

# VMware vRealize Orchestrator 설치 및 구성

2021년 4월 15일

vRealize Orchestrator 8.4

다음 VMware 웹 사이트에서 최신 기술 문서를 확인할 수 있습니다.

<https://docs.vmware.com/kr/>

**VMware, Inc.**  
3401 Hillview Ave.  
Palo Alto, CA 94304  
[www.vmware.com](http://www.vmware.com)

**VMware 코리아**  
서울시 강남구  
영동대로 517  
아셈타워 13층  
(우) 06164  
전화: +82 2 3016 6500  
팩스: +82 2 3016 6501  
[www.vmware.com/kr](http://www.vmware.com/kr)

# 목차

## VMware vRealize Orchestrator 설치 및 구성 6

### 1 VMware vRealize Orchestrator 소개 7

- Orchestrator 플랫폼의 주요 기능 7
- vRealize Orchestrator 사용자 역할 9
- vRealize Orchestrator 아키텍처 10
- vRealize Orchestrator Plug-in 11

### 2 vRealize Orchestrator 시스템 요구 사항 12

- 기본 장치 구성 요소 12
- 하드웨어 요구 사항 13
- 최대 확장성 13
- 포트 및 끝점 13
- 브라우저 지원 14
- 국제화 지원 14

### 3 vRealize Orchestrator 구성 요소 설정 16

- vCenter Server 설정 16
- 인증 방법 16

### 4 vRealize Orchestrator 설치 18

- vRealize Orchestrator Appliance 다운로드 및 배포 18
- vRealize Orchestrator Appliance의 전원을 켜고 홈 페이지 열기 20
- vRealize Orchestrator Appliance에 대한 SSH 액세스 사용 또는 사용 안 함 21

### 5 초기 구성 22

- 독립형 vRealize Orchestrator 서버 구성 22
  - vRealize Automation 인증을 사용하여 독립형 vRealize Orchestrator 서버 구성 22
  - vSphere 인증을 사용하여 독립형 vRealize Orchestrator 서버 구성 23
- 라이선스 기반 vRealize Orchestrator 기능 사용 설정 25
- vRealize Orchestrator 데이터베이스 연결 26
- 인증서 관리 26
  - vRealize Orchestrator 인증서 관리 27
    - vRealize Orchestrator에 대한 사용자 지정 TLS 인증서 생성 27
    - vRealize Orchestrator에 대한 사용자 지정 TLS 인증서 설정 28
    - 제어 센터를 사용하여 신뢰할 수 있는 인증서 가져오기 30

vRealize Orchestrator Plug-in 구성	31
vRealize Orchestrator 플러그인 관리	31
vRealize Orchestrator 플러그인 설치 또는 업데이트	31
플러그인 삭제	32
vRealize Orchestrator 고가용성	32
최대 확장성	33
vRealize Orchestrator 클러스터 구성	33
vRealize Orchestrator 클러스터 노드 제거	35
독립형 vRealize Orchestrator 배포 확장	35
vRealize Orchestrator 클러스터 모니터링	37
고객 환경 개선 프로그램 구성	37
VMware가 수신하는 정보의 범주	37
고객 환경 향상 프로그램 가입 또는 탈퇴	37

## 6 vRealize Orchestrator API 서비스 사용 39

REST API를 통해 SSL 인증서 관리	39
REST API를 사용하여 TLS 인증서 삭제	40
REST API를 사용하여 TLS 인증서 가져오기	40
REST API를 사용하여 키 저장소 생성	41
REST API를 사용하여 키 저장소 삭제	42
REST API를 사용하여 키 추가	42

## 7 추가 구성 옵션 43

인증 재구성	43
인증 제공자 변경	43
인증 매개 변수 변경	44
워크플로 실행 속성 구성	44
vRealize Orchestrator 로그 파일	45
로깅 지속성	45
vRealize Orchestrator 로그 구성	46
vRealize Log Insight와 로깅 통합 구성	46
vRealize Orchestrator에서 Syslog 통합 생성 또는 덮어쓰기	47
vRealize Orchestrator에서 Syslog 통합 삭제	48
Kerberos 디버그 로깅 사용	48
Opentracing 및 Wavefront 확장 사용	49
Opentracing 확장 구성	50
Wavefront 확장 구성	51
vRealize Orchestrator에 대한 시간 동기화 사용	52
vRealize Orchestrator에 대한 시간 동기화 비활성화	53
vRealize Orchestrator Kubernetes CIDR 구성	53

## 8 구성 사용 사례 및 문제 해결 55

- 서버 vRealize Orchestrator 번호 확인 55
- vSphere Web Client용 vRealize Orchestrator 플러그인 구성 55
- 실행 중인 워크플로 취소 56
- vRealize Orchestrator 서버 디버깅 사용 57
- vRealize Orchestrator Appliance 디스크 크기 조정 58
- vRealize Orchestrator 서버의 힙 메모리 크기를 조정하는 방법 60
- Site Recovery Manager를 사용한 vRealize Orchestrator의 재해 복구 61
  - vSphere Replication에 대한 가상 시스템 구성 61
  - 보호 그룹 만들기 61
  - 복구 계획 만들기 63
  - 복구 계획을 폴더로 구성 64
  - 복구 계획 편집 65

## 9 시스템 속성 설정 66

- 워크플로 및 작업에 대한 서버 파일 시스템 액세스 설정 66
  - vRealize Orchestrator 시스템에 대한 쓰기 액세스 권한을 허용하는 js-io-rights.conf 파일의 규칙 66
  - 워크플로 및 작업에 대한 서버 파일 시스템 액세스 설정 67
- 워크플로 및 작업에 대한 운영 체제 명령의 액세스 설정 68
- Java 클래스에 JavaScript 액세스 설정 69
- 사용자 지정 시간 초과 속성 설정 70
- vRealize Orchestrator SQL 플러그인을 위한 JDBC 커넥터 추가 71

## 10 이후 작업 73

# VMware vRealize Orchestrator 설치 및 구성

"VMware vRealize Orchestrator 설치 및 구성"에서는 VMware® vRealize Orchestrator 설치 및 구성에 대한 정보와 지침을 제공합니다.

## 대상 사용자

이 정보는 가상 시스템 기술과 데이터 센터 작업에 익숙한 고급 vSphere 관리자 및 숙련된 시스템 관리자를 대상으로 제공됩니다.

# VMware vRealize Orchestrator 소개

# 1

VMware vRealize Orchestrator는 VMware 제품은 물론 타사 기술을 관리할 자동화, 구성 가능 프로세스를 만들고 실행할 수 있는 확장 가능한 워크플로 라이브러리를 제공하는 개발 및 프로세스 자동화 플랫폼입니다.

vRealize Orchestrator는 VMware 및 타사 애플리케이션(서비스 데스크, 변경 관리 시스템, IT 자산 관리 시스템 등) 모두의 관리 및 운영 작업을 자동화합니다.

본 장은 다음 항목을 포함합니다.

- Orchestrator 플랫폼의 주요 기능
- vRealize Orchestrator 사용자 역할
- vRealize Orchestrator 아키텍처
- vRealize Orchestrator Plug-in

## Orchestrator 플랫폼의 주요 기능

vRealize Orchestrator는 오케스트레이션 도구에 필요한 공통 기능을 제공하는 오케스트레이션 플랫폼, 하위 시스템의 제어를 통합하는 플러그인 아키텍처 그리고 워크플로 라이브러리라는 세 가지 별개의 계층으로 구성됩니다. vRealize Orchestrator는 새 플러그인 및 콘텐츠를 사용하여 확장할 수 있는 개방형 플랫폼이며 REST API를 통해 더 큰 아키텍처에 통합될 수 있습니다.

vRealize Orchestrator에는 워크플로 실행 및 관리에 도움이 되는 몇 가지 주요 기능이 포함되어 있습니다.

### 지속성

운영 등급 PostgreSQL 데이터베이스는 프로세스, 워크플로 상태 및 vRealize Orchestrator 구성과 같은 관련 정보를 저장하는 데 사용됩니다.

### 중앙 집중식 관리

vRealize Orchestrator는 프로세스를 관리하는 중앙 집중식 도구를 제공합니다. 전체 버전 기록을 가진 애플리케이션 서버 기반 플랫폼은 같은 스토리지 위치에 스크립트와 프로세스 관련 기본 형식을 저장할 수 있습니다. 이러한 방식으로 버전 관리가 되지 않고 서버에 적절한 변경 제어가 없는 스크립트를 방지할 수 있습니다.

## 검사점 설정

워크플로의 각 단계는 데이터베이스에 저장되므로 서버를 다시 시작해야 할 경우 데이터 손실이 예방됩니다. 이 기능은 특히 장기 실행 프로세스에 유용합니다.

## 제어 센터

제어 센터는 런타임 작업, 워크플로 모니터링 그리고 워크플로 실행 및 시스템 리소스 간 상관 관계에 대한 중앙 집중식 관리 인터페이스를 제공하여 vRealize Orchestrator 인스턴스의 관리 효율성을 높이는 웹 기반 포털입니다.

## 버전 관리

모든 vRealize Orchestrator 플랫폼 개체에는 관련 버전 기록이 있습니다. 버전 기록은 프로젝트 단계 또는 위치에 프로세스를 배포할 때 기본 변경 관리에 유용합니다.

## Git 통합

vRealize Orchestrator Client를 사용하면 Git 저장소를 통합하여 vRealize Orchestrator 콘텐츠의 버전 및 소스 제어를 더욱 개선할 수 있습니다. Git을 사용하면 여러 vRealize Orchestrator 인스턴스에서 워크플로 개발을 관리할 수 있습니다. "VMware vRealize Orchestrator 클라이언트 사용" 가이드에서 "vRealize Orchestrator 클라이언트에서 Git 사용"을 참조하십시오.

## 스크립팅 엔진

Mozilla Rhino JavaScript 엔진은 vRealize Orchestrator Client 플랫폼을 위한 빌딩 블록을 만드는 방법을 제공합니다. 스크립팅 엔진은 기본 버전 제어, 변수 유형 확인, 이름 공간 관리 및 예외 처리를 사용해 향상됩니다. 이 엔진은 다음 빌딩 블록에서 사용할 수 있습니다.

- 작업
- 워크플로
- 정책

## 워크플로 엔진

워크플로 엔진을 사용해 비즈니스 프로세스를 자동화할 수 있습니다. 워크플로 엔진은 워크플로에서 단계별 프로세스 자동화를 만들기 위해 다음 개체를 사용합니다.

- vRealize Orchestrator Client가 제공하는 워크플로 및 작업.
- 고객이 만든 사용자 지정 빌딩 블록.
- 플러그인이 vRealize Orchestrator Client에 추가한 개체.

사용자, 기타 워크플로, 스케줄 또는 정책이 워크플로를 시작할 수 있습니다.

## 정책 엔진

정책 엔진을 사용해 vRealize Orchestrator Client 서버 또는 플러그인된 기술을 모니터링하고 변경되는 조건에 반응해 이벤트를 생성할 수 있습니다. 정책은 플랫폼이나 플러그인에서 이벤트를 집계할 수 있으며 이는 통합된 기술에서 변경되는 조건을 처리하는 데 도움이 됩니다.



## vRealize Orchestrator Client

vRealize Orchestrator Client를 사용하여 워크플로를 생성, 실행, 편집 및 모니터링합니다. 또한 vRealize Orchestrator Client를 사용하여 작업, 구성, 정책 및 리소스 요소를 관리할 수 있습니다. "vRealize Orchestrator 클라이언트 사용" 을 참조하십시오.

## 개발 및 리소스

vRealize Orchestrator에서 사용할 수 있는 고유한 플러그인을 개발하는 데 도움이 되도록 vRealize Orchestrator 방문 페이지에서는 리소스에 빠르게 액세스할 수 있습니다. vRealize Orchestrator REST API를 사용하여 vRealize Orchestrator 서버에 요청을 보내는 방법에 대한 정보도 찾을 수 있습니다.

## 보안

vRealize Orchestrator는 다음과 같은 고급 보안 기능을 제공합니다.

- 서버 간에 가져오고 내보낸 콘텐츠에 서명하고 암호화하는 공용 키 인프라(PKI).
- 내보낸 콘텐츠의 보기, 편집 및 재배포 방법을 제어하는 디지털 저작권 관리(DRM).
- vRealize Orchestrator Client, vRealize Orchestrator 서버 및 웹 프론트 엔드에 대한 HTTPS 액세스 간에 암호화된 통신을 제공하는 TLS(Transport Layer Security).
- 고급 액세스 권한 관리를 통해 이러한 프로세스가 조작하는 프로세스 및 개체에 대한 액세스를 제어할 수 있습니다.

## 암호화

vRealize Orchestrator는 문자열 암호화를 위해 256비트 암호화 키를 사용하는 FIPS 준수 AES(Advanced Encryption Standard)를 사용합니다. 암호화 키는 임의로 생성되며 클러스터의 일부가 아닌 장치 전반에서 고유합니다. 클러스터의 모든 노드는 암호 키를 공유합니다.

## vRealize Orchestrator 사용자 역할

vRealize Orchestrator는 전역 사용자 역할의 특정 책임을 기반으로 다양한 도구와 인터페이스를 제공합니다. vRealize Orchestrator에는 관리자 그룹(**관리자**)의 일부인 전체 권한이 있는 사용자, 개발자(**워크플로 디자이너**), 문제 해결 중인 사용자(**뷰어**) 및 액세스가 제한된 사용자가 있을 수 있습니다.

vRealize Orchestrator 사용자 역할은 vRealize Orchestrator Client의 **역할 관리** 메뉴에서 관리됩니다. vRealize Orchestrator Client에서 사용자 역할을 구성하는 방법에 대한 자세한 내용은 "VMware vRealize Orchestrator 클라이언트 사용" 가이드의 "vRealize Orchestrator 클라이언트에서 역할 할당" 을 참조하십시오.

---

**참고** vRealize Automation에서 인증된 vRealize Orchestrator 배포의 경우 또는 vRealize Automation 라이선스를 사용하는 경우, 사용자 역할은 vRealize Automation 플랫폼의 ID 및 액세스 관리 서비스로 할당됩니다. "VMware vRealize Orchestrator 클라이언트 사용" 에서 "vRealize Automation에서 vRealize Orchestrator 클라이언트 역할 구성" 을 참조하십시오.

---

사용자 역할	설명
관리자	<p>이 사용자는 특정 그룹에서 생성된 콘텐츠를 포함하여 모든 vRealize Orchestrator 플랫폼 기능 및 콘텐츠에 대한 전체 액세스 권한을 갖습니다. 기본 관리자 사용자 책임은 다음과 같습니다.</p> <ul style="list-style-type: none"> <li>■ vRealize Orchestrator 설치 및 구성.</li> <li>■ vRealize Orchestrator Client에 사용자 추가, 역할 할당, 그룹 생성 및 삭제. "VMware vRealize Orchestrator 클라이언트 사용"에서 "vRealize Orchestrator 클라이언트에서 그룹 생성"을 참조하십시오.</li> <li>■ vRealize Orchestrator 환경의 개발자를 위해 Git 저장소와 통합 생성. "VMware vRealize Orchestrator 클라이언트 사용"에서 "Git 저장소에 대한 연결 구성"을 참조하십시오.</li> <li>■ 워크플로 유효성 검사 및 디버깅 워크플로 스크립트와 같은 기능을 통해 vRealize Orchestrator 환경 문제를 해결합니다.</li> </ul>
뷰어	<p>이 사용자는 모든 그룹 및 그룹 콘텐츠를 포함하여 모든 vRealize Orchestrator Client에 대한 읽기 전용 액세스 권한이 있습니다. 이 사용자는 볼 수 있지만 콘텐츠를 생성, 편집 또는 실행하거나 워크플로 실행, 워크플로 실행 로그 또는 패키지를 내보낼 수는 없습니다. 뷰어는 그룹 사용 권한으로 제한되지 않습니다.</p> <p><b>참고</b> 뷰어 역할은 vRealize Automation으로 인증된 vRealize Orchestrator 인스턴스에 대해서만 지원됩니다. 이 역할은 기본적으로 vRealize Automation 역할에 매핑되어 있지 않으므로 사용자에게 명시적으로 할당해야 합니다.</p>
워크플로 디자이너	<p>이 사용자는 개체를 생성 및 편집하여 vRealize Orchestrator 플랫폼 기능을 확장할 수 있습니다. 워크플로 디자이너는 vRealize Orchestrator Client의 관리 및 문제 해결 기능에 액세스할 수 없습니다. 기본 워크플로 디자이너 책임은 다음과 같습니다.</p> <ul style="list-style-type: none"> <li>■ 워크플로, 작업, 정책 및 구성 요소와 같은 vRealize Orchestrator 개체 생성, 편집, 실행 및 삭제.</li> <li>■ 워크플로 실행 스케줄링. "VMware vRealize Orchestrator 클라이언트 사용"에서 "vRealize Orchestrator 클라이언트에서 워크플로 예약"을 참조하십시오.</li> <li>■ 워크플로 개발자가 생성한 콘텐츠를 자신에게 할당된 그룹에 추가.</li> <li>■ vRealize Orchestrator 콘텐츠 인벤토리에 대한 로컬 변경 내용을 연결된 Git 저장소로 푸시합니다. "VMware vRealize Orchestrator 클라이언트 사용"에서 "변경 내용을 Git 저장소로 푸시"를 참조하십시오.</li> </ul>
제한된 권한을 가진 사용자	<p>할당된 역할이 없는 사용자는 vRealize Orchestrator Client에 로그인할 수 있지만 클라이언트 기능 및 콘텐츠에 대한 액세스는 제한됩니다. 그룹에 할당된 경우, 이 사용자는 해당 그룹에 포함된 콘텐츠를 살펴보고 실행할 수 있습니다.</p>

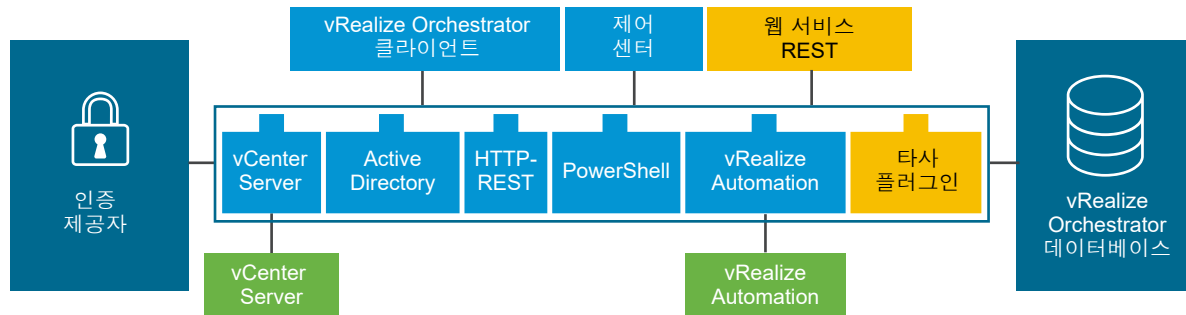
## vRealize Orchestrator 아키텍처

vRealize Orchestrator는 사용자가 오케스트레이션 프로세스를 자동화하는 워크플로를 생성하고 실행할 수 있는 워크플로 라이브러리와 워크플로 엔진을 포함하고 있습니다. 사용자는 vRealize Orchestrator가 일련의 플러그인을 통해 액세스하는 다양한 기술의 개체에서 워크플로를 실행합니다.

vRealize Orchestrator는 vCenter Server 및 vRealize Automation 플러그인을 포함하는 표준 플러그인 집합을 제공하므로, 플러그인이 노출된 다양한 환경에서 작업을 오케스트레이션할 수 있습니다.

