모든 대상에 대해 프록시를 바이패스합니다.

`networking.machineNetwork[].cidr` 필드에 의해 정의된 네트워크에 포함되어 있지 않은 작업자를 설치 구성에서 확장하려면 연결 문제를 방지하기 위해 이 목록에 해당 작업자를 추가해야 합니다.

`httpProxy` 또는 `httpsProxy` 필드가 설정되지 않은 경우 이 필드는 무시됩니다.

3. 매니페스트를 `migration-controller.yaml`로 저장합니다.
4. 업데이트된 매니페스트를 적용합니다.

```
$ oc replace -f migration-controller.yaml -n openshift-migration
```

9.2.4. MTC API를 사용하여 애플리케이션 마이그레이션

MTC(Migration Toolkit for Containers) API를 사용하여 명령줄에서 애플리케이션을 마이그레이션할 수 있습니다.

절차

1. 호스트 클러스터에 대한 **MigCluster** CR 매니페스트를 생성합니다.

```
$ cat << EOF | oc apply -f -
apiVersion: migration.openshift.io/v1alpha1
kind: MigCluster
metadata:
  name: <host_cluster>
  namespace: openshift-migration
```

```
spec:
  isHostCluster: true
EOF
```

2. 각 원격 클러스터에 대한 **Secret** 오브젝트 매니페스트를 생성합니다.

```
$ cat << EOF | oc apply -f -
apiVersion: v1
kind: Secret
metadata:
  name: <cluster_secret>
  namespace: openshift-config
type: Opaque
data:
  saToken: <sa_token>
EOF
```

1. 원격 클러스터의 base64로 인코딩된 **migration-controller** 서비스 계정(SA) 토큰을 지정합니다. 다음 명령을 실행하여 토큰을 가져올 수 있습니다.

```
$ oc sa get-token migration-controller -n openshift-migration | base64 -w 0
```

3. 각 원격 클러스터에 대해 **MigCluster** CR 매니페스트를 생성합니다.

```
$ cat << EOF | oc apply -f -
apiVersion: migration.openshift.io/v1alpha1
kind: MigCluster
metadata:
  name: <remote_cluster>
  namespace: openshift-migration
spec:
  exposedRegistryPath: <exposed_registry_route>
  insecure: false
  isHostCluster: false
  serviceAccountSecretRef:
    name: <remote_cluster_secret>
    namespace: openshift-config
  url: <remote_cluster_url>
EOF
```

1. 원격 클러스터의 **Cluster** CR을 지정합니다.
2. 선택 사항: 직접 이미지 마이그레이션을 위해 노출된 레지스트리 경로를 지정합니다.
3. **false**인 경우 SSL 확인이 활성화됩니다. CA 인증서가 필요하지 않거나 **true**인 경우 확인되지 않습니다.
4. 원격 클러스터의 **Secret** 오브젝트를 지정합니다.
5. 원격 클러스터의 URL을 지정합니다.

4. 모든 클러스터가 **Ready** 상태에 있는지 확인합니다.

■

```
$ oc describe cluster <cluster>
```

5. 복제 리포지토리의 **Secret** 오브젝트 매니페스트를 생성합니다.

```
$ cat << EOF | oc apply -f -
apiVersion: v1
kind: Secret
metadata:
  namespace: openshift-config
  name: <migstorage_creds>
type: Opaque
data:
  aws-access-key-id: <key_id_base64> 1
  aws-secret-access-key: <secret_key_base64> 2
EOF
```

- 1 base64 형식으로 키 ID를 지정합니다.
- 2 base64 형식으로 시크릿 키를 지정합니다.

AWS 인증 정보는 기본적으로 base64로 인코딩됩니다. 다른 스토리지 공급자의 경우 각 키로 다음 명령을 실행하여 인증 정보를 인코딩해야 합니다.

```
$ echo -n "<key>" | base64 -w 0 1
```

- 1 키 ID 또는 시크릿 키를 지정합니다. 두 키 모두 base64로 인코딩되어야 합니다.

6. 복제 리포지토리에 대한 **MigStorage** CR 매니페스트를 생성합니다.

```
$ cat << EOF | oc apply -f -
apiVersion: migration.openshift.io/v1alpha1
kind: MigStorage
metadata:
  name: <migstorage>
  namespace: openshift-migration
spec:
  backupStorageConfig:
    awsBucketName: <bucket> 1
    credsSecretRef:
      name: <storage_secret> 2
      namespace: openshift-config
  backupStorageProvider: <storage_provider> 3
  volumeSnapshotConfig:
    credsSecretRef:
      name: <storage_secret> 4
      namespace: openshift-config
  volumeSnapshotProvider: <storage_provider> 5
EOF
```

- 1 버킷 이름을 지정합니다.
- 2 오브젝트 스토리지의 **Secrets** CR을 지정합니다. 오브젝트 스토리지의 **Secrets** CR에 저장된 인증 정보가 올바른지 확인해야 합니다.

- 3 스토리지 공급자를 지정합니다.
- 4 선택 사항: 스냅샷을 사용하여 데이터를 복사하는 경우 오브젝트 스토리지의 **Secrets** CR을 지정합니다. 오브젝트 스토리지의 **Secrets** CR에 저장된 인증 정보가 올바른지 확인해야 합니다.
- 5 선택 사항: 스냅샷을 사용하여 데이터를 복사하는 경우 스토리지 공급자를 지정합니다.

7. **MigStorage** CR이 **Ready** 상태에 있는지 확인합니다.

```
$ oc describe migstorage <migstorage>
```

8. **MigPlan** CR 매니페스트를 생성합니다.

```
$ cat << EOF | oc apply -f -
apiVersion: migration.openshift.io/v1alpha1
kind: MigPlan
metadata:
  name: <migplan>
  namespace: openshift-migration
spec:
  destMigClusterRef:
    name: <host_cluster>
    namespace: openshift-migration
  indirectImageMigration: true 1
  indirectVolumeMigration: true 2
  migStorageRef:
    name: <migstorage> 3
    namespace: openshift-migration
  namespaces:
    - <source_namespace_1> 4
    - <source_namespace_2>
    - <source_namespace_3>:<destination_namespace> 5
  srcMigClusterRef:
    name: <remote_cluster> 6
    namespace: openshift-migration
EOF
```

- 1 **false**인 경우 직접 이미지 마이그레이션이 활성화됩니다.
- 2 **false**인 경우 직접 볼륨 마이그레이션이 활성화됩니다.
- 3 **MigStorage** CR 인스턴스의 이름을 지정합니다.
- 4 하나 이상의 소스 네임스페이스를 지정합니다. 기본적으로 대상 네임스페이스의 이름은 동일합니다.
- 5 대상 네임스페이스가 소스 네임스페이스와 다른 경우 대상 네임스페이스를 지정합니다.
- 6 소스 클러스터 **MigCluster** 인스턴스의 이름을 지정합니다.

9. **MigPlan** 인스턴스가 **Ready** 상태인지 확인합니다.

```
$ oc describe migplan <migplan> -n openshift-migration
```

10. **MigPlan** 인스턴스에 정의된 마이그레이션을 시작하도록 **MigMigration** CR 매니페스트를 생성합니다.

```
$ cat << EOF | oc apply -f -
apiVersion: migration.openshift.io/v1alpha1
kind: MigMigration
metadata:
  name: <migmigration>
  namespace: openshift-migration
spec:
  migPlanRef:
    name: <migplan> 1
    namespace: openshift-migration
  quiescePods: true 2
  stage: false 3
  rollback: false 4
EOF
```

- 1 **MigPlan** CR 이름을 지정합니다.
- 2 **true**인 경우 마이그레이션 전에 소스 클러스터의 포드가 중지됩니다.
- 3 애플리케이션을 중지하지 않고 대부분의 데이터를 복사하는 단계 마이그레이션이 **true**인 경우 수행됩니다.
- 4 **true**인 경우 완료된 마이그레이션이 롤백됩니다.

11. **MigMigration** CR의 진행 상황을 확인하여 마이그레이션을 확인합니다.

```
$ oc watch migmigration <migmigration> -n openshift-migration
```

출력은 다음과 유사합니다.

출력 예

```
Name:      c8b034c0-6567-11eb-9a4f-0bc004db0fbc
Namespace: openshift-migration
Labels:    migration.openshift.io/migplan-name=django
Annotations: openshift.io/touch: e99f9083-6567-11eb-8420-0a580a81020c
API Version: migration.openshift.io/v1alpha1
Kind:      MigMigration
...
Spec:
  Mig Plan Ref:
    Name:      migplan
    Namespace: openshift-migration
  Stage:      false
Status:
  Conditions:
    Category:      Advisory
    Last Transition Time: 2021-02-02T15:04:09Z
```

```

Message:      Step: 19/47
Reason:      InitialBackupCreated
Status:      True
Type:       Running
Category:    Required
Last Transition Time: 2021-02-02T15:03:19Z
Message:      The migration is ready.
Status:      True
Type:       Ready
Category:    Required
Durable:     true
Last Transition Time: 2021-02-02T15:04:05Z
Message:      The migration registries are healthy.
Status:      True
Type:       RegistriesHealthy
Itinerary:   Final
Observed Digest:
7fae9d21f15979c71ddc7dd075cb97061895caac5b936d92fae967019ab616d5
Phase:      InitialBackupCreated
Pipeline:
  Completed: 2021-02-02T15:04:07Z
  Message:   Completed
  Name:     Prepare
  Started:  2021-02-02T15:03:18Z
  Message:  Waiting for initial Velero backup to complete.
  Name:     Backup
  Phase:    InitialBackupCreated
  Progress:
    Backup openshift-migration/c8b034c0-6567-11eb-9a4f-0bc004db0fbc-wpc44: 0 out of
    estimated total of 0 objects backed up (5s)
  Started:  2021-02-02T15:04:07Z
  Message:  Not started
  Name:     StageBackup
  Message:  Not started
  Name:     StageRestore
  Message:  Not started
  Name:     DirectImage
  Message:  Not started
  Name:     DirectVolume
  Message:  Not started
  Name:     Restore
  Message:  Not started
  Name:     Cleanup
Start Timestamp: 2021-02-02T15:03:18Z
Events:
Type Reason Age      From      Message
---- -
Normal Running 57s      migmigration_controller Step: 2/47
Normal Running 57s      migmigration_controller Step: 3/47
Normal Running 57s (x3 over 57s) migmigration_controller Step: 4/47
Normal Running 54s      migmigration_controller Step: 5/47
Normal Running 54s      migmigration_controller Step: 6/47
Normal Running 52s (x2 over 53s) migmigration_controller Step: 7/47
Normal Running 51s (x2 over 51s) migmigration_controller Step: 8/47

```

```
Normal Ready 50s (x12 over 57s) migmigration_controller The migration is ready.
Normal Running 50s migmigration_controller Step: 9/47
Normal Running 50s migmigration_controller Step: 10/47
```

9.2.5. 상태 마이그레이션

MTC(Migration Toolkit for Containers)를 사용하여 애플리케이션 상태를 구성하는 PVC(영구 볼륨 클레임)를 마이그레이션하여 반복 가능한 상태 전용 마이그레이션을 수행할 수 있습니다. 마이그레이션 계획에서 다른 PVC를 제외하여 지정된 PVC를 마이그레이션합니다. PVC를 매핑하여 소스 및 대상 PVC가 동기화되었는지 확인할 수 있습니다. 영구 볼륨(PV) 데이터는 대상 클러스터에 복사됩니다. PV 참조는 이동되지 않으며 애플리케이션 포드는 소스 클러스터에서 계속 실행됩니다.

상태 마이그레이션은 OpenShift Gitops와 같은 외부 CD 메커니즘과 함께 사용하도록 특별히 설계되었습니다. MTC를 사용하여 상태를 마이그레이션하는 동안 GitOps를 사용하여 애플리케이션 매니페스트를 마이그레이션할 수 있습니다.

CI/CD 파이프라인이 있는 경우 대상 클러스터에 배포하여 상태 비저장 구성 요소를 마이그레이션할 수 있습니다. 그런 다음 MTC를 사용하여 상태 저장 구성 요소를 마이그레이션할 수 있습니다.

클러스터 간에 또는 동일한 클러스터 내에서 상태 마이그레이션을 수행할 수 있습니다.



중요

상태 마이그레이션은 애플리케이션 상태를 구성하는 구성 요소만 마이그레이션합니다. 전체 네임스페이스를 마이그레이션하려면 stage 또는 cutover 마이그레이션을 사용합니다.

사전 요구 사항

- 소스 클러스터의 애플리케이션 상태는 **PersistentVolumeClaim s** 를 통해 **프로비저닝된 PersistentVolume** 에 유지됩니다.
- 애플리케이션의 매니페스트는 소스와 대상 클러스터에서 모두 액세스할 수 있는 중앙 리포지토리에서 사용할 수 있습니다.

절차

1. 소스에서 대상 클러스터로 영구 볼륨 데이터를 마이그레이션합니다.
이 단계는 필요한 만큼 여러 번 수행할 수 있습니다. 소스 애플리케이션이 계속 실행됩니다.
2. 소스 애플리케이션을 격리합니다.
소스 클러스터에서 직접 워크로드 리소스의 복제본을 **0** 으로 설정하거나 GitHub에서 매니페스트를 업데이트하고 Argo CD 애플리케이션을 다시 동기화하여 이를 수행할 수 있습니다.
3. 대상 클러스터에 애플리케이션 매니페스트를 복제합니다.
Argo CD를 사용하여 애플리케이션 매니페스트를 대상 클러스터에 복제할 수 있습니다.
4. 나머지 볼륨 데이터를 소스에서 대상 클러스터로 마이그레이션합니다.
최종 데이터 마이그레이션을 수행하여 상태 마이그레이션 프로세스 중에 애플리케이션이 생성한 새 데이터를 마이그레이션합니다.
5. 복제된 애플리케이션이 정지된 상태인 경우 정지되지 않습니다.
6. DNS 레코드를 대상 클러스터로 전환하여 사용자 트래픽을 마이그레이션된 애플리케이션으로 리디렉션합니다.



참고

MTC 1.6 상태 마이그레이션을 수행할 때 자동으로 애플리케이션을 격리할 수 없습니다. PV 데이터만 마이그레이션할 수 있습니다. 따라서 애플리케이션 정지 또는 정지를 위해 CD 메커니즘을 사용해야 합니다.

MTC 1.7은 명시적 단계 및 Cutover 흐름을 소개합니다. 스테이징을 사용하여 필요에 따라 초기 데이터 전송을 여러 번 수행할 수 있습니다. 그런 다음 소스 애플리케이션이 자동으로 초기화되는 cutover를 수행할 수 있습니다.

