OpenShift Dedicated 4

인증 및 권한 부여

OpenShift Dedicated 보안.

Last Updated: 2024-09-07

OpenShift Dedicated 4 인증 및 권한 부여

OpenShift Dedicated 보안.

법적 공지

초록

이 문서에서는 OpenShift Dedicated 클러스터 보안에 대한 정보를 제공합니다.

차례

1장. 인증 및 권한 부여 개요

1.1. OPENSHIFT DEDICATED 인증 및 권한 부여에 대한 일반 용어집

이 용어집은 OpenShift Dedicated 인증 및 권한 부여에 사용되는 일반적인 용어를 정의합니다.

인증

인증은 OpenShift Dedicated 클러스터에 대한 액세스를 결정하고 인증된 사용자만 OpenShift Dedicated 클러스터에 액세스하도록 합니다.

권한 부여

권한 부여는 식별된 사용자에게 요청된 작업을 수행할 수 있는 권한이 있는지 여부를 결정합니다.

Bearer 토큰

Bearer 토큰은 헤더 **Authorization: Bearer <token>**을 사용하여 API에 인증하는 데 사용됩니다.

구성 맵

구성 맵에서는 구성 데이터를 Pod에 삽입하는 방법을 제공합니다. 구성 맵에 저장된 데이터를 **ConfigMap** 유형의 볼륨에서 참조할 수 있습니다. Pod에서 실행되는 애플리케이션에서는 이 데이터를 사용할 수 있습니다.

컨테이너

소프트웨어와 모든 종속 항목으로 구성된 경량 및 실행 가능한 이미지입니다. 컨테이너는 운영 체제를 가상화하므로 데이터 센터, 퍼블릭 또는 프라이빗 클라우드 또는 로컬 호스트에서 컨테이너를 실행할 수 있습니다.

CR(사용자 정의 리소스)

CR은 Kubernetes API의 확장입니다.

group

그룹은 사용자 집합입니다. 그룹은 여러 사용자에게 한 번 권한을 부여하는 데 유용합니다.

HTPasswd

htpasswd는 HTTP 사용자 인증을 위해 사용자 이름 및 암호를 저장하는 파일을 업데이트합니다.

Keystone

Keystone은 ID, 토큰, 카탈로그 및 정책 서비스를 제공하는 RHOSP(Red Hat OpenStack Platform) 프로젝트입니다.

LDAP(Lightweight Directory Access Protocol)

LDAP는 사용자 정보를 쿼리하는 프로토콜입니다.

네임스페이스

네임스페이스는 모든 프로세스에 표시되는 특정 시스템 리소스를 격리합니다. 네임스페이스 내에서 해당 네임스페이스의 멤버인 프로세스만 해당 리소스를 볼 수 있습니다.

노드

노드는 OpenShift Dedicated 클러스터의 작업자 머신입니다. 노드는 VM(가상 머신) 또는 물리적 머신입니다.

OAuth 클라이언트

OAuth 클라이언트는 bearer 토큰을 얻는 데 사용됩니다.

OAuth 서버

OpenShift Dedicated 컨트롤 플레인에는 구성된 ID 공급자의 사용자 ID를 결정하고 액세스 토큰을 생성하는 기본 제공 OAuth 서버가 포함되어 있습니다.

OpenID Connect

OpenID Connect는 사용자가 SSO(Single Sign-On)를 사용하여 OpenID 공급자를 사용하는 사이트에 액세스하도록 사용자를 인증하는 프로토콜입니다.

Pod

Pod는 Kubernetes에서 가장 작은 논리 단위입니다. Pod는 작업자 노드에서 실행할 하나 이상의 컨테이너로 구성됩니다.

일반 사용자

처음 로그인하거나 API를 통해 클러스터에서 자동으로 생성되는 사용자입니다.

요청 헤더

요청 헤더는 HTTP 요청 컨텍스트에 대한 정보를 제공하는 데 사용되는 HTTP 헤더이므로 서버가 요청의 응답을 추적할 수 있습니다.

RBAC(역할 기반 액세스 제어)

클러스터 사용자와 워크로드가 역할을 실행하는 데 필요한 리소스에만 액세스할 수 있도록 하는 주요 보안 제어입니다.

서비스 계정

서비스 계정은 클러스터 구성 요소 또는 애플리케이션에서 사용합니다.

시스템 사용자

클러스터가 설치될 때 자동으로 생성되는 사용자입니다.

사용자

사용자는 API에 요청할 수 있는 엔티티입니다.

1.2. OPENSHIFT DEDICATED의 인증 정보

OpenShift Dedicated 클러스터에 대한 액세스를 제어하려면 **dedicated-admin** 역할의 관리자는 사용자 인증을 구성하고 승인된 사용자만 클러스터에 액세스할 수 있습니다.

OpenShift Dedicated 클러스터와 상호 작용하려면 사용자가 먼저 OpenShift Dedicated API에 대해 인증해야 합니다. OpenShift Dedicated API에 대한 요청에 OAuth 액세스 토큰 또는 X.509 클라이언트 인증서를 제공하여 인증할 수 있습니다.

참고

유효한 액세스 토큰 또는 인증서가 없으면 요청이 인증되지 않고 HTTP 401 오류가 발생합니다.

관리자는 ID 공급자를 구성하여 인증을 구성할 수 있습니다. OpenShift Dedicated에서 지원되는 모든 ID 공급자 를 정의하여 클러스터에 추가할 수 있습니다.

1.3. OPENSHIFT DEDICATED의 권한 부여 정보

권한 부여는 식별된 사용자에게 요청된 작업을 수행할 수 있는 권한을 갖는지 여부를 결정하는 것입니다.

관리자는 권한을 정의하고 규칙, 역할, 바인딩과 같은 RBAC 오브젝트 를 사용하여 사용자에게 할당할 수 있습니다. OpenShift Dedicated에서 권한 부여 작동 방식을 이해하려면 권한 평가 를 참조하십시오.

프로젝트 및 네임스페이스 를 통해 OpenShift Dedicated 클러스터에 대한 액세스를 제어할 수도 있습니다.

클러스터에 대한 사용자 액세스 제어와 함께 Pod에서 수행할 수 있는 작업과 SCC(보안 컨텍스트 제약 조건) 를 사용하여 액세스할 수 있는 리소스를 제어할 수도 있습니다.

다음 작업을 통해 OpenShift Dedicated에 대한 권한 부여를 관리할 수 있습니다.

- 로컬 및 클러스터 역할 및 바인딩을 확인합니다.
- 로컬 역할을 생성하고 사용자 또는 그룹에 할당합니다.
- 사용자 또는 그룹에 클러스터 역할 할당: OpenShift Dedicated에는 기본 클러스터 역할 세트가 포함되어 있습니다. 사용자 또는 그룹에 추가할 수 있습니다.
- 사용자에게 관리자 권한 부여: 사용자에게 **전용 관리자** 권한을 부여할 수 있습니다.
- 서비스 계정 생성: 서비스 계정은 일반 사용자의 자격 증명을 공유하지 않고도 API 액세스를 유연하게 제어할 수 있는 방법을 제공합니다. 사용자는 애플리케이션에서 서비스 계정을 생성하고 OAuth 클라이언트로 사용할 수 있습니다.
- 범위 지정 토큰: 범위가 지정된 토큰은 특정 작업만 수행할 수 있는 특정 사용자로 식별하는 토큰입니다. 범위가 지정된 토큰을 생성하여 일부 권한을 다른 사용자 또는 서비스 계정에 위임할 수 있습니다.
- LDAP 그룹 동기화: LDAP 서버에 저장된 그룹을 OpenShift Dedicated 사용자 그룹과 동기화 하여 한 곳에서 사용자 그룹을 관리할 수 있습니다.

2장. 인증 이해

사용자가 OpenShift Dedicated와 상호 작용하려면 먼저 클러스터에 인증해야 합니다. 인증 계층은 OpenShift Dedicated API에 대한 요청과 관련된 사용자를 식별합니다. 그런 다음 권한 부여 계층에서 요청한 사용자에 대한 정보로 요청의 허용 여부를 결정합니다.

2.1. 사용자

OpenShift Dedicated의 *사용자*는 OpenShift Dedicated API에 요청할 수 있는 엔티티입니다. OpenShift Dedicated **User** 오브젝트는 시스템에서 역할을 추가하여 권한을 부여할 수 있는 작업자 또는 작업자 그룹을 나타냅니다. 일반적으로 OpenShift Dedicated와 상호 작용하는 개발자 또는 관리자 계정을 나타냅니다.

다음과 같이 여러 유형의 사용자가 존재할 수 있습니다.

사용자 유형	설명
Regular users	이는 대부분의 대화형 OpenShift Dedicated 사용자를 표시하는 방법입니다. 일반 사용자는 처음 로그인할 때 시스템에서 자동으로 생성되며 API를 통해 생성할 수도 있습니다. 일반 사용자는 **User** 오브젝트로 표시됩니다. 예:**joe alice**
System users	대부분의 시스템 사용자는 주로 인프라와 API 간의 안전한 상호 작용을 목적으로 인프라가 정의될 때 자동으로 생성됩니다. 여기에는 클러스터 관리자(전체 액세스 권한 보유), 노드별 사용자, 라우터 및 레지스트리용 사용자를 비롯하여 기타 다양한 사용자가 포함됩니다. 마지막으로, 인증되지 않은 요청에 기본적으로 사용되는 **anonymous** 시스템 사용자가 있습니다. 예: **system:admin system:openshift-registry system:node:node1.example.com**
Service accounts	프로젝트와 관련된 특수한 시스템 사용자입니다. 일부는 프로젝트가 처음 생성될 때 자동으로 생성되지만 프로젝트 관리자가 각 프로젝트 콘텐츠에 대한 액세스 권한을 정의하기 위해 추가로 생성할 수도 있습니다. 서비스 계정은 **ServiceAccount** 오브젝트로 표시됩니다. 예: **system:serviceaccount:default:deployer system:serviceaccount:foo:builder**

각 사용자는 OpenShift Dedicated에 액세스하려면 어떤 방식으로든 인증해야 합니다. 인증되지 않았거나 인증이 유효하지 않은 API 요청은 **anonymous** 시스템 사용자의 요청으로 인증됩니다. 인증 후 정책에 따라 사용자가 수행할 수 있는 작업이 결정됩니다.

2.2. 그룹

사용자는 하나 이상의 *그룹*에 할당될 수 있으며, 각 그룹은 특정 사용자 집합을 나타냅니다. 그룹은 권한 부여 정책을 관리하여 여러 사용자에게 한꺼번에 권한을 부여할 때 유용합니다(예: 사용자에게 개별적으로 오브젝트 액세스 권한을 부여하는 대신 특정 프로젝트 내의 여러 오브젝트에 대한 액세스 허용).

명시적으로 정의된 그룹 외에도 클러스터에서 자동으로 프로비저닝하는 시스템 그룹 또는 *가상 그룹*이 있습니다.

다음과 같은 기본 가상 그룹이 가장 중요합니다.

가상 그룹	설명
system:authenticated	인증된 모든 사용자와 자동으로 연결됩니다.
system:authenticated:oauth	OAuth 액세스 토큰을 사용하여 인증된 모든 사용자와 자동으로 연결됩니다.
system:unauthenticated	인증되지 않은 모든 사용자와 자동으로 연결됩니다.

2.3. API 인증

OpenShift Dedicated API에 대한 요청은 다음 방법을 사용하여 인증됩니다.