또한 vRealize Orchestrator는 외부 타사 애플리케이션을 오케스트레이션 플랫폼에 연결하기 위한 개방형 아키텍처를 제공합니다. 사용자는 직접 정의한 플러그인된 기술의 개체를 워크플로에서 실행할 수 있습니다. vRealize Orchestrator는 사용자 계정을 관리하기 위해 인증 제공자에 연결하며, 실행하는 워크플로의 정보를 저장하기 위해 미리 구성된 PostgreSQL 데이터베이스에 연결합니다. vRealize Orchestrator Client 또는 웹 서비스를 통해 vRealize Orchestrator, 여기에 노출되는 개체 및 vRealize Orchestrator 워크플로에 액세스할 수 있습니다. vRealize Orchestrator 워크플로 및 서비스의 모니터링과 구성은 vRealize Orchestrator Client 및 제어 센터를 통해 수행됩니다.

그림 1-1. VMware vRealize Orchestrator 아키텍처



## vRealize Orchestrator Plug-in

플러그인을 통해 vRealize Orchestrator를 사용하여 외부 기술과 애플리케이션에 액세스하고 제어할 수 있습니다. vRealize Orchestrator 플러그인에서 외부 기술을 노출시킴으로써 해당 외부 기술의 개체와 기능에 액세스하는 워크플로에서 개체와 기능을 통합할 수 있습니다.

플러그인을 사용하여 액세스할 수 있는 외부 기술에는 가상화 관리 도구, 이메일 시스템, 데이터베이스, 디렉토리 서비스, 원격 제어 인터페이스 등이 포함됩니다.

vRealize Orchestrator는 VMware vCenter Server API 및 이메일 기능과 같은 기술을 워크플로에 통합하는 데 사용할 수 있는 표준 플러그인 집합을 제공합니다. 플러그인을 사용하여 새로운 IT 서비스 배포를 자동화하거나 기존 인프라 및 애플리케이션 서비스의 기능을 조정할 수 있습니다. 또한 vRealize Orchestrator 개방형 플러그인 아키텍처를 사용하여 다른 애플리케이션에 액세스하는 플러그인을 개발할 수 있습니다.

VMware가 개발하는 vRealize Orchestrator 플러그인은 .vmoapp 파일로 배포됩니다. VMware에서 개발하고 배포하는 vRealize Orchestrator 플러그인에 대한 자세한 내용은 [vRealize Orchestrator 외부 플러그인](#)을 참조하십시오. 타사 vRealize Orchestrator 플러그인에 대한 자세한 내용은 [VMware Marketplace](#)를 참조하십시오.

# vRealize Orchestrator 시스템 요구 사항

## 2

시스템은 vRealize Orchestrator가 제대로 작동하기 위해 필요한 기술적 요구 사항을 충족해야 합니다.

지원되는 vCenter Server, vSphere Web Client, vRealize Automation 및 기타 VMware 솔루션 목록은 [VMware 제품 상호 운용성 매트릭스](#)를 참조하십시오.

본 장은 다음 항목을 포함합니다.

- [vRealize Orchestrator Appliance 구성 요소](#)
- [vRealize Orchestrator Appliance의 하드웨어 요구 사항](#)
- [vRealize Orchestrator 최대 확장성](#)
- [vRealize Orchestrator 포트 및 끝점](#)
- [vRealize Orchestrator에서 지원하는 브라우저](#)
- [국제화 및 현지화 지원 수준](#)

## vRealize Orchestrator Appliance 구성 요소

vRealize Orchestrator Appliance는 컨테이너에서 실행되는 Photon 기반 가상 장치입니다.

vRealize Orchestrator Appliance에는 다음과 같은 구성 요소가 포함됩니다.

- 인프라 수준 Kubernetes 계층.
- 미리 구성된 PostgreSQL 데이터베이스.
- 코어 vRealize Orchestrator 서비스: 서버 서비스, 제어 센터 서비스 및 오케스트레이션 UI 서비스.

기본 vRealize Orchestrator Appliance 데이터베이스 구성은 즉시 사용할 수 있습니다.

---

**참고** 운영 환경에서 vRealize Orchestrator Appliance를 사용하려면 vRealize Orchestrator 서버가 vRealize Automation 또는 vSphere를 통해 인증하도록 구성해야 합니다. [독립형 vRealize Orchestrator 서버 구성](#)의 내용을 참조하십시오.

---

## vRealize Orchestrator Appliance의 하드웨어 요구 사항

vRealize Orchestrator Appliance는 컨테이너에서 실행되는 미리 구성된 Photon 기반 가상 시스템입니다. 해당 장치를 배포하기 전에 시스템이 최소 하드웨어 요구 사항을 충족하는지 확인합니다.

vRealize Orchestrator Appliance의 하드웨어 요구 사항은 다음과 같습니다.

- CPU 4개
- 12 GB 메모리
- 200 GB 하드 디스크

기본 메모리 크기를 줄이지 마십시오.vRealize Orchestrator 서버에는 8GB 이상의 사용 가능한 메모리가 필요합니다.

## vRealize Orchestrator 최대 확장성

확장성 제한 표에는 vRealize Orchestrator 8.x 배포에서 권장되는 최대값이 요약되어 있습니다.

구성 요소	확장 목표	추가 정보
가상 시스템	35,000	
vCenter Server 연결	10	<a href="#">vCenter Server</a> 설정의 내용을 참조하십시오.
클러스터의 능동 노드	3	<a href="#">vRealize Orchestrator 클러스터 구성</a> 의 내용을 참조하십시오.
동시에 실행 중인 워크플로	노드당 300	<a href="#">워크플로 실행 속성 구성</a> 의 내용을 참조하십시오.
실행 대기 중인 워크플로	노드당 10,000	
보존된 워크플로 실행	노드당 100	
로그 이벤트 만료 일수	15	

## vRealize Orchestrator 포트 및 끝점

vRealize Orchestrator Kubernetes 서비스에는 두 개의 끝점과 여러 개의 기본 네트워크 포트가 포함됩니다.

### vRealize Orchestrator 네트워크 포트

포트 443을 통해 vRealize Orchestrator에 액세스할 수 있습니다. 443 포트는 설치 중에 생성되는 자체 서명된 인증서로 보안되며 사용자가 교체할 수 없습니다. 외부 로드 밸런서를 사용하는 경우 포트 443에서 밸런싱되도록 설정해야 합니다.

모든 vRealize Orchestrator 포트를 보려면 [포트 및 프로토콜](#) 도구를 참조하십시오.

## vRealize Orchestrator 끝점

다음 끝점에서 vRealize Orchestrator 클라이언트 및 제어 센터 서비스에 액세스할 수 있습니다.

서비스	끝점
vRealize Orchestrator 클라이언트	<code>https://your_orchestrator_FQDN/orchestration-ui</code>
제어 센터	<code>https://your_orchestrator_FQDN/vco-controlcenter</code>

## vRealize Orchestrator에서 지원하는 브라우저

브라우저에서 vRealize Orchestrator를 지원하는지 확인합니다.

vRealize Orchestrator Client 및 제어 센터에 액세스하려면 다음 브라우저 중 하나를 사용해야 합니다.

- Microsoft Edge
- Mozilla Firefox
- Google Chrome

## 국제화 및 현지화 지원 수준

vRealize Orchestrator 제어 센터와 vRealize Orchestrator Client에는 영어 이외의 운영 체제에 대한 지원, 영어 이외의 데이터 형식, 제어 센터 및 클라이언트 사용자 인터페이스에 대한 다국어 지원이 포함됩니다.

vRealize Orchestrator 제어 센터와 vRealize Orchestrator Client는 영어 이외의 운영 체제, 영어 이외의 입력 및 출력 사용을 지원하며 영어 이외의 데이터 형식(예: 날짜, 시간 및 숫자)을 지원합니다.

vRealize Orchestrator와 vRealize Orchestrator Client 사용자 인터페이스는 다음과 같은 언어로 현지화되었습니다.

- 스페인어
- 프랑스어
- 독일어
- 중국어 번체
- 중국어 간체
- 한국어
- 일본어
- 이탈리아어
- 네덜란드어
- 포르투갈어(브라질)

- 러시아어

# vRealize Orchestrator 구성 요소 설정

## 3

vRealize Orchestrator Appliance를 다운로드하고 배포하면 vRealize Orchestrator 서버가 미리 구성됩니다. 배포 후 서비스가 자동으로 시작됩니다.

vRealize Orchestrator 설정의 가용성 및 확장성을 강화하려면 다음 지침을 따르십시오.

- 인증 제공자를 설치 및 구성하고 vRealize Orchestrator가 이 제공자와 작동하도록 구성합니다. [독립형 vRealize Orchestrator 서버 구성](#)의 내용을 참조하십시오.
- 클러스터링된 vRealize Orchestrator 환경의 경우, 로드 밸런싱 서버를 설치 및 구성하고 vRealize Orchestrator 서버 간에 워크로드를 분산하도록 구성합니다.

본 장은 다음 항목을 포함합니다.

- [vCenter Server 설정](#)
- [인증 방법](#)

## vCenter Server 설정

vRealize Orchestrator 설정에서 vCenter Server 인스턴스의 수를 늘리면 vRealize Orchestrator가 더 많은 세션을 관리하게 됩니다. 활성 세션이 너무 많으면 10개 이상의 vCenter Server 연결이 발생할 때 vRealize Orchestrator에서 시간 초과가 발생할 수 있습니다.

지원되는 vCenter Server 버전 목록은 [VMware 제품 상호 운용성 매트릭스](#)를 참조하십시오.

---

**참고** 네트워크의 대역폭과 지연 시간이 충분한 경우, vRealize Orchestrator 설정의 여러 가상 시스템에서 여러 vCenter Server 인스턴스를 실행할 수 있습니다. vRealize Orchestrator와 vCenter Server 간의 통신을 개선하기 위해 LAN을 사용 중인 경우 반드시 100Mb 회선을 사용해야 합니다.

---

## 인증 방법

사용자 권한을 인증 및 관리하려면 vRealize Orchestrator가 vRealize Automation 또는 vSphere 서버 인스턴스에 연결되어 있어야 합니다.



vRealize Orchestrator Appliance를 다운로드하고 배포할 때 vRealize Automation 또는 vSphere 인증으로 서버를 구성해야 합니다. [독립형 vRealize Orchestrator 서버 구성](#)의 내용을 참조하십시오.

---

**참고** vRealize Automation을 사용한 vRealize Orchestrator 8.x 인증은 vRealize Automation 8.x에서만 지원됩니다.

---

# vRealize Orchestrator 설치

# 4

vRealize Orchestrator는 서버 구성 요소와 클라이언트 구성 요소로 이루어집니다.

vRealize Orchestrator를 사용하려면 vRealize Orchestrator Appliance를 배포하고 vRealize Orchestrator 서버를 구성해야 합니다.

vRealize Orchestrator 제어 센터를 사용하여 기본 vRealize Orchestrator 구성 설정을 변경할 수 있습니다.

본 장은 다음 항목을 포함합니다.

- [vRealize Orchestrator Appliance 다운로드 및 배포](#)

## vRealize Orchestrator Appliance 다운로드 및 배포

vRealize Orchestrator 콘텐츠 및 서비스에 액세스하려면 먼저 vRealize Orchestrator Appliance를 다운로드하고 배포해야 합니다.

사전 요구 사항

- 실행 중인 vCenter Server 인스턴스가 있는지 확인합니다. vCenter Server 버전은 6.0 이상이어야 합니다.
- vRealize Orchestrator Appliance를 배포하는 호스트가 하드웨어 최소 요구 사항을 충족하는지 확인합니다. [vRealize Orchestrator Appliance의 하드웨어 요구 사항](#)의 내용을 참조하십시오.
- 시스템이 분리되어 있고 인터넷에 연결되지 않은 경우 VMware 웹 사이트에서 해당 장치에 대한 .ova 파일을 다운로드해야 합니다.

절차

- 1 vSphere Web Client에 **관리자**로 로그인합니다.
- 2 데이터 센터, 폴더, 클러스터, 리소스 풀 또는 호스트와 같이 가상 시스템의 유효한 상위 개체인 인벤토리 개체를 선택합니다.
- 3 **작업 > OVF 템플릿 배포**를 선택합니다.
- 4 .ova 파일의 URL 또는 파일 경로를 입력하고 **다음**을 클릭합니다.

- 5 vRealize Orchestrator Appliance의 이름 및 위치를 입력하고 **다음**을 클릭합니다.
- 6 호스트, 클러스터, 리소스 풀 또는 vApp을 장치가 실행되는 대상으로 선택하고 **다음**을 클릭합니다.
- 7 배포 세부 정보를 검토하고 **다음**을 클릭합니다.
- 8 라이선스 계약 조건에 동의하고 **다음**을 클릭합니다.
- 9 vRealize Orchestrator Appliance에 사용할 스토리지 형식을 선택합니다.

형식	설명
느리게 비워지는 씩 프로비저닝	기본 씩 형식의 가상 디스크를 생성합니다. 가상 디스크에 필요한 공간은 가상 디스크 생성 중에 할당됩니다. 물리적 디바이스에 남아 있는 데이터는 생성 동안에 지워지지 않지만 나중에 가상 시스템에서 처음으로 쓰는 경우, 해당 데이터는 요구대로 비워집니다.
빠르게 비워지는 씩 프로비저닝	Fault Tolerance와 같은 클러스터링 기능을 지원합니다. 가상 디스크에 필요한 공간은 가상 디스크 생성 중에 할당됩니다. 물리적 디바이스에 남아 있는 모든 데이터는 가상 디스크를 생성하는 동안 비워집니다. 다른 형식의 디스크를 생성하는 것보다 이 형식의 디스크를 생성하는 것이 더 오래 걸릴 수도 있습니다.
썸 프로비저닝된 형식	하드 디스크 공간을 절약합니다. 썸 디스크의 경우 선택하는 디스크 크기 값에 기반하여 디스크가 필요로 하는 만큼의 데이터스토어 공간을 프로비저닝합니다. 썸 디스크는 먼저 작은 크기부터 시작합니다. 초기 작업을 위해 이 디스크에 필요한 데이터스토어 공간 만큼의 크기만 사용합니다.

- 10 **다음**을 클릭합니다.
- 11 네트워크 설정을 구성하고 **루트** 암호를 입력합니다.

vRealize Orchestrator Appliance의 네트워크 설정을 구성할 때는 IPv4 프로토콜을 사용해야 합니다. DHCP 및 정적 네트워크 구성 모두에 대해 vRealize Orchestrator Appliance의 FQDN(정규화된 도메인 이름)을 추가해야 합니다.

배포된 vRealize Orchestrator Appliance의 셸에 표시된 호스트 이름이 *photon-machine*인 경우 위의 네트워크 구성 요구 사항이 충족되지 않습니다.

- 12 (선택 사항) vRealize Orchestrator Appliance에 대한 추가 네트워크 설정(예: SSH 액세스 활성화)을 구성합니다.

**참고** Kubernetes 네트워크를 구성할 때 내부 클러스터 CIDR 및 내부 서비스 CIDR의 값은 1024개 이상의 호스트를 허용해야 합니다. 이러한 요구 사항 때문에 네트워크 마스크 값은 22 이하여야 합니다. 22보다 큰 네트워크 마스크 값이 유효하지 않습니다. Kubernetes 네트워크 속성의 기본값은 다음과 같습니다.

Kubernetes network property	Default value	Property description
Kubernetes 내부 클러스터 CIDR	10.244.0.0/22	Kubernetes 클러스터 내부에서 실행되는 포드에 사용되는 CIDR입니다.
Kubernetes 내부 서비스 CIDR	10.244.4.0/22	Kubernetes 클러스터 내부의 Kubernetes 서비스에 사용되는 CIDR입니다.

**참고** 배포 후에 Kubernetes CIDR 네트워크 속성을 변경할 수도 있습니다. [vRealize Orchestrator Kubernetes CIDR 구성](#)의 내용을 참조하십시오.

- 13 (선택 사항)** vRealize Orchestrator Appliance에 대해 FIPS 모드를 사용하도록 설정하려면 **FIPS 모드**를 **엄격**으로 설정합니다.

**참고** FIPS 140-2 사용은 새로운 vRealize Orchestrator 환경에서만 지원됩니다. 환경에서 FIPS 모드를 사용하도록 설정하려면 설치 중에 설정해야 합니다.

- 14** 다음을 클릭합니다.

- 15 완료 준비** 페이지를 검토하고 **마침**을 클릭합니다.

결과

vRealize Orchestrator Appliance가 배포되었습니다.

다음에 수행할 작업

vRealize Orchestrator Appliance 명령줄에 **루트**로 로그인하고 정방향 또는 역방향 DNS 조회를 수행할 수 있는지 확인합니다.

- 정방향 DNS 조회를 수행하려면 `nslookup your_orchestrator_FQDN` 명령을 실행합니다. 이 명령은 vRealize Orchestrator Appliance IP 주소를 반환해야 합니다.
- 역방향 DNS 조회를 수행하려면 `nslookup your_orchestrator_IP` 명령을 실행합니다. 이 명령은 vRealize Orchestrator Appliance FQDN을 반환해야 합니다.

**참고** 배포 중에 SSH를 사용하도록 설정하지 않은 경우 vSphere Web Client의 가상 시스템 콘솔에서 DNS 조회를 수행할 수도 있습니다.

## vRealize Orchestrator Appliance의 전원을 켜고 홈 페이지 열기

독립형 vRealize Orchestrator Appliance를 사용하려면 먼저 전원을 켜야 합니다.

절차

- 1 vSphere Web client에 **관리자**로 로그인합니다.
- 2 vRealize Orchestrator Appliance를 마우스 오른쪽 버튼으로 클릭하고 **전원 > 전원 켜기**를 선택합니다.

- 3 웹 브라우저에서 OVA 배포 중에 구성한 vRealize Orchestrator Appliance 가상 시스템의 호스트 주소로 이동합니다.

`https://your_orchestrator_FQDN/vco.`

## vRealize Orchestrator Appliance에 대한 SSH 액세스 사용 또는 사용 안 함

vRealize Orchestrator Appliance에 대한 SSH 액세스를 사용하거나 사용하지 않도록 설정할 수 있습니다.

사전 요구 사항

- vRealize Orchestrator Appliance를 다운로드하고 배포합니다.
- vRealize Orchestrator Appliance가 가동되어 실행 중인지 확인합니다.

절차

- 1 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 SSH 액세스를 사용하도록 설정하려면 `/usr/bin/toggle-ssh enable` 명령을 실행합니다.
- 3 SSH 액세스를 사용하지 않도록 설정하려면 `/usr/bin/toggle-ssh disable` 명령을 실행합니다.

## 초기 구성

# 5

vRealize Orchestrator로 작업 자동화 및 시스템과 애플리케이션 관리를 시작하기 전에 vRealize Orchestrator 제어 센터를 사용하여 외부 인증 제공자를 구성해야 합니다. 라이선스 및 인증서 정보 관리, 플러그인 설치 및 vRealize Orchestrator 클러스터 상태 모니터링과 같은 추가 구성 작업에도 vRealize Orchestrator 제어 센터를 사용할 수 있습니다.

본 장은 다음 항목을 포함합니다.

- 독립형 vRealize Orchestrator 서버 구성
- 라이선스 기반 vRealize Orchestrator 기능 사용 설정
- vRealize Orchestrator 데이터베이스 연결
- 인증서 관리
- vRealize Orchestrator Plug-in 구성
- vRealize Orchestrator 고가용성
- 고객 환경 개선 프로그램 구성

### 독립형 vRealize Orchestrator 서버 구성

vRealize Orchestrator Appliance는 미리 구성된 Photon 기반 가상 시스템이지만 vRealize Orchestrator 제어 센터 및 vRealize Orchestrator Client의 전체 기능에 액세스하려면 먼저 인증 제공자를 구성해야 합니다.

### vRealize Automation 인증을 사용하여 독립형 vRealize Orchestrator 서버 구성

vRealize Orchestrator Appliance를 사용하도록 준비하려면 호스트 설정과 인증 제공자를 구성해야 합니다. vRealize Automation으로 인증하도록 vRealize Orchestrator를 구성할 수 있습니다. vRealize Automation 8.x에서 vRealize Automation 인증을 사용합니다.