사전 요구 사항

- 소스 클러스터의 애플리케이션 상태는 **PersistentVolumeClaim s** 를 통해 **프로비저닝된 PersistentVolume** 에 유지됩니다.
- 애플리케이션의 매니페스트는 소스와 대상 클러스터에서 모두 액세스할 수 있는 중앙 리포지토리에서 사용할 수 있습니다.

절차

1. 소스에서 대상 클러스터로 영구 볼륨 데이터를 마이그레이션합니다.
이 단계는 필요한 만큼 여러 번 수행할 수 있습니다. 소스 애플리케이션이 계속 실행됩니다.
2. 소스 애플리케이션을 격리합니다.
소스 클러스터에서 직접 워크로드 리소스의 복제본을 **0** 으로 설정하거나 GitHub에서 매니페스트를 업데이트하고 Argo CD 애플리케이션을 다시 동기화하여 이를 수행할 수 있습니다.
3. 대상 클러스터에 애플리케이션 매니페스트를 복제합니다.
Argo CD를 사용하여 애플리케이션 매니페스트를 대상 클러스터에 복제할 수 있습니다.
4. 나머지 볼륨 데이터를 소스에서 대상 클러스터로 마이그레이션합니다.
최종 데이터 마이그레이션을 수행하여 상태 마이그레이션 프로세스 중에 애플리케이션이 생성한 새 데이터를 마이그레이션합니다.
5. 복제된 애플리케이션이 정지된 상태인 경우 정지되지 않습니다.
6. DNS 레코드를 대상 클러스터로 전환하여 사용자 트래픽을 마이그레이션된 애플리케이션으로 리디렉션합니다.



참고

MTC 1.6 상태 마이그레이션을 수행할 때 자동으로 애플리케이션을 격리할 수 없습니다. PV 데이터만 마이그레이션할 수 있습니다. 따라서 애플리케이션 정지 또는 정지를 위해 CD 메커니즘을 사용해야 합니다.

MTC 1.7은 명시적 단계 및 Cutover 흐름을 소개합니다. 스테이징을 사용하여 필요에 따라 초기 데이터 전송을 여러 번 수행할 수 있습니다. 그런 다음 소스 애플리케이션이 자동으로 초기화되는 cutover를 수행할 수 있습니다.

추가 리소스

- 상태 마이그레이션에 대한 [마이그레이션에서 PVC 제외](#) 를 참조하십시오.
- 대상 클러스터 의 [프로비저닝된 PVC로 소스 PV 데이터를 마이그레이션하기 위한 PVC 매핑](#) 을 참조하십시오.

- 애플리케이션 상태를 구성하는 [Kubernetes 오브젝트](#) 를 마이그레이션하려면 Kubernetes 오브젝트 마이그레이션을 참조하십시오.

9.3. 마이그레이션 후크

마이그레이션의 다른 단계에서 각 후크가 실행되고 단일 마이그레이션 계획에 최대 4개의 마이그레이션 후크를 추가할 수 있습니다. 마이그레이션 후크는 애플리케이션 정지 사용자 정의, 지원되지 않는 데이터 유형을 수동으로 마이그레이션 및 마이그레이션 후 애플리케이션 업데이트와 같은 작업을 수행합니다.

마이그레이션 후크는 다음 마이그레이션 단계 중 하나에서 소스 또는 대상 클러스터에서 실행됩니다.

- **PreBackup:** 소스 클러스터에서 리소스를 백업하기 전
- **PostBackup:** 소스 클러스터에서 리소스를 백업한 후
- **PreRestore:** 대상 클러스터에서 리소스가 복원되기 전
- **PostRestore:** 대상 클러스터에서 리소스가 복원된 후

기본 Ansible 이미지 또는 사용자 정의 후크 컨테이너로 실행되는 Ansible 플레이북을 생성하여 후크를 생성할 수 있습니다.

Ansible 플레이북

Ansible 플레이북은 후크 컨테이너에 구성 맵으로 마운트됩니다. 후크 컨테이너는 **MigPlan** 사용자 정의 리소스에 지정된 클러스터, 서비스 계정 및 네임스페이스를 사용하여 작업으로 실행됩니다. 작업은 기본 6번의 재시도 한도에 도달하거나 성공적으로 완료될 때까지 계속 실행됩니다. 이는 초기 포드가 제거되거나 종료된 경우에도 계속됩니다.

기본 Ansible 런타임 이미지는 registry.redhat.io/rhmtc/openshift-migration-hook-runner-rhel7:1.7 입니다. 이 이미지는 Ansible Runner 이미지를 기반으로 하며 Ansible Kubernetes 리소스에 대해 **python-openshift** 및 업데이트된 **oc** 바이너리를 포함합니다.

사용자 정의 후크 컨테이너

기본 Ansible 이미지 대신 사용자 정의 후크 컨테이너를 사용할 수 있습니다.

9.3.1. 마이그레이션 후크의 Ansible 플레이북 작성

마이그레이션 후크로 사용할 Ansible 플레이북을 작성할 수 있습니다. MTC 웹 콘솔을 사용하거나 **MigPlan** 사용자 정의 리소스(CR) 매니페스트에서 **spec.hooks** 매개변수의 값을 지정하여 후크가 마이그레이션 계획에 추가됩니다.

Ansible 플레이북은 후크 컨테이너에 구성 맵으로 마운트됩니다. 후크 컨테이너는 **MigPlan** CR에 지정된 클러스터, 서비스 계정 및 네임스페이스를 사용하여 작업으로 실행됩니다. 후크 컨테이너는 클러스터에서 실행되기 전에 작업에 인증이 필요하지 않도록 지정된 서비스 계정 토큰을 사용합니다.

9.3.1.1. Ansible 모듈

Ansible **shell** 모듈을 사용하여 **oc** 명령을 실행할 수 있습니다.

shell 모듈 예

```
- hosts: localhost
gather_facts: false
tasks:
```

```
- name: get pod name
  shell: oc get po --all-namespaces
```

k8s_info와 같은 **kubernetes.core** 모듈을 사용하여 Kubernetes 리소스와 상호 작용할 수 있습니다.

k8s_facts 모듈 예

```
- hosts: localhost
  gather_facts: false
  tasks:
  - name: Get pod
    k8s_info:
      kind: pods
      api: v1
      namespace: openshift-migration
      name: "{{ lookup('env', 'HOSTNAME') }}"
      register: pods

  - name: Print pod name
    debug:
      msg: "{{ pods.resources[0].metadata.name }}"
```

fail 모듈을 사용하여 0이 아닌 종료 상태가 정상적으로 생성되지 않는 경우 후크의 성공 또는 실패 여부를 확인할 수 있습니다. 후크는 작업으로 실행되며 후크의 성공 또는 실패 상태는 작업 컨테이너의 종료 상태를 기반으로 합니다.

fail 모듈 예

```
- hosts: localhost
  gather_facts: false
  tasks:
  - name: Set a boolean
    set_fact:
      do_fail: true

  - name: "fail"
    fail:
      msg: "Cause a failure"
      when: do_fail
```

9.3.1.2. 환경 변수

MigPlan CR 이름과 마이그레이션 네임스페이스는 환경 변수로 후크 컨테이너에 전달됩니다. 이러한 변수는 **lookup** 플러그인을 사용하여 액세스할 수 있습니다.

환경 변수 예

```
- hosts: localhost
  gather_facts: false
  tasks:
  - set_fact:
      namespaces: "{{ (lookup('env', 'MIGRATION_NAMESPACES')).split(',') }}"

  - debug:
```

```
msg: "{{ item }}"
with_items: "{{ namespaces }}"

- debug:
  msg: "{{ lookup('env', 'MIGRATION_PLAN_NAME') }}"
```

9.4. 마이그레이션 계획 옵션

MigPlan CR(사용자 정의 리소스)에서 구성 요소를 제외, 편집 및 매핑할 수 있습니다.

9.4.1. 리소스 제외

마이그레이션하기 위해 리소스 로드를 줄이거나 다른 도구를 사용하여 이미지 또는 PV를 마이그레이션하기 위해 MTC(Migration Toolkit for Containers) 마이그레이션 계획에서 리소스(예: 이미지 스트림, 영구 볼륨(PV) 또는 서브스크립션)를 제외할 수 있습니다.

기본적으로 MTC는 서비스 카탈로그 리소스 및 OLM(Operator Lifecycle Manager) 리소스를 마이그레이션에서 제외합니다. 이러한 리소스는 현재 마이그레이션에 지원되지 않는 서비스 카탈로그 API 그룹 및 OLM API 그룹의 일부입니다.

절차

1. **MigrationController** 사용자 지정 매니페스트를 편집합니다.

```
$ oc edit migrationcontroller <migration_controller> -n openshift-migration
```

2. 특정 리소스를 제외하도록 매개변수를 추가하여 **사양** 섹션을 업데이트합니다. 자체 제외 매개변수가 없는 리소스에 **additional_excluded_resources** 매개변수를 추가합니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigrationController
metadata:
  name: migration-controller
  namespace: openshift-migration
spec:
  disable_image_migration: true 1
  disable_pv_migration: true 2
  additional_excluded_resources: 3
  - resource1
  - resource2
  ...
```

- 1 마이그레이션에서 이미지 스트림을 제외하려면 **disable_image_migration: true**를 추가합니다. **MigrationController** 포드가 다시 시작되면 **imagestreams**가 **main.yml**의 **excluded_resources** 목록에 추가됩니다.
- 2 마이그레이션 계획에서 PV를 제외하려면 **disable_pv_migration: true**를 추가합니다. **MigrationController** 포드가 다시 시작되면 **persistentvolumes** 및 **persistentvolumeclaims**가 **main.yml**의 **excluded_resources** 목록에 추가됩니다. PV 마이그레이션을 비활성화하면 마이그레이션 계획을 생성할 때 PV 검색도 비활성화됩니다.
- 3 **additional_excluded_resources** 목록에 제외하려는 OpenShift Container Platform 리소스를 추가할 수 있습니다.

3. 변경 사항이 적용되도록 **MigrationController** 포드가 다시 시작될 때까지 2분 정도 기다립니다.
4. 리소스가 제외되었는지 확인합니다.

```
$ oc get deployment -n openshift-migration migration-controller -o yaml | grep EXCLUDED_RESOURCES -A1
```

출력에는 제외된 리소스가 포함됩니다.

출력 예

```
name: EXCLUDED_RESOURCES
value:
resource1,resource2,imagetags,templateinstances,clusterserviceversions,packagemanifests,sul
scriptions,servicebrokers,servicebindings,serviceclasses,serviceinstances,serviceplans,imaget
ams,persistentvolumes,persistentvolumeclaims
```

9.4.2. 네임스페이스 매핑

MigPlan CR(사용자 정의 리소스)에서 네임스페이스를 매핑하는 경우 마이그레이션 중에 네임스페이스의 UID 및 GID 범위가 복사되므로 소스 또는 대상 클러스터에서 네임스페이스가 복제되지 않아야 합니다.

두 개의 소스 네임스페이스가 동일한 대상 네임스페이스에 매핑됨

```
spec:
  namespaces:
  - namespace_2
  - namespace_1:namespace_2
```

소스 네임스페이스를 동일한 이름의 네임스페이스에 매핑하려면 매핑을 생성할 필요가 없습니다. 기본적으로 소스 네임스페이스와 대상 네임스페이스의 이름은 동일합니다.

잘못된 네임 스페이스 매핑

```
spec:
  namespaces:
  - namespace_1:namespace_1
```

올바른 네임 스페이스 참조

```
spec:
  namespaces:
  - namespace_1
```

9.4.3. 영구 볼륨 클레임 제외

마이그레이션할 PVC를 제외하고 상태 마이그레이션에 대해 PVC(영구 볼륨 클레임)를 선택합니다. PV(영구 볼륨)가 검색된 후 **MigPlan** 사용자 정의 리소스(CR)의 **spec.persistentVolumes.pvc.selection.action** 매개변수를 설정하여 PVC를 제외합니다.

사전 요구 사항

- **MigPlan** CR이 **Ready** 상태입니다.

절차

- **spec.persistentVolumes.pvc.selection.action** 매개변수를 **MigPlan** CR에 추가하고 건너뛰도록 설정합니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: MigPlan
metadata:
  name: <migplan>
  namespace: openshift-migration
spec:
  ...
  persistentVolumes:
  - capacity: 10Gi
    name: <pv_name>
    pvc:
  ...
  selection:
    action: skip

```

9.4.4. 영구 볼륨 클레임 매핑

PVC를 매핑하여 소스 클러스터에서 PV(영구 볼륨 클레임) 데이터를 **MigPlan** CR의 대상 클러스터에 이미 프로비저닝한 PVC(영구 볼륨 클레임)로 마이그레이션할 수 있습니다. 이 매핑을 사용하면 마이그레이션된 애플리케이션의 대상 PVC가 소스 PVC와 동기화됩니다.

PV가 검색된 후 **MigPlan** CR(사용자 정의 리소스)에서 **spec.persistentVolumes.pvc.name** 매개변수를 업데이트하여 PVC를 매핑합니다.

사전 요구 사항

- **MigPlan** CR이 **Ready** 상태입니다.