OAuth 액세스 토큰

- <namespace_route>/oauth/ **authorize 및 *<namespace_route>* / oauth /token** 끝점을 사용하여 OpenShift Dedicated OAuth 서버에서 가져옵니다.
- ***Authorization: Bearer...*** *헤더로 전송됩니다.*
- *WebSocket 요청의 경우* ***base64url.bearer.authorization.k8s.io.<base64url-encoded-token>*** *형식의 WebSocket 하위 프로토콜 헤더로 전송됩니다.*

X.509 클라이언트 인증서

- *API 서버에 대한 HTTPS 연결이 필요합니다.*
- *신뢰할 수 있는 인증 기관 번들과 대조하여 API 서버에서 확인합니다.*
- *API 서버는 인증서를 작성하고 컨트롤러에 분배하여 자체적으로 인증합니다.*

유효하지 않은 액세스 토큰 또는 유효하지 않은 인증서가 있는 요청은 ***401*** *오류와 함께 인증 계층에서 거부됩니다.*

액세스 토큰이나 인증서가 없는 경우 인증 계층은 ***system:anonymous*** *가상 사용자 및* ***system:unauthenticated*** *가상 그룹을 요청에 할당합니다. 그러면 권한 부여 계층에서 익명 사용자가 할 수 있는 요청(있는 경우)을 결정합니다.*

2.3.1. OpenShift Dedicated OAuth 서버

OpenShift Dedicated 마스터에는 내장 OAuth 서버가 포함되어 있습니다. 사용자는 API 인증을 위해 OAuth 액세스 토큰을 가져옵니다.

사용자가 새 OAuth 토큰을 요청하면 OAuth 서버는 구성된 ID 공급자를 사용하여 요청한 사람의 ID를 확인합니다.

그런 다음 해당 ID와 매핑되는 사용자를 결정하고 그 사용자를 위한 액세스 토큰을 만들어 제공합니다.

2.3.1.1. OAuth 토큰 요청

OAuth 토큰을 요청할 때마다 토큰을 받고 사용할 OAuth 클라이언트를 지정해야 합니다. 다음 OAuth 클라이언트는 OpenShift Dedicated API를 시작할 때 자동으로 생성됩니다.

OAuth 클라이언트	사용법
openshift-browser-client	대화형 로그인을 처리할 수 있는 사용자 에이전트를 사용하여 **<namespace_route>/oauth/token/request**에서 토큰을 요청합니다. [1]
openshift-challenging-client	**WWW-Authenticate** 챌린지를 처리할 수 있는 사용자 에이전트로 토큰을 요청합니다.

1. ***<namespace_route>**는 네임스페이스 경로를 나타냅니다. 다음 명령을 실행하여 확인할 수 있습니다.*

```
$ oc get route oauth-openshift -n openshift-authentication -o json | jq .spec.host
```

*OAuth 토큰에 대한 모든 요청에는 **<namespace_route>/oauth/authorize**에 대한 요청이 포함됩니다. 대부분의 인증 통합에서는 이 끝점 앞에 인증 프록시를 배치하거나 백업 ID 공급자에 대한 인증 정보를 검증하도록 OpenShift Dedicated를 구성합니다. **<namespace_route>/oauth/authorize**에 대한 요청은 CLI와 같은 대화형 로그인 페이지를 표시할 수 없는 사용자 에이전트에서 발생할 수 있습니다. 따라서 OpenShift Dedicated에서는 대화형 로그인 흐름 외에도 **WWW-Authenticate** 챌린지를 사용한 인증을 지원합니다.*

*인증 프록시를 **<namespace_route>/oauth/authorize** 끝점 앞에 배치하면 대화형 로그인 페이지를 표시하거나 대화형 로그인 flows로 리디렉션하는 대신 인증되지 않은 브라우저 이외의 사용자 에이전트 **WWW-Authenticate** 챌린지를 보냅니다.*

참고

*브라우저 클라이언트에 대한 CSRF(Cross-Site Request Forgery) 공격을 방지하려면 **X-CSRF-Token** 헤더가 요청에 있는 경우에만 기본 인증 챌린지를 보냅니다. 기본 **WWW-Authenticate** 챌린지를 받을 것으로 예상되는 클라이언트는 이 헤더를 비어 있지 않은 값으로 설정해야 합니다.*

*인증 프록시가 **WWW-Authenticate** 챌린지를 지원할 수 없거나 OpenShift Dedicated가 WWW-Authenticate 챌린지를 지원하지 않는 ID 공급자를 사용하도록 구성된 경우 브라우저를 사용하여 **<namespace_route>/oauth/token/request** 에서 수동으로 토큰을 가져와야 합니다.*

3장. 사용자 소유 OAUTH 액세스 토큰 관리

사용자는 자체 OAuth 액세스 토큰을 검토하고 더 이상 필요하지 않은 항목을 삭제할 수 있습니다.

3.1. 사용자 소유 OAUTH 액세스 토큰 나열

사용자 소유 OAuth 액세스 토큰을 나열할 수 있습니다. 토큰 이름은 민감하지 않으며 로그인에 사용할 수 없습니다.

절차

- 모든 사용자 소유 OAuth 액세스 토큰 나열:

```
$ oc get useroauthaccesstokens
```

출력 예

```
NAME      CLIENT NAME                CREATED              EXPIRES
REDIRECT URI                                  SCOPES
<token1>  openshift-challenging-client  2021-01-11T19:25:35Z  2021-01-12 19:25:35
+0000 UTC  https://oauth-openshift.apps.example.com/oauth/token/implicit  user:full
<token2>  openshift-browser-client  2021-01-11T19:27:06Z  2021-01-12 19:27:06 +0000
UTC  https://oauth-openshift.apps.example.com/oauth/token/display  user:full
<token3>  console                  2021-01-11T19:26:29Z  2021-01-12 19:26:29 +0000 UTC
https://console-openshift-console.apps.example.com/auth/callback  user:full
```

- 특정 OAuth 클라이언트의 사용자 소유자 OAuth 액세스 토큰 나열:

```
$ oc get useroauthaccesstokens --field-selector=clientName="console"
```

출력 예

```
NAME      CLIENT NAME                CREATED              EXPIRES
REDIRECT URI                                  SCOPES
<token3>  console                  2021-01-11T19:26:29Z  2021-01-12 19:26:29 +0000 UTC
https://console-openshift-console.apps.example.com/auth/callback  user:full
```

3.2. 사용자 소유 OAUTH 액세스 토큰의 세부 정보 보기

사용자 소유 OAuth 액세스 토큰의 세부 정보를 볼 수 있습니다.

절차

- 사용자 소유 OAuth 액세스 토큰의 세부 정보 설명:

```
$ oc describe useroauthaccesstokens <token_name>
```

출력 예

```
Name:                <token_name> 1
Namespace:
```

```
Labels:                 <none>
Annotations:            <none>
API Version:            oauth.openshift.io/v1
Authorize Token:        sha256~Ksckkug-9Fg_RWn_AUysPoIg-_HqmFI9zUL_CgD8wr8
Client Name:            openshift-browser-client (2)
Expires In:             86400 (3)
Inactivity Timeout Seconds:  317 (4)
Kind:                   UserOAuthAccessToken
Metadata:
  Creation Timestamp:  2021-01-11T19:27:06Z
  Managed Fields:
    API Version:  oauth.openshift.io/v1
    Fields Type:  FieldsV1
    fieldsV1:
      f:authorizeToken:
      f:clientName:
      f:expiresIn:
      f:redirectURI:
      f:scopes:
      f:userName:
      f:userUID:
    Manager:        oauth-server
    Operation:      Update
    Time:           2021-01-11T19:27:06Z
  Resource Version:  30535
  Self Link:         /apis/oauth.openshift.io/v1/useroauthaccesstokens/<token_name>
  UID:               f9d00b67-ab65-489b-8080-e427fa3c6181
Redirect URI:        https://oauth-openshift.apps.example.com/oauth/token/display
Scopes:
  user:full (5)
User Name:  <user_name> (6)
User UID:   82356ab0-95f9-4fb3-9bc0-10f1d6a6a345
Events:     <none>
```

(1) 토큰의 sha256 해시인 토큰 이름입니다. 토큰 이름은 민감하지 않으며 로그인에 사용할 수 없습니다.

(2) 토큰이 시작된 위치를 설명하는 클라이언트 이름입니다.

(3) 이 토큰이 만료되기 전 생성 후 값(초)입니다.

(4) OAuth 서버에 대한 토큰 비활성 타임아웃이 설정된 경우 이 토큰을 더 이상 사용할 수 없기 전 생성 후 값(초)입니다.

(5) 이 토큰의 범위입니다.

(6) 이 토큰과 연관된 사용자 이름입니다.

3.3. 사용자 소유 OAUTH 액세스 토큰 삭제

oc logout 명령은 활성 세션에 대해 OAuth 토큰만 무효화합니다. 다음 절차에 따라 더 이상 필요하지 않은 사용자 소유 OAuth 토큰을 삭제할 수 있습니다.

OAuth 액세스 토큰을 삭제하면 토큰을 사용하는 모든 세션에서 사용자를 로그아웃합니다.

절차

- *사용자 소유 OAuth 액세스 토큰 삭제:*

```
$ oc delete useroauthaccesstokens <token_name>
```

출력 예

```
useroauthaccesstoken.oauth.openshift.io "<token_name>" deleted
```

3.4. 클러스터 역할에 인증되지 않은 그룹 추가

클러스터 관리자는 클러스터 역할 바인딩을 생성하여 OpenShift Dedicated의 다음 클러스터 역할에 인증되지 않은 사용자를 추가할 수 있습니다. 인증되지 않은 사용자는 비공용 클러스터 역할에 액세스할 수 없습니다. 이 작업은 필요한 경우에만 특정 사용 사례에서 수행해야 합니다.

인증되지 않은 사용자를 다음 클러스터 역할에 추가할 수 있습니다.

- ***system:scope-impersonation***
- ***system:webhook***
- ***system:oauth-token-deleter***
- ***self-access-reviewer***

중요

인증되지 않은 액세스를 수정할 때 항상 조직의 보안 표준을 준수하는지 확인하십시오.