사전 요구 사항

- 최신 버전의 vRealize Orchestrator Appliance를 다운로드하여 배포합니다. [vRealize Orchestrator Appliance 다운로드 및 배포](#)의 내용을 참조하십시오.

- vRealize Automation 8.x를 설치 및 구성하고 vRealize Automation 서버가 실행되고 있는지 확인합니다. vRealize Automation 설명서를 참조하십시오.

---

**중요** vRealize Automation 인증 제공자의 제품 버전은 vRealize Orchestrator 배포의 제품 버전과 일치해야 합니다. 예를 들어 vRealize Orchestrator 8.4 배포를 인증하려면 vRealize Automation 8.4 배포를 사용해야 합니다.

---

클러스터를 생성하려는 경우 다음 작업을 수행하십시오.

- vRealize Orchestrator의 여러 인스턴스에서 트래픽을 분배하기 위해 로드 밸런서를 설정합니다. [VMware vRealize Orchestrator 8.x 로드 밸런싱 가이드](#)를 참조하십시오.

#### 절차

- 1 제어 센터에 액세스하여 구성 마법사를 시작합니다.
  - a `https://your_orchestrator_FQDN/vco-controlcenter`로 이동합니다.
  - b OVA 배포 동안 입력한 암호를 사용하여 **root**로 로그인합니다.
- 2 인증 제공자를 구성합니다.
  - a **인증 제공자 구성** 페이지의 **인증 모드** 드롭다운 메뉴에서 **vRealize Automation**을 선택합니다.
  - b **호스트 주소** 텍스트 상자에서 vRealize Automation 호스트 주소를 입력하고 **연결**을 클릭합니다.  
vRealize Automation 호스트 주소는 `https://your_vra_hostname` 형식이어야 합니다.
  - c **인증서 수락**을 클릭합니다.
  - d vRealize Orchestrator를 구성할 vRealize Automation 조직 소유자의 자격 증명을 입력합니다.  
**등록**을 클릭합니다.
  - e **변경 내용 저장**을 클릭합니다.  
구성이 성공적으로 저장되었다는 메시지가 표시됩니다.

#### 결과

vRealize Orchestrator 서버 구성이 성공적으로 완료되었습니다.

다음에 수행할 작업

- **CSP가 라이선싱** 페이지에서 구성된 라이선스 제공자인지 확인합니다.
- **구성 검증** 페이지에서 노드가 올바르게 구성되었는지 확인합니다.

---

**참고** 인증 제공자의 구성에 따라 2분 후에 vRealize Orchestrator 서버가 자동으로 다시 시작됩니다. 인증 직후에 구성을 확인하면 잘못된 구성 상태가 반환될 수 있습니다.

---

## vSphere 인증을 사용하여 독립형 vRealize Orchestrator 서버 구성

vSphere 인증 모드를 사용하여 vRealize Orchestrator 서버를 vCenter Single Sign-On 서버에 등록합니다. vCenter Single Sign-On 인증을 vCenter Server 6.0 이상과 함께 사용합니다.

## 사전 요구 사항

- 최신 버전의 vRealize Orchestrator Appliance를 다운로드하여 배포합니다. [vRealize Orchestrator Appliance 다운로드 및 배포](#)의 내용을 참조하십시오.
- vCenter Single Sign-On을 실행 중인 vCenter Server를 설치하고 구성합니다. vSphere 설명서를 참조하십시오.

클러스터를 생성하려는 경우 다음 작업을 수행하십시오.

- vRealize Orchestrator의 여러 인스턴스에서 트래픽을 분배하기 위해 로드 밸런서를 설정합니다. [VMware vRealize Orchestrator 8.x 로드 밸런싱 가이드](#)를 참조하십시오.

## 절차

- 1 제어 센터에 액세스하여 구성 마법사를 시작합니다.
  - a [https://your\\_orchestrator\\_FQDN/vco-controlcenter](https://your_orchestrator_FQDN/vco-controlcenter)로 이동합니다.
  - b OVA 배포 동안 입력한 암호를 사용하여 **root**로 로그인합니다.
- 2 인증 제공자를 구성합니다.
  - a **인증 제공자 구성** 페이지에서 **vSphere(인증 모드 드롭다운 메뉴에서)**를 선택합니다.
  - b **호스트 주소** 텍스트 상자에 vCenter Single Sign-On을 포함하는 Platform Services Controller 인스턴스의 FQDN(정규화된 도메인 이름) 또는 IP 주소를 입력하고 **연결**을 클릭합니다.

---

**참고** 외부 Platform Services Controller 인스턴스 또는 로드 밸런서의 뒤에 있는 여러 Platform Services Controller 인스턴스를 사용하는 경우 vCenter Single Sign-On 도메인을 공유하는 모든 Platform Services Controller의 인증서를 수동으로 가져와야 합니다.

---

**참고** 다른 vSphere Client를 구성된 vRealize Orchestrator 환경과 통합하려면 vRealize Orchestrator에 등록된 것과 동일한 Platform Services Controller를 사용하도록 vSphere를 구성해야 합니다. 고가용성 vRealize Orchestrator 환경의 경우, vRealize Orchestrator 로드 밸런서 서버 뒤에 PCS 인스턴스를 복제해야 합니다.

---

- c 인증 제공자의 인증서 정보를 검토하고 **인증서 수락**을 클릭합니다.
- d vCenter Single Sign-On 도메인에 대한 로컬 관리자 계정의 자격 증명을 입력합니다. **등록**을 클릭합니다.  
기본적으로 이 계정은 **administrator@vsphere.local**이며 기본 테넌트 이름은 **vsphere.local**입니다.
- e **관리자 그룹** 텍스트 상자에서 관리자 그룹의 이름을 입력하고 **검색**을 클릭합니다.  
예를 들어 **vsphere.local\vcadmins**와 같습니다.



f 사용할 관리 그룹을 선택합니다.

g **변경 내용 저장**을 클릭합니다.

구성이 성공적으로 저장되었다는 메시지가 표시됩니다.

## 결과

vRealize Orchestrator 서버 구성이 성공적으로 완료되었습니다.

다음에 수행할 작업

- **CIS가 라이선싱** 페이지에서 구성된 라이선스 제공자인지 확인합니다.
- **구성 검증** 페이지에서 노드가 올바르게 구성되었는지 확인합니다.

---

**참고** 인증 제공자의 구성에 따라 2분 후에 vRealize Orchestrator 서버가 자동으로 다시 시작됩니다. 인증 직후에 구성을 확인하면 잘못된 구성 상태가 반환될 수 있습니다.

---

## 라이선스 기반 vRealize Orchestrator 기능 사용 설정

특정 vRealize Orchestrator 기능에 대한 액세스는 vRealize Orchestrator 배포에 적용된 라이선스에 기 반합니다.

인증 후에 해당 인증 제공자에 기반하여 vRealize Orchestrator 인스턴스에 라이선스가 할당됩니다. 라이선스는 다음 vRealize Orchestrator 기능에 대한 액세스를 제어합니다.

- Git 통합
- 역할 관리
- 다중 언어 지원(Python, Node.js 및 PowerShell)

제어 센터의 **라이선스** 페이지에서 vRealize Orchestrator 서버의 라이선스를 수동으로 변경할 수 있습니다.

---

**참고** 라이선스 유형에 관계없이 동일한 라이선스를 적용할 수 있는 vRealize Orchestrator 배포 수에는 제한이 없습니다. vRealize Automation 라이선스의 경우 배포되고 구성된 vRealize Automation 환경이 필요하지 않습니다.

---

인증	라이선스	Git 통합	역할 관리	다중 언어 지원
vSphere	vSphere vCloud Suite Standard	아니요	아니요	아니요
vSphere	vRealize Automation vRealize Suite Advanced 또는 Enterprise vCloud Suite Advanced 또는 Enterprise	예	예	예
vRealize Automation	vRealize Automation vRealize Suite Advanced 또는 Enterprise vCloud Suite Advanced 또는 Enterprise	예	vRealize Orchestrator 인증에 사용되는 vRealize Automation 인스턴스에서 역할이 관리됩니다.	예

**참고** vRealize Suite Standard 라이선스에는 vRealize Automation이 포함되지 않으므로 vRealize Orchestrator 기능에 대한 액세스를 지원하지 않습니다.

## vRealize Orchestrator 데이터베이스 연결

vRealize Orchestrator 서버에는 데이터 저장을 위한 데이터베이스가 필요합니다.

배포된 vRealize Orchestrator Appliance에는 vRealize Orchestrator 서버에서 데이터를 저장하는 데 사용하는 미리 구성된 PostgreSQL 데이터베이스가 포함되어 있습니다.

사용자는 PostgreSQL 데이터베이스에 액세스할 수 없습니다.

## 인증서 관리

특정 서버용으로 발급되고 서버 공개 키에 대한 정보가 포함된 인증서를 사용하여 vRealize Orchestrator에서 생성된 모든 요소에 서명하고 신뢰성을 보장할 수 있습니다. 클라이언트가 서버에서 일반적으로 패키지인 요소를 수신하면 클라이언트는 사용자의 ID를 확인하고 서명의 신뢰 여부를 결정합니다.

### ■ vRealize Orchestrator 인증서 관리

vRealize Orchestrator 인증서는 vRealize Orchestrator 제어 센터 또는 vRealize Orchestrator Client에 있는 **인증서** 페이지를 통해 *ssl\_Trust\_Manager* 태그가 지정된 워크플로를 사용하여 관리할 수 있습니다.

## vRealize Orchestrator 인증서 관리

vRealize Orchestrator 인증서는 vRealize Orchestrator 제어 센터 또는 vRealize Orchestrator Client에 있는 **인증서** 페이지를 통해 *ssl\_Trust\_Manager* 태그가 지정된 워크플로를 사용하여 관리할 수 있습니다.

### 인증서를 Orchestrator 신뢰 저장소에 가져오기

vRealize Orchestrator 제어 센터는 보안 연결을 사용하여 vCenter Server, 관계형 데이터베이스 관리 시스템(RDBMS), LDAP, Single Sign-On 및 기타 서버와 통신합니다. URL 또는 PEM-인코드 파일에서 필수 TLS 인증서를 가져올 수 있습니다. 서버 인스턴스에 대해 TLS 연결을 사용할 때마다 **인증서** 페이지의 **신뢰할 수 있는 인증서** 탭에서 해당 인증서를 가져오고 해당 TLS 인증서를 가져와야 합니다.

vRealize Orchestrator에서 URL 주소 또는 PEM-인코드 파일로부터 TLS 인증서를 로드할 수 있습니다.

옵션	설명
URL 또는 프록시 URL에서 가져오기	원격 서버의 URL입니다. <b>https://your_server_IP_address</b> 또는 <b>your_server_IP_address:port</b>
파일에서 가져오기	PEM 인코딩된 인증서 파일의 경로입니다. <b>참고</b> vRealize Orchestrator Client에서 <b>파일에서 신뢰할 수 있는 인증서 가져오기</b> 워크플로를 실행하여 신뢰할 수 있는 인증서를 가져올 수도 있습니다. 이 워크플로를 통해 가져온 파일은 DER 인코딩 형식이어야 합니다.

인증서 가져오기에 대한 자세한 내용은 **제어 센터를 사용하여 신뢰할 수 있는 인증서 가져오기** 항목을 참조하십시오.

### 패키지 서명 인증서

vRealize Orchestrator 서버에서 내보낸 패키지는 디지털로 서명되어 있습니다. 패키지 서명에 사용될 새 인증서를 가져오고, 내보내고 또는 생성합니다. 패키지 서명 인증서는 암호화된 통신 및 Orchestrator 패키지 서명을 보장하는 데 사용되는 디지털 ID 양식입니다.

vRealize Orchestrator Appliance는 장치의 네트워크 설정을 기준으로 자동 생성된 패키지 서명 인증서를 포함합니다. 장치의 네트워크 설정이 변경된 경우 수동으로 새 패키지 서명 인증서를 생성해야 합니다. 새 패키지 서명 인증서를 생성하면, 이후에 내보낸 모든 패키지는 새 인증서로 서명됩니다.

### vRealize Orchestrator에 대한 사용자 지정 TLS 인증서 생성

vRealize Orchestrator Appliance를 사용하여 환경에 대한 새 TLS 인증서를 생성하거나 기존 사용자 지정 인증서를 설정할 수 있습니다.

vRealize Orchestrator Appliance는 장치의 네트워크 설정을 기준으로 자동 생성된 TLS(Trusted Layer Security) 인증서를 포함합니다. 장치의 네트워크 설정이 변경된 경우 수동으로 새 인증서를 생성해야 합니다. 인증서 체인을 생성하여 암호화된 통신을 보장하고 패키지에 서명을 제공할 수 있습니다. 그러나, 수신자는 사용자로 할당된 타사 제공자가 아니라 사용자의 서버에 의해 실제로 발급되고 자체 서명된 패키지인지 확인할 수 없습니다. 서버 ID를 증명하려면 CA(인증 기관)에 의해 서명된 인증서를 사용합니다.

vRealize Orchestrator는 환경에 고유한 서버 인증서를 생성합니다. 개인 키는 vRealize Orchestrator 데이터베이스의 `vmo_keystore` 테이블에 저장됩니다.

---

**참고** 기존 사용자 지정 TLS 인증서를 사용하도록 vRealize Orchestrator Appliance를 구성하려면 [vRealize Orchestrator에 대한 사용자 지정 TLS 인증서 설정](#)의 내용을 참조하십시오.

---

#### 사전 요구 사항

vRealize Orchestrator Appliance에 대한 SSH 액세스가 가능하도록 설정되었는지 확인합니다. [vRealize Orchestrator Appliance에 대한 SSH 액세스 사용 또는 사용 안 함](#)의 내용을 참조하십시오.

#### 절차

- 1 SSH를 통해 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 `vracli certificate ingress --generate auto --set stdin` 명령을 실행합니다.
- 3 사용자 지정 인증서를 vRealize Orchestrator Appliance에 적용하려면 배포 스크립트를 실행합니다.
  - a `/opt/scripts/` 디렉토리로 이동합니다.

```
cd /opt/scripts/
```

- b `./deploy.sh` 스크립트를 실행합니다.

---

**중요** 배포 스크립트를 중단하지 마십시오. 스크립트 실행이 완료되면 다음 메시지가 표시됩니다.

```
Prelude가 배포되었습니다. 액세스하려면 your_orchestrator_address로 이동합니다.
```

---

#### 다음에 수행할 작업

새 인증서 체인이 적용되었는지 확인하려면 `vracli certificate ingress --list` 명령을 실행합니다.

### vRealize Orchestrator에 대한 사용자 지정 TLS 인증서 설정

vRealize Orchestrator Appliance에 대한 사용자 지정 TLS 인증서를 설정합니다.

vRealize Orchestrator Appliance는 장치의 네트워크 설정을 기준으로 자동 생성된 TLS(Trusted Layer Security) 인증서를 포함합니다.

기존 사용자 지정 TLS 인증서를 사용하도록 vRealize Orchestrator Appliance를 구성할 수 있습니다. 로컬 시스템에서 vRealize Orchestrator Appliance로 관련 PEM 파일을 가져와서 인증서를 설정할 수 있습니다. 인증서 체인을 vRealize Orchestrator Appliance에 직접 복사하여 사용자 지정 TLS 인증서를 설정할 수도 있습니다. 두 절차 모두 `./deploy.sh` 스크립트를 실행해야 vRealize Orchestrator 배포에서 새 TLS 인증서를 사용할 수 있습니다.

새 사용자 지정 TLS 인증서 생성에 대한 자세한 내용은 [vRealize Orchestrator에 대한 사용자 지정 TLS 인증서 생성](#) 항목을 참조하십시오.

## 사전 요구 사항

- vRealize Orchestrator Appliance에 대한 SSH 액세스가 가능하도록 설정되었는지 확인합니다.  
vRealize Orchestrator Appliance에 대한 SSH 액세스 사용 또는 사용 안 함의 내용을 참조하십시오.
- TLS 인증서가 포함된 PEM 파일에 다음 구성 요소가 설정된 순서대로 포함되어 있는지 확인합니다.
  - a 인증서의 개인 키.
  - b 기본 인증서입니다.
  - c 해당되는 경우, CA(인증 기관) 중간 인증서 또는 인증서.
  - d 루트 CA 인증서.

예를 들어 TLS 인증서는 다음과 같은 구조로 이루어질 수 있습니다.

```
-----BEGIN RSA PRIVATE KEY-----
<Private Key>
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
<Primary TLS certificate>
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
<Intermediate certificate>
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
<Root CA certificate>
-----END CERTIFICATE-----
```

## 절차

- 1 PEM 파일을 vRealize Orchestrator Appliance로 가져와서 인증서를 설정합니다.
  - a SSH 셸에서 보안 복사(SCP) 명령을 실행하여 로컬 시스템에서 인증서 PEM을 가져옵니다.  
Linux의 경우 터미널 SCP 명령을 사용할 수 있습니다.

```
scp ~/PEM_local_filepath/your_cert_file.PEM root@orchestrator_FQDN_or_IP:/
PEM_orchestrator_filepath/your_cert_file.PEM
```

Windows의 경우 PuTTY 클라이언트 PSCP 명령을 사용할 수 있습니다.

```
pscp C:\PEM_local_filepath\your_cert_file.PEM root@<orchestrator_FQDN_or_IP>:/
PEM_orchestrator_filepath/your_cert_file.PEM
```

- b SSH를 통해 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
  - c `vracli certificate ingress --set your_cert_file.PEM` 명령을 실행합니다.

- 2 (선택 사항) 인증서 체인을 장치에 직접 복사하여 인증서를 설정합니다.
  - a SSH를 통해 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
  - b `vracli certificate ingress --set stdin` 명령을 실행합니다.
  - c 인증서 체인을 복사하여 붙여넣고 **Ctrl+D**를 누릅니다.
- 3 새 TLS 인증서를 적용하려면 배포 스크립트를 실행합니다.
  - a `/opt/scripts/` 디렉토리로 이동합니다.

```
cd /opt/scripts/
```

- b `./deploy.sh` 스크립트를 실행합니다.

**중요** 배포 스크립트를 중단하지 마십시오. 스크립트 실행이 완료되면 다음 메시지가 표시됩니다.

Prelude가 배포되었습니다. 액세스하려면 [https://your\\_orchestrator\\_FQDN](https://your_orchestrator_FQDN)으로 이동합니다.

## 결과

vRealize Orchestrator Appliance에 대해 사용자 지정 TLS 인증서를 설정했습니다.

다음에 수행할 작업

새 인증서 체인이 적용되었는지 확인하려면 `vracli certificate ingress --list` 명령을 실행합니다.

## 제어 센터를 사용하여 신뢰할 수 있는 인증서 가져오기

다른 서버와 안전하게 통신하려면 vRealize Orchestrator 서버가 해당 서버의 ID를 확인할 수 있어야 합니다. 이를 위해 원격 엔터티의 TLS 인증서를 vRealize Orchestrator 신뢰 저장소로 가져와야 할 수 있습니다. 인증서를 신뢰하기 위해 특정 URL에 대한 연결을 설정하는 방법으로 인증서를 신뢰 저장소로 가져오거나 PEM- 인코드 파일 형태로 직접 가져올 수 있습니다.

## 절차

- 1 제어 센터에 **root**로 로그인합니다.
- 2 **인증서** 페이지로 이동합니다.
- 3 **신뢰할 수 있는 인증서**를 선택하고 **가져오기**를 클릭합니다.
- 4 파일에서 인증서를 가져오려면 **PEM-인코드 파일에서 가져오기**를 선택합니다.
- 5 인증서 파일을 찾아보고 **가져오기**를 클릭합니다.
- 6 URL 주소에서 인증서를 가져오려면 **URL에서 가져오기**를 선택합니다.
- 7 인증서가 저장된 URL 주소를 입력하고 **가져오기**를 클릭합니다.

## 결과

원격 서버 인증서를 vRealize Orchestrator 신뢰 저장소로 가져오기가 완료되었습니다.