절차

- **MigPlan** CR에서 **spec.persistentVolumes.pvc.name** 매개변수를 업데이트합니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: MigPlan
metadata:
  name: <migplan>
  namespace: openshift-migration
spec:
  ...
  persistentVolumes:
  - capacity: 10Gi
    name: <pv_name>
    pvc:
      name: <source_pvc>:<destination_pvc> ❶

```

- ❶ 소스 클러스터에서 PVC와 대상 클러스터에서 PVC를 지정합니다. 대상 PVC가 없으면 생성됩니다. 이 매핑을 사용하여 마이그레이션 중에 PVC 이름을 변경할 수 있습니다.

9.4.5. 영구 볼륨 속성 편집

MigPlan CR(사용자 정의 리소스)을 생성한 후 **MigrationController** CR은 PV(영구 볼륨)를 검색합니다. **spec.persistentVolumes** 블록 및 **status.destStorageClasses** 블록이 **MigPlan** CR에 추가됩니다.

spec.persistentVolumes.selection 블록에서 값을 편집할 수 있습니다.

spec.persistentVolumes.selection 블록 외부의 값을 변경하는 경우 **MigPlan** CR이 **MigrationController** CR에 의해 조정될 때 값을 덮어씁니다.

참고

spec.persistentVolumes.selection.storageClass 매개변수의 기본값은 다음 논리에 따라 결정됩니다.

1. 소스 클러스터 PV가 Gluster 또는 NFS인 경우 **accessMode: ReadWriteMany** 또는 **ceph rbd**의 기본값은 **cephfs** 로, **accessMode: ReadWriteOnce** 의 경우 기본값은 **ceph fs**입니다.
2. PV가 Gluster와 NFS가 아니거나 **cephfs** 또는 **cephrbd** 를 사용할 수 없는 경우 기본값은 동일한 프로비저너의 스토리지 클래스입니다.
3. 동일한 프로비저너의 스토리지 클래스를 사용할 수 없는 경우 기본값은 대상 클러스터의 기본 스토리지 클래스입니다.

storageClass 값은 **MigPlan** CR의 **status.destStorageClasses** 블록에서 **name** 매개변수 값으로 변경할 수 있습니다.

storageClass 값이 비어 있으면 마이그레이션 후 PV에 스토리지 클래스가 없습니다. 예를 들어 대상 클러스터의 NFS 볼륨으로 PV를 이동하려는 경우 이 옵션이 적합합니다.

사전 요구 사항

- **MigPlan** CR이 **Ready** 상태입니다.

절차

- **MigPlan** CR에서 **spec.persistentVolumes.selection** 값을 편집합니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigPlan
metadata:
  name: <migplan>
  namespace: openshift-migration
spec:
  persistentVolumes:
    - capacity: 10Gi
      name: pvc-095a6559-b27f-11eb-b27f-021bddcaf6e4
      proposedCapacity: 10Gi
      pvc:
        accessModes:
          - ReadWriteMany
        hasReference: true
        name: mysql
        namespace: mysql-persistent
      selection:
        action: <copy> 1
```

```

copyMethod: <filesystem> 2
verify: true 3
storageClass: <gp2> 4
accessMode: <ReadWriteMany> 5
storageClass: cephfs

```

- 1 허용되는 값은 **move, copy, skip** 입니다. 하나의 작업만 지원되는 경우 기본값은 지원되는 작업입니다. 여러 작업이 지원되는 경우 기본값은 **copy** 입니다.
- 2 허용되는 값은 스냅샷 과 파일 시스템입니다. 기본값은 **filesystem** 입니다.
- 3 MTC 웹 콘솔에서 파일 시스템 복사에 대한 확인 옵션을 선택하면 **verify** 매개변수가 표시됩니다. **false** 로 설정할 수 있습니다.
- 4 **MigPlan** CR의 **status.destStorageClasses** 블록에서 **name** 매개변수 값으로 기본값을 변경할 수 있습니다. 값을 지정하지 않으면 마이그레이션 후 PV에 스토리지 클래스가 없습니다.
- 5 허용되는 값은 **ReadWriteOnce** 및 **ReadWriteMany** 입니다. 이 값을 지정하지 않으면 기본값은 소스 클러스터 PVC의 액세스 모드입니다. **MigPlan** CR에서 액세스 모드만 편집할 수 있습니다. MTC 웹 콘솔을 사용하여 편집할 수 없습니다.

9.4.6. MTC 웹 콘솔에서 스토리지 클래스 변환

동일한 클러스터 내에서 마이그레이션하여 PV(영구 볼륨)의 스토리지 클래스를 변환할 수 있습니다. 이렇게 하려면 MTC(Migration Toolkit for Containers) 웹 콘솔에서 마이그레이션 계획을 생성하고 실행해야 합니다.

사전 요구 사항

- MTC가 실행 중인 클러스터에 **cluster-admin** 권한이 있는 사용자로 로그인해야 합니다.
- MTC 웹 콘솔에 클러스터를 추가해야 합니다.

절차

1. OpenShift Container Platform 웹 콘솔의 왼쪽 탐색 창에서 **프로젝트**를 클릭합니다.
2. 프로젝트 목록에서 프로젝트를 클릭합니다.
프로젝트 세부 정보 페이지가 열립니다.
3. **DeploymentConfig** 이름을 클릭합니다. 실행 중인 포드의 이름을 기록해 둡니다.
4. 프로젝트의 YAML 탭을 엽니다. PV를 찾아 해당 PVC(영구 볼륨 클레임)의 이름을 확인합니다.
5. MTC 웹 콘솔에서 **마이그레이션 계획**을 클릭합니다.
6. **마이그레이션 계획** 추가를 클릭합니다.
7. **계획** 이름을 입력합니다.
마이그레이션 계획 이름에는 3~63자의 소문자 영숫자(**a-z, 0-9**)가 포함되어야 하며 공백이나 밑줄(_)을 포함해서는 안 됩니다.
8. **마이그레이션 유형** 메뉴에서 **스토리지 클래스 변환**을 선택합니다.

9. 소스 클러스터 목록에서 스토리지 클래스 변환을 위해 원하는 클러스터를 선택합니다.
10. 다음을 클릭합니다.
네임스페이스 페이지가 열립니다.
11. 필요한 프로젝트를 선택합니다.
12. 다음을 클릭합니다.
영구 볼륨 페이지가 열립니다. 페이지에 기본적으로 선택된 프로젝트의 PV가 표시됩니다.
13. 각 PV에 대해 원하는 대상 스토리지 클래스를 선택합니다.
14. 다음을 클릭합니다.
마법사에서 새 마이그레이션 계획을 검증하고 준비가 완료되었음을 표시합니다.
15. 단기를 클릭합니다.
새 계획이 마이그레이션 계획 페이지에 나타납니다.
16. 변환을 시작하려면 새 계획의 옵션 메뉴를 클릭합니다.
Migrations 에서 두 가지 옵션, Stage 및 Cutover 가 표시됩니다.



참고

애플리케이션의 PVC 참조를 잘라내십시오.

단계 마이그레이션은 애플리케이션의 PVC 참조를 업데이트하지 않습니다.

17. 원하는 옵션을 선택합니다.
선택한 옵션에 따라 Stage 마이그레이션 또는 Cutover 마이그레이션 알림이 표시됩니다.
18. 마이그레이션을 클릭합니다.
선택한 옵션에 따라 Stage started 또는 Cutover started 메시지가 표시됩니다.
19. 현재 마이그레이션 상태를 보려면 Migrations 열에서 번호를 클릭합니다.
마이그레이션 페이지가 열립니다.
20. 현재 마이그레이션에 대한 자세한 내용과 진행 상황을 모니터링하려면 Type (유형) 열에서 마이그레이션을 선택합니다.
마이그레이션 세부 정보 페이지가 열립니다. DirectVolume 단계로 마이그레이션이 진행되고 단계의 상태가 Running Rsync Pod가 되어 영구 볼륨 데이터를 마이그레이션 하면 세부 정보 보기를 클릭하고 복사본의 세부 정보를 확인할 수 있습니다.
21. 이동 경로 표시줄에서 Stage 또는 Cutover 를 클릭하고 모든 단계가 완료될 때까지 기다립니다.
22. OpenShift Container Platform 웹 콘솔의 PersistentVolumeClaims 탭을 엽니다.
대상 스토리지 클래스를 사용하는 새 PVC는 초기 PVC의 이름으로 표시되지만 새 PVC로 끝낼 수 있습니다.
23. 왼쪽 탐색 창에서 포드 를 클릭합니다. 프로젝트의 포드가 다시 실행되고 있는지 확인합니다.

추가 리소스

- 동작 및 복사 작업에 대한 자세한 내용은 [MTC 워크플로를](#) 참조하십시오.
- 건너뛰기 작업에 대한 자세한 내용은 [마이그레이션에서 PVC 제외](#)를 참조하십시오.

- 파일 시스템 및 스냅샷 복사 방법에 대한 자세한 내용은 [데이터 복사 방법](#) 정보를 참조하십시오.

9.4.7. MTC API를 사용하여 Kubernetes 오브젝트의 상태 마이그레이션 수행

모든 PV 데이터를 마이그레이션한 후 MTC(Migration Toolkit for Containers) API를 사용하여 애플리케이션을 구성하는 Kubernetes 오브젝트의 일회성 상태 마이그레이션을 수행할 수 있습니다.

MigPlan 사용자 정의 리소스(CR) 필드를 구성하여 Kubernetes 리소스 목록을 추가 라벨 선택기와 함께 제공하여 이러한 리소스를 추가로 필터링한 다음 **MigMigration** CR을 생성하여 마이그레이션을 수행하여 수행합니다. **MigPlan** 리소스는 마이그레이션 후 종료됩니다.



참고

Kubernetes 리소스를 선택하는 것은 API 전용 기능입니다. **MigPlan** CR을 업데이트하고 CLI를 사용하여 **MigMigration** CR을 생성해야 합니다. MTC 웹 콘솔은 Kubernetes 오브젝트 마이그레이션을 지원하지 않습니다.



참고

마이그레이션 후 **MigPlan** CR의 **closed** 매개변수가 **true**로 설정됩니다. 이 **MigPlan** CR에 대해 다른 **MigMigration** CR을 생성할 수 없습니다.

다음 옵션 중 하나를 사용하여 **MigPlan** CR에 Kubernetes 오브젝트를 추가합니다.

- Kubernetes 오브젝트를 **includedResources** 섹션에 추가합니다. **MigPlan** CR에 포함된 **Resources** 필드가 지정되면 계획에서 **group-kind** 목록을 입력으로 사용합니다. 목록에 있는 리소스만 마이그레이션에 포함됩니다.
- 선택적 **labelSelector** 매개변수를 추가하여 **MigPlan**에 포함된 **Resources**를 필터링합니다. 이 필드를 지정하면 라벨 선택기와 일치하는 리소스만 마이그레이션에 포함됩니다. 예를 들어 **app: frontend** 레이블을 필터로 사용하여 **Secret** 및 **ConfigMap** 리소스 목록을 필터링할 수 있습니다.

절차

1. Kubernetes 리소스를 포함하도록 **MigPlan** CR을 업데이트하고, 선택적으로 **labelSelector** 매개변수를 추가하여 포함된 리소스를 필터링합니다.
 - a. Kubernetes 리소스를 포함하도록 **MigPlan** CR을 업데이트하려면 다음을 수행합니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigPlan
metadata:
  name: <migplan>
  namespace: openshift-migration
spec:
  includedResources:
    - kind: <kind> 1
      group: ""
    - kind: <kind>
      group: ""
```

1. Kubernetes 오브젝트를 지정합니다(예: **Secret** 또는 **ConfigMap**).

- b. 선택 사항: **labelSelector** 매개변수를 추가하여 포함된 리소스를 필터링하려면 다음을 수행합니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: MigPlan
metadata:
  name: <migplan>
  namespace: openshift-migration
spec:
  includedResources:
    - kind: <kind> 1
      group: ""
    - kind: <kind>
      group: ""
  ...
  labelSelector:
    matchLabels:
      <label> 2

```

- 1 Kubernetes 오브젝트를 지정합니다(예: **Secret** 또는 **ConfigMap**).
- 2 마이그레이션할 리소스의 레이블을 지정합니다(예 : **app: frontend**).

2. 선택한 Kubernetes 리소스를 마이그레이션하기 위해 **MigMigration** CR을 생성합니다. **migPlanRef**:에서 올바른 **MigPlan** 이 참조되었는지 확인합니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: MigMigration
metadata:
  generateName: <migplan>
  namespace: openshift-migration
spec:
  migPlanRef:
    name: <migplan>
    namespace: openshift-migration
  stage: false

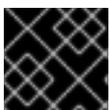
```

9.5. 마이그레이션 컨트롤러 옵션

마이그레이션 계획 제한을 편집하거나, 대규모 마이그레이션 및 성능 향상을 위해 **MigrationController** CR(사용자 정의 리소스)에서 캐시된 Kubernetes 클라이언트를 활성화하고, 영구 볼륨 크기 조정을 활성화할 수 있습니다.

9.5.1. 대규모 마이그레이션에 대한 제한 증가

대규모 마이그레이션을 위해 MTC(Migration Toolkit for Containers)로 마이그레이션 오브젝트 및 컨테이너 리소스에 대한 마이그레이션 컨트롤러 제한을 늘릴 수 있습니다.



중요

프로덕션 환경에서 마이그레이션을 수행하기 전에 이러한 변경 사항을 테스트해야 합니다.

절차

1. **MigrationController** 사용자 지정 (CR) 매니페스트를 편집합니다.

```
$ oc edit migrationcontroller -n openshift-migration
```

2. 다음 매개 변수를 업데이트합니다.

```
...
mig_controller_limits_cpu: "1" 1
mig_controller_limits_memory: "10Gi" 2
...
mig_controller_requests_cpu: "100m" 3
mig_controller_requests_memory: "350Mi" 4
...
mig_pv_limit: 100 5
mig_pod_limit: 100 6
mig_namespace_limit: 10 7
...
```

- 1 **MigrationController** CR에서 사용할 수 있는 CPU 수를 지정합니다.
- 2 **MigrationController** CR에서 사용할 수 있는 메모리 양을 지정합니다.
- 3 **MigrationController** CR 요청에 사용할 수 있는 CPU 단위 수를 지정합니다. **100m**은 0.1 CPU 단위(100 * 1e-3)를 나타냅니다.
- 4 **MigrationController** CR 요청에 사용할 수 있는 메모리 양을 지정합니다.
- 5 마이그레이션할 수 있는 영구 볼륨 수를 지정합니다.
- 6 마이그레이션할 수 있는 포드 수를 지정합니다.
- 7 마이그레이션할 수 있는 네임스페이스 수를 지정합니다.

3. 업데이트된 매개 변수를 사용하여 변경 사항을 확인하는 마이그레이션 계획을 생성합니다.
마이그레이션 계획이 **MigrationController** CR 제한을 초과하는 경우 MTC 콘솔은 마이그레이션 계획을 저장할 때 경고 메시지를 표시합니다.

9.5.2. 직접 볼륨 마이그레이션의 영구 볼륨 크기 조정 활성화

대상 클러스터에서 디스크 공간이 부족하지 않도록 직접 볼륨 마이그레이션의 PV(영구 볼륨) 크기 조정을 활성화할 수 있습니다.

PV의 디스크 사용량이 구성된 수준에 도달하면 **MigrationController** CR(사용자 정의 리소스)은 PVC(영구 볼륨 클레임)의 요청된 스토리지 용량을 실제 프로비저닝된 용량과 비교합니다. 그런 다음 대상 클러스터에 필요한 공간을 계산합니다.

pv_resizing_threshold 매개 변수는 PV 크기 조정을 사용할 시기를 결정합니다. 기본 임계값은 **3%**입니다. 즉 PV의 디스크 사용량이 **97%**를 초과하면 PV 크기가 조정됩니다. PV 크기 조정이 디스크 사용량이 낮은 수준에서 발생하도록 이 임계값을 늘릴 수 있습니다.

PVC 용량은 다음 기준에 따라 계산됩니다.

- PVC의 요청된 스토리지 용량(**spec.resources.requests.storage**)이 실제 프로비저닝된 용량(**status.capacity.storage**)과 같지 않으면 더 큰 값이 사용됩니다.

- PV가 PVC를 통해 프로비저닝되고 나중에 PV 및 PVC 용량이 더 이상 일치하지 않도록 변경된 경우 더 큰 값이 사용됩니다.

사전 요구 사항

- **MigrationController** CR에서 명령을 실행할 수 있도록 PVC를 실행 중인 하나 이상의 pod에 연결해야 합니다.

절차

1. 호스트 클러스터에 로그인합니다.
2. **MigrationController** CR의 패치를 적용하여 PV 크기 조정을 활성화합니다.