사전 요구 사항

- ***cluster-admin** 역할의 사용자로 클러스터에 액세스할 수 있어야 합니다.*
- *OpenShift CLI(**oc**)가 설치되어 있습니다.*

절차

1. ***add-<cluster_role>-unauth.yaml** 이라는 YAML 파일을 생성하고 다음 콘텐츠를 추가합니다.*

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
 annotations:
   rbac.authorization.kubernetes.io/autoupdate: "true"
 name: <cluster_role>access-unauthenticated
roleRef:
 apiGroup: rbac.authorization.k8s.io
 kind: ClusterRole
 name: <cluster_role>
subjects:
 - apiGroup: rbac.authorization.k8s.io
   kind: Group
   name: system:unauthenticated
```

2. *다음 명령을 실행하여 구성을 적용합니다.*

```
$ oc apply -f add-<cluster_role>.yaml
```

4 장. ID 공급자 구성

OpenShift Dedicated 클러스터가 생성된 후 사용자가 클러스터에 액세스하기 위해 로그인하는 방법을 결정하도록 ID 공급자를 구성해야 합니다.

4.1. ID 공급자 이해

OpenShift Dedicated에는 기본 제공 OAuth 서버가 포함되어 있습니다. 개발자와 관리자는 OAuth 액세스 토큰을 가져와 API 인증을 수행합니다. 관리자는 클러스터를 설치한 후 ID 공급자를 지정하도록 OAuth를 구성할 수 있습니다. ID 공급자를 구성하면 사용자가 클러스터에 로그인하고 액세스할 수 있습니다.

4.1.1. 지원되는 ID 공급자

다음 유형의 ID 공급자를 구성할 수 있습니다.

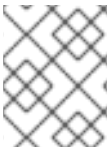
ID 공급자	설명
GitHub 또는 GitHub Enterprise	GitHub 또는 GitHub Enterprise의 OAuth 인증 서버에 대해 사용자 이름 및 암호의 유효성을 검사하도록 GitHub ID 공급자를 구성합니다.
GitLab	GitLab.com 또는 기타 GitLab 인스턴스를 ID 공급자로 사용하도록 GitLab ID 공급자를 구성합니다.
Google	Google 의 OpenID Connect 통합을 사용하여 GoogleID 공급자를 구성합니다.
LDAP	단순 바인드 인증을 사용하여 LDAPv3 서버에 대해 사용자 이름 및 암호의 유효성을 확인하도록 LDAP ID 공급자를 구성합니다.
OpenID Connect	인증 코드 흐름을 사용하여 OIDC ID 공급자와 통합하도록 OpenID Connect(OIDC) ID 공급자를 구성합니다.
htpasswd	단일 정적 관리 사용자에 대해 htpasswd ID 공급자를 구성합니다. 사용자로 클러스터에 로그인하여 문제를 해결할 수 있습니다.  **중요** htpasswd ID 공급자 옵션은 단일 정적 관리 사용자를 생성할 수 있도록만 포함됩니다. htpasswd는 OpenShift Dedicated의 일반 ID 공급자로 지원되지 않습니다. 단일 사용자를 구성하는 단계는 *htpasswd ID 공급자 구성*을 참조하십시오.

4.1.2. ID 공급자 매개변수

다음 매개변수는 모든 ID 공급자에 공통입니다.

매개변수	설명
name	공급자 사용자 이름에 접두어로 공급자 이름을 지정하여 ID 이름을 만듭니다.

매개변수	설명
mappingMethod	사용자가 로그인할 때 새 ID를 사용자에게 매핑하는 방법을 정의합니다. 다음 값 중 하나를 입력하십시오. **claim** 기본값입니다. 사용자에게 ID의 기본 사용자 이름을 프로비저닝합니다. 해당 사용자 이름의 사용자가 이미 다른 ID에 매핑되어 있는 경우 실패합니다. **lookup** 기존 ID, 사용자 ID 매핑 및 사용자를 조회하지만 사용자 또는 ID를 자동으로 프로비저닝하지는 않습니다. 클러스터 관리자는 이를 통해 수동으로 또는 외부 프로세스를 사용하여 ID 및 사용자를 설정할 수 있습니다. 이 방법을 사용하려면 사용자를 수동으로 프로비저닝해야 합니다. **add** 사용자에게 ID의 기본 사용자 이름을 프로비저닝합니다. 해당 사용자 이름을 가진 사용자가 이미 존재하는 경우 ID가 기존 사용자에게 매핑되고 그 사용자의 기존 ID 매핑에 추가됩니다. 동일한 사용자 집합을 식별하고 동일한 사용자 이름에 매핑되는 ID 공급자를 여럿 구성한 경우 필요합니다.



참고

*ID 공급자를 추가하거나 변경할 때 **mappingMethod** 매개변수를 **add**로 설정하면 새 공급자의 ID를 기존 사용자에게 매핑할 수 있습니다.*

4.2. GITHUB ID 공급자 구성

GitHub 또는 GitHub Enterprise의 OAuth 인증 서버에 대해 사용자 이름 및 암호의 유효성을 검사하고 OpenShift Dedicated 클러스터에 액세스하도록 GitHub ID 공급자를 구성합니다. OAuth는 OpenShift Dedicated와 GitHub 또는 GitHub Enterprise 간의 토큰 교환 흐름을 용이하게 합니다.

주의

GitHub 인증을 구성하면 사용자가 GitHub 자격 증명을 사용하여 OpenShift Dedicated에 로그인할 수 있습니다. GitHub 사용자 ID가 있는 사람이 OpenShift Dedicated 클러스터에 로그인하지 못하도록 특정 GitHub 조직 또는 팀의 사용자만 액세스를 제한해야 합니다.

사전 요구 사항

- *OAuth 애플리케이션은 GitHub 조직 관리자가 GitHub 조직 설정 내에서 직접 생성해야 합니다.*
- *GitHub 조직 또는 팀은 GitHub 계정에 설정됩니다.*

프로세스

1. *OpenShift Cluster Manager 에서 **Cluster List** 페이지로 이동하여 ID 공급자를 구성하는 데 필요한 클러스터를 선택합니다.*

2. ***액세스 제어*** 탭을 클릭합니다.
3. ***ID 공급자 추가***를 클릭합니다.

참고

클러스터 생성 후 표시된 경고 메시지에서 **Oauth 구성 추가** 링크를 클릭하여 ID 공급자를 구성할 수도 있습니다.

4. 드롭다운 메뉴에서 **GitHub** 를 선택합니다.
5. ID 공급자의 고유 이름을 입력합니다. 이 이름은 나중에 변경할 수 없습니다.
   - 제공된 필드에 **OAuth 콜백 URL** 이 자동으로 생성됩니다. 이를 사용하여 GitHub 애플리케이션을 등록합니다.

     ```
     https://oauth-openshift.apps.<cluster_name>.
     <cluster_domain>/oauth2callback/<idp_provider_name>
     ```

     예를 들면 다음과 같습니다.

     ```
     https://oauth-openshift.apps.openshift-cluster.example.com/oauth2callback/github
     ```

6. GitHub에 애플리케이션을 등록 합니다.
7. OpenShift Dedicated로 돌아가서 드롭다운 메뉴에서 매핑 방법을 선택합니다. 대부분의 경우 **클레임** 이 권장됩니다.
8. GitHub에서 제공하는 **클라이언트 ID** 및 **클라이언트 시크릿을** 입력합니다.
9. ***호스트 이름을 입력합니다***. GitHub Enterprise의 호스팅 인스턴스를 사용하는 경우 호스트 이름을 입력해야 합니다.
10. 선택 사항: CA(인증 기관) 파일을 사용하여 구성된 GitHub Enterprise URL에 대한 서버 인증서의 유효성을 확인할 수 있습니다. **찾아보기** 를 클릭하여 **CA 파일을** 찾아서 ID 공급자에 연결합니다.
11. ***조직 또는 팀 사용을*** 선택하여 특정 GitHub 조직 또는 GitHub 팀에 대한 액세스를 제한합니다.
12. 액세스를 제한할 조직 또는 팀의 이름을 입력합니다. **추가 추가** 를 클릭하여 사용자가 멤버로 속할 수 있는 여러 조직 또는 팀을 지정합니다.
13. ***확인을*** 클릭합니다.

검증

- 이제 구성된 ID 공급자가 **클러스터 목록** 페이지의 **액세스 제어** 탭에 표시됩니다.

4.3. GITLAB ID 공급자 구성

GitLab.com 또는 기타 GitLab 인스턴스를 ID 공급자로 사용하도록 GitLab ID 공급자를 구성합니다.

사전 요구 사항

- GitLab 버전 7.7.0~11.0을 사용하는 경우 OAuth 통합을 사용하여 연결합니다. GitLab 버전 11.1 이상을 사용하는 경우 OAuth 대신 OpenID Connect (OIDC)를 사용하여 연결할 수 있습니다.

프로세스

1. OpenShift Cluster Manager 에서 **Cluster List** 페이지로 이동하여 ID 공급자를 구성하는 데 필요한 클러스터를 선택합니다.
2. **액세스 제어** 탭을 클릭합니다.
3. **ID 공급자 추가**를 클릭합니다.

참고

클러스터 생성 후 표시된 경고 메시지에서 **Oauth 구성 추가** 링크를 클릭하여 ID 공급자를 구성할 수도 있습니다.

4. 드롭다운 메뉴에서 **GitLab** 을 선택합니다.
5. ID 공급자의 고유 이름을 입력합니다. 이 이름은 나중에 변경할 수 없습니다.
   - 제공된 필드에 **OAuth 콜백 URL** 이 자동으로 생성됩니다. 이 URL 을 GitLab에 제공합니다.

   ```
   https://oauth-openshift.apps.<cluster_name>.
   <cluster_domain>/oauth2callback/<idp_provider_name>
   ```

   예를 들면 다음과 같습니다.

   ```
   https://oauth-openshift.apps.openshift-cluster.example.com/oauth2callback/gitlab
   ```

6. GitLab에 새 애플리케이션을 추가합니다.
7. OpenShift Dedicated로 돌아가서 드롭다운 메뉴에서 매핑 방법을 선택합니다. 대부분의 경우 **클레임** 이 권장됩니다.
8. GitLab에서 제공하는 **클라이언트 ID** 및 **클라이언트 시크릿을** 입력합니다.
9. GitLab 공급자의 **URL** 을 입력합니다.
10. 선택 사항: CA(인증 기관) 파일을 사용하여 구성된 GitLab URL에 대한 서버 인증서의 유효성을 확인할 수 있습니다. **찾아보기** 를 클릭하여 **CA 파일을** 찾아서 ID 공급자에 연결합니다.
11. **확인을** 클릭합니다.

검증

- 이제 구성된 ID 공급자가 **클러스터 목록** 페이지의 **액세스 제어** 탭에 표시됩니다.

4.4. GOOGLE ID 공급자 구성

사용자가 Google 자격 증명으로 인증할 수 있도록 Google ID 공급자를 구성합니다.

주의

*Google을 ID 공급자로 사용하면 모든 Google 사용자가 서버 인증을 수행할 수 있습니다. **hostedDomain** 구성 속성을 사용하여 특정 호스트 도메인의 멤버 인증을 제한할 수 있습니다.*

프로세스

1. *OpenShift Cluster Manager 에서 **Cluster List** 페이지로 이동하여 ID 공급자를 구성하는 데 필요한 클러스터를 선택합니다.*
2. ***액세스 제어** 탭을 클릭합니다.*
3. ***ID 공급자 추가**를 클릭합니다.*

참고

*클러스터 생성 후 표시된 경고 메시지에서 **Oauth 구성 추가** 링크를 클릭하여 ID 공급자를 구성할 수도 있습니다.*

4. *드롭다운 메뉴에서 **Google** 을 선택합니다.*
5. *ID 공급자의 고유 이름을 입력합니다. 이 이름은 나중에 변경할 수 없습니다.*
   - *제공된 필드에 **OAuth 콜백 URL** 이 자동으로 생성됩니다. 이 URL을 Google에 제공합니다.*

     ```
     https://oauth-openshift.apps.<cluster_name>.
     <cluster_domain>/oauth2callback/<idp_provider_name>
     ```

     예를 들면 다음과 같습니다.

     ```
     https://oauth-openshift.apps.openshift-cluster.example.com/oauth2callback/google
     ```

6. *Google 의 OpenID Connect 통합을 사용하여 Google ID 공급자를 구성합니다.*
7. *OpenShift Dedicated로 돌아가서 드롭다운 메뉴에서 매핑 방법을 선택합니다. 대부분의 경우 **클레임** 이 권장됩니다.*
8. *등록된 Google 프로젝트의 **클라이언트 ID** 와 Google에서 발행한 **클라이언트 시크릿을** 입력합니다.*
9. *사용자를 Google Apps 도메인으로 제한하려면 호스팅 도메인을 입력합니다.*
10. ***확인을** 클릭합니다.*

검증

- *이제 구성된 ID 공급자가 **클러스터 목록** 페이지의 **액세스 제어** 탭에 표시됩니다.*

4.5. LDAP ID 공급자 구성

단순 바인드 인증을 사용하여 LDAPv3 서버에 대해 사용자 이름 및 암호의 유효성을 확인하도록 LDAP ID 공급자를 구성합니다.

사전 요구 사항

- *LDAP ID 공급자를 구성할 때 구성된 **LDAP URL** 을 입력해야 합니다. 구성된 URL은 RFC 2255 URL로, 사용할 LDAP 호스트 및 검색 매개변수를 지정합니다. URL 구문은 다음과 같습니다.*