## vRealize Orchestrator Plug-in 구성

vRealize Orchestrator Appliance는 미리 설치된 기본 플러그인 라이브러리에 대한 액세스를 제공합니다. 기본 vRealize Orchestrator 플러그인은 vRealize Orchestrator Client에서 실행되는 플러그인 관련 워크플로를 통해 구성됩니다.

기본 vRealize Orchestrator 플러그인에는 구성 워크플로가 있습니다. vRealize Orchestrator Client에서 이러한 워크플로를 실행하여 관리용 끝점을 등록할 수 있습니다.

구성 워크플로에는 *Configuration* 태그가 있습니다. 예를 들어 AMQP 브로커 및 구독을 관리하는 데 사용되는 워크플로에 액세스하려면 워크플로 라이브러리의 검색 텍스트 상자에 *AMQP* 및 *Configuration* 태그를 입력합니다.

## vRealize Orchestrator 플러그인 관리

vRealize Orchestrator 제어 센터의 **플러그인 관리** 페이지에서 vRealize Orchestrator에 설치된 모든 플러그인 목록을 확인하고 기본 관리 작업을 수행할 수 있습니다.

### 플러그인 설치 또는 업그레이드

vRealize Orchestrator 플러그인을 사용하여 vRealize Orchestrator 서버를 다른 소프트웨어 제품과 통합할 수 있습니다. vRealize Orchestrator는 사전 설치된 기본 플러그인 집합이 함께 제공됩니다. 사용자 지정 플러그인을 설치하여 vRealize Orchestrator 플랫폼의 기능을 한층 확장할 수 있습니다.

vRealize Orchestrator의 **플러그인 관리** 페이지에서 플러그인을 설치하거나 업그레이드할 수 있습니다. 사용할 수 있는 파일 확장명은 **.vmoapp**입니다.

vRealize Orchestrator 플러그인 설치 또는 업그레이드에 대한 자세한 내용은 [vRealize Orchestrator 플러그인 설치 또는 업데이트](#) 항목을 참조하십시오.

### 플러그인 로깅 수준 변경

vRealize Orchestrator의 로깅 수준을 변경하는 대신 특정 플러그인에 대해서만 로깅 수준을 변경할 수 있습니다.

### 플러그인 사용 안 함

플러그인 이름 옆에 있는 **플러그인 사용** 옵션을 선택 취소하여 플러그인을 사용하지 않도록 설정할 수 있습니다.

이 작업은 플러그인 파일을 제거하지 않습니다. vRealize Orchestrator에서 플러그인 설치 제거에 대한 자세한 내용은 [플러그인 삭제](#) 항목을 참조하십시오.

## vRealize Orchestrator 플러그인 설치 또는 업데이트

vRealize Orchestrator 제어 센터를 사용하여 타사 플러그인을 설치하거나 업데이트할 수 있습니다.

사전 요구 사항

플러그인의 `.dar` 또는 `.vmoapp` 파일을 다운로드합니다.

---

**참고** vRealize Orchestrator 플러그인에 선호하는 파일 형식은 `.vmoapp`입니다.

---

절차

- 1 제어 센터에 **root**로 로그인합니다.
- 2 **플러그인 관리** 페이지를 선택합니다.
- 3 **찾아보기**를 클릭하고 설치하거나 업데이트할 플러그인의 `.dar` 또는 `.vmoapp` 파일을 선택합니다.
- 4 **업로드**를 클릭합니다.
- 5 플러그인 정보를 검토하고 최종 사용자 라이선스 계약에 동의한 후(해당하는 경우) **설치**를 클릭합니다.

플러그인이 설치 또는 업데이트되고 vRealize Orchestrator 서버 서비스가 다시 시작됩니다.

다음에 수행할 작업

**플러그인 관리** 페이지에 올바른 플러그인 정보가 나열되어 있는지 확인합니다.

## 플러그인 삭제


제어 센터를 통해 vRealize Orchestrator Appliance에서 타사 플러그인을 삭제할 수 있습니다.

---

**참고** vRealize Orchestrator 8.0부터는 더 이상 vRealize Orchestrator Client에서 플러그인 패키지를 수동으로 삭제하지 않습니다.

---

절차

- 1 제어 센터에 **root**로 로그인합니다.
- 2 **플러그인 관리**를 선택합니다.
- 3 삭제할 플러그인을 찾아서 삭제 아이콘()을 클릭합니다.
- 4 플러그인을 삭제할지 확인하고 **삭제**를 클릭합니다.

결과

vRealize Orchestrator Appliance에서 플러그인을 삭제했습니다.

## vRealize Orchestrator 고가용성

vRealize Orchestrator 서비스의 가용성을 높이려면 클러스터의 여러 vRealize Orchestrator 서버 인스턴스를 공유 데이터베이스와 함께 시작합니다. vRealize Orchestrator는 클러스터의 일부로 작동하도록 구성되지 않는 한 단일 인스턴스로 작동합니다.



서버 및 플러그인 구성이 동일한 여러 vRealize Orchestrator 서버 인스턴스는 클러스터에서 함께 작동하며 단일 데이터베이스를 공유합니다.

모든 vRealize Orchestrator 서버 인스턴스는 하트비트를 교환하여 서로 통신합니다. 각 하트비트는 노드가 특정 시간 간격으로 클러스터의 공유 데이터베이스에 기록하는 타임 스탬프입니다. 네트워크 문제, 무응답 데이터베이스 서버 또는 과부하로 인해 vRealize Orchestrator 클러스터 노드의 응답이 중지될 수 있습니다. 활성 vRealize Orchestrator 서버 인스턴스가 페일오버 시간 초과 기간 내에 하트비트를 보내지 못하면 응답 없음으로 간주됩니다. 페일오버 시간 초과는 페일오버 하트비트 수와 하트비트 간격 값을 곱한 값과 동일합니다. 이는 신뢰할 수 없는 노드에 대한 정의로 사용되고 사용 가능한 리소스 및 운영 로드에 따라 사용자 지정할 수 있습니다.

데이터베이스와의 연결이 끊어지면 vRealize Orchestrator 노드는 대기 모드로 전환되고 데이터베이스 연결이 복구되기 전까지 같은 모드를 유지합니다. 클러스터의 다른 노드는 스크립팅 가능한 작업 또는 워크플로 호출과 같은 완료되지 않은 지난 항목에서 모든 중단된 워크플로를 재개하여 활성 작업을 제어합니다.

vRealize Orchestrator 클러스터의 상태는 vRealize Orchestrator 제어 센터의 **Orchestrator 클러스터 관리** 페이지에서 모니터링할 수 있습니다. 이 페이지를 사용하여 클러스터 하트비트, 페일오버 하트비트 수 및 활성 vRealize Orchestrator 노드 수를 구성할 수도 있습니다.

## vRealize Orchestrator 최대 확장성

확장성 제한 표에는 vRealize Orchestrator 8.x 배포에서 권장되는 최대값이 요약되어 있습니다.

구성 요소	확장 목표	추가 정보
가상 시스템	35,000	
vCenter Server 연결	10	<a href="#">vCenter Server 설정</a> 의 내용을 참조하십시오.
클러스터의 능동 노드	3	<a href="#">vRealize Orchestrator 클러스터 구성</a> 의 내용을 참조하십시오.
동시에 실행 중인 워크플로	노드당 300	<a href="#">워크플로 실행 속성 구성</a> 의 내용을 참조하십시오.
실행 대기 중인 워크플로	노드당 10,000	
보존된 워크플로 실행	노드당 100	
로그 이벤트 만료 일수	15	

## vRealize Orchestrator 클러스터 구성

세 개의 노드를 배포하고 클러스터로 연결하여 고가용성으로 실행되도록 새 vRealize Orchestrator 배포를 구성할 수 있습니다.

vRealize Orchestrator 클러스터는 공통 PostgreSQL 데이터베이스를 공유하는 세 개의 vRealize Orchestrator 인스턴스로 구성됩니다. 구성된 vRealize Orchestrator 클러스터의 데이터베이스는 비동기 모드에서만 실행할 수 있습니다.

vRealize Orchestrator 클러스터를 생성하려면 클러스터의 기본 노드로 사용할 vRealize Orchestrator 인스턴스를 하나 선택해야 합니다. 기본 노드를 구성한 후 여기에 보조 노드를 조인합니다.

생성된 vRealize Orchestrator 클러스터는 자동 페일오버로 미리 구성됩니다.

---

**참고** 자동 페일오버가 실패하면 데이터베이스 데이터가 손실될 수 있습니다.

---

#### 사전 요구 사항

- 세 개의 독립형 vRealize Orchestrator 인스턴스를 다운로드하고 배포합니다. [vRealize Orchestrator Appliance 다운로드 및 배포](#)의 내용을 참조하십시오.

---

**참고** 클러스터링된 vRealize Orchestrator 환경을 생성하는 데 사용할 수 있는 권장 노드 수는 세 개입니다.

---

- 모든 vRealize Orchestrator 노드에 SSH 액세스를 사용하도록 설정되어 있는지 확인합니다. [vRealize Orchestrator Appliance에 대한 SSH 액세스 사용 또는 사용 안 함](#)의 내용을 참조하십시오.
- 로드 밸런서 서버를 구성합니다. [VMware vRealize Orchestrator 8.x 로드 밸런싱 가이드](#)를 참조하십시오.

#### 절차

##### 1 기본 노드를 구성합니다.

- SSH를 통해 기본 노드의 vRealize Orchestrator Appliance에 **root**로 로그인합니다.
- 클러스터 로드 밸런서 서버를 구성하려면 `vracli load-balancer set load_balancer_FQDN` 명령을 실행합니다.
- 기본 노드의 제어 센터에 로그인하고 **호스트 설정**을 선택합니다.
- 변경**을 클릭하고 연결된 로드 밸런서 서버의 호스트 주소를 설정합니다.
- 인증 제공자를 구성합니다. [독립형 vRealize Orchestrator 서버 구성](#)의 내용을 참조하십시오.

##### 2 보조 노드를 기본 노드에 조인합니다.

- SSH를 통해 보조 노드의 vRealize Orchestrator Appliance에 **root**로 로그인합니다.
- 보조 노드를 기본 노드에 조인하려면 `vracli cluster join primary_node_hostname_or_IP` 명령을 실행합니다.
- 기본 노드의 루트 암호를 입력합니다.
- 다른 보조 노드에 대해서도 이 절차를 반복합니다.

- (선택 사항) 기본 노드가 사용자 지정 인증서를 사용하는 경우에는 장치에서 인증서를 설정하거나 새 인증서를 생성해야 합니다. [vRealize Orchestrator에 대한 사용자 지정 TLS 인증서 생성](#)의 내용을 참조하십시오.

---

**참고** 인증서 체인이 포함된 파일은 PEM-인코드 파일이어야 합니다.

---

#### 4 클러스터 배포를 마칩니다.

- a SSH를 통해 기본 노드의 vRealize Orchestrator Appliance에 **root**로 로그인합니다.
- b 모든 노드가 준비 상태인지 확인하려면 `kubectl -n prelude get nodes` 명령을 실행합니다.
- c `/opt/scripts/deploy.sh` 스크립트를 실행하고 배포를 마칠 때까지 기다립니다.

#### 결과

vRealize Orchestrator 클러스터가 생성되었습니다. 클러스터를 생성한 후에는 로드 밸런서 서버의 FQDN 주소에서만 vRealize Orchestrator 환경에 액세스할 수 있습니다.

**참고** 로드 밸런서의 루트 암호를 사용해야만 클러스터의 제어 센터에 액세스할 수 있기 때문에 루트 암호가 다른 경우 클러스터 노드의 구성을 편집할 수 없습니다. 이 노드의 구성을 편집하려면 로드 밸런서에서 제거하고 제어 센터에서 구성을 편집한 다음, 노드를 로드 밸런서에 다시 추가합니다.

#### 다음에 수행할 작업

vRealize Orchestrator 클러스터의 상태를 모니터링하려면 제어 센터에 로그인하고 **Orchestrator 클러스터 관리** 페이지를 선택합니다. [vRealize Orchestrator 클러스터 모니터링](#)의 내용을 참조하십시오.

## vRealize Orchestrator 클러스터 노드 제거

vRealize Orchestrator를 삭제하여 클러스터 용량을 줄일 수 있습니다.

vRealize Orchestrator 클러스터에서 노드를 제거한 후에는 해당 노드가 더 이상 작동하지 않습니다. 이 노드를 다시 사용하려면 vCenter Server에서 해당 vRealize Orchestrator Appliance를 삭제하고 다시 배포해야 합니다. [vRealize Orchestrator Appliance 다운로드 및 배포](#)의 내용을 참조하십시오.

#### 사전 요구 사항

vRealize Orchestrator 클러스터를 생성합니다. [vRealize Orchestrator 클러스터 구성](#)의 내용을 참조하십시오.

#### 절차

- 1 제거하려는 노드의 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 vRealize Orchestrator에서 노드를 제거하려면 `vracli cluster leave` 명령을 실행합니다.
- 3 나머지 노드 중 하나의 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 4 `kubectl -n prelude get nodes` 명령을 실행하고 제거된 노드가 더 이상 클러스터의 일부가 아닌지 확인합니다.

## 독립형 vRealize Orchestrator 배포 확장

구성된 vRealize Orchestrator 배포를 확장하여 가용성 및 확장성을 높일 수 있습니다.

## 사전 요구 사항

- vRealize Orchestrator 인스턴스를 다운로드, 배포 및 구성합니다. 자세한 내용은 [vRealize Orchestrator Appliance 다운로드 및 배포](#) 및 독립형 vRealize Orchestrator 서버 구성에 나와 있습니다.
- 두 개의 추가 vRealize Orchestrator 인스턴스를 다운로드하고 배포합니다. [vRealize Orchestrator Appliance 다운로드 및 배포](#)의 내용을 참조하십시오.
- 로드 밸런서 서버를 구성합니다. [VMware vRealize Orchestrator 8.x 로드 밸런싱 가이드](#)를 참조하십시오.

## 절차

## 1 기본 노드를 구성합니다.

- a 구성된 vRealize Orchestrator 배포의 제어 센터에 **root**로 로그인합니다.
- b **인증 제공자 구성**을 선택하고 인증 제공자를 등록 취소합니다.
- c **호스트 설정**을 선택하고 로드 밸런서 서버의 호스트 이름을 입력합니다.
- d **인증 제공자 구성**을 선택하고 인증 제공자를 다시 등록합니다.
- e 구성된 인스턴스의 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- f vRealize Orchestrator 인스턴스의 모든 서비스를 중지하려면 `/opt/scripts/deploy.sh --onlyClean` 명령을 실행합니다.
- g 로드 밸런서를 설정하려면 `vracli load-balancer set load_balancer_FQDN`을 실행합니다.
- h (선택 사항) vRealize Orchestrator 인스턴스가 사용자 지정 인증서를 사용하는 경우 `vracli certificate ingress --set your_cert_file.pem` 명령을 실행합니다.

---

**참고** 인증서 체인이 포함된 파일은 PEM-인코드 파일이어야 합니다.

---

## 2 보조 노드를 구성된 인스턴스에 조인합니다.

- a 보조 노드의 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- b 보조 노드를 구성된 인스턴스에 조인하려면 `vracli cluster join primary_node_hostname_or_IP` 명령을 실행합니다.
- c 다른 보조 노드에 대해 반복합니다.

## 3 확장 프로세스를 마칩니다.

- a 구성된 인스턴스의 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- b `/opt/scripts/deploy.sh`를 실행하고 스크립트를 마칠 때까지 기다립니다.

## 결과

vRealize Orchestrator 배포를 확장했습니다.

## vRealize Orchestrator 클러스터 모니터링

vRealize Orchestrator 제어 센터를 통해 기존 vRealize Orchestrator 클러스터를 모니터링할 수 있습니다.

제어 센터의 **Orchestrator 클러스터 관리** 페이지에서 클러스터에 조인된 vRealize Orchestrator 인스턴스의 구성 동기화 상태를 모니터링할 수 있습니다.

구성 동기화 상태	설명
실행 중	vRealize Orchestrator 서비스를 사용할 수 있으며 요청을 수락할 수 있습니다.
대기	vRealize Orchestrator 서비스가 다음과 같은 이유로 요청을 처리할 수 없습니다. <ul style="list-style-type: none"> <li>■ 노드는 HA(고가용성) 클러스터의 일부이며 주 노드에 장애가 발생할 때까지 대기 모드를 유지합니다.</li> <li>■ 서비스가 데이터베이스, 인증 제공자 및 vRealize Orchestrator 인스턴스 라이선스에 대한 유효한 연결과 같은 구성 필수 요건을 확인할 수 없습니다.</li> </ul>
서비스 상태 검색 실패	vRealize Orchestrator 서버 서비스가 중지되었거나 네트워크 문제가 있으므로 이 서비스에 연결할 수 없습니다.
다시 시작 대기 중	제어 센터가 구성 변경 사항을 감지하여 vRealize Orchestrator 서버가 자동으로 다시 시작됩니다.

## 고객 환경 개선 프로그램 구성

CEIP(고객 환경 향상 프로그램)에 참여하도록 선택하면 VMware는 VMware 제품과 서비스의 품질, 안정성 및 기능 향상시키는 데 도움이 되는 익명 정보를 받습니다.

## VMware가 수신하는 정보의 범주

CEIP(고객 환경 향상 프로그램)은 VMware가 해당 제품 및 서비스를 향상시키고 문제를 수정할 수 있도록 하는 정보를 VMware에 제공합니다.

CEIP를 통해 수집되는 데이터에 대한 세부 정보와 VMware에서 해당 정보를 사용하는 목적은 신뢰 및 보장 센터(<http://www.vmware.com/trustvmware/ceip.html>)에 명시되어 있습니다. 이 제품에 대한 CEIP에 가입하거나 탈퇴하려면 [고객 환경 향상 프로그램 가입 또는 탈퇴](#)를 참조하십시오.

## 고객 환경 향상 프로그램 가입 또는 탈퇴

고객 환경 향상 프로그램은 vRealize Orchestrator Appliance 명령줄에서 가입할 수 있습니다.

### 절차

- 1 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 고객 환경 향상 프로그램에 가입하려면 **vracli ceip on** 명령을 실행합니다.

- 3 고객 환경 향상 프로그램 정보를 검토하고 `vracli ceip on --acknowledge-ceip` 명령을 실행합니다.
- 4 vRealize Orchestrator 서비스를 다시 시작합니다.
  - a 서버 서비스를 다시 시작하려면 `kubectl -n prelude exec -it your_vro_pod -c vco-server-app /bin/bash` 명령을 실행합니다.
  - b 서비스를 중지하려면 `kill 1` 명령을 실행합니다.
  - c 제어 센터 서비스를 다시 시작하려면 `kubectl -n prelude exec -it your_vro_pod -c vco-controlcenter-app /bin/bash` 명령을 실행합니다.
  - d 서비스를 중지하려면 `kill 1` 명령을 실행합니다.
- 5 고객 환경 향상 프로그램을 탈퇴하려면 `vracli ceip off` 명령을 실행합니다.
- 6 서비스를 다시 시작하는 단계를 반복합니다.

# vRealize Orchestrator API 서비스 사용

## 6

제어 센터를 사용한 vRealize Orchestrator 구성 외에도 장치에 저장된 vRealize Orchestrator REST API, 제어 센터 REST API 또는 명령줄 유틸리티를 사용하여 vRealize Orchestrator 서버 구성 설정을 수정할 수 있습니다.

구성 플러그인은 기본적으로 vRealize Orchestrator 패키지에 포함되어 있습니다. 구성 플러그인 워크플로는 vRealize Orchestrator 워크플로 라이브러리 또는 vRealize Orchestrator REST API를 통해 액세스할 수 있습니다. 이 워크플로를 통해 신뢰할 수 있는 인증서 및 vRealize Orchestrator 서버의 키 저장소 설정을 변경할 수 있습니다. 제공되는 모든 vRealize Orchestrator REST API 서비스 호출에 대한 자세한 내용은 [https://your\\_orchestrator\\_FQDN/vco/api/docs](https://your_orchestrator_FQDN/vco/api/docs)에서 "vRealize Orchestrator 서버 API" 설명서를 참조하십시오.