```
$ oc patch migrationcontroller migration-controller -p '{"spec": {"enable_dvm_pv_resizing":true}}' \
--type='merge' -n openshift-migration
```

1 PV 크기 조정을 비활성화하려면 값을 **false**로 설정합니다.

3. 선택 사항: **pv_resizing_threshold** 매개변수를 업데이트하여 임계값을 늘립니다.

```
$ oc patch migrationcontroller migration-controller -p '{"spec":{"pv_resizing_threshold":41}}' \
--type='merge' -n openshift-migration
```

1 기본값은 **3**입니다.

임계값을 초과하면 **MigPlan** CR 상태에 다음 상태 정보 메시지가 표시됩니다.

```
status:
  conditions:
  ...
  - category: Warn
    durable: true
    lastTransitionTime: "2021-06-17T08:57:01Z"
    message: 'Capacity of the following volumes will be automatically adjusted to avoid disk
    capacity issues in the target cluster: [pvc-b800eb7b-cf3b-11eb-a3f7-0eae3e0555f3]'
    reason: Done
    status: "False"
    type: PvCapacityAdjustmentRequired
```



참고

AWS gp2 스토리지의 경우 gp2에서 볼륨 사용량과 크기를 계산하는 방식 때문에 **pv_resizing_threshold**가 42% 이상인 경우 이 메시지가 표시되지 않습니다. ([BZ#1973148](#))

9.5.3. 캐시된 Kubernetes 클라이언트 활성화

마이그레이션 중에 성능이 향상되도록 **MigrationController CR** (사용자 정의 리소스)에서 캐시된 Kubernetes 클라이언트를 활성화할 수 있습니다. 서로 다른 지역에 있는 클러스터 간에 마이그레이션하거나 네트워크 지연 시간이 큰 경우 가장 큰 성능 이점이 표시됩니다.



참고

예를 들어 직접 볼륨 마이그레이션에 대한 Rsync 백업이나 Velero 백업 및 복원과 같은 위임된 작업에서는 캐시된 클라이언트의 성능이 향상되지 않습니다.

MigrationController CR에서 **MigCluster CR**과 상호 작용하는 데 필요한 모든 API 리소스를 캐시하므로 캐시된 클라이언트에는 추가 메모리가 필요합니다. 일반적으로 API 서버로 전송되는 요청은 대신 캐시로 이동합니다. 캐시는 업데이트를 위해 API 서버를 모니터링합니다.

캐시된 클라이언트를 활성화한 후 **OOMKilled** 오류가 발생하면 **MigrationController CR**의 메모리 제한과 요청을 늘릴 수 있습니다.

절차

1. 다음 명령을 실행하여 캐시된 클라이언트를 활성화합니다.

```
$ oc -n openshift-migration patch migrationcontroller migration-controller --type=json --patch \
'[{ "op": "replace", "path": "/spec/mig_controller_enable_cache", "value": true}]'
```

2. 선택 사항: 다음 명령을 실행하여 **MigrationController CR** 메모리 제한을 늘립니다.

```
$ oc -n openshift-migration patch migrationcontroller migration-controller --type=json --patch \
'[{ "op": "replace", "path": "/spec/mig_controller_limits_memory", "value": <10Gi>}]'
```

3. 선택 사항: 다음 명령을 실행하여 **MigrationController CR** 메모리 요청을 늘립니다.

```
$ oc -n openshift-migration patch migrationcontroller migration-controller --type=json --patch \
'[{ "op": "replace", "path": "/spec/mig_controller_requests_memory", "value": <350Mi>}]'
```

10장. 문제 해결

이 섹션에서는 MTC(Migration Toolkit for Containers) 문제 해결을 위한 리소스에 대해 설명합니다.

알려진 문제는 [MTC 릴리스 노트](#)를 참조하십시오.

10.1. MTC 워크플로

MTC(Migration Toolkit for Containers) 웹 콘솔 또는 Kubernetes API를 사용하여 Kubernetes 리소스, 영구 볼륨 데이터 및 내부 컨테이너 이미지를 OpenShift Container Platform 4.9로 마이그레이션할 수 있습니다.

(MTC)는 다음 리소스를 마이그레이션합니다.

- 마이그레이션 계획에 지정된 네임스페이스입니다.
- 네임스페이스가 지정된 리소스: MTC가 네임스페이스를 마이그레이션하면 서비스 또는 포트와 같은 해당 네임스페이스와 연결된 모든 오브젝트와 리소스가 마이그레이션됩니다. 또한 네임스페이스에 존재하지만 클러스터 수준에 없는 리소스가 클러스터 수준에 존재하는 리소스에 따라 달라지는 경우 MTC가 두 개의 리소스 모두를 마이그레이션합니다.
예를 들어 SCC(보안 컨텍스트 제약 조건)는 클러스터 수준에 존재하는 리소스이며, 서비스 계정(SA)은 네임스페이스 수준에 존재하는 리소스입니다. MTC가 마이그레이션하는 네임스페이스에 SA가 있는 경우 MTC는 SA에 연결된 모든 SCC를 자동으로 찾고 해당 SCC도 마이그레이션합니다. 마찬가지로 MTC는 네임스페이스의 영구 볼륨 클레임에 연결된 영구 볼륨을 마이그레이션합니다.



참고

리소스에 따라 클러스터 범위 리소스를 수동으로 마이그레이션해야 할 수 있습니다.

- CR(사용자 정의 리소스) 및 CRD(사용자 정의 리소스 정의): MTC는 네임스페이스 수준에서 CR 및 CRD를 자동으로 마이그레이션합니다.

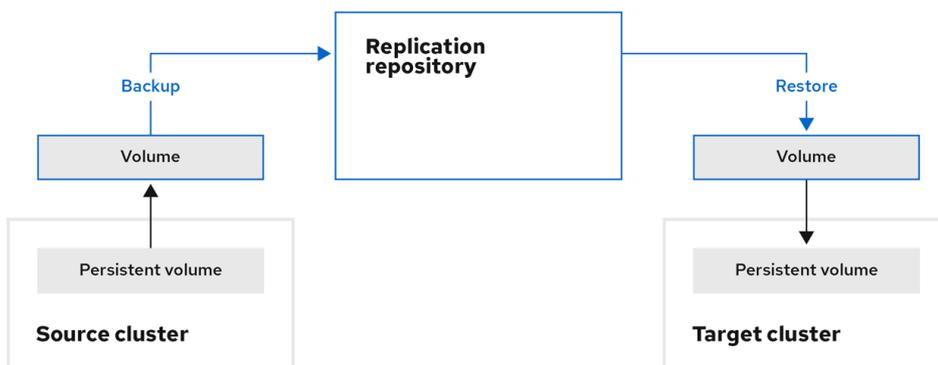
MTC 웹 콘솔을 사용하여 애플리케이션을 마이그레이션하는 데는 다음 단계가 포함됩니다.

1. 모든 클러스터에 Migration Toolkit for Containers Operator를 설치합니다.
인터넷 액세스가 제한되거나 없는 제한된 환경에서 Migration Toolkit for Containers Operator를 설치할 수 있습니다. 소스 및 대상 클러스터는 상호 액세스 권한 및 미러 레지스트리에 대한 네트워크 액세스 권한이 있어야 합니다.
2. MTC가 데이터를 마이그레이션하는 데 사용하는 중간 오브젝트 스토리지인 복제 리포지토리를 구성합니다.
소스 및 대상 클러스터는 마이그레이션 중에 복제 리포지토리에 대한 네트워크 액세스 권한이 있어야 합니다. 프록시 서버를 사용하는 경우 복제 리포지토리와 클러스터 간의 네트워크 트래픽을 허용하도록 해당 서버를 구성해야 합니다.
3. MTC 웹 콘솔에 소스 클러스터를 추가합니다.
4. MTC 웹 콘솔에 복제 리포지토리를 추가합니다.
5. 다음 데이터 마이그레이션 옵션 중 하나를 사용하여 마이그레이션 계획을 생성합니다.
 - **복사:** MTC는 소스 클러스터에서 복제 리포지토리로, 복제 리포지토리에서 대상 클러스터로 데이터를 복사합니다.



참고

직접 이미지 마이그레이션 또는 직접 볼륨 마이그레이션을 사용하는 경우 소스 클러스터에서 대상 클러스터로 이미지 또는 볼륨이 직접 복사됩니다.



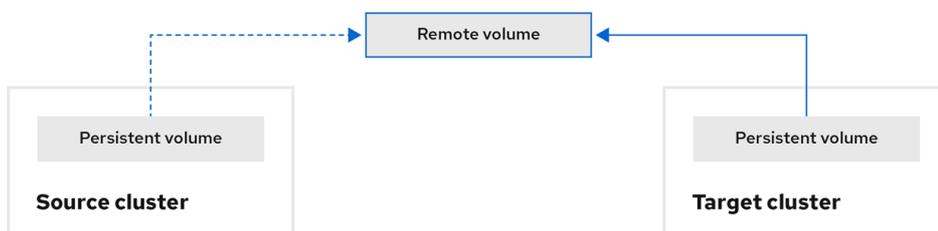
OpenShift_45_1019

- **이동:** MTC는 소스 클러스터에서 원격 볼륨(예: NFS)을 마운트 해제하고 원격 볼륨을 가리키는 대상 클러스터에 PV 리소스를 만든 다음 대상 클러스터에 원격 볼륨을 마운트합니다. 대상 클러스터에서 실행되는 애플리케이션은 소스 클러스터와 동일한 원격 볼륨을 사용합니다. 소스 및 대상 클러스터가 원격 볼륨에 액세스할 수 있어야 합니다.



참고

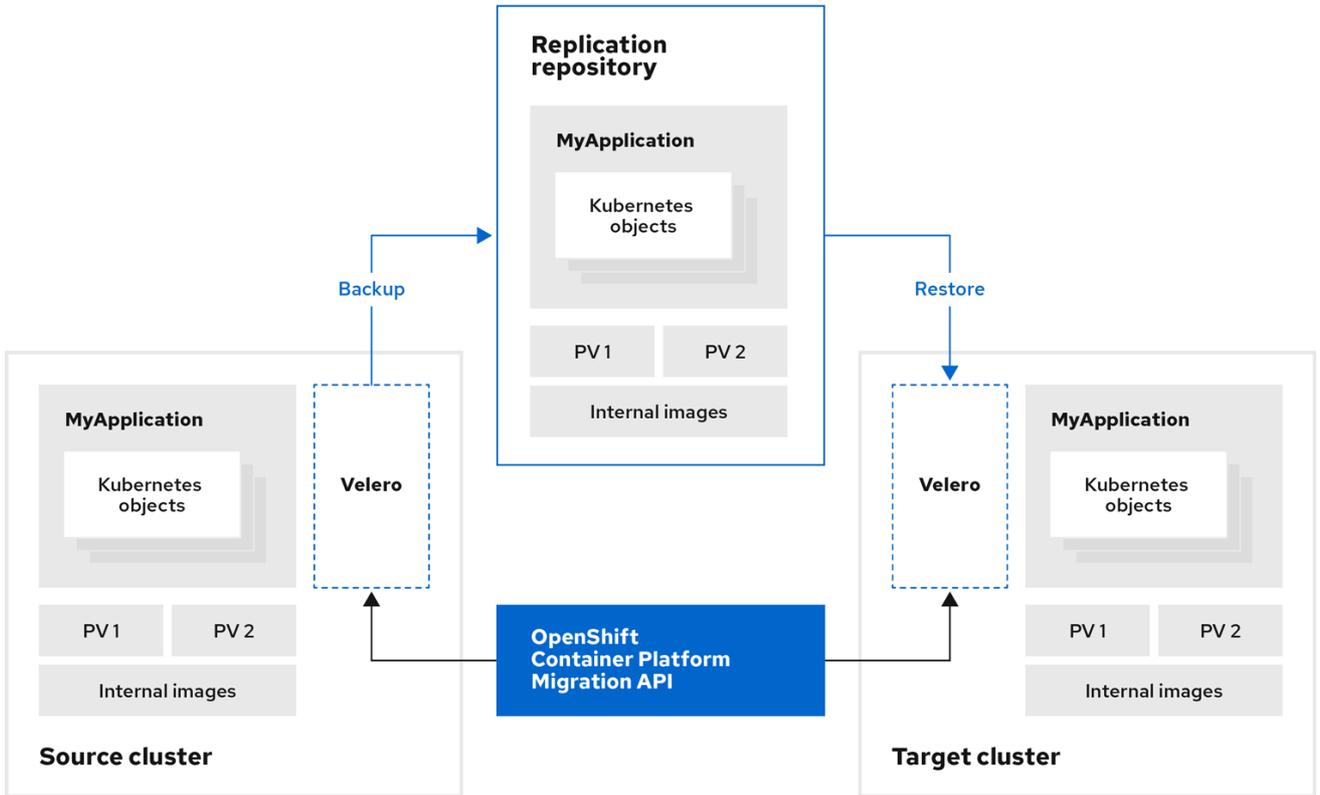
이 다이어그램에는 복제 리포지토리가 나타나지 않지만 마이그레이션에는 필요합니다.



OpenShift_45_1019

6. 다음 옵션 중 하나를 사용하여 마이그레이션 계획을 실행합니다.

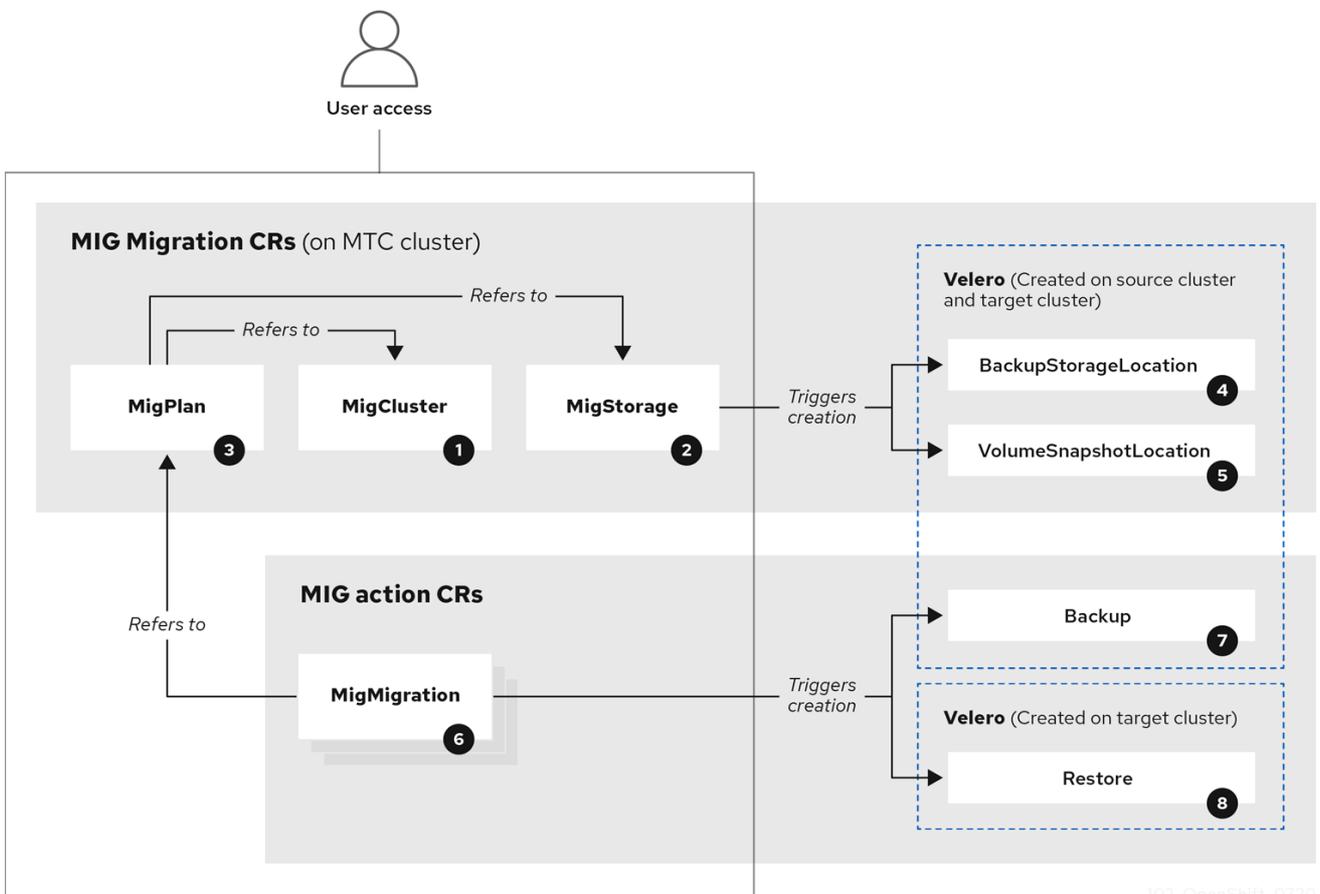
- **스테이지 (Stage)**는 애플리케이션을 중지하지 않고 데이터를 대상 클러스터에 복사합니다. 스테이지 마이그레이션을 마이그레이션 전에 대부분의 데이터가 대상에 복사되도록 여러 번 실행할 수 있습니다. 하나 이상의 단계적 마이그레이션을 실행하면 컷오버 마이그레이션 기간이 단축됩니다.
- **컷오버 (Cutover)**는 소스 클러스터에서 애플리케이션을 중지하고 리소스를 대상 클러스터로 이동합니다.
선택 사항: 마이그레이션 중에 소스 클러스터에서 트랜잭션 중지 확인란의 선택을 취소할 수 있습니다.



OpenShift_45_1019

MTC 사용자 정의 리소스 정보

MTC(Migration Toolkit for Containers)는 다음과 같은 사용자 정의 리소스(CR)를 생성합니다.



102_OpenShift_0720

- 1 **MigCluster** (구성, MTC 클러스터): 클러스터 정의
- 2 **MigStorage** (구성, MTC 클러스터): 스토리지 정의
- 3 **MigPlan** (구성, MTC 클러스터): 마이그레이션 계획

MigPlan CR은 마이그레이션 중인 소스 및 대상 클러스터, 복제 리포지토리 및 네임스페이스를 설명합니다. 0, 1 또는 많은 **MigMigration** CR과 연관됩니다.



참고

MigPlan CR을 삭제하면 연결된 **MigMigration** CR이 삭제됩니다.

- 4 **BackupStorageLocation** (구성, MTC 클러스터): **Velero** 백업 오브젝트의 위치
- 5 **VolumeSnapshotLocation** (구성, MTC 클러스터): **Velero** 볼륨 스냅샷의 위치
- 6 **MigMigration** (작업, MTC 클러스터): 데이터를 준비하거나 마이그레이션할 때마다 마이그레이션이 생성됩니다. 각 **MigMigration** CR은 **MigPlan** CR과 연결되어 있습니다.
- 7 **백업** (작업, 소스 클러스터): 마이그레이션 계획을 실행할 때 **MigMigration** CR은 각 소스 클러스터에 두 개의 **Velero** 백업 CR을 생성합니다.
 - Kubernetes 오브젝트의 백업 CR #1
 - PV 데이터용 백업 CR #2
- 8 **복원** (작업, 대상 클러스터): 마이그레이션 계획을 실행할 때 **MigMigration** CR은 대상 클러스터에 두 개의 **Velero** 복원 CR을 생성합니다.
 - PV 데이터에 대한 CR #1 복원(백업 CR #2 사용)
 - Kubernetes 오브젝트에 대한 CR #2 복원(백업 CR #1 사용)

10.2. MTC 사용자 정의 리소스 매니페스트

MTC(Migration Toolkit for Containers)는 다음 사용자 정의 리소스(CR) 매니페스트를 사용하여 애플리케이션을 마이그레이션합니다.

10.2.1. DirectImageMigration

DirectImageMigration CR은 소스 클러스터에서 대상 클러스터로 이미지를 직접 복사합니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: DirectImageMigration
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: <direct_image_migration>
spec:
```

```

srcMigClusterRef:
  name: <source_cluster>
  namespace: openshift-migration
destMigClusterRef:
  name: <destination_cluster>
  namespace: openshift-migration
namespaces: ❶
  - <source_namespace_1>
  - <source_namespace_2>:<destination_namespace_3> ❷