```
ldap://host:port/basedn?attribute?scope?filter
```

URL 구성 요소	설명
ldap	일반 LDAP의 경우 **ldap** 문자열을 사용합니다. 보안 LDAP(LDAPS)의 경우 대신 **ldaps**를 사용합니다.
host:port	LDAP 서버의 이름 및 포트입니다. ldap의 경우 기본값은 **localhost:389** 이고 LDAPS의 경우 **localhost:636**입니다.
basedn	모든 검색을 시작해야 하는 디렉터리 분기의 DN입니다. 적어도 디렉터리 트리의 맨 위에 있어야 하지만 디렉터리에 하위 트리를 지정할 수도 있습니다.
attribute	검색할 속성입니다. RFC 2255에서는 쉼표로 구분된 속성 목록을 사용할 수 있지만 제공되는 속성 수와 관계없이 첫 번째 속성만 사용됩니다. 속성이 제공되지 않는 경우 기본값은 **uid**를 사용하는 것입니다. 사용할 하위 트리의 모든 항목에서 고유한 속성을 선택하는 것이 좋습니다.
scope	검색 범위입니다. **one** 또는 **sub**일 수 있습니다. 범위가 제공되지 않는 경우 기본값은 **sub** 범위를 사용하는 것입니다.
filter	유효한 LDAP 검색 필터입니다. 제공하지 않는 경우 기본값은 **(objectClass=*)**입니다.

검색을 수행할 때 속성, 필터, 제공된 사용자 이름을 결합하여 다음과 같은 검색 필터가 생성됩니다.