### ■ REST API를 사용하여 TLS 인증서 및 키 저장소 관리

제어 센터를 사용하여 TLS 인증서를 관리하는 것 외에, 구성 플러그인에서 워크플로를 실행하거나 REST API를 사용하여 신뢰할 수 있는 인증서 및 키 저장소를 관리할 수도 있습니다.

## REST API를 사용하여 TLS 인증서 및 키 저장소 관리

제어 센터를 사용하여 TLS 인증서를 관리하는 것 외에, 구성 플러그인에서 워크플로를 실행하거나 REST API를 사용하여 신뢰할 수 있는 인증서 및 키 저장소를 관리할 수도 있습니다.

구성 플러그인에는 TLS 인증서 및 키 저장소를 가져오고 삭제하는 워크플로가 포함되어 있습니다. 이러한 워크플로는 vRealize Orchestrator Client에서 **라이브러리 > 워크플로 > SSL 신뢰 관리자 및 라이브러리 > 워크플로 > 키 저장소**로 이동하여 액세스할 수 있습니다. vRealize Orchestrator REST API를 사용하여 이러한 워크플로를 실행할 수도 있습니다.

제어 센터 REST API는 vRealize Orchestrator 서버를 구성할 수 있도록 리소스에 대한 액세스를 제공합니다. 타사 시스템에서 제어 센터 REST API를 사용하여 vRealize Orchestrator 구성을 자동화할 수 있습니다. 제어 센터 REST API의 루트 끝점은 [https://your\\_orchestrator\\_FQDN/vco/api](https://your_orchestrator_FQDN/vco/api)입니다. 제어 센터 REST API에 대해 수행할 수 있는 사용 가능한 모든 서비스 호출에 대한 자세한 내용은 "vRealize Orchestrator 제어 센터 API" 설명서([https://your\\_orchestrator\\_FQDN/vco-controlcenter/docs](https://your_orchestrator_FQDN/vco-controlcenter/docs))를 참조하십시오.

## REST API를 사용하여 TLS 인증서 삭제

구성 플러그인의 신뢰할 수 있는 인증서 삭제 워크플로를 실행하거나 REST API를 사용하여 TLS 인증서를 삭제할 수 있습니다.

### 절차

- 1 신뢰할 수 있는 인증서 삭제 워크플로의 워크플로 서비스 URL에서 GET 요청을 만듭니다.

```
GET https://{orchestrator_host}:{port}/vco/api/workflows?conditions=name=Delete trusted certificate
```

- 2 정의의 URL에서 GET 요청을 만들어서 신뢰할 수 있는 인증서 삭제 워크플로 정의를 검색합니다.

```
GET https://{orchestrator_host}:{port}/vco/api/workflows/8a70a326-ffd7-4fef-97e0-2002ac49f5bd
```

- 3 신뢰할 수 있는 인증서 삭제 워크플로의 실행 개체를 보유하는 URL에서 POST 요청을 만듭니다.

```
POST https://{orchestrator_host}:{port}/vco/api/workflows/8a70a326-ffd7-4fef-97e0-2002ac49f5bd/executions/
```

- 4 삭제하려는 인증서의 이름을 요청 본문의 실행 컨텍스트 요소에서 신뢰할 수 있는 인증서 삭제 워크플로의 입력 매개 변수로 제공합니다.

## REST API를 사용하여 TLS 인증서 가져오기

구성 플러그인에서 워크플로를 실행하거나 REST API를 사용하여 TLS 인증서를 가져올 수 있습니다.

파일 또는 URL에서 신뢰할 수 있는 인증서를 가져올 수 있습니다. [제어 센터를 사용하여 신뢰할 수 있는 인증서 가져오기](#)의 내용을 참조하십시오.

### 절차

- 1 워크플로 서비스의 URL에서 GET 요청을 만듭니다.

옵션	설명
파일에서 신뢰할 수 있는 인증서 가져오기	파일에서 신뢰할 수 있는 인증서를 가져옵니다.
URL에서 신뢰할 수 있는 인증서 가져오기	URL 주소에서 신뢰할 수 있는 인증서를 가져옵니다.
프록시 서버를 사용하여 URL에서 신뢰할 수 있는 인증서 가져오기	프록시 서버를 사용하여 URL 주소에서 신뢰할 수 있는 인증서를 가져옵니다.
인증서 별칭이 포함된 URL에서 신뢰할 수 있는 인증서 가져오기	URL 주소에서 인증서 별칭이 있는 신뢰할 수 있는 인증서를 가져옵니다.

파일에서 신뢰할 수 있는 인증서를 가져오려면 다음 GET 요청을 만듭니다.

```
GET https://{orchestrator_host}:{port}/vco/api/workflows?conditions=name=Import trusted certificate from a file
```



- 2 정의의 URL에서 GET 요청을 만들어 워크플로 정의를 검색합니다.

파일 워크플로에서 신뢰할 수 있는 인증서 가져오기에 대한 정의를 검색하려면 다음 GET 요청을 만듭니다.

```
GET https://{orchestrator_host}:{port}/vco/api/workflows/93a7bb21-0255-4750-9293-2437abe9d2e5
```

- 3 워크플로의 실행 개체를 보유하는 URL에서 POST 요청을 만듭니다.

파일 워크플로에서 신뢰할 수 있는 인증서를 가져오려면 다음 POST 요청을 만듭니다.

```
POST https://{orchestrator_host}:{port}/vco/api/workflows/93a7bb21-0255-4750-9293-2437abe9d2e5/
executions
```

- 4 요청 본문의 실행 컨텍스트 요소에 워크플로의 입력 매개 변수에 대한 값을 제공합니다.

매개 변수	설명
<b>cer</b>	TLS 인증서를 가져오려는 CER 파일입니다. 이 매개 변수는 파일 워크플로에서 신뢰할 수 있는 인증서 가져오기에 적용됩니다.
<b>url</b>	TLS 인증서를 가져오려는 URL입니다. HTTPS가 아닌 서비스의 경우 지원되는 형식은 <code>IP_address_or_DNS_name:port</code> 입니다. 이 매개 변수는 URL 워크플로에서 신뢰할 수 있는 인증서 가져오기에 적용됩니다.

## REST API를 사용하여 키 저장소 생성

구성 플러그인의 키 저장소 생성 워크플로를 실행하거나 REST API를 사용하여 키 저장소를 생성할 수 있습니다.

### 절차

- 1 키 저장소 생성 워크플로의 워크플로 서비스 URL에서 GET 요청을 만듭니다.

```
GET https://{orchestrator_host}:{port}/vco/api/workflows?conditions=name=Create a keystore
```

- 2 정의의 URL에서 GET 요청을 만들어 키 저장소 생성 워크플로 정의를 검색합니다.

```
GET https://{orchestrator_host}:{port}/vco/api/workflows/6c301bff-e8fe-4ae0-ad08-5318178594b3/
```

- 3 키 저장소 생성 워크플로의 실행 개체를 보유하는 URL에서 POST 요청을 만듭니다.

```
POST https://{orchestrator_host}:{port}/vco/api/workflows/6c301bff-e8fe-4ae0-ad08-5318178594b3/
executions/
```

- 4 요청 본문의 실행 컨텍스트 요소에 키 저장소 생성 워크플로의 입력 매개 변수로 생성하려는 키 저장소 이름을 제공합니다.

## REST API를 사용하여 키 저장소 삭제

구성 플러그인의 키 저장소 삭제 워크플로를 실행하거나 REST API를 사용하여 키 저장소를 삭제할 수 있습니다.

### 절차

- 1 키 저장소 삭제 워크플로의 워크플로 서비스 URL에서 GET 요청을 만듭니다.

```
GET https://{orchestrator_host}:{port}/vco/api/workflows?conditions=name=Delete a keystore
```

- 2 정의의 URL에서 GET 요청을 만들어 키 저장소 삭제 워크플로 정의를 검색합니다.

```
GET https://{orchestrator_host}:{port}/vco/api/workflows/7a3389eb-1fab-4d77-860b-81b66bb45b86/
```

- 3 키 저장소 삭제 워크플로의 실행 개체를 보유하는 URL에서 POST 요청을 만듭니다.

```
POST https://{orchestrator_host}:{port}/vco/api/workflows/7a3389eb-1fab-4d77-860b-81b66bb45b86/
executions/
```

- 4 요청 본문의 실행 컨텍스트 요소에 키 저장소 삭제 워크플로의 입력 매개 변수로 삭제하려는 키 저장소 이름을 제공합니다.

## REST API를 사용하여 키 추가

구성 플러그인의 키 추가 워크플로를 실행하거나 REST API를 사용하여 키를 추가할 수 있습니다.

### 절차

- 1 키 추가 워크플로의 워크플로 서비스 URL에서 GET 요청을 만듭니다.

```
GET https://{orchestrator_host}:{port}/vco/api/workflows?conditions=name=Add key
```

- 2 정의의 URL에서 GET 요청을 만들어 키 추가 워크플로 정의를 검색합니다.

```
GET https://{orchestrator_host}:{port}/vco/api/workflows/6c301bff-e8fe-4ae0-ad08-5318178594b3/
```

- 3 키 추가 워크플로의 실행 개체를 보유하는 URL에서 POST 요청을 만듭니다.

```
POST https://{orchestrator_host}:{port}/vco/api/workflows/6c301bff-e8fe-4ae0-ad08-5318178594b3/
executions/
```

- 4 요청 본문의 실행 컨텍스트 요소에서 키 저장소, 키 별칭, PEM-인코드 키, 인증서 체인 및 키 암호를 키 추가 워크플로의 입력 매개 변수로 제공합니다.

# 추가 구성 옵션

## 7

제어 센터를 사용하여 기본 vRealize Orchestrator 동작을 변경할 수 있습니다.

본 장은 다음 항목을 포함합니다.

- 인증 재구성
- 워크플로 실행 속성 구성
- vRealize Orchestrator 로그 파일
- Opentracing 및 Wavefront 확장 사용
- vRealize Orchestrator에 대한 시간 동기화 사용
- vRealize Orchestrator에 대한 시간 동기화 비활성화
- vRealize Orchestrator Kubernetes CIDR 구성

## 인증 재구성

제어 센터의 초기 구성 중에 인증 방법을 설정한 후에는 언제든지 인증 제공자 또는 구성한 매개 변수를 변경할 수 있습니다.

## 인증 제공자 변경

인증 모드 또는 인증 제공자 연결 설정을 변경하려면, 먼저 기존 인증 제공자를 등록 취소해야 합니다.

절차

- 1 제어 센터에 **root**로 로그인합니다.
- 2 **인증 제공자 구성** 페이지에서 호스트 주소 텍스트 상자 옆에 있는 **등록 취소** 버튼을 클릭하여 사용 중인 인증 제공자의 등록을 취소합니다.

결과

인증 제공자를 성공적으로 등록 취소했습니다.

다음에 수행할 작업

제어 센터에서 인증을 재구성합니다. [독립형 vRealize Orchestrator 서버 구성](#)의 내용을 참조하십시오.

## 인증 매개 변수 변경

제어 센터에서 vSphere를 인증 제공자로 사용하는 경우, vRealize Orchestrator 관리자 그룹의 기본 테넌트를 변경할 수 있습니다.

사전 요구 사항

vSphere를 vRealize Orchestrator 배포에 대한 인증 제공자로 구성합니다. [vSphere 인증을 사용하여 독립형 vRealize Orchestrator 서버 구성](#)의 내용을 참조하십시오.

---

**참고** vRealize Automation 인증에는 이러한 매개 변수가 포함되지 않습니다.

---

절차

- 1 제어 센터에 **root**로 로그인합니다.
- 2 **인증 제공자 구성**을 선택합니다.
- 3 **기본 테넌트** 텍스트 상자 옆에 있는 **변경** 버튼을 클릭합니다.
- 4 테넌트 이름을 바꿉니다.
- 5 **변경** 버튼(**관리자 그룹** 텍스트 상자 옆에 있음)을 클릭합니다.

---

**참고** 관리자 그룹을 재구성하지 않으면 비어 있게 되며 더 이상 제어 센터에 액세스할 수 없습니다.

---

- 6 관리자 그룹의 이름을 입력하고 **검색**을 클릭합니다.
- 7 관리자 그룹을 선택합니다.
- 8 관리자 그룹을 변경합니다.
- 9 인증 매개 변수 편집을 마치려면 **변경 사항 저장**을 클릭합니다.

## 워크플로 실행 속성 구성

기본적으로 노드별로 최대 300개의 워크플로를 실행할 수 있으며 현재 실행 중인 워크플로의 개수가 이에 도달하면 최대 10,000개의 워크플로를 대기열에 추가할 수 있습니다.

vRealize Orchestrator 노드가 동시에 300개 이상의 워크플로를 실행해야 하는 경우 보류 중인 워크플로는 대기열에 추가됩니다. 활성 워크플로 실행이 완료되면 대기열의 다음 워크플로가 실행되기 시작합니다. 대기 중인 워크플로가 최대 개수에 도달하면 보류 중인 워크플로 중 하나가 실행되기 전까지는 다음 워크플로 실행이 실패합니다.

제어 센터의 **고급 옵션** 페이지에서 워크플로 실행 속성을 구성할 수 있습니다.

옵션	설명
안전 모드 사용	안전 모드를 사용하면 실행 중인 모든 워크플로가 취소되고 다음 vRealize Orchestrator 노트 시작 시 재개되지 않습니다.
동시 실행 워크플로 개수	동시에 실행되는 워크플로의 수입니다. 기본값은 노드당 300개 워크플로입니다.
대기열에서 실행 중인 워크플로의 최대 개수	사용할 수 없는 상태가 되기 전까지 vRealize Orchestrator 서버가 수락하는 워크플로 실행 요청의 개수입니다. 기본값은 노드당 10,000개 워크플로입니다.
워크플로별로 유지되는 실행의 최대 개수	워크플로당 기록으로 유지되는 완료된 워크플로 실행의 최대 수입니다. 개수를 초과하면 가장 오래된 워크플로 실행이 삭제됩니다. 기본값은 워크플로당 100개의 실행입니다.
로그 이벤트 만료 일수	로그 이벤트가 제거되기 전에 데이터베이스에서 유지되는 기간(일)입니다. 기본값은 15일입니다.

## vRealize Orchestrator 로그 파일

VMware 기술 지원은 사용자의 지원 요청이 제출되면 정기적으로 진단 정보를 요청합니다. 해당 제품이 실행되는 호스트에서 진단된 이 정보에는 제품 관련 로그 및 구성 파일이 포함됩니다.

vRealize Orchestrator Appliance 로그는 `/data/vco/usr/lib/vco/app-server/logs/` 디렉토리에 저장됩니다. 장치 명령줄에 로그인하고 `vraccli log-bundle` 명령을 실행하여 vRealize Orchestrator Appliance 배포의 로그를 내보냅니다. 생성된 로그 번들이 vRealize Orchestrator Appliance의 루트 폴더에 저장됩니다.

## 로깅 지속성

모든 종류의 vRealize Orchestrator 스크립트(예: 워크플로, 정책 또는 작업)에 정보를 기록할 수 있습니다. 이러한 정보에는 유형과 수준이 있습니다. 유형은 영구 또는 비영구입니다. 수준은 디버그, 정보, 주의, 오류, 추적 및 심각이 될 수 있습니다.

표 7-1. 영구 및 비영구 로그 생성

로그 수준	영구 유형	비영구 유형
디버그	<code>Server.debug("short text", "long text");</code>	<code>System.debug("text")</code>
정보	<code>Server.log("short text", "long text");</code>	<code>System.log("text");</code>
주의	<code>Server.warn("short text", "long text");</code>	<code>System.warn("text");</code>
오류	<code>Server.error("short text", "long text");</code>	<code>System.error("text");</code>

## 영구 로그

영구 로그(서버 로그)는 지난 워크플로 실행 로그를 추적하고 vRealize Orchestrator 데이터베이스에 저장됩니다.

## 비영구 로그

비영구 로그(시스템 로그)를 사용하여 스크립트를 생성하는 경우 vRealize Orchestrator 서버는 실행 중인 모든 vRealize Orchestrator 애플리케이션에 이 로그에 대한 알림을 제공하지만 이 정보는 데이터베이스에 저장되지 않습니다. 애플리케이션을 다시 시작하면 로그 정보가 손실됩니다. 비영구 로그는 디버깅 목적 및 라이브 정보에 사용됩니다. 시스템 로그를 보려면 vRealize Orchestrator Client에서 완료된 워크플로 실행을 선택하고 **로그** 탭을 선택해야 합니다.

## vRealize Orchestrator 로그 구성

제어 센터의 **로그 구성** 페이지에서 필요한 서버 로그 및 스크립팅 로그의 수준을 설정할 수 있습니다. 하루에 두 로그 중 하나가 여러 번 생성되는 경우 문제가 발생하는 원인을 찾기 어렵습니다.

서버 로그 및 스크립팅 로그의 기본 로그 수준은 정보입니다. 로그 수준을 변경하면 서버가 로그에 입력되는 모든 새 메시지와 데이터베이스의 활성 연결 수에 영향을 줍니다. 로깅의 자세한 정도는 내림차순으로 감소합니다.

**경고** 로그 수준을 디버그 또는 모두로 설정해야만 문제를 디버그할 수 있습니다. 성능에 심각한 영향을 줄 수 있기 때문에 운영 환경에서는 이러한 설정을 사용하지 마십시오.

## vRealize Orchestrator 로그 생성

vRealize Orchestrator Appliance 명령줄에 **root**로 로그인하고 **vraccli log-bundle** 명령을 실행하여 배포의 로그를 내보낼 수 있습니다. 생성된 로그 번들은 장치의 루트 폴더에 저장됩니다.

.

**참고** 클러스터에 둘 이상의 vRealize Orchestrator 인스턴스가 있으면, log-bundle은 클러스터에 있는 모든 vRealize Orchestrator 인스턴스의 로그를 포함합니다.

## vRealize Log Insight와 로깅 통합 구성

vRealize Log Insight 서버에 로깅 정보를 보내도록 vRealize Orchestrator를 구성할 수 있습니다.

vRealize Orchestrator Appliance 명령줄을 통해 vRealize Log Insight 서버에 대한 로깅 통합을 구성할 수 있습니다.

**참고** 원격 Syslog 서버와의 로깅 통합 구성에 대한 자세한 내용은 [vRealize Orchestrator에서 Syslog 통합 생성 또는 덮어쓰기](#) 항목을 참조하십시오.

### 사전 요구 사항

- vRealize Log Insight 서버를 구성합니다. "vRealize Log Insight 설명서"를 참조하십시오.
- vRealize Log Insight 버전이 4.7.1 이상인지 확인합니다.

### 절차

- 1 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.

- 2 vRealize Log Insight와 로깅 통합을 구성하려면 `vraccli vrli set vRLI_FQDN` 명령을 실행합니다.

**참고** vRealize Orchestrator 인스턴스가 자체 서명된 인증서를 사용하는 경우 선택적 `-k` 또는 `--insecure` 인수를 포함하여 SSL 인증을 사용하지 않도록 설정할 수 있습니다.

다음에 수행할 작업

vRealize Log Insight 구성 옵션에 대한 자세한 내용을 보려면 `vraccli vrli -h` 명령을 실행하십시오.

## vRealize Orchestrator에서 Syslog 통합 생성 또는 덮어쓰기

하나 이상의 원격 Syslog 서버에 로깅 정보를 보내도록 vRealize Orchestrator를 구성할 수 있습니다.

`vraccli remote-syslog set` 명령은 Syslog 통합을 생성하거나 기존 통합을 덮어쓰는 데 사용됩니다.

vRealize Orchestrator 원격 Syslog 통합은 세 가지 연결 유형을 지원합니다.

- UDP 연결.
- TLS를 사용하지 않는 TCP 연결.