```

❶ 마이그레이션할 이미지를 포함하는 하나 이상의 네임스페이스입니다. 기본적으로 대상 네임스페이스의 이름은 소스 네임스페이스와 동일합니다.

❷ 다른 이름으로 대상 네임스페이스의 소스 네임스페이스에 매핑합니다.

10.2.2. DirectImageStreamMigration

DirectImageStreamMigration CR은 소스 클러스터에서 대상 클러스터로 직접 이미지 스트림 참조를 복사합니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: DirectImageStreamMigration
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: <direct_image_stream_migration>
spec:
  srcMigClusterRef:
    name: <source_cluster>
    namespace: openshift-migration
  destMigClusterRef:
    name: <destination_cluster>
    namespace: openshift-migration
  imageStreamRef:
    name: <image_stream>
    namespace: <source_image_stream_namespace>
  destNamespace: <destination_image_stream_namespace>

```

10.2.3. DirectVolumeMigration

DirectVolumeMigration CR은 소스 클러스터에서 대상 클러스터로 직접 PV(영구 볼륨)를 복사합니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: DirectVolumeMigration
metadata:
  name: <direct_volume_migration>
  namespace: openshift-migration
spec:
  createDestinationNamespaces: false ❶
  deleteProgressReportingCRs: false ❷
  destMigClusterRef:
    name: <host_cluster> ❸
    namespace: openshift-migration

```

```

persistentVolumeClaims:
- name: <pvc> 4
  namespace: <pvc_namespace>
srcMigClusterRef:
  name: <source_cluster>
  namespace: openshift-migration

```

- 1 대상 클러스터의 PV에 네임스페이스를 만들려면 **true**로 설정합니다.
- 2 마이그레이션 후 **DirectVolumeMigrationProgress** CR을 삭제하려면 **true**로 설정합니다. 문제 해결을 위해 **DirectVolumeMigrationProgress** CR이 유지되기 위한 기본값은 **false**입니다.
- 3 대상 클러스터가 호스트 클러스터가 아닌 경우 클러스터 이름을 업데이트합니다.
- 4 마이그레이션할 하나 이상의 PVC를 지정합니다.

10.2.4. DirectVolumeMigrationProgress

DirectVolumeMigrationProgress CR은 **DirectVolumeMigration** CR의 진행 상황을 보여줍니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: DirectVolumeMigrationProgress
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: <direct_volume_migration_progress>
spec:
  clusterRef:
    name: <source_cluster>
    namespace: openshift-migration
  podRef:
    name: <rsync_pod>
    namespace: openshift-migration

```

10.2.5. MigAnalytic

MigAnalytic CR은 이미지 수, Kubernetes 리소스 및 관련 **MigPlan** CR에서 영구 볼륨 (PV) 용량을 수집합니다.

수집하는 데이터를 구성할 수 있습니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: MigAnalytic
metadata:
  annotations:
    migplan: <migplan>
  name: <miganalytic>
  namespace: openshift-migration
  labels:
    migplan: <migplan>
spec:
  analyzeImageCount: true 1
  analyzeK8SResources: true 2

```

```
analyzePVCcapacity: true 3
listImages: false 4
listImagesLimit: 50 5
migPlanRef:
  name: <migplan>
  namespace: openshift-migration
```

- 1 선택 사항: 이미지 수를 반환합니다.
- 2 선택 사항: Kubernetes 리소스의 수, 종류 및 API 버전을 반환합니다.
- 3 선택 사항: PV 용량을 반환합니다.
- 4 이미지 이름 목록을 반환합니다. 기본값은 **false**이므로 출력이 과도하게 길지 않습니다.
- 5 선택 사항: **listImages**가 **true**인 경우 최대 이미지 이름 수를 지정합니다.

10.2.6. MigCluster

MigCluster CR은 호스트, 로컬 또는 원격 클러스터를 정의합니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigCluster
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: <host_cluster> 1
  namespace: openshift-migration
spec:
  isHostCluster: true 2
  # The 'azureResourceGroup' parameter is relevant only for Microsoft Azure.
  azureResourceGroup: <azure_resource_group> 3
  caBundle: <ca_bundle_base64> 4
  insecure: false 5
  refresh: false 6
  # The 'restartRestic' parameter is relevant for a source cluster.
  restartRestic: true 7
  # The following parameters are relevant for a remote cluster.
  exposedRegistryPath: <registry_route> 8
  url: <destination_cluster_url> 9
  serviceAccountSecretRef:
    name: <source_secret> 10
    namespace: openshift-config
```

- 1 **migration-controller** pod가 이 클러스터에서 실행되지 않는 경우 클러스터 이름을 업데이트합니다.
- 2 **migration-controller** pod는 **true**인 경우 이 클러스터에서 실행됩니다.
- 3 Microsoft Azure만 해당: 리소스 그룹을 지정합니다.
- 4 선택 사항: 자체 서명된 CA 인증서에 대한 인증서 번들을 생성하고 **insecure** 매개변수 값이 **false**인 경우 base64 인코딩 인증서 번들을 지정합니다.

- 5 SSL 확인을 비활성화하려면 **true**로 설정합니다.
- 6 클러스터를 확인하려면 **true**로 설정합니다.
- 7 **Stage** pod가 생성된 후 소스 클러스터에서 **Restic** pod를 다시 시작하려면 **true**로 설정합니다.
- 8 원격 클러스터 및 직접 이미지 마이그레이션만 해당: 공용 보안 레지스트리 경로를 지정합니다.
- 9 원격 클러스터만 해당: URL을 지정합니다.
- 10 원격 클러스터만 해당: **Secret** 오브젝트의 이름을 지정합니다.

10.2.7. MigHook

MigHook CR은 마이그레이션의 지정된 단계에서 사용자 정의 코드를 실행하는 마이그레이션 후크를 정의합니다. 최대 4개의 마이그레이션 후크를 생성할 수 있습니다. 각 후크는 마이그레이션의 다른 단계에서 실행됩니다.

후크 이름, 런타임 기간, 사용자 정의 이미지 및 후크를 실행할 클러스터를 구성할 수 있습니다.

후크의 마이그레이션 단계 및 네임스페이스는 **MigPlan** CR에 구성됩니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigHook
metadata:
  generateName: <hook_name_prefix> 1
  name: <mighook> 2
  namespace: openshift-migration
spec:
  activeDeadlineSeconds: 1800 3
  custom: false 4
  image: <hook_image> 5
  playbook: <ansible_playbook_base64> 6
  targetCluster: source 7
```

- 1 선택 사항: 각 마이그레이션 후크마다 고유한 이름이 있도록 이 매개변수의 값에 고유한 해시가 추가됩니다. **name** 매개변수의 값을 지정할 필요가 없습니다.
- 2 **generateName** 매개변수 값을 지정하지 않으면 마이그레이션 후크 이름을 지정합니다.
- 3 선택 사항: 후크를 실행할 수 있는 최대 초를 지정합니다. 기본값은 **1800**입니다.
- 4 **true**인 경우 후크는 사용자 정의 이미지입니다. 사용자 지정 이미지는 Ansible을 포함하거나 다른 프로그래밍 언어로 작성할 수 있습니다.
- 5 사용자 지정 이미지를 지정합니다(예: **quay.io/konveyor/hook-runner:latest**). **custom**가 **true**인 경우 필수 항목입니다.
- 6 base64로 인코딩된 Ansible 플레이북입니다. **custom**가 **false**인 경우 필수 항목입니다.
- 7 후크를 실행할 클러스터를 지정합니다. 유효한 값은 **source** 또는 **destination**입니다.

10.2.8. MigMigration

MigMigration CR은 **MigPlan** CR을 실행합니다.

단계 또는 증분 마이그레이션을 실행하거나 진행 중인 마이그레이션을 취소하거나 완료된 마이그레이션을 롤백하도록 **MigMigration** CR을 구성할 수 있습니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigMigration
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: <migmigration>
  namespace: openshift-migration
spec:
  canceled: false 1
  rollback: false 2
  stage: false 3
  quiescePods: true 4
  keepAnnotations: true 5
  verify: false 6
  migPlanRef:
    name: <migplan>
    namespace: openshift-migration
```

- 1 진행 중인 마이그레이션을 취소하려면 **true**로 설정합니다.
- 2 완료된 마이그레이션을 롤백하려면 **true**로 설정합니다.
- 3 단계적 마이그레이션을 실행하려면 **true**로 설정합니다. 데이터가 증분적으로 복사되고 소스 클러스터의 pod가 중지되지 않습니다.
- 4 마이그레이션 중에 애플리케이션을 중지하려면 **true**로 설정합니다. **backup** 단계 후에 소스 클러스터의 pod는 **0**으로 조정됩니다.
- 5 마이그레이션 프로세스 중에 적용되는 레이블 및 주석을 유지하려면 **true**로 설정합니다.
- 6 대상 클러스터에서 마이그레이션된 pod의 상태를 확인하고 **Running** 상태가 아닌 pod의 이름을 반환하려면 **true**로 설정합니다.

10.2.9. MigPlan

MigPlan CR은 마이그레이션 계획의 매개변수를 정의합니다.

대상 네임스페이스, 후크 단계, 직접 또는 간접 마이그레이션을 구성할 수 있습니다.



참고

기본적으로 대상 네임스페이스의 이름은 소스 네임스페이스와 동일합니다. 다른 대상 네임스페이스를 구성하는 경우 마이그레이션 중에 UID 및 GID 범위가 복사되므로 소스 또는 대상 클러스터에서 네임스페이스가 복제되지 않도록 해야 합니다.