```
(&(<filter>)(<attribute>=<username>))
```

중요

*LDAP 디렉터리에서 검색에 인증이 필요한 경우 항목을 검색하는 데 사용할 **bindDN** 및 **bindPassword**를 지정하십시오.*

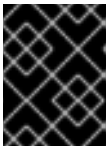
프로세스

1. *OpenShift Cluster Manager 에서 **Cluster List** 페이지로 이동하여 ID 공급자를 구성하는 데 필요한 클러스터를 선택합니다.*
2. ***액세스 제어** 탭을 클릭합니다.*
3. ***ID 공급자 추가**를 클릭합니다.*

참고

*클러스터 생성 후 표시된 경고 메시지에서 **Oauth 구성 추가** 링크를 클릭하여 ID 공급자를 구성할 수도 있습니다.*

4. *드롭다운 메뉴에서 **LDAP** 를 선택합니다.*
5. *ID 공급자의 고유 이름을 입력합니다. 이 이름은 나중에 변경할 수 없습니다.*
6. *드롭다운 메뉴에서 매핑 방법을 선택합니다. 대부분의 경우 **클레임** 이 권장됩니다.*
7. ***LDAP URL** 을 입력하여 사용할 LDAP 검색 매개변수를 지정합니다.*
8. *선택 사항: **바인딩 DN** 및 **바인딩 암호를** 입력합니다.*
9. *LDAP 속성을 ID에 매핑할 속성을 입력합니다.*
    - *값을 사용자 **ID** 로 사용해야 하는 ID 특성을 입력합니다. **추가 추가** 를 클릭하여 여러 ID 속성을 추가합니다.*
    - *선택 사항: 값이 표시 이름으로 사용해야 하는 **Preferred username** 특성을 입력합니다. **추가 추가** 를 클릭하여 선호하는 사용자 이름 속성을 여러 개 추가합니다.*
    - *선택 사항: 값을 **이메일** 주소로 사용해야 하는 이메일 속성을 입력합니다. **추가 추가** 를 클릭하여 여러 이메일 속성을 추가합니다.*
10. *선택 사항: **고급 옵션 표시를** 클릭하여 LDAP ID 공급자에 CA(인증 기관) 파일을 추가하여 구성된 URL 에 대한 서버 인증서의 유효성을 검사합니다. **찾아보기** 를 클릭하여 **CA 파일을** 찾아서 ID 공급자에 연결합니다.*
11. *선택 사항: 고급 옵션에 따라 LDAP 공급자를 비보안으로 만들 수 **있습니다**. 이 옵션을 선택하면 CA 파일을 사용할 수 없습니다.*



중요

*비보안 LDAP 연결(ldap:// 또는 포트 389)을 사용하는 경우 구성 마법사에서 **Insecure** 옵션을 선택해야 합니다.*

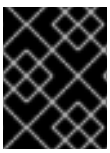
12. ***확인을** 클릭합니다.*

검증

- *이제 구성된 ID 공급자가 **클러스터 목록** 페이지의 **액세스 제어** 탭에 표시됩니다.*

4.6. OPENID ID 공급자 구성

인증 코드 흐름을 사용하여 OpenID Connect ID 공급자와 통합하도록 OpenID ID 공급자를 구성합니다.



중요

OpenShift Dedicated 의 Authentication Operator 는 구성된 OpenID Connect ID 공급자가 OpenID Connect 검색 사양을 구현해야 합니다.

*클레임은 OpenID ID 공급자에서 반환된 JWT **id_token** 에서 읽고, 지정된 경우 발급자 URL 에서 반환된 JSON 에서 읽습니다.*

사용자 ID 로 사용할 하나 이상의 클레임을 구성해야 합니다.

또한 사용자의 기본 사용자 이름, 표시 이름, 이메일 주소로 사용할 클레임을 나타낼 수도 있습니다. 여러 클레임이 지정되는 경우 비어 있지 않은 값이 있는 첫 번째 클레임이 사용됩니다. 표준 클레임은 다음과 같습니다.

클레임	설명
preferred_username	사용자를 프로비저닝할 때 사용하는 기본 사용자 이름입니다. 사용자가 사용하고자 하는 약칭입니다(예: **janedoe**). 일반적으로 인증 시스템의 사용자 로그인 또는 사용자 이름에 해당하는 값입니다(예: 사용자 이름 또는 이메일).
email	이메일 주소입니다.
name	표시 이름입니다.

자세한 내용은 OpenID 클레임 설명서 를 참조하십시오.

사전 요구 사항

- *OpenID Connect 를 구성하기 전에 OpenShift Dedicated 클러스터에서 사용하려는 Red Hat 제품 또는 서비스의 설치 사전 요구 사항을 확인하십시오.*

프로세스

1. *OpenShift Cluster Manager 에서 **Cluster List** 페이지로 이동하여 ID 공급자를 구성하는 데 필요한 클러스터를 선택합니다.*
2. ***액세스 제어** 탭을 클릭합니다.*
3. ***ID 공급자 추가**를 클릭합니다.*

참고

*클러스터 생성 후 표시된 경고 메시지에서 **Oauth 구성 추가** 링크를 클릭하여 ID 공급자를 구성할 수도 있습니다.*

4. *드롭다운 메뉴에서 **OpenID** 를 선택합니다.*
5. *ID 공급자의 고유 이름을 입력합니다. 이 이름은 나중에 변경할 수 없습니다.*
   - *제공된 필드에 **OAuth 콜백 URL** 이 자동으로 생성됩니다.*

     ```
     https://oauth-openshift.apps.<cluster_name>.
     <cluster_domain>/oauth2callback/<idp_provider_name>
     ```

     예를 들면 다음과 같습니다.

> https://oauth-openshift.apps.openshift-cluster.example.com/oauth2callback/openid

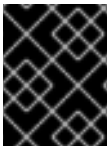
6. 인증 요청을 생성하는 단계에 따라 OpenID ID 공급자에 새 OpenID Connect 클라이언트를 등록합니다.
7. OpenShift Dedicated로 돌아가서 드롭다운 메뉴에서 매핑 방법을 선택합니다. 대부분의 경우 **클레임** 이 권장됩니다.
8. OpenID에서 제공하는 **클라이언트 ID** 및 **클라이언트 시크릿을** 입력합니다.
9. **발급자 URL** 을 입력합니다. 이는 OpenID 공급자가 발급자 식별자로 어설션하는 URL 입니다. URL 쿼리 매개변수 또는 조각이 없는 https 체계를 사용해야 합니다.
10. 값을 이메일 주소로 사용해야 하는 **Email** 특성을 입력합니다. **추가 추가** 를 클릭하여 여러 이메일 속성을 추가합니다.
11. 기본 사용자 이름으로 값을 사용해야 하는 **Name** 특성을 입력합니다. **추가 추가** 를 클릭하여 선호하는 사용자 이름을 여러 개 추가합니다.
12. 표시 이름으로 값을 사용해야 하는 **Preferred username** 특성을 입력합니다. **추가 추가** 를 클릭하여 여러 표시 이름을 추가합니다.
13. 선택 사항: **고급 옵션 표시를** 클릭하여 OpenID ID 공급자에 CA(인증 기관) 파일을 추가합니다.
14. 선택 사항: 고급 옵션 아래에서 **추가 범위를** 추가할 수 있습니다. 기본적으로 **OpenID** 범위가 요청됩니다.
15. **확인을** 클릭합니다.

검증

- 이제 구성된 ID 공급자가 **클러스터 목록** 페이지의 **액세스 제어** 탭에 표시됩니다.

4.7. HTPASSWD ID 공급자 구성

클러스터 관리 권한이 있는 정적 사용자를 생성하도록 htpasswd ID 공급자를 구성합니다. 사용자로 클러스터에 로그인하여 문제를 해결할 수 있습니다.



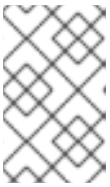
중요

htpasswd ID 공급자 옵션은 단일 정적 관리 사용자를 생성할 수 있도록만 포함됩니다. htpasswd는 OpenShift Dedicated의 일반 ID 공급자로 지원되지 않습니다.

프로세스

1. OpenShift Cluster Manager 에서 **클러스터 목록** 페이지로 이동하여 클러스터를 선택합니다.
2. **액세스 제어 → ID 공급자를** 선택합니다.
3. **ID 공급자 추가를** 클릭합니다.
4. **Identity Provider** 드롭다운 메뉴에서 **HTPasswd** 를 선택합니다.
5. ID 공급자의 **이름** 필드에 고유한 이름을 추가합니다.

6. 정적 사용자에 대해 제안된 사용자 이름과 암호를 사용하거나 직접 만듭니다.



참고

다음 단계에서 **추가** 를 선택하면 이 단계에서 정의된 인증 정보가 표시되지 않습니다. 인증 정보가 손실되면 ID 공급자를 다시 생성하고 인증 정보를 다시 정의해야 합니다.

7. **Add** 를 선택하여 htpasswd ID 공급자와 단일 정적 사용자를 생성합니다.

8. 정적 사용자에게 클러스터를 관리할 수 있는 권한을 부여합니다.

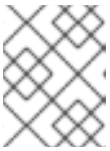
   a. **액세스 제어 → 클러스터 역할 및 액세스** 에서 **사용자 추가** 를 선택합니다.

   b. 이전 단계에서 생성한 정적 사용자의 **사용자 ID** 를 입력합니다.

   c. **Add user** 를 선택하여 사용자에게 관리 권한을 부여합니다.

검증

- 구성된 htpasswd ID 공급자는 **액세스 제어 → ID 공급자** 페이지에 표시됩니다.



참고

ID 공급자를 생성한 후 동기화가 일반적으로 2분 내에 완료됩니다. htpasswd ID 공급자를 사용할 수 있게 되면 사용자로 클러스터에 로그인할 수 있습니다.

- 단일 관리자는 **액세스 제어 → 클러스터 역할 및 액세스** 페이지에 표시됩니다. 사용자의 관리 그룹 멤버십도 표시됩니다.

4.8. 클러스터에 액세스

ID 공급자를 구성한 후 사용자는 Red Hat OpenShift Cluster Manager 에서 클러스터에 액세스할 수 있습니다.

사전 요구 사항

- OpenShift Cluster Manager 에 로그인했습니다.
- OpenShift Dedicated 클러스터를 생성하셨습니다.
- 클러스터의 ID 공급자를 구성했습니다.
- 구성된 ID 공급자에 사용자 계정을 추가했습니다.

프로세스

1. OpenShift Cluster Manager 에서 액세스할 클러스터를 클릭합니다.
2. **Open Console** 을 클릭합니다.
3. ID 공급자를 클릭하고 클러스터에 로그인할 수 있는 인증 정보를 제공합니다.
4. **Open console** 을 클릭하여 클러스터의 웹 콘솔을 엽니다.

5. *ID 공급자를 클릭하고 클러스터에 로그인할 수 있는 인증 정보를 제공합니다. 공급자가 제공하는 권한 부여 요청을 완료합니다.*

5장. 관리 역할 및 사용자 관리

5.1. 관리 역할 이해

5.1.1. cluster-admin 역할

CCO(Customer Cloud Subscription)가 있는 OpenShift Dedicated 클러스터의 관리자는 **cluster-admin** 역할에 액세스할 수 있습니다. 클러스터를 생성한 사용자는 최대 관리자 권한을 가지도록 **cluster-admin** 사용자 역할을 계정에 추가할 수 있습니다. 이러한 권한은 클러스터를 생성할 때 사용자 계정에 자동으로 할당되지 않습니다. cluster-admin 역할을 사용하여 계정에 로그인하는 동안 사용자는 대부분 무제한으로 클러스터를 제어하고 구성할 수 있습니다. 클러스터 불안정을 방지하기 위해 Webhook로 차단되거나 OpenShift Cluster Manager 에서 관리되고 클러스터 내 변경 사항을 덮어쓰기 위해 Webhook로 차단되는 몇 가지 구성이 있습니다. cluster-admin 역할의 사용에는 Red Hat과의 부록 4 계약에 나열된 제한 사항이 적용됩니다. 가장 좋은 방법은 **cluster-admin** 사용자 수를 가능한 한 적은 수로 제한합니다.

5.1.2. dedicated-admin 역할

OpenShift Dedicated 클러스터의 관리자는 조직의 클러스터의 모든 사용자 생성 프로젝트에 대한 추가 권한 및 액세스 권한이 있습니다. **dedicated-admin** 역할을 사용하여 계정에 로그인하는 동안 개발자 CLI 명령(**oc** 명령 아래)을 사용하면 프로젝트 전체에서 오브젝트에 대한 가시성 및 관리 기능을 향상시킬 수 있지만 관리자 CLI 명령(**oc adm** 명령 아래)을 사용하면 추가 작업을 완료할 수 있습니다.

참고

계정에는 이러한 권한이 증가했지만 실제 클러스터 유지 관리 및 호스트 구성은 여전히 OpenShift Operations 팀에서 수행합니다.

5.2. OPENSHIFT DEDICATED 관리자 관리

관리자 역할은 클러스터의 **cluster-admin** 또는 **dedicated-admin** 그룹을 사용하여 관리됩니다. 이 그룹의 기존 멤버는 OpenShift Cluster Manager 를 통해 멤버십을 편집할 수 있습니다.

5.2.1. 사용자 추가

프로세스

1. **클러스터 세부 정보** 페이지 및 **액세스 제어** 탭으로 이동합니다.
2. **클러스터 역할 및 액세스** 탭을 선택하고 **사용자 추가** 를 클릭합니다.
3. 사용자 이름을 입력하고 그룹을 선택합니다.
4. **사용자 추가를** 클릭합니다.

참고

cluster-admin 그룹에 사용자를 추가하는 데 몇 분이 걸릴 수 있습니다.

5.2.2. 사용자 제거

프로세스

1. *클러스터 세부 정보 페이지 및 **액세스 제어** 탭으로 이동합니다.*

2. *사용자 및 그룹 조합 오른쪽에 있는 옵션 메뉴*  *를 클릭하고 **삭제** 를 클릭합니다.*

6장. RBAC를 사용하여 권한 정의 및 적용

6.1. RBAC 개요

RBAC(역할 기반 액세스 제어) 오브젝트에 따라 사용자가 프로젝트 내에서 지정된 작업을 수행할 수 있는지가 결정됩니다.

***dedicated-admin** 역할의 관리자는 클러스터 역할 및 바인딩을 사용하여 OpenShift Dedicated 플랫폼 자체 및 모든 프로젝트에 대한 다양한 액세스 수준이 있는 사용자를 제어할 수 있습니다.*

개발자는 로컬 역할 및 바인딩을 사용하여 프로젝트에 액세스할 수 있는 사용자를 제어할 수 있습니다. 권한 부여는 인증과 별도의 단계이며, 여기서는 조치를 수행할 사용자의 신원을 파악하는 것이 더 중요합니다.

권한 부여는 다음을 사용하여 관리합니다.

권한 부여 오브젝트	설명
규칙	오브젝트 집합에 허용되는 동사 집합입니다. 예를 들면 사용자 또는 서비스 계정의 Pod **생성** 가능 여부입니다.
역할	규칙 모음입니다. 사용자와 그룹을 여러 역할에 연결하거나 바인딩할 수 있습니다.
바인딩	역할이 있는 사용자 및/또는 그룹 간 연결입니다.

권한 부여를 제어하는 두 가지 수준의 RBAC 역할 및 바인딩이 있습니다.

RBAC 수준	설명
클러스터 RBAC	모든 프로젝트에 적용할 수 있는 역할 및 바인딩입니다. *클러스터 역할*은 클러스터 전체에 존재하며 *클러스터 역할 바인딩*은 클러스터 역할만 참조할 수 있습니다.
지역 RBAC	지정된 프로젝트에 적용되는 역할 및 바인딩입니다. *로컬 역할*은 단일 프로젝트에만 존재하지만 로컬 역할 바인딩은 클러스터 및 로컬 역할을 모두 참조할 수 있습니다.

클러스터 역할 바인딩은 클러스터 수준에 존재하는 바인딩입니다. 역할 바인딩은 프로젝트 수준에 있습니다. 해당 사용자가 프로젝트를 보려면 로컬 역할 바인딩을 사용하여 클러스터 역할 보기를 사용자에게 바인딩해야 합니다. 클러스터 역할이 특정 상황에 필요한 권한 집합을 제공하지 않는 경우에만 로컬 역할을 생성하십시오.

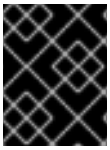
이러한 2단계 계층 구조로 인해 클러스터 역할로는 여러 프로젝트에서 재사용하고, 로컬 역할로는 개별 프로젝트 내에서 사용자 정의할 수 있습니다.

평가 중에는 클러스터 역할 바인딩과 로컬 역할 바인딩이 모두 사용됩니다. 예를 들면 다음과 같습니다.

1. *클러스터 전체의 "허용" 규칙을 확인합니다.*
2. *로컬 바인딩된 "허용" 규칙을 확인합니다.*
3. *기본적으로 거부합니다.*

6.1.1. 기본 클러스터 역할

OpenShift Dedicated에는 클러스터 전체 또는 로컬로 사용자 및 그룹에 바인딩할 수 있는 기본 클러스터 역할 세트가 포함되어 있습니다.



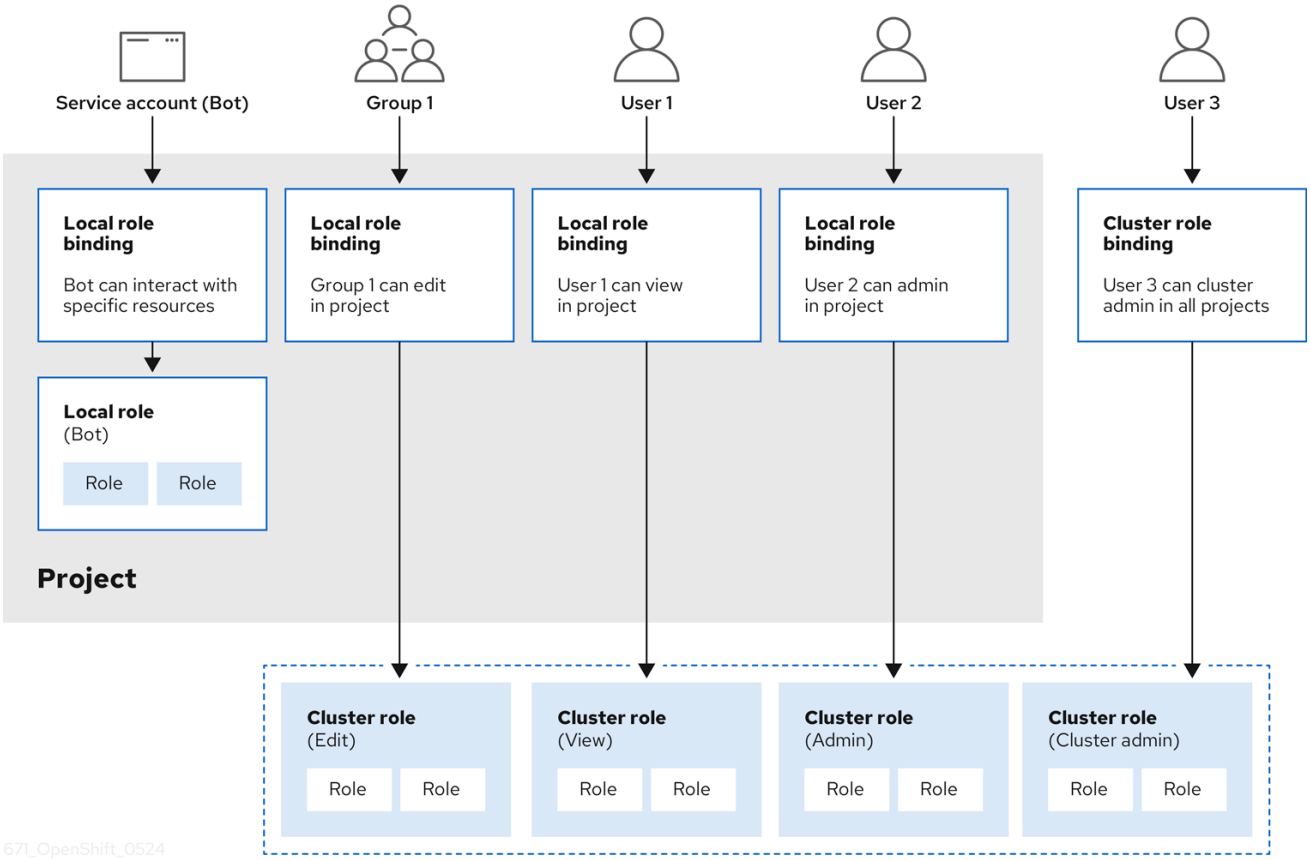
중요

기본 클러스터 역할을 수동으로 수정하지 않는 것이 좋습니다. 이러한 시스템 역할에 대한 수정으로 인해 클러스터가 제대로 작동하지 않을 수 있습니다.

기본 클러스터 역할	설명
admin	프로젝트 관리자입니다. 로컬 바인딩에 사용되는 경우 **admin**은 프로젝트의 모든 리소스를 보고 할당량을 제외한 프로젝트의 모든 리소스를 수정할 수 있는 권한이 있습니다.
basic-user	프로젝트 및 사용자에 대한 기본 정보를 가져올 수 있는 사용자입니다.
cluster-admin	모든 프로젝트에서 모든 작업을 수행할 수 있는 슈퍼 유저입니다. 로컬 바인딩을 통해 사용자에게 바인딩하면 할당량은 물론 프로젝트의 모든 리소스에 대한 모든 조치를 완전히 제어할 수 있습니다.
cluster-status	기본 클러스터 상태 정보를 가져올 수 있는 사용자입니다.
cluster-reader	대부분의 오브젝트를 가져오거나 볼 수 있지만 수정할 수는 없는 사용자입니다.
edit	프로젝트에서 대부분의 오브젝트를 수정할 수 있지만 역할이나 바인딩을 보거나 수정할 권한은 없는 사용자입니다.
self-provisioner	자체 프로젝트를 만들 수 있는 사용자입니다.
view	수정할 수는 없지만 프로젝트의 오브젝트를 대부분 볼 수 있는 사용자입니다. 역할 또는 바인딩을 보거나 수정할 수 없습니다.

*로컬 바인딩과 클러스터 바인딩의 차이점에 유의하십시오. 예를 들어 로컬 역할 바인딩을 사용하여 **cluster-admin** 역할을 사용자에게 바인딩하는 경우, 이 사용자에게 클러스터 관리자 권한이 있는 것처럼 보일 수 있습니다. 사실은 그렇지 않습니다. 프로젝트의 사용자에게 **cluster-admin**을 바인딩하면 해당 프로젝트에 대해서만 슈퍼 관리자 권한이 사용자에게 부여됩니다. 해당 사용자에게는 클러스터 역할 **admin**의 권한을 비롯하여 해당 프로젝트에 대한 속도 제한 편집 기능과 같은 몇 가지 추가 권한이 있습니다. 이 바인딩은 실제 클러스터 관리자에게 바인딩된 클러스터 역할 바인딩이 나열되지 않는 웹 콘솔 UI로 인해 혼동될 수 있습니다. 그러나 **cluster-admin**을 로컬로 바인딩하는 데 사용할 수 있는 로컬 역할 바인딩은 나열됩니다.*

아래에는 클러스터 역할, 로컬 역할, 클러스터 역할 바인딩, 로컬 역할 바인딩, 사용자, 그룹, 서비스 계정 간의 관계가 설명되어 있습니다.



주의

***get pods/exec,pods/*, get ** 규칙은 역할에 적용될 때 실행 권한을 부여합니다. 최소 권한 원칙을 적용하고 사용자 및 에이전트에 필요한 최소 RBAC 권한만 할당합니다. 자세한 내용은 RBAC 규칙 실행 권한을 참조하십시오.*

6.1.2. 권한 부여 평가

OpenShift Dedicated는 다음을 사용하여 권한 부여를 평가합니다.

ID

사용자 이름 및 사용자가 속한 그룹 목록입니다.

작업

수행하는 작업입니다. 대부분의 경우 다음으로 구성됩니다.

- ***프로젝트**: 액세스하는 프로젝트입니다. 프로젝트는 추가 주석이 있는 쿠버네티스 네임스페이스로, 사용자 커뮤니티가 다른 커뮤니티와 별도로 컨텐츠를 구성하고 관리할 수 있습니다.*
- ***동사**: 작업 자체를 나타내며 **get**, **list**, **create**, **update**, **delete**, **deletecollection** 또는 **watch**에 해당합니다.*
- ***리소스 이름**: 액세스하는 API 끝점입니다.*

바인딩

전체 바인딩 목록으로, 역할이 있는 사용자 또는 그룹 간 연결을 나타냅니다.

OpenShift Dedicated는 다음 단계를 사용하여 권한 부여를 평가합니다.

1. ID 및 프로젝트 범위 작업은 사용자 또는 해당 그룹에 적용되는 모든 바인딩을 찾는 데 사용됩니다.
2. 바인딩은 적용되는 모든 역할을 찾는 데 사용됩니다.
3. 역할은 적용되는 모든 규칙을 찾는 데 사용됩니다.
4. 일치하는 규칙을 찾기 위해 작업을 각 규칙에 대해 확인합니다.
5. 일치하는 규칙이 없으면 기본적으로 작업이 거부됩니다.

작은 정보

사용자 및 그룹을 동시에 여러 역할과 연결하거나 바인딩할 수 있습니다.

프로젝트 관리자는 CLI를 사용하여 각각 연결된 동사 및 리소스 목록을 포함하여 로컬 역할 및 바인딩을 볼 수 있습니다.

중요

프로젝트 관리자에게 바인딩된 클러스터 역할은 로컬 바인딩을 통해 프로젝트에서 제한됩니다. **cluster-admin** 또는 **system:admin**에 부여되는 클러스터 역할과 같이 클러스터 전체에 바인딩되지 않습니다.

클러스터 역할은 클러스터 수준에서 정의된 역할이지만 클러스터 수준 또는 프로젝트 수준에서 바인딩할 수 있습니다.

6.1.2.1. 클러스터 역할 집계

기본 admin, edit, view 및 cluster-reader 클러스터 역할은 새 규칙이 생성될 때 각 역할에 대한 클러스터 규칙이 동적으로 업데이트되는 클러스터 역할 집계 를 지원합니다. 이 기능은 사용자 정의 리소스를 생성하여 쿠버네티스 API를 확장한 경우에만 관련이 있습니다.

6.2. 프로젝트 및 네임스페이스

쿠버네티스 네임스페이스는 클러스터의 리소스 범위를 지정하는 메커니즘을 제공합니다. 쿠버네티스 설명서에 네임스페이스에 대한 자세한 정보가 있습니다.

네임스페이스는 다음에 대한 고유 범위를 제공합니다.

- 기본 이름 지정 충돌을 피하기 위해 이름이 지정된 리소스
- 신뢰할 수 있는 사용자에게 위임된 관리 권한
- 커뮤니티 리소스 사용을 제한하는 기능

시스템에 있는 대부분의 오브젝트는 네임스페이스에 따라 범위가 지정되지만, 노드 및 사용자를 비롯한 일부는 여기에 해당하지 않으며 네임스페이스가 없습니다.

프로젝트는 추가 주석이 있는 쿠버네티스 네임스페이스이며, 일반 사용자용 리소스에 대한 액세스를 관리하는 가장 중요한 수단입니다. 사용자 커뮤니티는 프로젝트를 통해 다른 커뮤니티와 별도로 콘텐츠를 구

성하고 관리할 수 있습니다. 사용자는 관리자로부터 프로젝트에 대한 액세스 권한을 부여받아야 합니다. 프로젝트를 생성하도록 허용된 경우 자신의 프로젝트에 액세스할 수 있는 권한이 자동으로 제공됩니다.

*프로젝트에는 별도의 **name**, **displayName**, **description**이 있을 수 있습니다.*

- *필수 항목인 **name**은 프로젝트의 고유 식별자이며 CLI 도구 또는 API를 사용할 때 가장 잘 보입니다. 최대 이름 길이는 63자입니다.*
- *선택적 **displayName**은 프로젝트가 웹 콘솔에 표시되는 방법입니다(기본값: **name**).*
- *선택적 **description**은 프로젝트에 대한 보다 자세한 설명으로, 웹 콘솔에서도 볼 수 있습니다.*

각 프로젝트의 범위는 다음과 같습니다.

오브젝트	설명
Objects	Pod, 서비스, 복제 컨트롤러 등입니다.
Policies	사용자는 오브젝트에서 이 규칙에 대해 작업을 수행할 수 있거나 수행할 수 없습니다.
Constraints	제한할 수 있는 각 종류의 오브젝트에 대한 할당량입니다.
Service accounts	서비스 계정은 프로젝트의 오브젝트에 지정된 액세스 권한으로 자동으로 작동합니다.

***dedicated-admin** 역할의 관리자는 프로젝트를 생성하고 프로젝트에 대한 관리 권한을 사용자 커뮤니티의 모든 멤버에게 위임할 수 있습니다. **dedicated-admin** 역할의 관리자는 개발자가 자신의 프로젝트를 만들 수 있도록 허용할 수도 있습니다.*

개발자와 관리자는 CLI 또는 웹 콘솔을 사용하여 프로젝트와 상호 작용할 수 있습니다.

6.3. 기본 프로젝트

*OpenShift Dedicated에는 여러 기본 프로젝트가 제공되며 **openshift-** 로 시작하는 프로젝트가 사용자에게 가장 중요합니다. 이러한 프로젝트는 Pod 및 기타 인프라 구성 요소로 실행되는 마스터 구성 요소를 호스팅합니다. 중요 Pod 주석 이 있는 네임스페이스에 생성된 Pod는 중요한 Pod로 간주되며, kubelet의 승인이 보장됩니다. 이러한 네임스페이스에서 마스터 구성 요소용으로 생성된 Pod는 이미 중요로 표시되어 있습니다.*

중요

기본 프로젝트에서 워크로드를 실행하거나 기본 프로젝트에 대한 액세스를 공유하지 마세요. 기본 프로젝트는 핵심 클러스터 구성 요소를 실행하기 위해 예약되어 있습니다.

*다음 기본 프로젝트는 높은 권한이 있는 것으로 간주됩니다. **default**, **kube-public**, **kube-system**, **openshift**, **openshift-infra**, **openshift-node** 및 **openshift.io/run-level** 레이블이 **0** 또는 **1** 로 설정된 기타 시스템 생성 프로젝트입니다. Pod 보안 승인, 보안 컨텍스트 제약 조건, 클러스터 리소스 할당량 및 이미지 참조 확인과 같은 승인 플러그인에 의존하는 기능은 높은 권한 있는 프로젝트에서 작동하지 않습니다.*

6.4. 클러스터 역할 및 바인딩 보기

*oc CLI에서 **oc describe** 명령을 사용하여 클러스터 역할 및 바인딩을 볼 수 있습니다.*

사전 요구 사항

- *oc CLI를 설치합니다.*
- *클러스터 역할 및 바인딩을 볼 수 있는 권한을 얻습니다.*

프로세스

1. *클러스터 역할 및 관련 규칙 집합을 보려면 다음을 수행합니다.*