**참고** TLS를 사용하지 않고 Syslog 통합을 생성하려면 `vraccli remote-syslog set` 명령에 `--disable-ssl` 플래그를 추가합니다.

- TLS를 사용한 TCP 연결.

vRealize Log Insight와의 로깅 통합 구성에 대한 자세한 내용은 [vRealize Log Insight와 로깅 통합 구성](#) 항목을 참조하십시오.

사전 요구 사항

하나 이상의 원격 Syslog 서버를 구성합니다.

절차

- 1 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 Syslog 서버에 대해 통합을 생성하려면 `vraccli remote-syslog set` 명령을 실행합니다.

```
vraccli remote-syslog set -id name_of_integration protocol_type://syslog_URL_or_FQDN:syslog_port
```

**참고** `vraccli remote-syslog set` 명령에 포트를 입력하지 않으면 포트는 기본값인 514로 설정됩니다.

**참고** Syslog 구성에 인증서를 추가할 수 있습니다. 인증서 파일을 추가하려면 `--ca-file` 플래그를 사용합니다. 인증서를 일반 텍스트로 추가하려면 `--ca-cert` 플래그를 사용합니다.

- 3 (선택 사항) 기존 Syslog 통합을 덮어쓰려면 `vracli remote-syslog set`를 실행하고 `-id` 플래그 값을 덮어쓸 통합의 이름으로 설정합니다.

---

**참고** 기본적으로 vRealize Orchestrator Appliance는 Syslog 통합을 덮어쓸 것인지 확인을 요청합니다. 확인 요청을 건너뛰려면 `-f` 또는 `--force` 플래그를 `vracli remote-syslog set` 명령에 추가합니다.

---

다음에 수행할 작업

장치에서 현재 Syslog 통합을 검토하려면 `vracli remote-syslog` 명령을 실행합니다.

## vRealize Orchestrator에서 Syslog 통합 삭제

`vracli remote-syslog unset` 명령을 실행하여 vRealize Orchestrator Appliance에서 Syslog 통합을 삭제할 수 있습니다.

사전 요구 사항

vRealize Orchestrator Appliance에서 Syslog 통합을 하나 이상 생성합니다. [vRealize Orchestrator에서 Syslog 통합 생성 또는 덮어쓰기](#)의 내용을 참조하십시오.

절차

- 1 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 vRealize Orchestrator Appliance에서 Syslog 통합을 삭제합니다.
  - a 특정 Syslog 통합을 삭제하려면 `vracli remote-syslog unset -id Integration_name` 명령을 실행합니다.
  - b vRealize Orchestrator Appliance에서 모든 Syslog 통합을 삭제하려면 `-id` 플래그 없이 `vracli remote-syslog unset` 명령을 실행합니다.

---

**참고** 기본적으로 vRealize Orchestrator Appliance는 모든 Syslog 통합을 삭제할 것인지 확인을 요청합니다. 확인 요청을 건너뛰려면 `-f` 또는 `--force` 플래그를 `vracli remote-syslog unset` 명령에 추가합니다.

---

## Kerberos 디버그 로깅 사용

플러그인에 사용되는 Kerberos 구성 파일을 수정하여 vRealize Orchestrator 플러그인 문제를 해결할 수 있습니다.

Kerberos 구성 파일은 vRealize Orchestrator Appliance의 `/data/vco/usr/lib/vco/app-server/conf/` 디렉토리에 있습니다.

절차

- 1 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 `kubect1 -n prelude edit deployment vco-app` 명령을 실행합니다.



- 3 배포 파일에서 `-Djava.security.krb5.conf=/usr/lib/vco/app-server/conf/krb5.conf` 문자열을 찾아서 편집합니다.

```
-Djava.security.krb5.conf=/usr/lib/vco/app-server/conf/krb5.conf -Dsun.security.krb5.debug=true'
```

- 4 변경 사항을 저장하고 파일 편집기를 종료합니다.

- 5 `kubectl -n prelude get pods` 명령을 실행합니다.

모든 포드가 실행될 때까지 기다립니다.

- 6 Kerberos 디버그 로깅을 사용하도록 설정되었는지 확인합니다.

```
kubectl -n prelude log {vco_app_name} -c vco-server-app | grep krb5
```

로그에 다음과 유사한 메시지가 포함되어 있는지 확인합니다.

```
kubectl -n prelude log vco-app-5c965f9b9d-v8srd -c vco-server-app | grep krb5
12:23:05,417 INFO 011N:75 - Sysprop: java.security.krb5.conf = /usr/lib/vco/app-server/conf/krb5.conf
12:23:05,421 INFO 011N:75 - Sysprop: sun.security.krb5.debug = true
2019-10-22 12:23:38.521+0000 [Thread-19] INFO {} [011N] Sysprop: java.security.krb5.conf = /usr/lib/vco/app-server/conf/krb5.conf
2019-10-22 12:23:38.525+0000 [Thread-19] INFO {} [011N] Sysprop: sun.security.krb5.debug = true
Java config name: /usr/lib/vco/app-server/conf/krb5.conf
EType: sun.security.krb5.internal.crypto.Aes256CtsHmacSha1EType
```

## Opentracing 및 Wavefront 확장 사용

vRealize Orchestrator용 Opentracing 및 Wavefront 확장은 vRealize Orchestrator 환경에 대한 데이터를 수집하는 도구를 제공합니다. 이 데이터를 사용하여 vRealize Orchestrator 시스템 및 워크플로 문제를 해결할 수 있습니다.

Opentracing 및 Wavefront 확장을 사용하도록 vRealize Orchestrator를 구성하려면 먼저 vRealize Orchestrator Appliance에서 사용하도록 설정해야 합니다.

사전 요구 사항

- vRealize Orchestrator Appliance SSH 서비스를 사용하도록 설정되어 있는지 확인합니다. [vRealize Orchestrator Appliance에 대한 SSH 액세스 사용 또는 사용 안 함](#)의 내용을 참조하십시오.
- Opentracing 또는 Wavefront 이전 버전의 확장을 사용하도록 설정한 경우, 현재 버전을 사용하도록 설정하기 전에 해당 버전을 제거해야 합니다. 예를 들어 이전에 Wavefront 확장 버전 8.1.0을 사용하도록 설정한 경우 `rm /data/vco/usr/lib/vco/app-server/extensions/wavefront-8.1.0.jar` 명령을 실행해야 합니다.

절차

- 1 SSH를 통해 vRealize Orchestrator Appliance에 **root**로 로그인합니다.

- 2 사용 가능한 확장을 모두 나열하려면 `ls /data/vco/usr/lib/vco/app-server/extensions/` 명령을 실행합니다.

- 3 Opentracing 확장을 사용하도록 설정하려면 다음 명령을 실행합니다.

```
mv /data/vco/usr/lib/vco/app-server/extensions/opentracing-8.4.0.jar.inactive /
data/vco/usr/lib/vco/app-server/extensions/opentracing-8.4.0.jar
```

- 4 Wavefront 확장을 사용하도록 설정하려면 다음 명령을 실행합니다.

```
mv /data/vco/usr/lib/vco/app-server/extensions/wavefront-8.4.0.jar.inactive /data/vco/usr/lib/vco/
app-server/extensions/wavefront-8.4.0.jar
```

- 5 제어 센터에 로그인하고 확장이 **확장 속성** 페이지에 나타나는지 확인합니다.

다음에 수행할 작업

**확장 속성** 페이지에서 vRealize Orchestrator와 Opentracing 및 Wavefront 통합을 구성합니다. 자세한 내용은 [Opentracing 확장 구성](#) 및 [Wavefront 확장 구성](#)에 나와 있습니다.

## Opentracing 확장 구성

Opentracing 확장은 워크플로 실행에 대한 데이터를 Jaeger 서버로 보냅니다. 데이터에는 워크플로 상태, 입력 및 출력 매개 변수, 워크플로 실행을 시작한 사용자 및 워크플로 ID 데이터가 포함됩니다.

사전 요구 사항

- vRealize Orchestrator Appliance에서 Opentracing을 사용하도록 설정되어 있는지 확인합니다. [Opentracing](#) 및 [Wavefront 확장 사용](#)의 내용을 참조하십시오.
- Opentracing 확장에서 사용할 Jaeger 서버를 배포합니다. 자세한 내용은 [Jaeger 시작 설명서](#)를 참조하십시오.

절차

- 1 제어 센터에 **root**로 로그인합니다.
- 2 **확장 속성** 페이지를 선택합니다.
- 3 Opentracing 확장을 선택합니다.
- 4 Jaeger 서버 호스트 주소와 포트를 입력합니다.

---

**참고** 서버 주소를 입력하기 전에 슬래시 2개("/")를 삽입합니다.

---

- 5 **저장**을 클릭합니다.

결과

vRealize Orchestrator에 대한 Opentracing 확장을 구성했습니다.

다음에 수행할 작업

- **Opentracing** 확장에서 수집한 데이터를 포함하는 **Jaeger UI**에 액세스하려면 구성 중에 입력한 호스트 주소에 방문합니다.
- **서비스** 옵션에서 **워크플로**를 선택합니다.
- 표시할 데이터를 지정하려면 **태그** 옵션을 사용합니다. 예를 들어, 실패한 워크플로에 대한 데이터를 보려면 **status=failed**를 입력합니다.

## Wavefront 확장 구성

Wavefront 확장을 사용하여 vRealize Orchestrator 시스템 및 워크플로에 대한 메트릭 데이터를 수집합니다.

사전 요구 사항

- 1 vRealize Orchestrator Appliance에서 Wavefront를 사용하도록 설정되어 있는지 확인합니다. [Opentracing 및 Wavefront 확장 사용](#)의 내용을 참조하십시오.
- 2 Wavefront 인증서를 가져옵니다.
  - a vRealize Orchestrator 제어 센터에 **root**로 로그인합니다.
  - b **인증서** 페이지를 선택합니다.
  - c **가져오기** 드롭다운 메뉴를 클릭하고 **URL에서 가져오기**를 선택합니다.
  - d Wavefront URL을 입력하고 **가져오기**를 클릭합니다.
- 3 Wavefront 프록시를 구성합니다. 자세한 내용은 [Wavefront 프록시 설치 및 구성](#)을 참조하십시오.

절차

- 1 vRealize Orchestrator 제어 센터에 **root**로 로그인합니다.
- 2 **확장 속성** 페이지를 선택합니다.
- 3 Wavefront 확장을 선택합니다.
- 4 Wavefront 속성을 구성합니다.

옵션	설명
프록시	Wavefront 프록시 주소입니다.
호스트	선택 사항. Wavefront 호스트 주소입니다.
토큰	선택 사항. Wavefront API 토큰입니다. Wavefront API 토큰 생성에 대한 자세한 내용은 <a href="#">API 토큰 생성</a> 을 참조하십시오.
접두사	Wavefront로 전송되는 각 메트릭에 대한 접두사 레이블을 추가합니다. 접두사 레이블은 점 기호로 구분됩니다.

- 5 (선택 사항) **다음에 시작할 때 기본 대시보드 보내기**를 참조하십시오.
- 6 **저장**을 클릭합니다.

## 결과

vRealize Orchestrator에 대한 Wavefront 확장을 구성했습니다.

다음에 수행할 작업

- Wavefront에서 수집한 메트릭에 액세스하려면 구성 중에 입력한 주소에서 대시보드에 액세스합니다.
- vRealize Orchestrator 환경에서 특정 이벤트에 대한 알림을 받기 위해 Wavefront 경고를 사용할 수 있습니다. 자세한 내용은 [Wavefront 경고 설명서](#)를 참조하십시오.

## vRealize Orchestrator에 대한 시간 동기화 사용

vRealize Orchestrator Appliance 명령줄을 사용하여 vRealize Orchestrator 배포에서 시간 동기화를 사용하도록 설정할 수 있습니다.

NTP(Network Time Protocol) 통신 프로토콜을 사용하여 독립형 또는 클러스터링된 vRealize Orchestrator 배포에 대한 시간 동기화를 구성할 수 있습니다. vRealize Orchestrator은 상호 배타적인 두 가지 NTP 구성을 지원합니다.

NTP 구성	설명
ESXi	<p>이 구성은 vRealize Orchestrator Appliance를 호스팅하는 ESXi 서버가 NTP 서버와 동기화되는 경우 사용할 수 있습니다. 클러스터링된 배포를 사용하는 경우 모든 ESXi 호스트가 NTP 서버와 동기화되어야 합니다. ESXi용 NTP 구성에 대한 자세한 내용은 <a href="#">vSphere Web Client를 사용하여 ESXi 호스트에서 NTP(Network Time Protocol) 구성</a>을 참조하십시오.</p> <p><b>참고</b> vRealize Orchestrator 배포가 NTP 서버와 동기화되지 않은 ESXi 호스트로 마이그레이션되는 경우 클럭 드리프트(clock drift)가 발생할 수 있습니다.</p>
systemd	<p>이 구성은 systemd-timesyncd 데몬을 사용하여 vRealize Orchestrator 배포의 클럭을 동기화합니다.</p> <p><b>참고</b> 기본적으로 systemd-timesyncd 데몬을 사용하도록 설정되어 있지만 NTP 서버 없이 구성되어 있습니다. vRealize Orchestrator Appliance에서 동적 IP 구성을 사용하는 경우, 장치는 DHCP 프로토콜이 수신한 NTP 서버를 사용할 수 있습니다.</p>

## 절차

- 1 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 ESXi에서 NTP를 사용하도록 설정합니다.
  - a `vracli ntp esxi` 명령을 실행합니다.
  - b (선택 사항) NTP 구성의 상태를 확인하려면 `vracli ntp status` 명령을 실행합니다.

3 systemd에서 NTP를 사용하도록 설정합니다.

- a `vracli ntp systemd --set FQDN_or_IP_of_systemd_server` 명령을 실행합니다.

**참고** systemd NTP 서버의 네트워크 주소를 쉽표로 구분하여 여러 서버를 추가할 수 있습니다. 각 네트워크 주소는 작은따옴표로 묶어야 합니다. 예를 들면 `vracli ntp systemd --set 'ntp_address_1','ntp_address_2'`와 같습니다.

- b (선택 사항) NTP 구성의 상태를 확인하려면 `vracli ntp status` 명령을 실행합니다.

#### 결과

vRealize Orchestrator 배포에 대한 시간 동기화를 사용하도록 설정했습니다.

다음에 수행할 작업

NTP 서버와 vRealize Orchestrator 배포 사이에 시간 차이가 10분을 넘으면 NTP 구성이 실패할 수 있습니다. 이 문제를 해결하려면 vRealize Orchestrator Appliance를 재부팅합니다.

## vRealize Orchestrator에 대한 시간 동기화 비활성화

vRealize Orchestrator Appliance 명령줄을 사용하여 vRealize Orchestrator 배포에서 NTP(Network Time Protocol) 시간 동기화를 비활성화할 수 있습니다.

`vracli ntp reset` 명령을 실행하여 vRealize Orchestrator Appliance의 NTP 구성을 기본 상태로 재설정할 수도 있습니다.

#### 사전 요구 사항

ESXi 또는 systemd와 시간 동기화를 구성했는지 확인합니다. [vRealize Orchestrator에 대한 시간 동기화 사용](#)의 내용을 참조하십시오.

#### 절차

- 1 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 ESXi 또는 systemd의 시간 동기화를 비활성화하려면 `vracli ntp disable` 명령을 실행합니다.
- 3 (선택 사항) NTP 구성의 상태를 확인하려면 `vracli ntp status` 명령을 실행합니다.

## vRealize Orchestrator Kubernetes CIDR 구성

배포 후에 Kubernetes CIDR(클래스 없는 인터넷 도메인 라우팅) 서브넷 마스크를 변경할 수 있습니다.

vRealize Orchestrator Appliance는 Kubernetes 클러스터를 구성하고 실행합니다. 이 클러스터의 포트 및 서비스는 각각 내부 클러스터 CIDR 및 내부 서비스 CIDR로 나타내는 별도의 IPv4 서브넷에 배포됩니다. OVF 배포 중에 설정된 서브넷 마스크의 기본값은 다음과 같습니다.

Kubernetes network property	Default value	Property description
cluster-cidr	10.244.0.0/22	Kubernetes 클러스터 내부에서 실행되는 포트에 사용되는 CIDR입니다.
service-cidr	10.244.4.0/22	Kubernetes 클러스터 내부의 Kubernetes 서비스에 사용되는 CIDR입니다.

기본 CIDR 네트워크 주소는 사용 중일 수 있는 외부 전용 네트워크와 충돌할 수 있습니다. 이러한 시나리오에서는 vRealize Orchestrator Appliance 배포 도중 또는 이후에 이러한 CIDR 값의 구성을 변경할 수 있습니다.

**참고** 장치 배포 중에 CIDR 구성을 변경하는 방법에 대한 자세한 내용은 [vRealize Orchestrator Appliance 다운로드 및 배포](#) 항목을 참조하십시오.

#### 사전 요구 사항

- CIDR 주소 값이 1024개 이상의 호스트를 지원하는지 확인합니다.
- 내부 클러스터 CIDR 및 내부 서비스 CIDR은 동일한 서브넷 값을 공유하지 않아야 합니다.
- 서브넷 중 하나의 CIDR 값에 다른 서브넷에 추가하려는 값을 포함할 수 없습니다.

**참고** 예를 들어, cluster-cidr 값은 service-cidr 속성의 서브넷 값도 포함하기 때문에 **10.244.4.0/22 10.244.4.0/24**일 수 없습니다. 각 서브넷 값은 별도로 추가해야 합니다.

#### 절차

- 1 vRealize Orchestrator Appliance에 **루트**로 로그인합니다.
- 2 `vracli upgrade exec -y --prepare --profile k8s-subnets` 명령을 실행합니다.
- 3 VM(가상 시스템) 스냅샷을 생성하여 vRealize Orchestrator 배포를 백업합니다. [가상 시스템의 스냅샷 생성](#)을 참조하십시오.

**경고** vRealize Orchestrator 8.x는 현재 메모리 스냅샷을 지원하지 않습니다. vRealize Orchestrator 배포의 스냅샷을 생성하기 전에 **가상 시스템 메모리 스냅샷** 옵션이 비활성화되어 있는지 확인합니다.

- 4 `vracli network k8s-subnets` 명령을 실행하여 클러스터 CIDR 및 서비스 CIDR 서브넷의 값을 변경합니다.

```
vracli network k8s-subnets --cluster-cidr <CIDR_value> --service-cidr <CIDR_value>
```

- 5 CIDR 구성 프로세스를 완료하려면 `vracli upgrade exec` 명령을 실행합니다.

# 구성 사용 사례 및 문제 해결

# 8

구성 사용 사례는 vRealize Orchestrator 서버의 특정 구성 요구 사항을 충족하기 위해 수행할 수 있는 작업 흐름과 문제를 이해하고 해결하기 위한 문제 해결 항목을 제공합니다.

본 장은 다음 항목을 포함합니다.

- 서버 vRealize Orchestrator 번호 확인
- vSphere Web Client용 vRealize Orchestrator 플러그인 구성
- 실행 중인 워크플로 취소
- vRealize Orchestrator 서버 디버깅 사용
- vRealize Orchestrator Appliance 디스크 크기 조정
- vRealize Orchestrator 서버의 힙 메모리 크기를 조정하는 방법
- Site Recovery Manager를 사용한 vRealize Orchestrator의 재해 복구

## 서버 vRealize Orchestrator 번호 확인

특정 시나리오에서는 vRealize Orchestrator 배포의 서버 빌드 번호 확인이 필요할 수 있습니다.

[https://your\\_orchestrator\\_FQDN/vco/api/about](https://your_orchestrator_FQDN/vco/api/about)으로 이동하여 vRealize Orchestrator 서버 빌드 번호를 확인할 수 있습니다. 서버 빌드 번호는 <ns2:build-number> 태그에 표시됩니다.

서버 빌드 번호를 확인하면 VMware 지원에 로그인한 지원 요청(SR)에 대한 추가 정보 제공과 같은 사용 사례에 유용합니다. 서버 빌드 번호를 확인하여 vRealize Orchestrator가 최신 버전으로 업그레이드되었는지 확인할 수 있습니다.