```
apiVersion: migration.openshift.io/v1alpha1
kind: MigPlan
metadata:
```

```

labels:
  controller-tools.k8s.io: "1.0"
name: <migplan>
namespace: openshift-migration
spec:
  closed: false ❶
  srcMigClusterRef:
    name: <source_cluster>
    namespace: openshift-migration
  destMigClusterRef:
    name: <destination_cluster>
    namespace: openshift-migration
  hooks: ❷
    - executionNamespace: <namespace> ❸
      phase: <migration_phase> ❹
      reference:
        name: <hook> ❺
        namespace: <hook_namespace> ❻
        serviceAccount: <service_account> ❼
  indirectImageMigration: true ❽
  indirectVolumeMigration: false ❾
  migStorageRef:
    name: <migstorage>
    namespace: openshift-migration
  namespaces:
    - <source_namespace_1> ❿
    - <source_namespace_2>
    - <source_namespace_3>:<destination_namespace_4> ⓫
  refresh: false ⓬

```

- ❶ true인 경우 마이그레이션이 완료되었습니다. 이 **MigPlan** CR에 대해 다른 **MigMigration** CR을 생성할 수 없습니다.
- ❷ 선택 사항: 최대 4개의 마이그레이션 후크를 지정할 수 있습니다. 각 후크는 다른 마이그레이션 단계에서 실행되어야 합니다.
- ❸ 선택 사항: 후크를 실행할 네임스페이스를 지정합니다.
- ❹ 선택 사항: 후크가 실행되는 마이그레이션 단계를 지정합니다. 하나의 후크를 하나의 단계에 할당할 수 있습니다. 유효한 값은 **PreBackup**, **PostBackup**, **PreRestore** 및 **PostRestore**입니다.
- ❺ 선택 사항: **MigHook** CR의 이름을 지정합니다.
- ❻ 선택 사항: **MigHook** CR의 네임스페이스를 지정합니다.
- ❼ 선택 사항: **cluster-admin** 권한이 있는 서비스 계정을 지정합니다.
- ❽ true인 경우 직접 이미지 마이그레이션이 비활성화됩니다. 이미지는 소스 클러스터에서 복제 리포지토리로, 복제 리포지토리에서 대상 클러스터로 복사됩니다.
- ❾ true인 경우 직접 볼륨 마이그레이션이 비활성화됩니다. PV는 소스 클러스터에서 복제 리포지토리로, 복제 리포지토리에서 대상 클러스터로 복사됩니다.
- ❿ 하나 이상의 소스 네임스페이스를 지정합니다. 소스 네임스페이스만 지정하는 경우 대상 네임스페이스는 동일합니다.

11 대상 네임스페이스가 소스 네임스페이스와 다른 경우 지정합니다.

12 MigPlan CR이 **true**인 경우 검증됩니다.

10.2.10. MigStorage

MigStorage CR은 복제 리포지토리의 오브젝트 스토리지를 설명합니다.

Amazon Web Services(AWS), Microsoft Azure, Google Cloud Storage, Multi-Cloud Object Gateway 및 일반 S3 호환 클라우드 스토리지를 지원합니다.

AWS 및 스냅샷 복사 방법에는 추가 매개 변수가 있습니다.

```

apiVersion: migration.openshift.io/v1alpha1
kind: MigStorage
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: <migstorage>
  namespace: openshift-migration
spec:
  backupStorageProvider: <backup_storage_provider> 1
  volumeSnapshotProvider: <snapshot_storage_provider> 2
  backupStorageConfig:
    awsBucketName: <bucket> 3
    awsRegion: <region> 4
    credsSecretRef:
      namespace: openshift-config
      name: <storage_secret> 5
    awsKmsKeyId: <key_id> 6
    awsPublicUrl: <public_url> 7
    awsSignatureVersion: <signature_version> 8
  volumeSnapshotConfig:
    awsRegion: <region> 9
    credsSecretRef:
      namespace: openshift-config
      name: <storage_secret> 10
  refresh: false 11

```

1 스토리지 공급자를 지정합니다.

2 스냅샷 복사 방법만: 스토리지 공급자를 지정합니다.

3 AWS만 해당: 버킷 이름을 지정합니다.

4 AWS만 해당: 버킷 리전을 지정합니다(예: **us-east-1**).

5 스토리지를 위해 생성한 **Secret** 오브젝트의 이름을 지정합니다.

6 AWS만 해당: AWS 키 관리 서비스를 사용하는 경우 키의 고유 식별자를 지정합니다.

7 AWS만 해당: AWS 버킷에 대한 공용 액세스 권한이 부여된 경우 버킷 URL을 지정합니다.

- 8 AWS만 해당: 버킷에 대한 요청을 인증하기 위해 AWS 서명 버전을 지정합니다(예: 4).
- 9 스냅샷 복사 방법만 해당: 클러스터의 지역 리전을 지정합니다.
- 10 스냅샷 복사 방법만 해당: 스토리지를 위해 생성한 **Secret** 오브젝트의 이름을 지정합니다.
- 11 클러스터를 확인하려면 **true**로 설정합니다.

10.3. 로그 및 디버깅 툴

이 섹션에서는 문제 해결에 사용할 수 있는 로그 및 디버깅 툴에 대해 설명합니다.

10.3.1. 마이그레이션 계획 리소스 보기

MTC 웹 콘솔 및 CLI(명령줄 인터페이스)를 사용하여 마이그레이션 계획 리소스를 확인하여 실행 중인 마이그레이션을 모니터링하거나 실패한 마이그레이션 문제를 해결할 수 있습니다.

절차

1. MTC 웹 콘솔에서 **마이그레이션 계획**을 클릭합니다.
2. 마이그레이션 계획 옆에 있는 **마이그레이션** 번호를 클릭하면 **마이그레이션** 페이지가 표시됩니다.
3. 마이그레이션을 클릭하여 **마이그레이션 세부 정보**를 확인합니다.
4. **마이그레이션 리소스**를 확장하여 목록 보기에서 마이그레이션 리소스 및 해당 상태를 봅니다.



참고

실패한 마이그레이션 문제를 해결하려면 실패한 상위 수준 리소스부터 시작한 다음 리소스 트리를 하위 수준 리소스로 이동합니다.

5. 리소스 옆에 있는 옵션 메뉴  를 클릭하고 다음 옵션 중 하나를 선택합니다.
 - **oc describe** 복사 명령은 명령을 클립보드에 복사합니다.
 - 관련 클러스터에 로그인한 다음 명령을 실행합니다.
리소스의 조건 및 이벤트는 YAML 형식으로 표시됩니다.
 - **oc logs** 복사 명령은 명령을 클립보드에 복사합니다.
 - 관련 클러스터에 로그인한 다음 명령을 실행합니다.
리소스가 로그 필터링을 지원하는 경우 필터링된 로그가 표시됩니다.
 - **JSON 보기**는 웹 브라우저에서 JSON 형식으로 리소스 데이터를 표시합니다.
데이터는 **oc get <resource>** 명령의 출력과 동일합니다.

10.3.2. 마이그레이션 계획 로그 보기

마이그레이션 계획에 대한 집계된 로그를 볼 수 있습니다. MTC 웹 콘솔을 사용하여 클립보드에 명령을 복사한 다음 CLI(명령줄 인터페이스)에서 명령을 실행합니다.

명령은 다음 pod의 필터링된 로그를 표시합니다.

- **Migration Controller**
- **Velero**
- **Restic**
- **Rsync**
- **Stunnel**
- **Registry**

절차

1. MTC 웹 콘솔에서 **마이그레이션 계획**을 클릭합니다.
2. 마이그레이션 계획 옆에 있는 **마이그레이션** 번호를 클릭합니다.
3. **로그 보기**를 클릭합니다.
4. 복사 아이콘을 클릭하여 **oc logs** 명령을 클립보드에 복사합니다.
5. 해당 클러스터에 로그인하고 CLI에 명령을 입력합니다.
마이그레이션 계획에 대해 집계된 로그가 표시됩니다.

10.3.3. 마이그레이션 로그 리더 사용

마이그레이션 로그 리더를 사용하여 모든 마이그레이션 로그에 대한 필터링된 보기를 표시할 수 있습니다.

프로세스

1. **mig-log-reader** 포드를 가져옵니다.

```
$ oc -n openshift-migration get pods | grep log
```

2. 단일 마이그레이션 로그를 표시하려면 다음 명령을 입력합니다.

```
$ oc -n openshift-migration logs -f <mig-log-reader-pod> -c color 1
```

1 **-c plain** 옵션은 색상 없이 로그를 표시합니다.

10.3.4. 성능 지표 액세스

MigrationController CR(사용자 정의 리소스)은 지표를 기록하고 클러스터 내부 모니터링 스토리지로 가져옵니다. PromQL(Prometheus Query Language)을 사용하여 마이그레이션 성능 문제를 진단하여 지표를 쿼리할 수 있습니다. Migration Controller Pod가 다시 시작되면 모든 메트릭이 재설정됩니다.

OpenShift Container Platform 웹 콘솔을 사용하여 성능 지표에 액세스하고 쿼리를 실행할 수 있습니다.

프로세스

1. OpenShift Container Platform 웹 콘솔에서 **모니터링** → **메트릭**을 클릭합니다.
2. PromQL 쿼리를 입력하고 표시할 시간 창을 선택한 다음 **Run Queries(쿼리 실행)**를 클릭합니다. 웹 브라우저에 모든 결과가 표시되지 않으면 Prometheus 콘솔을 사용합니다.

10.3.4.1. 제공된 지표

MigrationController CR(사용자 정의 리소스)은 **MigMigration** CR 수 및 해당 API 요청에 대한 지표를 제공합니다.

10.3.4.1.1. cam_app_workload_migrations

이 메트릭은 시간 경과에 따른 **MigMigration** CR 수입입니다. 마이그레이션 상태 변경과 함께 API 요청 정보를 수집하기 위해 the **mtc_client_request_count** 및 **mtc_client_request_elapsed** 지표를 확인하는 데 유용합니다. 이 지표는 Telemetry에 포함되어 있습니다.

표 10.1. cam_app_workload_migrations metric

쿼리 가능한 라벨 이름	라벨 값 샘플	레이블 설명
status	실행 중,유휴 상태,실패했습니다.완료	MigMigration CR의 상태
type	단계, 최종	MigMigration CR의 유형

10.3.4.1.2. mtc_client_request_count

이 지표는 **MigrationController** 가 실행한 Kubernetes API 요청의 누적 수입입니다. Telemetry에는 포함되어 있지 않습니다.

표 10.2. mtc_client_request_count 메트릭

쿼리 가능한 라벨 이름	라벨 값 샘플	레이블 설명
cluster	https://migcluster-url:443	요청이 발행된 클러스터
component	MigPlan, MigCluster	요청을 발급한 하위 컨트롤러 API
함수	(*ReconcileMigPlan).Reconcile	요청이 발행된 기능
kind	SecretList,패포	요청이 발행된 Kubernetes 종류

10.3.4.1.3. mtc_client_request_elapsed

이 지표는 **MigrationController** 가 발행한 Kubernetes API 요청의 누적 대기 시간(밀리초)입니다. Telemetry에는 포함되어 있지 않습니다.

표 10.3. mtc_client_request_elapsed 메트릭

쿼리 가능한 라벨 이름	라벨 값 샘플	레이블 설명
cluster	https://cluster-url.com:443	요청이 발행된 클러스터
component	migplan, migcluster	요청을 발급한 하위 컨트롤러 API
함수	(*ReconcileMigPlan).Reconcile	요청이 발행된 기능
kind	SecretList, 배포	요청이 발행된 Kubernetes 리소스

10.3.4.1.4. 유용한 쿼리

테이블에는 성능 모니터링에 사용할 수 있는 몇 가지 유용한 쿼리가 나열되어 있습니다.

표 10.4. 유용한 쿼리

쿼리	설명
mtc_client_request_count	발급된 API 요청 수, 요청 유형별로 정렬
sum(mtc_client_request_count)	발행되는 총 API 요청 수
mtc_client_request_elapsed	API 요청 대기 시간, 요청 유형별로 정렬
sum(mtc_client_request_elapsed)	API 요청에 대한 총 대기 시간
sum(mtc_client_request_elapsed) / sum(mtc_client_request_count)	평균 API 요청 대기 시간
mtc_client_request_elapsed / mtc_client_request_count	요청 유형별로 정렬된 API 요청의 평균 대기 시간
cam_app_workload_migrations{status="running"} * 100	요청 수와 함께 더 쉽게 볼 수 있도록 실행 중인 마이그레이션 수, 100을 곱한 값

10.3.5. must-gather 툴 사용

must-gather 툴을 사용하여 MTC 사용자 정의 리소스에 대한 로그, 메트릭 및 정보를 수집할 수 있습니다.

must-gather 데이터는 모든 고객 사례에 첨부되어야 합니다.

1시간 또는 24시간 동안 데이터를 수집하고 Prometheus 콘솔을 사용하여 데이터를 볼 수 있습니다.

사전 요구 사항

- **cluster-admin** 역할의 사용자로 OpenShift Container Platform 클러스터에 로그인해야 합니다.
- OpenShift CLI(**oc**)가 설치되어 있어야 합니다.

절차

1. **must-gather** 데이터를 저장하려는 디렉터리로 이동합니다.
2. 다음 데이터 수집 옵션 중 하나에 대해 **oc adm must-gather** 명령을 실행합니다.

- 지난 시간 동안 데이터를 수집하려면 다음을 수행하십시오.

```
$ oc adm must-gather --image=registry.redhat.io/rhmtc/openshift-migration-must-gather-rhel8:v1.7
```

데이터는 **must-gather/must-gather.tar.gz** 로 저장됩니다. [Red Hat 고객 포털](#) 에서 해당 지원 사례에 이 파일을 업로드할 수 있습니다.

- 지난 24 시간 동안 데이터를 수집하려면 다음을 수행하십시오.

```
$ oc adm must-gather --image=registry.redhat.io/rhmtc/openshift-migration-must-gather-rhel8:v1.7 \
  -- /usr/bin/gather_metrics_dump
```

이 작업에는 오랜 시간이 걸릴 수 있습니다. 데이터는 **must-gather/metrics/prom_data.tar.gz** 로 저장됩니다.

Prometheus 콘솔을 사용하여 메트릭 데이터 보기

Prometheus 콘솔을 사용하여 지표 데이터를 볼 수 있습니다.

절차

1. **prom_data.tar.gz** 파일의 압축을 풉니다.