```
$ oc describe clusterrole.rbac
```

출력 예

```
Name:         admin
Labels:       kubernetes.io/bootstrapping=rbac-defaults
Annotations:  rbac.authorization.kubernetes.io/autoupdate: true
PolicyRule:
  Resources                                  Non-Resource URLs  Resource Names  Verbs
  ---------                                  -----------------  --------------  -----
  .packages.apps.redhat.com                  []                 []              [* create update
patch delete get list watch]
  imagestreams                               []                 []              [create delete
deletecollection get list patch update watch create get list watch]
  imagestreams.image.openshift.io            []                 []              [create delete
deletecollection get list patch update watch create get list watch]
  secrets                                    []                 []              [create delete deletecollection
get list patch update watch get list watch create delete deletecollection patch update]
  buildconfigs/webhooks                      []                 []              [create delete
deletecollection get list patch update watch get list watch]
  buildconfigs                               []                 []              [create delete
deletecollection get list patch update watch get list watch]
  buildlogs                                  []                 []              [create delete deletecollection
get list patch update watch get list watch]
  deploymentconfigs/scale                    []                 []              [create delete
deletecollection get list patch update watch get list watch]
  deploymentconfigs                          []                 []              [create delete
deletecollection get list patch update watch get list watch]
  imagestreamimages                          []                 []              [create delete
deletecollection get list patch update watch get list watch]
  imagestreammappings                        []                 []              [create delete
deletecollection get list patch update watch get list watch]
  imagestreamtags                            []                 []              [create delete
deletecollection get list patch update watch get list watch]
  processedtemplates                         []                 []              [create delete
deletecollection get list patch update watch get list watch]
  routes                                     []                 []              [create delete deletecollection
get list patch update watch get list watch]
  templateconfigs                            []                 []              [create delete
deletecollection get list patch update watch get list watch]
  templateinstances                          []                 []              [create delete
deletecollection get list patch update watch get list watch]
  templates                                  []                 []              [create delete
```