## vSphere Web Client용 vRealize Orchestrator 플러그인 구성

vSphere Web Client용 vRealize Orchestrator 플러그인을 사용하려면 vRealize Orchestrator를 vCenter Server의 확장으로 등록해야 합니다.

vRealize Orchestrator 서버를 vCenter Single Sign-On에 등록하고 vCenter Server와 작동하도록 구성한 후, vRealize Orchestrator를 vCenter Server의 확장으로 등록해야 합니다.

## 사전 요구 사항

- vRealize Orchestrator Appliance에서 SSH 액세스가 가능하도록 설정되어 있는지 확인합니다.  
vRealize Orchestrator Appliance에 대한 SSH 액세스 사용 또는 사용 안 함의 내용을 참조하십시오.
- 관리되는 vCenter Server가 인증에 사용하는 것과 동일한 Platform Services Controller에 vSphere 인증을 사용하여 vRealize Orchestrator를 등록해야 합니다.
- vco-plugin.zip을 vRealize Orchestrator Appliance에 복사합니다.
  - a VMware Technology Network에서 .vco-plugin.zip 파일을 다운로드합니다.
  - b SSH 클라이언트를 엽니다.

---

**참고** Linux 또는 MacOS 환경의 경우 터미널 명령줄 인터페이스를 사용할 수 있습니다.  
Windows 환경의 경우 PuTTY 클라이언트를 사용할 수 있습니다.

---

- c vco-plugin.zip 파일을 복사하려면 보안 복사 명령을 실행합니다.

```
For Linux/MacOS: scp ~/<zip_download_dir>/vco-plugin.zip root@<orchestrator_FQDN_or_IP>:/data/vco/usr/lib/vco/downloads/vco-plugin.zip
```

```
For Windows: pscp C:\<zip_download_dir>\vco-plugin.zip root@<orchestrator_FQDN_or_IP>:/data/vco/usr/lib/vco/downloads/vco-plugin.zip
```

## 절차

- 1 vRealize Orchestrator Client에 로그인합니다.
- 2 라이브러리 > 워크플로로 이동합니다.
- 3 vCenter Orchestrator를 vCenter Server 확장으로 등록 워크플로를 검색하여 실행을 클릭합니다.
- 4 vRealize Orchestrator를 등록할 vCenter Server 인스턴스를 선택합니다.
- 5 https://your\_orchestrator\_FQDN 또는 요청을 vRealize Orchestrator 서버 노드로 리디렉션하는 로드 밸런서의 서비스 URL을 입력합니다.
- 6 실행을 클릭합니다.

## 실행 중인 워크플로 취소

vRealize Orchestrator 제어 센터를 사용하여 제대로 완료되지 않은 워크플로를 취소할 수 있습니다.

## 절차

- 1 제어 센터에 root로 로그인합니다.
- 2 문제 해결을 클릭합니다.



### 3 실행 중인 워크플로를 취소합니다.

옵션	설명
모든 워크플로 실행 취소	해당 워크플로에 대한 모든 토큰을 취소하려면 워크플로 ID를 입력합니다.
ID별로 워크플로 실행 취소	취소하려는 모든 토큰 ID를 입력합니다. 각 ID는 쉼표로 구분합니다.
실행 중인 모든 워크플로 취소	서버에서 실행 중인 모든 워크플로를 취소합니다.

**참고** 실행 스레드를 즉시 취소하는 신뢰할 만한 방법이 없으므로 ID별로 워크플로를 취소하는 작업은 실패할 수 있습니다.

#### 결과

다음 서버 시작 시, 워크플로는 취소된 상태로 설정됩니다.

## vRealize Orchestrator 서버 디버깅 사용

플러그인을 개발할 때 vRealize Orchestrator 서버를 디버그 모드에서 시작하여 문제를 디버깅할 수 있습니다.

#### 사전 요구 사항

로컬 시스템에 Kubernetes 명령줄 도구를 설치하고 구성합니다. [kubectl 설치 및 설정](#)을 참조하십시오.

#### 절차

- 1 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 `kubectl -n prelude edit deployment vco-app` 명령을 실행합니다.
- 3 `vco-server-app` 컨테이너에 디버그 환경 변수를 추가하여 배포 YAML 파일을 편집합니다. 변수는 `vco-server-app` 컨테이너의 `env` 섹션에 추가해야 합니다.

```
containers:
  - command:
    ...
    env:
      - name: DEBUG_PORT
        value: "your_desired_debug_port"
    ...
  name: vco-server-app
  ...
```

**참고** `env` 섹션에 디버그 환경 변수를 추가할 때 이전 예제에 나와 있는 것처럼 YAML 들여쓰기 형식을 따라야 합니다.

- 4 변경 내용을 배포 파일에 저장합니다.

배포 파일에 대한 편집이 성공하면 `deployment.extensions/vco-app edited` 메시지를 받게 됩니다.

5 `vracli dev kubeconfig` 명령을 실행하여 Kubernetes 구성 파일을 생성합니다.

`kubeconfig`는 개발자 환경이기 때문에 계속할지 여부를 묻는 메시지가 표시됩니다. 계속하려면 **yes**를 입력하고 중지하려면 **no**를 입력합니다.

6 생성된 구성 파일의 콘텐츠를 `apiVersion: v1`에서 `client-key-data` 콘텐츠까지 포함하여 복사합니다.

7 생성된 Kubernetes 구성 파일을 로컬 시스템에 저장합니다.

8 vRealize Orchestrator Appliance에서 로그아웃합니다.

9 로컬 시스템에서 디버그 모드 구성을 완료합니다.

a 명령줄 셸을 엽니다.

b `KUBECONFIG` 환경 변수를 저장된 구성 파일에 바인딩합니다.

---

**참고** 이 예는 Linux 환경을 기반으로 합니다.

---

```
export KUBECONFIG=/file/path/fileName
```

c 서비스가 실행 중인지 확인하려면 `kubectl cluster-info` 명령을 실행합니다.

d 디버그 모드 구성을 완료하려면 다음 Kubernetes API 요청을 수행합니다.

---

**참고** `localhost_debug_port` 변수 값은 IDE(통합 개발 환경)의 원격 디버깅 구성에 설정된 포트입니다. `vro_debug_port` 변수의 값은 이 절차의 3단계에서 생성됩니다.

---

```
kubectl port-forward pod/vco_app_pod_ID localhost_debug_port:vro_debug_port
```

---

**중요** 디버깅 도구를 구성할 때 포트 전달 명령을 수행한 로컬 시스템의 DNS 및 IP 설정을 제공합니다.

---

결과

vRealize Orchestrator Appliance에 대한 서버 디버깅이 구성되었습니다.

## vRealize Orchestrator Appliance 디스크 크기 조정

vSphere에서 vRealize Orchestrator Appliance 가상 시스템의 디스크 크기 설정을 편집하여 vRealize Orchestrator Appliance의 디스크 크기를 수정할 수 있습니다.

사전 요구 사항

vRealize Orchestrator Appliance SSH 서비스를 사용하도록 설정되어 있는지 확인합니다. [vRealize Orchestrator Appliance에 대한 SSH 액세스 사용 또는 사용 안 함](#)의 내용을 참조하십시오.

## 절차

- 1 vRealize Orchestrator Appliance에서 현재 사용 가능한 디스크 공간을 확인합니다.

**참고** vRealize Orchestrator Appliance 디스크에는 최소 20%의 사용 가능한 디스크 공간이 필요합니다.

- a SSH를 통해 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- b `vracli disk-mgr` 명령을 실행합니다.

- 2 vSphere에서 vRealize Orchestrator Appliance 가상 시스템의 디스크 크기를 조정합니다.

- a vSphere Client에 **관리자**로 로그인합니다.
- b 가상 시스템을 오른쪽 버튼으로 클릭하고 **설정 편집**을 선택합니다.
- c **가상 하드웨어** 탭에서 **하드 디스크**를 확장하여 디스크 설정을 보고 변경한 후 **확인**을 클릭합니다.  
vSphere 가상 시스템의 디스크 크기 변경에 대한 자세한 내용은 "vSphere 가상 시스템 관리"에서 "가상 디스크 구성 변경"을 참조하십시오.

- 3 Photon OS에서 자동 크기 조정을 트리거합니다.

- a SSH를 통해 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- b `/var/vmware/prelude/disk-management` 디렉토리로 이동합니다.

```
cd /var/vmware/prelude/disk-management
```

- c `disk_stats` 파일을 열고 파일의 콘텐츠를 수정합니다.

```
/dev/sda: 1
/dev/sdb: 1
/dev/sdc: 1
/dev/sdd: 1
```

- d `vracli disk-mgr resize` 명령을 실행합니다.

**참고** `/var/log/vmware/prelude/disk_resize.log`에서 디스크 크기 조정 절차의 진행률을 추적할 수 있습니다.

vRealize Orchestrator Appliance 디스크의 크기가 조정되었습니다.

- 4 `disk-mgr` 명령을 실행하여 디스크의 크기 조정 절차가 성공했는지 확인합니다.

```
vracli disk-mgr
```

다음에 수행할 작업

디스크 크기 조정 절차 문제를 해결하려면 [KB 79925](#)를 참조하십시오.

## vRealize Orchestrator 서버의 힙 메모리 크기를 조정하는 방법

values.yaml 파일을 편집하여 vRealize Orchestrator 서버의 힙 메모리 크기를 조정할 수 있습니다.

vRealize Orchestrator 서버의 힙 메모리 크기를 조정하여 오케스트레이션 환경에서 변화하는 워크로드를 관리할 수 있습니다. 예를 들어 여러 vCenter Server를 관리할 계획이면 vRealize Orchestrator 배포의 힙 메모리를 늘릴 수 있습니다.

사전 요구 사항

- SSH를 통해 vRealize Orchestrator Appliance에 액세스할 수 있도록 설정합니다. [vRealize Orchestrator Appliance에 대한 SSH 액세스 사용 또는 사용 안 함](#)의 내용을 참조하십시오.
- vRealize Orchestrator가 배포된 가상 시스템의 RAM을 다음에 적절한 증분까지 늘립니다. vSphere에서 가상 시스템의 RAM을 늘리는 방법에 대한 자세한 내용은 "vSphere 가상 시스템 관리"에서 "메모리 구성 변경"을 참조하십시오.

절차

- 1 SSH를 통해 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 /opt/charts/vco/ 디렉토리로 이동합니다.
- 3 기본 편집기를 사용하여 values.yaml 파일을 편집합니다.

```
vi values.yaml
```

- 4 serverMemoryLimit, serverMemoryRequest 및 serverJvmHeapMax 매개 변수를 수정합니다.
  - a serverJvmHeapMax 매개 변수를 편집하여 힙 메모리 값을 설정합니다.
  - b serverMemoryLimit 및 serverMemoryRequest 매개 변수 값을 업데이트합니다.

**경고** serverMemoryLimit 매개 변수 값은 serverJvmHeapMax 매개 변수에 설정된 값보다 2GB 높은 값이어야 합니다. serverMemoryRequest 매개 변수 값은 serverJvmHeapMax 매개 변수에 설정된 값보다 1GB 높은 값이어야 합니다. 메모리 구성의 예는 다음과 같습니다:

```
serverMemoryLimit: 8G
serverMemoryRequest: 7G
serverJvmHeapMax: 6G
```

**참고** 클러스터링된 환경의 경우 클러스터의 모든 노드에서 이전 단계를 수행합니다.

- 5 변경 내용을 values.yaml 파일에 저장하고 /opt/scripts 디렉토리로 이동합니다.
- 6 deploy.sh 명령을 실행합니다.

결과

vRealize Orchestrator 서버의 힙 메모리 크기가 변경되었습니다.

## Site Recovery Manager를 사용한 vRealize Orchestrator의 재해 복구

vRealize Orchestrator를 보호하려면 Site Recovery Manager를 구성해야 합니다. Site Recovery Manager에 대한 일반 구성 작업을 완료하여 해당 보호를 구현합니다.

### 환경 준비

Site Recovery Manager 구성을 시작하기 전에 다음 사전 요구 사항을 충족하는지 확인하십시오.

- 보호된 사이트 및 복구 사이트에 vSphere 6.0 이상이 설치되어 있는지 확인합니다.
- Site Recovery Manager 8.1 이상을 사용하는지 확인합니다.
- vRealize Orchestrator가 구성되어 있는지 확인합니다.

### vSphere Replication에 대한 가상 시스템 구성

Site Recovery Manager를 사용하려면 vSphere Replication에 대한 가상 시스템 또는 어레이 기반 복제를 구성해야 합니다.

다음 단계에 따라 필요한 가상 시스템에서 vSphere Replication를 활성화합니다.

#### 절차

- 1 vSphere Web Client에서 vSphere Replication이 활성화되어야 하는 가상 시스템을 선택하고 **작업 > 모든 vSphere Replication 작업 > 복제 구성**을 클릭합니다.
- 2 **복제 유형** 창에서 **vCenter Server로 복제**를 선택하고 **다음**을 클릭합니다.
- 3 **대상 사이트** 창에서 복구 사이트용 vCenter를 선택하고 **다음**을 클릭합니다.
- 4 **복제 서버** 창에서 vSphere Replication서버를 선택하고 **다음**을 클릭합니다.
- 5 **대상 위치** 창에서 **편집**을 클릭하고 복제된 파일이 저장될 대상 데이터스토어를 선택한 후 **다음**을 클릭합니다.
- 6 **복제 옵션** 창의 기본 설정을 유지하고 **다음**을 클릭합니다.
- 7 **복구 설정** 창에서 **복구 지점 목표(RPO)** 및 **특정 시점 인스턴스**에 대한 시간을 입력하고 **다음**을 클릭합니다.
- 8 **완료 준비** 창에서 설정 내용을 확인하고 **완료**를 클릭합니다.
- 9 이 단계를 vSphere Replication이 활성화되어야 하는 모든 가상 시스템에서 반복합니다.

### 보호 그룹 만들기

보호 그룹을 만들어 Site Recovery Manager에서 가상 시스템을 보호할 수 있습니다.

보호 그룹을 폴더로 구성할 수 있습니다. **보호 그룹** 탭에는 보호 그룹 이름이 표시되지만 보호 그룹이 있는 폴더는 표시되지 않습니다. 이름이 같은 두 개의 보호 그룹이 서로 다른 폴더에 있으면 구분하기 어려울 수 있습니다. 따라서 모든 폴더에서 보호 그룹 이름이 고유한지 확인하십시오. 사용자에 따라 일부 폴더에 대한 보기 권한이 없는 환경에서는 보호 그룹 이름의 고유성을 위해 보호 그룹을 폴더에 배치하지 마십시오.

보호 그룹을 만드는 경우 작업이 예상대로 완료될 때까지 기다립니다. **Site Recovery Manager**가 보호 그룹을 만들고 그룹에서 가상 시스템의 보호가 성공적으로 수행되는지 확인합니다.

#### 사전 요구 사항

다음 작업 중 하나를 수행했는지 확인하십시오.

- 어레이 기반 복제를 구성한 데이터스토어에 가상 시스템을 포함했습니다.
- "스토리지 정책 보호 그룹의 사전 요구 사항"의 요구 사항을 충족하며 "Site Recovery Manager 관리" 가이드에서 "스토리지 정책 보호 그룹에 대한 제한"을 검토했습니다.
- 가상 시스템에 vSphere Replication을 구성했습니다.
- 위 사항 전부 또는 일부의 조합을 수행했습니다.

#### 절차

- 1 vSphere Client 또는 vSphere Web Client에서 **Site Recovery > Site Recovery 열기**를 클릭합니다.
- 2 Site Recovery 홈 탭에서 사이트 쌍을 선택하고 **세부 정보 보기**를 클릭합니다.
- 3 **보호 그룹** 탭을 선택하고 **새로 만들기**를 클릭하여 보호 그룹을 생성합니다.
- 4 [이름 및 방향] 페이지에서 보호 그룹의 이름과 설명을 입력하고 방향을 선택한 후 **다음**을 클릭합니다.
- 5 [보호 그룹 유형] 페이지에서 보호 그룹 유형을 선택하고 **다음**을 클릭합니다.

옵션	작업
어레이 기반 복제 보호 그룹 만들기	데이터스토어 그룹(어레이 기반 복제)을 선택하고 어레이 쌍을 선택합니다.
vSphere Replication 보호 그룹 만들기	개별 VM(vSphere Replication)을 선택합니다.
스토리지 정책 보호 그룹 만들기	스토리지 정책(어레이 기반 복제)을 선택합니다.

- 6 데이터스토어 그룹, 가상 시스템 또는 스토리지 정책을 선택하여 보호 그룹에 추가합니다.

옵션	작업
어레이 기반 복제 보호 그룹	데이터스토어 그룹을 선택하고 <b>다음</b> 을 클릭합니다. 데이터스토어 그룹을 선택하면 그룹에 포함된 가상 시스템이 가상 시스템 테이블에 나타납니다.
vSphere Replication 보호 그룹	목록에서 가상 시스템을 선택하고 <b>다음</b> 을 클릭합니다. vSphere Replication에 대해 구성한 가상 시스템 중 아직 보호 그룹에 포함되지 않은 가상 시스템만 목록에 표시됩니다.
스토리지 정책 보호 그룹	목록에서 스토리지 정책을 선택하고 <b>다음</b> 을 클릭합니다.

## 7 필요한 경우 [복구 계획] 페이지에서 복구 계획에 보호 그룹을 추가할 수 있습니다.

옵션	작업
기존 복구 계획에 추가	기존 복구 계획에 보호 그룹을 추가합니다.
새 복구 계획에 추가	새 복구 계획에 보호 그룹을 추가합니다. 이 옵션을 선택하는 경우 복구 계획 이름을 입력해야 합니다.
지금은 복구 계획에 추가하지 마십시오.	보호 그룹을 복구 계획에 추가하지 않으려면 이 옵션을 선택합니다.

## 8 설정을 검토하고 **마침**을 클릭합니다.

**보호 그룹** 탭에서 보호 그룹 생성 진행률을 모니터링할 수 있습니다.

- 어레이 기반 복제 및 vSphere Replication 보호 그룹의 경우, Site Recovery Manager가 보호된 가상 시스템에 인벤토리 매핑을 적용했으면, 보호 그룹에 대한 보호 상태는 **정상**입니다.
- 스토리지 정책 보호 그룹의 경우 Site Recovery Manager가 스토리지 정책과 연결된 모든 가상 시스템을 성공적으로 보호했으면 보호 그룹의 보호 상태는 **정상**입니다.
- 어레이 기반 복제 및 vSphere Replication 보호 그룹의 경우, 인벤토리 매핑을 구성하지 않았거나 Site Recovery Manager가 인벤토리 매핑을 적용하지 못했으면 보호 그룹의 보호 상태는 **구성되지 않음**입니다.
- 스토리지 정책 보호 그룹의 경우, Site Recovery Manager가 스토리지 정책과 연결된 모든 가상 시스템을 보호할 수 없으면 보호 그룹의 보호 상태는 **구성되지 않음**입니다.

다음에 수행할 작업

어레이 기반 복제 및 vSphere Replication 보호 그룹의 경우, 보호 그룹의 보호 상태가 **구성되지 않음**이면 가상 시스템에 인벤토리 매핑을 적용합니다.

- 사이트 전체 인벤토리 매핑을 적용하거나 이미 설정한 인벤토리 매핑이 유효한지 확인하려면 "Site Recovery Manager 관리" 가이드에서 "인벤토리 매핑 구성"을 참조하십시오. 이러한 매핑을 모든 가상 시스템에 적용하려면 "Site Recovery Manager 관리" 가이드에서 "보호 그룹의 모든 멤버에 인벤토리 매핑 적용"을 참조하십시오.
- 보호 그룹의 각 가상 시스템에 인벤토리 매핑을 개별적으로 적용하려면 "Site Recovery Manager 관리" 가이드에서 "보호 그룹의 개별 가상 시스템에 대해 인벤토리 매핑 구성"을 참조하십시오.