```
$ tar -xvzf must-gather/metrics/prom_data.tar.gz
```

2. 로컬 Prometheus 인스턴스를 생성합니다.

```
$ make prometheus-run
```

이 명령은 Prometheus URL을 출력합니다.

출력 결과

```
Started Prometheus on http://localhost:9090
```

3. 웹 브라우저를 시작하고 URL로 이동하여 Prometheus 웹 콘솔을 사용하여 데이터를 확인합니다.
4. 데이터를 보고 나면 Prometheus 인스턴스 및 데이터를 삭제합니다.

```
$ make prometheus-cleanup
```

10.3.6. Velero CLI 툴을 사용하여 Velero 리소스 디버깅

Backup 및 **Restore** CR(사용자 정의 리소스)을 디버그하고 Velero CLI 툴을 사용하여 로그를 검색할 수 있습니다.

Velero CLI 툴은 OpenShift CLI 툴보다 자세한 정보를 제공합니다.

구문

oc exec 명령을 사용하여 Velero CLI 명령을 실행합니다.

```
$ oc -n openshift-migration exec deployment/velero -c velero -- ./velero \
  <backup_restore_cr> <command> <cr_name>
```

예

```
$ oc -n openshift-migration exec deployment/velero -c velero -- ./velero \
  backup describe 0e44ae00-5dc3-11eb-9ca8-df7e5254778b-2d8ql
```

도움말 옵션

velero --help 옵션을 사용하여 모든 Velero CLI 명령을 나열합니다.

```
$ oc -n openshift-migration exec deployment/velero -c velero -- ./velero \
  --help
```

Describe 명령

velero describe 명령을 사용하여 **Backup** 또는 **Restore** CR과 관련된 경고 및 오류 요약을 검색합니다.

```
$ oc -n openshift-migration exec deployment/velero -c velero -- ./velero \
  <backup_restore_cr> describe <cr_name>
```

예

```
$ oc -n openshift-migration exec deployment/velero -c velero -- ./velero \
  backup describe 0e44ae00-5dc3-11eb-9ca8-df7e5254778b-2d8ql
```

Logs 명령

velero logs 명령을 사용하여 **Backup** 또는 **Restore** CR의 로그를 검색합니다.

```
$ oc -n openshift-migration exec deployment/velero -c velero -- ./velero \
  <backup_restore_cr> logs <cr_name>
```

예

```
$ oc -n openshift-migration exec deployment/velero -c velero -- ./velero \
  restore logs ccc7c2d0-6017-11eb-afab-85d0007f5a19-x4lbf
```

10.3.7. 부분적인 마이그레이션 실패 디버깅

Velero CLI를 사용하여 **Restore** CR(사용자 정의 리소스) 로그를 검사하여 부분적인 마이그레이션 실패 경고 메시지를 디버깅할 수 있습니다.

부분적인 오류는 Velero가 마이그레이션에 실패하지 않는 문제가 발생하면 발생합니다. 예를 들어 CRD(사용자 정의 리소스 정의)가 누락되거나 소스 및 대상 클러스터에서 CRD 버전 간에 불일치가 있는 경우 마이그레이션이 완료되지만 CR은 대상 클러스터에서 생성되지 않습니다.

Velero 를 부분적인 오류로 기록한 다음 **Backup** CR에서 나머지 오브젝트를 처리합니다.

절차

1. **MigMigration** CR의 상태를 확인합니다.

```
$ oc get migmigration <migmigration> -o yaml
```

출력 예

```
status:
  conditions:
  - category: Warn
    durable: true
    lastTransitionTime: "2021-01-26T20:48:40Z"
    message: 'Final Restore openshift-migration/ccc7c2d0-6017-11eb-afab-85d0007f5a19-x4lbf: partially failed on destination cluster'
    status: "True"
    type: VeleroFinalRestorePartiallyFailed
  - category: Advisory
    durable: true
    lastTransitionTime: "2021-01-26T20:48:42Z"
    message: The migration has completed with warnings, please look at `Warn` conditions.
    reason: Completed
    status: "True"
    type: SucceededWithWarnings
```

2. Velero **describe** 명령을 사용하여 **Restore** CR의 상태를 확인합니다.

```
$ oc -n {namespace} exec deployment/velero -c velero -- ./velero \
restore describe <restore>
```

출력 예

```
Phase: PartiallyFailed (run 'velero restore logs ccc7c2d0-6017-11eb-afab-85d0007f5a19-x4lbf' for more information)

Errors:
  Velero: <none>
  Cluster: <none>
  Namespaces:
    migration-example: error restoring example.com/migration-example/migration-example:
the server could not find the requested resource
```

3. Velero **logs** 명령을 사용하여 **Restore** CR 로그를 확인합니다.

```
$ oc -n {namespace} exec deployment/velero -c velero -- ./velero \
restore logs <restore>
```

출력 예

```
time="2021-01-26T20:48:37Z" level=info msg="Attempting to restore migration-example:
migration-example" logSource="pkg/restore/restore.go:1107" restore=openshift-
migration/ccc7c2d0-6017-11eb-afab-85d0007f5a19-x4lbf
time="2021-01-26T20:48:37Z" level=info msg="error restoring migration-example: the server
could not find the requested resource" logSource="pkg/restore/restore.go:1170"
restore=openshift-migration/ccc7c2d0-6017-11eb-afab-85d0007f5a19-x4lbf
```

Restore CR 로그 오류 메시지인 **the server could not find the requested resource**은 부분적으로 실패한 마이그레이션의 원인을 나타냅니다.

10.3.8. 문제 해결을 위해 MTC 사용자 정의 리소스 사용

다음 MTC(Migration Toolkit for Containers) 사용자 정의 리소스(CR)를 확인하여 마이그레이션 실패 문제를 해결할 수 있습니다.

- **MigCluster**
- **MigStorage**
- **MigPlan**
- **BackupStorageLocation**
BackupStorageLocation CR에는 CR을 생성한 MTC 인스턴스를 식별하는 **migrationcontroller** 레이블이 포함되어 있습니다.

```
labels:
  migrationcontroller: ebe13bee-c803-47d0-a9e9-83f380328b93
```

- **VolumeSnapshotLocation**
VolumeSnapshotLocation CR에는 CR을 생성한 MTC 인스턴스를 식별하는 **migrationcontroller** 레이블이 포함되어 있습니다.

```
labels:
  migrationcontroller: ebe13bee-c803-47d0-a9e9-83f380328b93
```

- **MigMigration**
- **Backup**
MTC는 대상 클러스터에서 PV(영구 볼륨)를 **Retain**으로 마이그레이션한 PV(영구 볼륨)의 회수 정책을 변경합니다. **Backup** CR에는 원래 회수 정책을 나타내는 **openshift.io/orig-reclaim-policy** 주석이 포함되어 있습니다. 마이그레이션된 PV의 회수 정책을 수동으로 복원할 수 있습니다.
- **Restore**

절차

1. **openshift-migration** 네임스페이스에 **MigMigration** CR을 나열합니다.

```
$ oc get migmigration -n openshift-migration
```

출력 예

```
NAME                                AGE
88435fe0-c9f8-11e9-85e6-5d593ce65e10  6m42s
```

2. **MigMigration** CR을 검사합니다.

```
$ oc describe migmigration 88435fe0-c9f8-11e9-85e6-5d593ce65e10 -n openshift-migration
```

출력은 다음 예제와 유사합니다.

MigMigration 예제 출력

```

name:      88435fe0-c9f8-11e9-85e6-5d593ce65e10
namespace: openshift-migration
labels:    <none>
annotations: touch: 3b48b543-b53e-4e44-9d34-33563f0f8147
apiVersion: migration.openshift.io/v1alpha1
kind:      MigMigration
metadata:
  creationTimestamp: 2019-08-29T01:01:29Z
  generation:       20
  resourceVersion:  88179
  selfLink:         /apis/migration.openshift.io/v1alpha1/namespaces/openshift-
migration/migmigrations/88435fe0-c9f8-11e9-85e6-5d593ce65e10
  uid:              8886de4c-c9f8-11e9-95ad-0205fe66cbb6
spec:
  migPlanRef:
    name:      socks-shop-mig-plan
    namespace: openshift-migration
  quiescePods: true
  stage:      false
status:
  conditions:
    category:      Advisory
    durable:       True
    lastTransitionTime: 2019-08-29T01:03:40Z
    message:       The migration has completed successfully.
    reason:        Completed
    status:        True
    type:          Succeeded
    phase:         Completed
    startTimestamp: 2019-08-29T01:01:29Z
  events:         <none>

```

PV 데이터를 설명하는 Velero 백업 CR #2 예제 출력

```

apiVersion: velero.io/v1
kind: Backup
metadata:
  annotations:
    openshift.io/migrate-copy-phase: final
    openshift.io/migrate-quiesce-pods: "true"
    openshift.io/migration-registry: 172.30.105.179:5000
    openshift.io/migration-registry-dir: /socks-shop-mig-plan-registry-44dd3bd5-c9f8-11e9-95ad-
0205fe66cbb6
    openshift.io/orig-reclaim-policy: delete
  creationTimestamp: "2019-08-29T01:03:15Z"
  generateName: 88435fe0-c9f8-11e9-85e6-5d593ce65e10-
  generation: 1
  labels:
    app.kubernetes.io/part-of: migration
    migmigration: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
    migration-stage-backup: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6

```

```

  velero.io/storage-location: myrepo-vpzq9
  name: 88435fe0-c9f8-11e9-85e6-5d593ce65e10-59gb7
  namespace: openshift-migration
  resourceVersion: "87313"
  selfLink: /apis/velero.io/v1/namespaces/openshift-migration/backups/88435fe0-c9f8-11e9-85e6-5d593ce65e10-59gb7
  uid: c80dbbc0-c9f8-11e9-95ad-0205fe66cbb6
spec:
  excludedNamespaces: []
  excludedResources: []
  hooks:
    resources: []
  includeClusterResources: null
  includedNamespaces:
  - sock-shop
  includedResources:
  - persistentvolumes
  - persistentvolumeclaims
  - namespaces
  - imagestreams
  - imagestreamtags
  - secrets
  - configmaps
  - pods
  labelSelector:
    matchLabels:
      migration-included-stage-backup: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
  storageLocation: myrepo-vpzq9
  ttl: 720h0m0s
  volumeSnapshotLocations:
  - myrepo-wv6fx
status:
  completionTimestamp: "2019-08-29T01:02:36Z"
  errors: 0
  expiration: "2019-09-28T01:02:35Z"
  phase: Completed
  startTimestamp: "2019-08-29T01:02:35Z"
  validationErrors: null
  version: 1
  volumeSnapshotsAttempted: 0
  volumeSnapshotsCompleted: 0
  warnings: 0

```

Kubernetes 리소스를 설명하는 Velero 복원 CR #2 예제 출력

```

apiVersion: velero.io/v1
kind: Restore
metadata:
  annotations:
    openshift.io/migrate-copy-phase: final
    openshift.io/migrate-quiesce-pods: "true"
    openshift.io/migration-registry: 172.30.90.187:5000
    openshift.io/migration-registry-dir: /socks-shop-mig-plan-registry-36f54ca7-c925-11e9-825a-06fa9fb68c88
  creationTimestamp: "2019-08-28T00:09:49Z"

```

```

generateName: e13a1b60-c927-11e9-9555-d129df7f3b96-
generation: 3
labels:
  app.kubernetes.io/part-of: migration
  migmigration: e18252c9-c927-11e9-825a-06fa9fb68c88
  migration-final-restore: e18252c9-c927-11e9-825a-06fa9fb68c88
name: e13a1b60-c927-11e9-9555-d129df7f3b96-gb8nx
namespace: openshift-migration
resourceVersion: "82329"
selfLink: /apis/velero.io/v1/namespaces/openshift-migration/restores/e13a1b60-c927-11e9-9555-
d129df7f3b96-gb8nx
uid: 26983ec0-c928-11e9-825a-06fa9fb68c88
spec:
  backupName: e13a1b60-c927-11e9-9555-d129df7f3b96-sz24f
  excludedNamespaces: null
  excludedResources:
    - nodes
    - events
    - events.events.k8s.io
    - backups.velero.io
    - restores.velero.io
    - resticrepositories.velero.io
  includedNamespaces: null
  includedResources: null
  namespaceMapping: null
  restorePVs: true
status:
  errors: 0
  failureReason: ""
  phase: Completed
  validationErrors: null
  warnings: 15

```

10.4. 일반적인 문제 및 우려 사항

이 섹션에서는 마이그레이션 중에 발생할 수 있는 일반적인 문제 및 우려 사항에 대해 설명합니다.

10.4.1. 직접 볼륨 마이그레이션이 완료되지 않음

직접 볼륨 마이그레이션이 완료되지 않으면 대상 클러스터에 소스 클러스터와 동일한 **node-selector** 주석이 없을 수 있습니다.

MTC(Migration Toolkit for Containers)는 보안 컨텍스트 제약 조건 및 스케줄링 요구 사항을 유지하기 위해 모든 주석이 있는 네임스페이스를 마이그레이션합니다. 직접 볼륨 마이그레이션 중에 MTC는 소스 클러스터에서 마이그레이션된 네임스페이스의 대상 클러스터에서 Rsync 전송 포드를 생성합니다. 대상 클러스터 네임스페이스에 소스 클러스터 네임스페이스와 동일한 주석이 없는 경우 Rsync 전송 포드를 예약할 수 없습니다. Rsync 포드는 **Pending** 상태로 유지됩니다.

다음 절차를 수행하여 이 문제를 확인하고 수정할 수 있습니다.

절차

1. **MigMigration** CR의 상태를 확인합니다.

```
$ oc describe migmigration <pod> -n openshift-migration
```

- 출력에는 다음 상태 메시지가 포함됩니다.

출력 예

```
Some or all transfer pods are not running for more than 10 mins on destination cluster
```

2. 소스 클러스터에서 마이그레이션된 네임스페이스의 세부 정보를 가져옵니다.

```
$ oc get namespace <namespace> -o yaml 1
```

- 1** 마이그레이션된 네임스페이스를 지정합니다.

3. 대상 클러스터에서 마이그레이션된 네임스페이스를 편집합니다.

```
$ oc edit namespace <namespace>
```

4. 다음 예와 같이 마이그레이션된 네임스페이스에 누락된 **openshift.io/node-selector** 주석을 추가합니다.