```
deletecollection get list patch update watch get list watch]
  deploymentconfigs.apps.openshift.io/scale        []        []        [create delete
deletecollection get list patch update watch get list watch]
  deploymentconfigs.apps.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  buildconfigs.build.openshift.io/webhooks        []        []        [create delete
deletecollection get list patch update watch get list watch]
  buildconfigs.build.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  buildlogs.build.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  imagestreamimages.image.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  imagestreammappings.image.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  imagestreamtags.image.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  routes.route.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  processedtemplates.template.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  templateconfigs.template.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  templateinstances.template.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  templates.template.openshift.io        []        []        [create delete
deletecollection get list patch update watch get list watch]
  serviceaccounts        []        []        [create delete
deletecollection get list patch update watch impersonate create delete deletecollection patch
update get list watch]
  imagestreams/secrets        []        []        [create delete
deletecollection get list patch update watch]
  rolebindings        []        []        [create delete
deletecollection get list patch update watch]
  roles        []        []        [create delete deletecollection
get list patch update watch]
  rolebindings.authorization.openshift.io        []        []        [create delete
deletecollection get list patch update watch]
  roles.authorization.openshift.io        []        []        [create delete
deletecollection get list patch update watch]
  imagestreams.image.openshift.io/secrets        []        []        [create delete
deletecollection get list patch update watch]
  rolebindings.rbac.authorization.k8s.io        []        []        [create delete
deletecollection get list patch update watch]
  roles.rbac.authorization.k8s.io        []        []        [create delete
deletecollection get list patch update watch]
  networkpolicies.extensions        []        []        [create delete
deletecollection patch update create delete deletecollection get list patch update watch get
list watch]
  networkpolicies.networking.k8s.io        []        []        [create delete
deletecollection patch update create delete deletecollection get list patch update watch get
list watch]
  configmaps        []        []        [create delete
deletecollection patch update get list watch]
  endpoints        []        []        [create delete
deletecollection patch update get list watch]
```

```
  persistentvolumeclaims                                  []                []              [create delete
deletecollection patch update get list watch]
  pods                                                    []                []              [create delete deletecollection
patch update get list watch]
  replicationcontrollers/scale                            []                []              [create delete
deletecollection patch update get list watch]
  replicationcontrollers                                  []                []              [create delete
deletecollection patch update get list watch]
  services                                                []                []              [create delete deletecollection
patch update get list watch]
  daemonsets.apps                                         []                []              [create delete
deletecollection patch update get list watch]
  deployments.apps/scale                                  []                []              [create delete
deletecollection patch update get list watch]
  deployments.apps                                        []                []              [create delete
deletecollection patch update get list watch]
  replicasets.apps/scale                                  []                []              [create delete
deletecollection patch update get list watch]
  replicasets.apps                                        []                []              [create delete
deletecollection patch update get list watch]
  statefulsets.apps/scale                                 []                []              [create delete
deletecollection patch update get list watch]
  statefulsets.apps                                       []                []              [create delete
deletecollection patch update get list watch]
  horizontalpodautoscalers.autoscaling                    []                []              [create delete
deletecollection patch update get list watch]
  cronjobs.batch                                          []                []              [create delete
deletecollection patch update get list watch]
  jobs.batch                                              []                []              [create delete
deletecollection patch update get list watch]
  daemonsets.extensions                                   []                []              [create delete
deletecollection patch update get list watch]
  deployments.extensions/scale                            []                []              [create delete
deletecollection patch update get list watch]
  deployments.extensions                                  []                []              [create delete
deletecollection patch update get list watch]
  ingresses.extensions                                    []                []              [create delete
deletecollection patch update get list watch]
  replicasets.extensions/scale                            []                []              [create delete
deletecollection patch update get list watch]
  replicasets.extensions                                  []                []              [create delete
deletecollection patch update get list watch]
  replicationcontrollers.extensions/scale                 []                []              [create delete
deletecollection patch update get list watch]
  poddisruptionbudgets.policy                             []                []              [create delete
deletecollection patch update get list watch]
  deployments.apps/rollback                               []                []              [create delete
deletecollection patch update]
  deployments.extensions/rollback                         []                []              [create delete
deletecollection patch update]
  catalogsources.operators.coreos.com                     []                []              [create update
patch delete get list watch]
  clusterserviceversions.operators.coreos.com             []                []              [create update
patch delete get list watch]
  installplans.operators.coreos.com                       []                []              [create update
patch delete get list watch]
```

```
  packagemanifests.operators.coreos.com                    []                []                [create update
patch delete get list watch]
  subscriptions.operators.coreos.com                       []                []                [create update
patch delete get list watch]
  buildconfigs/instantiate                                 []                []                [create]
  buildconfigs/instantiatebinary                           []                []                [create]
  builds/clone                                             []                []                [create]
  deploymentconfigrollbacks                                []                []                [create]
  deploymentconfigs/instantiate                            []                []                [create]
  deploymentconfigs/rollback                               []                []                [create]
  imagestreamimports                                       []                []                [create]
  localresourceaccessreviews                               []                []                [create]
  localsubjectaccessreviews                                []                []                [create]
  podsecuritypolicyreviews                                 []                []                [create]
  podsecuritypolicyselfsubjectreviews                      []                []                [create]
  podsecuritypolicysubjectreviews                          []                []                [create]
  resourceaccessreviews                                    []                []                [create]
  routes/custom-host                                       []                []                [create]
  subjectaccessreviews                                     []                []                [create]
  subjectrulesreviews                                      []                []                [create]
  deploymentconfigrollbacks.apps.openshift.io              []                []                [create]
  deploymentconfigs.apps.openshift.io/instantiate          []                []                [create]
  deploymentconfigs.apps.openshift.io/rollback             []                []                [create]
  localsubjectaccessreviews.authorization.k8s.io           []                []                [create]
  localresourceaccessreviews.authorization.openshift.io    []                []                [create]
  localsubjectaccessreviews.authorization.openshift.io     []                []                [create]
  resourceaccessreviews.authorization.openshift.io         []                []                [create]
  subjectaccessreviews.authorization.openshift.io          []                []                [create]
  subjectrulesreviews.authorization.openshift.io           []                []                [create]
  buildconfigs.build.openshift.io/instantiate              []                []                [create]
  buildconfigs.build.openshift.io/instantiatebinary        []                []                [create]
  builds.build.openshift.io/clone                          []                []                [create]
  imagestreamimports.image.openshift.io                    []                []                [create]
  routes.route.openshift.io/custom-host                    []                []                [create]
  podsecuritypolicyreviews.security.openshift.io           []                []                [create]
  podsecuritypolicyselfsubjectreviews.security.openshift.io []               []                [create]
  podsecuritypolicysubjectreviews.security.openshift.io    []                []                [create]
  jenkins.build.openshift.io                               []                []                [edit view view admin
edit view]
  builds                                                   []                []                [get create delete
deletecollection get list patch update watch get list watch]
  builds.build.openshift.io                                []                []                [get create delete
deletecollection get list patch update watch get list watch]
  projects                                                 []                []                [get delete get delete get patch
update]
  projects.project.openshift.io                            []                []                [get delete get delete
get patch update]
  namespaces                                               []                []                [get get list watch]
  pods/attach                                              []                []                [get list watch create delete
deletecollection patch update]
  pods/exec                                                []                []                [get list watch create delete
deletecollection patch update]
  pods/portforward                                         []                []                [get list watch create
delete deletecollection patch update]
  pods/proxy                                               []                []                [get list watch create delete
deletecollection patch update]
```

```
  services/proxy                                  []                  []                  [get list watch create delete
deletecollection patch update]
  routes/status                                   []                  []                  [get list watch update]
  routes.route.openshift.io/status                []                  []                  [get list watch update]
  appliedclusterresourcequotas                    []                  []                  [get list watch]
  bindings                                        []                  []                  [get list watch]
  builds/log                                      []                  []                  [get list watch]
  deploymentconfigs/log                           []                  []                  [get list watch]
  deploymentconfigs/status                        []                  []                  [get list watch]
  events                                          []                  []                  [get list watch]
  imagestreams/status                             []                  []                  [get list watch]
  limitranges                                     []                  []                  [get list watch]
  namespaces/status                               []                  []                  [get list watch]
  pods/log                                        []                  []                  [get list watch]
  pods/status                                     []                  []                  [get list watch]
  replicationcontrollers/status                   []                  []                  [get list watch]
  resourcequotas/status                           []                  []                  [get list watch]
  resourcequotas                                  []                  []                  [get list watch]
  resourcequotausages                             []                  []                  [get list watch]
  rolebindingrestrictions                         []                  []                  [get list watch]
  deploymentconfigs.apps.openshift.io/log         []                  []                  [get list watch]
  deploymentconfigs.apps.openshift.io/status      []                  []                  [get list watch]
  controllerrevisions.apps                        []                  []                  [get list watch]
  rolebindingrestrictions.authorization.openshift.io  []              []                  [get list watch]
  builds.build.openshift.io/log                   []                  []                  [get list watch]
  imagestreams.image.openshift.io/status          []                  []                  [get list watch]
  appliedclusterresourcequotas.quota.openshift.io  []                 []                  [get list watch]
  imagestreams/layers                             []                  []                  [get update get]
  imagestreams.image.openshift.io/layers          []                  []                  [get update get]
  builds/details                                  []                  []                  [update]
  builds.build.openshift.io/details               []                  []                  [update]


Name:         basic-user
Labels:       <none>
Annotations:  openshift.io/description: A user that can get basic information about projects.
              rbac.authorization.kubernetes.io/autoupdate: true
PolicyRule:
 Resources                                  Non-Resource URLs  Resource Names  Verbs
  ---------                                 -----------------  --------------  -----
  selfsubjectrulesreviews                   []                 []              [create]
  selfsubjectaccessreviews.authorization.k8s.io  []            []              [create]
  selfsubjectrulesreviews.authorization.openshift.io  []       []              [create]
  clusterroles.rbac.authorization.k8s.io    []                 []              [get list watch]
  clusterroles                              []                 []              [get list]
  clusterroles.authorization.openshift.io   []                 []              [get list]
  storageclasses.storage.k8s.io             []                 []              [get list]
  users                                     []                 [~]             [get]
  users.user.openshift.io                   []                 [~]             [get]
  projects                                  []                 []              [list watch]
  projects.project.openshift.io             []                 []              [list watch]
  projectrequests                           []                 []              [list]
  projectrequests.project.openshift.io      []                 []              [list]

Name:         cluster-admin
Labels:       kubernetes.io/bootstrapping=rbac-defaults
```

```
Annotations:  rbac.authorization.kubernetes.io/autoupdate: true
PolicyRule:
Resources  Non-Resource URLs  Resource Names  Verbs
---------  -----------------  --------------  -----
*.*        []                 []              [*]
           [*]                []              [*]

...
```

2. 다양한 역할에 바인딩된 사용자 및 그룹을 표시하는 현재 클러스터 역할 바인딩 집합을 보려면 다음을 수행하십시오.