스토리지 정책 보호 그룹의 경우, 보호 그룹의 보호 상태가 **구성되지 않음**이면 "스토리지 정책 보호 그룹의 사전 요구 사항"을 충족하는지 확인하고 "Site Recovery Manager 관리" 가이드에서 "스토리지 정책 보호 그룹의 제한 사항"을 검토했는지 확인하십시오.

## 복구 계획 만들기

복구 계획을 만들어 Site Recovery Manager의 가상 시스템 복구 방식을 설정합니다.

## 절차

- 1 vSphere Client 또는 vSphere Web Client에서 **Site Recovery > Site Recovery 열기**를 클릭합니다.
- 2 Site Recovery 홈 탭에서 사이트 쌍을 선택하고 **세부 정보 보기**를 클릭합니다.
- 3 **복구 계획** 탭을 선택하고 **새로 만들기**를 클릭하여 복구 계획을 만듭니다.
- 4 계획의 이름, 설명 및 방향을 입력하고 폴더를 선택한 후 **다음**을 클릭합니다.
- 5 메뉴에서 그룹 유형을 선택합니다.

옵션	설명
개별 VM 또는 데이터스토어 그룹의 보호 그룹	이 옵션을 선택하여 어레이 기반 복제 및 vSphere Replication 보호 그룹을 포함하는 복구 계획을 작성합니다.
스토리지 정책 보호 그룹	이 옵션을 선택하여 스토리지 정책 보호 그룹을 포함하는 복구 계획을 작성합니다. 확장된 스토리지를 사용하는 경우에는 이 옵션을 선택합니다.

- 6 복구할 계획에 대한 하나 이상의 보호 그룹을 선택하고 **다음**을 클릭합니다.
- 7 **테스트 네트워크** 드롭다운 메뉴에서 테스트 복구 중에 사용할 네트워크를 선택한 후 **다음**을 클릭합니다.

사이트 수준 매핑이 없는 경우 기본 옵션인 **사이트 수준 매핑 사용**은 분리된 테스트 네트워크를 생성합니다.

- 8 요약 정보를 검토하고 **마침**을 클릭하여 복구 계획을 만듭니다.

## 복구 계획을 폴더로 구성

복구 계획에 대한 여러 사용자나 그룹의 액세스를 제어하기 위해 복구 계획을 폴더로 구성할 수 있습니다.

복구 계획이 많으면, 복구 계획을 폴더로 구성하는 것이 유용합니다. 복구 계획을 폴더에 배치하고, 폴더에 대한 사용 권한을 여러 사용자나 그룹에 서로 다르게 할당하여 복구 계획에 대한 액세스를 제한할 수 있습니다. 폴더에 사용 권한을 할당하는 방법에 대한 자세한 내용은 "Site Recovery Manager 관리" 가이드의 "Site Recovery Manager 역할 및 사용 권한 할당"을 참조하십시오.

## 절차

- 1 **Site Recovery** 홈 탭에서 사이트 쌍을 선택하고 **세부 정보 보기**를 클릭합니다.
- 2 **복구 계획** 탭을 클릭하고 왼쪽 창에서 **복구 계획**을 마우스 오른쪽 버튼으로 클릭한 후 **새 폴더**를 클릭합니다.
- 3 만들 폴더의 이름을 입력하고 **추가**를 클릭합니다.



#### 4 새 복구 계획 또는 기존 복구 계획을 폴더에 추가합니다.

옵션	설명
새 복구 계획 만들기	폴더를 마우스 오른쪽 버튼으로 클릭하고 <b>새 복구 계획</b> 을 선택합니다.
기존 복구 계획 추가	인벤토리 트리의 복구 계획을 마우스 오른쪽 버튼으로 클릭하고 <b>이동</b> 을 클릭합니다. 대상 폴더를 선택하고 <b>이동</b> 을 클릭합니다.

## 복구 계획 편집

복구 계획을 편집하여 해당 복구 계획을 만들 때 지정한 속성을 변경할 수 있습니다. 보호된 사이트 또는 복구 사이트에서 복구 계획을 편집할 수 있습니다.

### 절차

- 1 vSphere Client 또는 vSphere Web Client에서 **Site Recovery > Site Recovery 열기**를 클릭합니다.
- 2 **Site Recovery** 홈 탭에서 사이트 쌍을 선택하고 **세부 정보 보기**를 클릭합니다.
- 3 **복구 계획** 탭을 클릭하고 복구 계획을 마우스 오른쪽 버튼으로 클릭한 후 **편집**을 클릭합니다.
- 4 (선택 사항) 계획의 이름 또는 설명을 편집하고 **다음**을 클릭합니다.  
복구 계획의 방향과 위치는 변경할 수 없습니다.
- 5 (선택 사항) 복구 계획에 추가할 하나 이상의 보호 그룹을 선택 또는 선택 해제하거나 계획에서 보호 그룹을 제거하고 **다음**을 클릭합니다.
- 6 (선택 사항) 드롭다운 메뉴에서 복구 사이트의 다른 테스트 네트워크를 선택하고 **다음**을 클릭합니다.
- 7 요약 정보를 검토하고 **마침**을 클릭하여 복구 계획에 대해 지정된 변경 내용을 수행합니다.  
**최근 작업** 보기에서 계획 업데이트를 모니터링할 수 있습니다.

# 시스템 속성 설정

## 9

시스템 속성을 설정하여 기본 Orchestrator 동작을 변경할 수 있습니다.

본 장은 다음 항목을 포함합니다.

- 워크플로 및 작업에 대한 서버 파일 시스템 액세스 설정
- 워크플로 및 작업에 대한 운영 체제 명령의 액세스 설정
- Java 클래스에 JavaScript 액세스 설정
- 사용자 지정 시간 초과 속성 설정
- vRealize Orchestrator SQL 플러그인을 위한 JDBC 커넥터 추가

## 워크플로 및 작업에 대한 서버 파일 시스템 액세스 설정

vRealize Orchestrator에서 워크플로 및 작업은 특정 파일 시스템 디렉토리에 대해 액세스가 제한되어 있습니다. `js-io-rights.conf` 구성 파일을 수정하여 서버 파일 시스템의 다른 부분으로 액세스를 확장할 수 있습니다.

### vRealize Orchestrator 시스템에 대한 쓰기 액세스 권한을 허용하는 `js-io-rights.conf` 파일의 규칙

`js-io-rights.conf` 파일은 서버 파일 시스템의 정의된 디렉토리에 대한 쓰기 액세스 권한을 허용하는 규칙을 포함합니다.

#### `js-io-rights.conf` 파일의 필수 콘텐츠

`js-io-rights.conf` 파일의 각 줄은 다음 정보를 포함해야 합니다.

- 권한이 허용되는지 거부되는지를 나타내는 더하기(+) 또는 빼기(-) 기호
- 읽기(r), 쓰기(w) 및 실행(x) 권한 수준

- 권한을 적용할 경로.

**참고** `js-io-rights.conf` 파일의 루트 폴더는 항상 `/var/run/vco`입니다. vRealize Orchestrator Appliance 파일 시스템에서 이 폴더는 `/data/vco/var/run/vco` 아래에 있습니다. vRealize Orchestrator 파일 시스템에 대한 액세스 권한이 있는 모든 콘텐츠는 이 루트 폴더 아래에 매핑되어야 합니다.

## js-io-rights.conf 파일의 기본 콘텐츠

Orchestrator Appliance에 있는 `js-io-rights.conf` 구성 파일의 기본 콘텐츠는 다음과 같습니다.

```
-rwx /
+rwX /var/run/vco
+rx /etc/vco
-rwx /etc/vco/app-server/security/
+rx /var/log/vco/
```

기본 `js-io-rights.conf` 구성 파일에서 처음 두 줄은 다음 액세스 권한을 허용합니다.

**-rwx /**

파일 시스템에 대한 모든 액세스가 거부됩니다.

**+rwX /var/run/vco**

`/var/run/vco` 디렉토리에서 읽기, 쓰기 및 실행 액세스 권한이 허용됩니다.

## js-io-rights.conf 파일의 규칙

vRealize Orchestrator는 `js-io-rights.conf` 파일에 표시되는 순서대로 액세스 권한을 확인합니다. 각 줄은 이전 줄을 재정의할 수 있습니다.

**중요** `js-io-rights.conf` 파일에서 `+rwX /`를 설정하여 파일 시스템의 모든 부분에 대한 액세스 권한을 허용할 수 있습니다. 단, 그렇게 하면 보안 위험이 높습니다.

## 워크플로 및 작업에 대한 서버 파일 시스템 액세스 설정

워크플로 및 vRealize Orchestrator API가 액세스할 수 있는 서버 파일 시스템의 부분을 변경하려면 `js-io-rights.conf` 구성 파일을 수정합니다. `js-io-rights.conf` 파일은 워크플로가 vRealize Orchestrator 서버 파일 시스템에 액세스하려고 시도하면 생성됩니다.

절차

- 1 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- 2 `/data/vco/var/run/vco/` 디렉토리로 이동합니다.
- 3 텍스트 편집기에서 `js-io-rights.conf` 구성 파일을 엽니다.

- 필요한 줄을 `js-io-rights.conf` 파일에 추가하여 파일 시스템 영역에 대한 액세스를 허용하거나 거부합니다.

예를 들어 다음 줄은 `/data/vco/var/run/vco/noexec` 디렉토리에서 실행 권한을 거부합니다.

```
-x /data/vco/var/run/vco/noexec
```

`/data/vco/var/run/vco/noexec`는 실행 권한을 보유하지만, `/data/vco/var/run/vco/noexec/bar`는 보유하지 않습니다. 두 디렉토리 모두 읽기 및 쓰기가 가능합니다.

## 결과

워크플로 및 vRealize Orchestrator API에 대해 파일 시스템의 액세스 권한을 수정했습니다.

## 워크플로 및 작업에 대한 운영 체제 명령의 액세스 설정

vRealize Orchestrator API는 vRealize Orchestrator 서버 호스트 운영 체제에서 명령을 실행하는 스크립팅 클래스인 `Command`를 제공합니다. 서버 호스트에 대한 무단 액세스를 방지하기 위해 vRealize Orchestrator 애플리케이션에는 기본적으로 `Command` 클래스를 실행할 권한이 없습니다. vRealize Orchestrator 애플리케이션이 호스트 운영 체제에서 명령을 실행하는 데 권한이 필요한 경우 `Command` 스크립팅 클래스를 활성화할 수 있습니다.

vRealize Orchestrator 구성 시스템 속성을 설정하여 `Command` 클래스에 대한 사용 권한을 부여할 수 있습니다.

## 절차

- 제어 센터에 **root**로 로그인합니다.
- 시스템 속성**을 클릭합니다.
- 새로 만들기**를 클릭합니다.
- 키** 텍스트 상자에 **com.vmware.js.allow-local-process**를 입력합니다.
- 값** 텍스트 상자에 **true**를 입력합니다.
- 설명** 텍스트 상자에 시스템 속성에 대한 설명을 입력합니다.
- 추가**를 클릭합니다.
- 팝업 메뉴에서 **변경 내용 저장**을 클릭합니다.  
성공적으로 저장했다는 메시지가 표시됩니다.
- vRealize Orchestrator 서버가 다시 시작될 때까지 기다립니다.

## 결과

vRealize Orchestrator 서버 호스트 운영 체제에서 로컬 명령을 실행할 수 있는 권한이 vRealize Orchestrator 애플리케이션에 부여되었습니다.

**참고** `com.vmware.js.allow-local-process` 시스템 속성을 `true`로 설정하면 Command 스크립팅 클래스가 파일 시스템의 모든 위치에서 쓰기가 허용됩니다. 이 속성은 `js-io-rights.conf` 파일에서 Command 스크립팅 클래스 전용으로 설정한 모든 파일 시스템 액세스 권한을 재정의합니다. `js-io-rights.conf` 파일에서 설정한 파일 시스템 액세스 권한은 여전히 Command 외 모든 스크립팅 클래스에 적용됩니다.

## Java 클래스에 JavaScript 액세스 설정

기본적으로 vRealize Orchestrator에서는 제한된 Java 클래스 집합으로 JavaScript 액세스가 제한됩니다. 보다 광범위한 Java 클래스에 대한 JavaScript 액세스가 필요한 경우 vRealize Orchestrator 시스템 속성을 설정해야 합니다.

JavaScript 엔진에 Java 가상 시스템(JVM)에 대한 전체 액세스 권한을 허용하면 잠재적 보안 문제가 나타납니다. vRealize Orchestrator 서버를 실행하는 사용자가 액세스할 수 있는 모든 시스템 구성 요소에 잘못된 형식 또는 악성 스크립트가 액세스할 수 있습니다. 따라서 vRealize Orchestrator JavaScript 엔진은 기본적으로 `java.util.*` 패키지의 클래스에만 액세스할 수 있습니다.

JavaScript가 `java.util.*` 패키지 외부 클래스에 액세스하게 해야 한다면 JavaScript 액세스를 허용할 Java 패키지를 구성 파일에 나열하십시오. 그런 다음 이 파일을 가리키도록 `com.vmware.scripting.rhino-class-shutter-file` 시스템 속성을 설정하십시오.

### 절차

- 1 JavaScript 액세스를 허용할 Java 패키지 목록을 저장하려면 텍스트 구성 파일을 작성합니다.

예를 들어 JavaScript가 `java.net` 패키지의 모든 클래스와 `java.lang.Object` 클래스에 액세스하게 하려면 다음 내용을 파일에 추가합니다.

```
java.net.*
java.lang.Object
```

- 2 구성 파일의 이름을 입력합니다.
- 3 구성 파일을 `/data/vco/usr/lib/vco` 하위 디렉토리에 저장합니다.

**참고** 다른 디렉토리에는 구성 파일을 저장할 수 없습니다.

- 4 제어 센터에 **root**로 로그인합니다.
- 5 **시스템 속성**을 클릭합니다.
- 6 **새로 만들기**를 클릭합니다.
- 7 키 텍스트 상자에 `com.vmware.scripting.rhino-class-shutter-file`을 입력합니다.
- 8 값 텍스트 상자에 `vco/usr/lib/vco/your_configuration_file_subdirectory`를 입력합니다.

**9 설명** 텍스트 상자에 시스템 속성에 대한 설명을 입력합니다.

**10 추가**를 클릭합니다.

**11** 팝업 메뉴에서 **변경 내용 저장**을 클릭합니다.

성공적으로 저장했다는 메시지가 표시됩니다.

**12** vRealize Orchestrator 서버가 다시 시작될 때까지 기다립니다.

결과

JavaScript 엔진이 지정한 Java 클래스에 액세스할 수 있습니다.

## 사용자 지정 시간 초과 속성 설정

vCenter Server가 오버로드되면 응답을 vRealize Orchestrator 서버로 반환하는 시간이 기본적으로 설정된 20000밀리초보다 더 걸립니다. 이런 상황을 방지하려면 기본 시간 초과 기간을 늘리도록 vRealize Orchestrator 구성 파일을 수정해야 합니다.

특정 작업이 완료되기 전에 기본 시간 초과 기간이 만료되면 vRealize Orchestrator 서버 로그에 오류가 포함됩니다.

```
Operation 'getPropertyContent' total time : '5742228' for 1823 calls, mean time : '3149.0', min time : '0', max time : '32313' Timeout, unable to get property 'info' com.vmware.vmo.plugin.vi4.model.TimeoutException
```

절차

**1** 제어 센터에 **root**로 로그인합니다.

**2** **시스템 속성**을 클릭합니다.

**3** **새로 만들기**를 클릭합니다.

**4** 키 텍스트 상자에 **com.vmware.vmo.plugin.vi4.waitUpdatesTimeout**을 입력합니다.

**5** 값 텍스트 상자에 새 시간 초과 기간을 밀리초 단위로 입력합니다.

**6** (선택 사항) **설명** 텍스트 상자에 시스템 속성에 대한 설명을 입력합니다.

**7** **추가**를 클릭합니다.

**8** 팝업 메뉴에서 **변경 내용 저장**을 클릭합니다.

성공적으로 저장했다는 메시지가 표시됩니다.

**9** Orchestrator 서버를 다시 시작합니다.

결과

설정된 값이 기본 시간 초과로 설정된 값인 20000밀리초를 재정의합니다.

## vRealize Orchestrator SQL 플러그인을 위한 JDBC 커넥터 추가

이 예는 vRealize Orchestrator SQL 플러그인을 위한 MySQL 커넥터를 추가하는 방법을 보여줍니다.

절차

1 MySQL connector.jar 파일을 vRealize Orchestrator Appliance에 추가합니다.

- a SSH를 통해 vRealize Orchestrator Appliance 명령줄에 **root**로 로그인합니다.
- b /data/vco/var/run/vco 디렉토리로 이동합니다.

```
cd /data/vco/var/run/vco
```

c plugins/SQL/lib/ 디렉토리를 생성합니다.

```
mkdir -p plugins/SQL/lib/
```

d SCP(Secure Copy) 명령을 실행하여 로컬 시스템에서 /data/vco/var/run/vco/plugins/SQL/lib/ 디렉토리로 MySQL connector.jar 파일을 복사합니다.

```
scp ~/local_machine_dir/your_mysql_connector.jar root@orchestrator_FQDN_or_IP:/data/vco/var/run/vco/plugins/SQL/lib/
```

**참고** PSCP와 같은 대체 방법을 사용하여 connector.jar 파일을 vRealize Orchestrator Appliance에 복사할 수도 있습니다.

2 새 MySQL 속성을 제어 센터에 추가합니다.

- a 제어 센터에 **root**로 로그인합니다.
- b **시스템 속성**을 선택합니다.
- c **새로 만들기**를 클릭합니다.
- d **키**에서 **o11n.plugin.SQL.classpath**를 입력합니다.
- e **값**에서 **/var/run/vco/plugins/SQL/lib/your\_mysql\_connector.jar**을 입력합니다.

**참고** 값 텍스트 상자에는 여러 JDBC 커넥터가 포함될 수 있습니다. 각 JDBC 커넥터는 세미콜론(;)으로 구분됩니다. 예:

```
/var/run/vco/plugins/SQL/lib/your_mysql_connector.jar;/var/run/vco/plugins/SQL/lib/your_mssql_connector.jar;/var/run/vco/plugins/SQL/lib/your_other_connector.jar
```

- f (선택 사항) MySQL 시스템 속성에 대한 설명을 입력합니다.
- g **추가**를 클릭하고 vRealize Orchestrator 서버가 다시 시작될 때까지 기다립니다.

---

**참고** 다른 디렉토리에 JDBC connector.jar 파일을 저장하거나 `o11n.plugin.SQL.classpath` 속성에 다른 값을 설정하지 마십시오. 그렇지 않으면 vRealize Orchestrator 배포에서 JDBC 커넥터를 사용할 수 없게 됩니다.

---



vRealize Orchestrator를 설치하고 구성한 경우 vRealize Orchestrator를 사용하여 가상 환경 관리와 관련하여 자주 반복되는 프로세스를 자동화할 수 있습니다.

- vRealize Orchestrator Client에 로그인하여 vCenter Server 인벤토리 개체 또는 vRealize Orchestrator가 플러그인을 통해 액세스하는 기타 개체에서 워크플로를 실행하고 스케줄링합니다. "VMware vRealize Orchestrator 클라이언트 사용"을 참조하십시오.
- 표준 vRealize Orchestrator 워크플로를 복제 및 수정하고 vCenter Server의 작업을 자동화하도록 자체 작업 및 워크플로를 작성합니다.
- vRealize Orchestrator 플랫폼의 기능을 확장하려면 플러그인을 개발합니다.
- 원격 Git 저장소 통합을 사용하여 여러 vRealize Orchestrator 인스턴스에서 vRealize Orchestrator 인벤토리를 관리합니다. "VMware vRealize Orchestrator 클라이언트 사용"을 참조하십시오.
- vSphere Web Client를 사용하여 vSphere 인벤토리 개체에서 워크플로를 실행합니다.