```
apiVersion: v1
kind: Namespace
metadata:
  annotations:
    openshift.io/node-selector: "region=east"
...
```

5. 마이그레이션 계획을 다시 실행합니다.

10.4.2. 오류 메시지 및 해결 방법

이 섹션에서는 MTC(Migration Toolkit for Containers)에 발생할 수 있는 일반적인 오류 메시지와 근본적인 원인을 해결하는 방법을 설명합니다.

10.4.2.1. MTC 콘솔에 처음 액세스할 때 표시되는 CA 인증서 오류

MTC 콘솔에 액세스하려고 할 때 **CA certificate error** 메시지가 표시되면 클러스터 중 하나에서 자체 서명된 CA 인증서를 사용할 가능성이 높습니다.

이 문제를 해결하려면 오류 메시지에 표시된 **oauth-authorization-server** URL로 이동하여 인증서를 수락합니다. 이 문제를 영구적으로 해결하려면 웹 브라우저의 신뢰 저장소에 인증서를 추가합니다.

인증서를 승인한 후 **Unauthorized** 메시지가 표시되면 MTC 콘솔로 이동하여 웹 페이지를 새로 고칩니다.

10.4.2.2. MTC 콘솔의 OAuth 시간제한 오류

자체 서명 인증서를 허용한 후 MTC 콘솔에 **connection has timed out** 메시지가 표시되면 원인은 다음과 같습니다.

- OAuth 서버에 대한 네트워크 액세스 중단
- OpenShift Container Platform 콘솔에 대한 네트워크 액세스 중단

- **oauth-authorization-server** URL에 대한 액세스를 차단하는 프록시 구성입니다. 자세한 내용은 [OAuth 시간제한 오류로 인해 액세스할 수 없는 MTC 콘솔](#) 을 참조하십시오.

시간 초과 원인을 확인하려면 다음을 수행합니다.

- 브라우저 웹 검사를 사용하여 MTC 콘솔 웹 페이지를 확인합니다.
- **Migration UI** pod 로그에 오류가 있는지 확인합니다.

10.4.2.3. 알 수 없는 권한 오류로 서명된 인증서

자체 서명된 인증서를 사용하여 MTC(Migration Toolkit for Containers)의 클러스터 또는 복제 리포지토리를 보호하는 경우 **Certificate signed by unknown authority** 오류 메시지와 함께 인증서 확인에 실패할 수 있습니다.

사용자 정의 CA 인증서 번들 파일을 생성하고 클러스터 또는 복제 리포지토리를 추가할 때 MTC 웹 콘솔에 업로드할 수 있습니다.

절차

원격 끝점에서 CA 인증서를 다운로드하여 CA 번들 파일로 저장합니다.

```
$ echo -n | openssl s_client -connect <host_FQDN>:<port> \ 1
| sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > <ca_bundle.cert> 2
```

1 끝점의 호스트 FQDN 및 포트를 지정합니다(예: **api.my-cluster.example.com:6443**).

2 CA 번들 파일의 이름을 지정합니다.

10.4.2.4. Velero pod 로그의 백업 스토리지 위치 오류

Velero Backup 사용자 정의 리소스에 존재하지 않는 백업 스토리지 위치(BSL)에 대한 참조가 포함된 경우 **Velero Pod** 로그에 다음과 같은 오류 메시지가 표시될 수 있습니다.

```
$ oc logs <Velero_Pod> -n openshift-migration
```

출력 예

```
level=error msg="Error checking repository for stale locks" error="error getting backup storage
location: BackupStorageLocation.velero.io \"ts-dpa-1\" not found" error.file="/remote-
source/src/github.com/vmware-tanzu/velero/pkg/restic/repository_manager.go:259"
```

이러한 오류 메시지는 무시해도 됩니다. 누락된 BSL로 인해 마이그레이션이 실패하지는 않습니다.

10.4.2.5. Velero Pod 로그의 Pod 볼륨 백업 시간 초과 오류

Restic 시간제한으로 인해 마이그레이션이 실패하면 **Velero pod** 로그에 다음 오류가 표시됩니다.

```
level=error msg="Error backing up item" backup=velero/monitoring error="timed out waiting for all
PodVolumeBackups to complete"
error.file="/go/src/github.com/heptio/velero/pkg/restic/backupper.go:165"
error.function="github.com/heptio/velero/pkg/restic.(*backupper).BackupPodVolumes" group=v1
```

`restic_timeout`의 기본값은 1시간입니다. 값이 클수록 오류 메시지 반환이 지연될 수 있다는 점을 염두에 두고 대규모 마이그레이션의 경우 이 매개 변수를 늘릴 수 있습니다.

절차

1. OpenShift Container Platform 웹 콘솔에서 **Operator** → 설치된 **Operator**로 이동합니다.
2. **Migration Toolkit for Containers Operator**를 클릭합니다.
3. **MigrationController** 탭에서 **migration-controller**를 클릭합니다.
4. **YAML** 탭에서 다음 매개 변수 값을 업데이트합니다.

```
spec:
  restic_timeout: 1h 1
```

1 유효한 단위는 **h**(시간), **m**(분) 및 **s**(초)입니다(예: **3h30m15s**).

5. **저장**을 클릭합니다.

10.4.2.6. MigMigration 사용자 지정 리소스의 제한적 유효성 검사 오류

파일 시스템 데이터 복사 방법을 사용하여 영구 볼륨을 마이그레이션할 때 데이터 확인에 실패하면 **MigMigration** CR에 다음 오류가 표시됩니다.

출력 예

```
status:
  conditions:
  - category: Warn
    durable: true
    lastTransitionTime: 2020-04-16T20:35:16Z
    message: There were verify errors found in 1 Restic volume restores. See restore `<registry-example-migration-rvwcm>`
      for details 1
    status: "True"
    type: ResticVerifyErrors 2
```

- 1** 오류 메시지는 **Restore** CR 이름을 식별합니다.
- 2** **ResticVerifyErrors**는 확인 오류가 포함된 일반적인 오류 경고 유형입니다.



참고

데이터 확인 오류로 인해 마이그레이션 프로세스가 실패하지 않습니다.

Restore CR을 확인하여 데이터 확인 오류의 원인을 식별할 수 있습니다.

절차

1. 대상 클러스터에 로그인합니다.

2. **Restore** CR을 보기:

```
$ oc describe <registry-example-migration-rvwcm> -n openshift-migration
```

출력은 **PodVolumeRestore** 오류가 있는 영구 볼륨을 식별합니다.

출력 예

```
status:
  phase: Completed
  podVolumeRestoreErrors:
  - kind: PodVolumeRestore
    name: <registry-example-migration-rvwcm-98t49>
    namespace: openshift-migration
  podVolumeRestoreResticErrors:
  - kind: PodVolumeRestore
    name: <registry-example-migration-rvwcm-98t49>
    namespace: openshift-migration
```

3. **PodVolumeRestore** CR 보기:

```
$ oc describe <migration-example-rvwcm-98t49>
```

출력은 오류를 기록한 **Restic** pod를 식별합니다.

출력 예

```
completionTimestamp: 2020-05-01T20:49:12Z
errors: 1
resticErrors: 1
...
resticPod: <restic-nr2v5>
```

4. **Restic** pod 로그를 보고 오류를 찾습니다.

```
$ oc logs -f <restic-nr2v5>
```

10.4.2.7. **root_squash**가 활성화된 NFS 스토리지에서 마이그레이션할 때 **Restic** 권한 오류

NFS 스토리지에서 데이터를 마이그레이션 중이고 **root_squash**가 활성화된 경우 **Restic**이 **nfsnobody**에 매핑되고 마이그레이션을 수행할 수 있는 권한이 없습니다. **Restic** Pod 로그에 다음 오류가 표시됩니다.

출력 예

```
backup=openshift-migration/<backup_id> controller=pod-volume-backup error="fork/exec
/usr/bin/restic: permission denied" error.file="/go/src/github.com/vmware-
tanzu/velero/pkg/controller/pod_volume_backup_controller.go:280"
error.function="github.com/vmware-tanzu/velero/pkg/controller.
(*podVolumeBackupController).processBackup"
logSource="pkg/controller/pod_volume_backup_controller.go:280" name=<backup_id>
namespace=openshift-migration
```

Restic의 추가 그룹을 생성하고 **MigrationController** CR 매니페스트에 그룹 ID를 추가하여 이 문제를 해결할 수 있습니다.

절차

1. NFS 스토리지에서 Restic에 대한 보조 그룹을 생성합니다.
2. 그룹 소유권이 상속되도록 NFS 디렉터리에 **setgid** 비트를 설정합니다.
3. 소스 및 대상 클러스터의 **MigrationController** CR 매니페스트에 **restic_supplemental_groups** 매개변수를 추가합니다.

```
spec:
  restic_supplemental_groups: <group_id> 1
```

- 1 보조 그룹 ID를 지정합니다.

4. 변경 사항을 적용할 수 있도록 **Restic** Pod가 다시 시작될 때까지 기다립니다.

10.5. 마이그레이션 롤백

MTC 웹 콘솔 또는 CLI를 사용하여 마이그레이션을 롤백할 수 있습니다.

마이그레이션을 수동으로 롤백할 수도 있습니다.

10.5.1. MTC 웹 콘솔을 사용하여 마이그레이션 롤백

MTC(Migration Toolkit for Containers) 웹 콘솔을 사용하여 마이그레이션을 롤백할 수 있습니다.

참고

다음 리소스는 실패한 직접 볼륨 마이그레이션(DVM) 이후 디버깅을 위해 마이그레이션된 네임스페이스에 남아 있습니다.

- 구성 맵 (소스 및 대상 클러스터)
- **Secret** 오브젝트(소스 및 대상 클러스터)
- **Rsync** CR (소스 클러스터)

이러한 리소스는 롤백에 영향을 미치지 않습니다. 수동으로 삭제할 수 있습니다.

나중에 동일한 마이그레이션 계획을 성공적으로 실행하면 실패한 마이그레이션의 리소스가 자동으로 삭제됩니다.

마이그레이션 실패로 인해 애플리케이션이 중지된 경우 영구 볼륨의 데이터 손상을 방지하려면 마이그레이션을 롤백해야 합니다.

원래 애플리케이션이 소스 클러스터에서 계속 실행 중이므로 마이그레이션 중에 애플리케이션이 중지되지 않은 경우 롤백이 필요하지 않습니다.

절차

1. MTC 웹 콘솔에서 **마이그레이션 계획**을 클릭합니다.
2. 마이그레이션 계획 옆의 옵션 메뉴  를 클릭하고 **마이그레이션**에서 **롤백**을 선택합니다.
3. **롤백**을 클릭하고 롤백이 완료될 때까지 기다립니다.
마이그레이션 계획 세부 사항에서 **롤백 성공**이 표시됩니다.
4. 소스 클러스터의 OpenShift Container Platform 웹 콘솔에서 롤백이 성공했는지 확인합니다.
 - a. **홈** → **프로젝트**를 클릭합니다.
 - b. 마이그레이션된 프로젝트를 클릭하여 상태를 봅니다.
 - c. **경로** 섹션에서 **위치**를 클릭하여 해당되는 경우 애플리케이션이 작동하는지 확인합니다.
 - d. **워크로드** → **포드**를 클릭하여 포드가 마이그레이션된 네임스페이스에서 실행 중인지 확인합니다.
 - e. **스토리지** → **영구 볼륨**을 클릭하여 마이그레이션된 영구 볼륨이 올바르게 프로비저닝되었는지 확인합니다.

10.5.2. 명령줄 인터페이스를 사용하여 마이그레이션 롤백

명령줄 인터페이스에서 **MigMigration CR**(사용자 정의 리소스)을 생성하여 마이그레이션을 롤백할 수 있습니다.



참고

다음 리소스는 실패한 직접 볼륨 마이그레이션(DVM) 이후 디버깅을 위해 마이그레이션된 네임스페이스에 남아 있습니다.

- 구성 맵 (소스 및 대상 클러스터)
- **Secret** 오브젝트(소스 및 대상 클러스터)
- **Rsync CR** (소스 클러스터)

이러한 리소스는 롤백에 영향을 미치지 않습니다. 수동으로 삭제할 수 있습니다.

나중에 동일한 마이그레이션 계획을 성공적으로 실행하면 실패한 마이그레이션의 리소스가 자동으로 삭제됩니다.

마이그레이션 실패로 인해 애플리케이션이 중지된 경우 영구 볼륨의 데이터 손상을 방지하려면 마이그레이션을 롤백해야 합니다.

원래 애플리케이션이 소스 클러스터에서 계속 실행 중이므로 마이그레이션 중에 애플리케이션이 중지되지 않은 경우 롤백이 필요하지 않습니다.

절차

1. 다음 예제를 기반으로 **MigMigration CR**을 생성합니다.

```
$ cat << EOF | oc apply -f -
apiVersion: migration.openshift.io/v1alpha1
```

```
kind: MigMigration
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: <migmigration>
  namespace: openshift-migration
spec:
  ...
  rollback: true
  ...
  migPlanRef:
    name: <migplan> 1
    namespace: openshift-migration
EOF
```

1 연결된 **MigPlan** CR의 이름을 지정합니다.

2. MTC 웹 콘솔에서 마이그레이션된 프로젝트 리소스가 대상 클러스터에서 제거되었는지 확인합니다.
3. 마이그레이션된 프로젝트 리소스가 소스 클러스터에 있고 애플리케이션이 실행 중인지 확인합니다.

10.5.3. 마이그레이션 수동 롤백

stage Pod를 삭제하고 애플리케이션의 정지를 해제하여 실패한 마이그레이션을 수동으로 롤백할 수 있습니다.

동일한 마이그레이션 계획을 성공적으로 실행하면 실패한 마이그레이션의 리소스가 자동으로 삭제됩니다.



참고

다음 리소스는 실패한 직접 볼륨 마이그레이션(DVM) 후에도 마이그레이션된 네임스페이스에 남아 있습니다.

- 구성 맵 (소스 및 대상 클러스터)
- **Secret** 오브젝트(소스 및 대상 클러스터)
- **Rsync** CR (소스 클러스터)

이러한 리소스는 롤백에 영향을 미치지 않습니다. 수동으로 삭제할 수 있습니다.

절차

1. 모든 클러스터에서 **stage** Pod를 삭제합니다.

```
$ oc delete $(oc get pods -l migration.openshift.io/is-stage-pod -n <namespace>) 1
```

1 **MigPlan** CR에 지정된 네임스페이스입니다.

- 복제본을 사전 마이그레이션 번호로 확장하여 소스 클러스터에서 애플리케이션 정지를 해제합니다.

```
$ oc scale deployment <deployment> --replicas=<premigration_replicas>
```

Deployment CR의 **migration.openshift.io/preQuiesceReplicas** 주석에는 복제본의 사전 마이그레이션 수가 표시됩니다.

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
    migration.openshift.io/preQuiesceReplicas: "1"
```

- 애플리케이션 pod가 소스 클러스터에서 실행 중인지 확인합니다.

```
$ oc get pod -n <namespace>
```

추가 리소스

- [웹 콘솔을 사용하여 클러스터에서 Operator 삭제](#)