```
$ oc describe clusterrolebinding.rbac
```

출력 예

```
Name:         alertmanager-main
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  alertmanager-main
Subjects:
  Kind           Name             Namespace
  ----           ----             ---------
  ServiceAccount  alertmanager-main  openshift-monitoring


Name:         basic-users
Labels:       <none>
Annotations:  rbac.authorization.kubernetes.io/autoupdate: true
Role:
  Kind:  ClusterRole
  Name:  basic-user
Subjects:
  Kind   Name               Namespace
  ----   ----               ---------
  Group  system:authenticated


Name:         cloud-credential-operator-rolebinding
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  cloud-credential-operator-role
Subjects:
  Kind           Name     Namespace
  ----           ----     ---------
  ServiceAccount  default  openshift-cloud-credential-operator


Name:         cluster-admin
Labels:       kubernetes.io/bootstrapping=rbac-defaults
Annotations:  rbac.authorization.kubernetes.io/autoupdate: true
```

```
Role:
  Kind:  ClusterRole
  Name:  cluster-admin
Subjects:
  Kind   Name           Namespace
  ----   ----           ---------
  Group  system:masters


Name:         cluster-admins
Labels:       <none>
Annotations:  rbac.authorization.kubernetes.io/autoupdate: true
Role:
  Kind:  ClusterRole
  Name:  cluster-admin
Subjects:
  Kind   Name                Namespace
  ----   ----                ---------
  Group  system:cluster-admins
  User   system:admin


Name:         cluster-api-manager-rolebinding
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  cluster-api-manager-role
Subjects:
  Kind            Name     Namespace
  ----            ----     ---------
  ServiceAccount  default  openshift-machine-api

...
```

6.5. 로컬 역할 및 바인딩 보기

oc CLI에서 **oc describe** 명령을 사용하여 로컬 역할 및 바인딩을 볼 수 있습니다.

사전 요구 사항

- **oc** CLI를 설치합니다.
- 로컬 역할 및 바인딩을 볼 수 있는 권한을 얻습니다.
  - **admin** 기본 클러스터 역할이 로컬로 바인딩된 사용자는 해당 프로젝트의 역할 및 바인딩을 보고 관리할 수 있습니다.

절차

1. 현재 프로젝트의 다양한 역할에 바인딩된 사용자 및 그룹을 표시하는 현재의 로컬 역할 바인딩 집합을 보려면 다음을 실행합니다.

```
$ oc describe rolebinding.rbac
```

2. *다른 프로젝트에 대한 로컬 역할 바인딩을 보려면 명령에 **-n** 플래그를 추가합니다.*

```
$ oc describe rolebinding.rbac -n joe-project
```

출력 예

```
Name:         admin
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  admin
Subjects:
  Kind  Name       Namespace
  ----  ----       ---------
  User  kube:admin


Name:         system:deployers
Labels:       <none>
Annotations:  openshift.io/description:
              Allows deploymentconfigs in this namespace to rollout pods in
              this namespace.  It is auto-managed by a controller; remove
              subjects to disa...
Role:
  Kind:  ClusterRole
  Name:  system:deployer
Subjects:
  Kind           Name      Namespace
  ----           ----      ---------
  ServiceAccount  deployer  joe-project


Name:         system:image-builders
Labels:       <none>
Annotations:  openshift.io/description:
              Allows builds in this namespace to push images to this
              namespace.  It is auto-managed by a controller; remove subjects
              to disable.
Role:
  Kind:  ClusterRole
  Name:  system:image-builder
Subjects:
  Kind           Name     Namespace
  ----           ----     ---------
  ServiceAccount  builder  joe-project


Name:         system:image-pullers
Labels:       <none>
Annotations:  openshift.io/description:
              Allows all pods in this namespace to pull images from this
              namespace.  It is auto-managed by a controller; remove subjects
              to disable.
Role:
```

```
  Kind:  ClusterRole
  Name:  system:image-puller
Subjects:
  Kind   Name                            Namespace
  ----   ----                            ---------
  Group  system:serviceaccounts:joe-project
```

6.6. 사용자 역할 추가

oc adm 관리자 CLI를 사용하여 역할 및 바인딩을 관리할 수 있습니다.

사용자 또는 그룹에 역할을 바인딩하거나 추가하면 역할에 따라 사용자 또는 그룹에 부여되는 액세스 권한이 부여됩니다. **oc adm policy** 명령을 사용하여 사용자 및 그룹에 역할을 추가하거나 사용자 및 그룹으로부터 역할을 제거할 수 있습니다.

기본 클러스터 역할을 프로젝트의 로컬 사용자 또는 그룹에 바인딩할 수 있습니다.

절차

1. 특정 프로젝트의 사용자에게 역할을 추가합니다.

```
$ oc adm policy add-role-to-user <role> <user> -n <project>
```

예를 들면 다음을 실행하여 **joe** 프로젝트의 **alice** 사용자에게 **admin** 역할을 추가할 수 있습니다.

```
$ oc adm policy add-role-to-user admin alice -n joe
```

작은 정보

다음 YAML을 적용하여 사용자에게 역할을 추가할 수도 있습니다.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: admin-0
  namespace: joe
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: alice
```

2. 로컬 역할 바인딩을 보고 출력에 추가되었는지 확인합니다.

```
$ oc describe rolebinding.rbac -n <project>
```

예를 들어, **joe** 프로젝트의 로컬 역할 바인딩을 보려면 다음을 수행합니다.

```
$ oc describe rolebinding.rbac -n joe
```

출력 예

```
Name:         admin
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  admin
Subjects:
  Kind  Name        Namespace
  ----  ----        ---------
  User  kube:admin


Name:         admin-0
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  admin
Subjects:
  Kind  Name   Namespace
  ----  ----   ---------
  User  alice  (1)


Name:         system:deployers
Labels:       <none>
Annotations:  openshift.io/description:
              Allows deploymentconfigs in this namespace to rollout pods in
              this namespace.  It is auto-managed by a controller; remove
              subjects to disa...
Role:
  Kind:  ClusterRole
  Name:  system:deployer
Subjects:
  Kind           Name      Namespace
  ----           ----      ---------
  ServiceAccount  deployer  joe


Name:         system:image-builders
Labels:       <none>
Annotations:  openshift.io/description:
              Allows builds in this namespace to push images to this
              namespace.  It is auto-managed by a controller; remove subjects
              to disable.
Role:
  Kind:  ClusterRole
  Name:  system:image-builder
Subjects:
  Kind           Name     Namespace
  ----           ----     ---------
  ServiceAccount  builder  joe
```

```
Name:         system:image-pullers
Labels:       <none>
Annotations:  openshift.io/description:
                Allows all pods in this namespace to pull images from this
                namespace.  It is auto-managed by a controller; remove subjects
                to disable.
Role:
  Kind:  ClusterRole
  Name:  system:image-puller
Subjects:
  Kind   Name                         Namespace
  ----   ----                         ---------
  Group  system:serviceaccounts:joe
```

1 ***alice** 사용자가 **admins RoleBinding**에 추가되었습니다.*

6.7. 로컬 역할 생성

프로젝트의 로컬 역할을 생성하여 이 역할을 사용자에게 바인딩할 수 있습니다.

절차

1. *프로젝트의 로컬 역할을 생성하려면 다음 명령을 실행합니다.*

```
$ oc create role <name> --verb=<verb> --resource=<resource> -n <project>
```

이 명령에서는 다음을 지정합니다.

- ***<name>**: 로컬 역할 이름*
- ***<verb>**: 역할에 적용할 동사를 쉼표로 구분한 목록*
- ***<resource>**: 역할이 적용되는 리소스*
- ***<project>**: 프로젝트 이름*

*예를 들어, 사용자가 **blue** 프로젝트의 Pod를 볼 수 있는 로컬 역할을 생성하려면 다음 명령을 실행합니다.*

```
$ oc create role podview --verb=get --resource=pod -n blue
```

2. *새 역할을 사용자에게 바인딩하려면 다음 명령을 실행합니다.*

```
$ oc adm policy add-role-to-user podview user2 --role-namespace=blue -n blue
```

6.8. 로컬 역할 바인딩 명령

*다음 작업을 사용하여 로컬 역할 바인딩에 대한 사용자 또는 그룹의 연결된 역할을 관리하는 경우, **-n** 플래그를 사용하여 프로젝트를 지정할 수 있습니다. 지정하지 않으면 현재 프로젝트가 사용됩니다.*

다음 명령을 사용하여 로컬 RBAC를 관리할 수 있습니다.

표 6.1. 로컬 역할 바인딩 작업

명령	설명
$ oc adm policy who-can *<verb> <resource>*	리소스에 작업을 수행할 수 있는 사용자를 나타냅니다.
$ oc adm policy add-role-to-user *<role> <username>*	현재 프로젝트에서 지정된 사용자에게 지정된 역할을 바인딩합니다.
$ oc adm policy remove-role-from-user *<role> <username>*	현재 프로젝트에서 지정된 사용자로부터 지정된 역할을 제거합니다.
$ oc adm policy remove-user *<username>*	현재 프로젝트에서 지정된 사용자 및 해당 사용자의 역할을 모두 제거합니다.
$ oc adm policy add-role-to-group *<role> <groupname>*	현재 프로젝트에서 지정된 그룹에 지정된 역할을 바인딩합니다.
$ oc adm policy remove-role-from-group *<role> <groupname>*	현재 프로젝트에서 지정된 그룹의 지정된 역할을 제거합니다.
$ oc adm policy remove-group *<groupname>*	현재 프로젝트에서 지정된 그룹과 해당 그룹의 역할을 모두 제거합니다.

6.9. 클러스터 역할 바인딩 명령

*다음 작업을 사용하여 클러스터 역할 바인딩을 관리할 수도 있습니다. 클러스터 역할 바인딩에 네임스페이스가 아닌 리소스가 사용되므로 **-n** 플래그가 해당 작업에 사용되지 않습니다.*

표 6.2. 클러스터 역할 바인딩 작업

명령	설명
$ oc adm policy add-cluster-role-to-user *<role> <username>*	클러스터의 모든 프로젝트에 대해 지정된 사용자에게 지정된 역할을 바인딩합니다.
$ oc adm policy remove-cluster-role-from-user *<role> <username>*	클러스터의 모든 프로젝트에 대해 지정된 사용자로부터 지정된 역할을 제거합니다.
$ oc adm policy add-cluster-role-to-group *<role> <groupname>*	클러스터의 모든 프로젝트에 대해 지정된 역할을 지정된 그룹에 바인딩합니다.
$ oc adm policy remove-cluster-role-from-group *<role> <groupname>*	클러스터의 모든 프로젝트에 대해 지정된 그룹에서 지정된 역할을 제거합니다.

6.10. 사용자에게 관리자 권한 부여

*클러스터의 ID 공급자를 구성하고 ID 공급자에 사용자를 추가한 후 **dedicated-admin** 클러스터 권한을 사용자에게 부여할 수 있습니다.*

사전 요구 사항

- *OpenShift Cluster Manager 에 로그인했습니다.*
- *OpenShift Dedicated 클러스터를 생성하셨습니다.*
- *클러스터의 ID 공급자를 구성했습니다.*

프로세스

1. *OpenShift Cluster Manager 로 이동하여 클러스터를 선택합니다.*
2. ***액세스 제어** 탭을 클릭합니다.*
3. ***클러스터 역할 및 액세스** 탭에서 **사용자 추가** 를 클릭합니다.*
4. *ID 공급자 사용자의 사용자 ID를 입력합니다.*
5. ***사용자 추가** 를 클릭하여 **dedicated-admin** 클러스터 권한을 사용자에게 부여합니다.*

검증

- *권한을 부여하면 사용자가 클러스터의 OpenShift Cluster Manager 페이지에서 **Access Control→ Cluster Roles 및 Access** 에서 **dedicated-admins** 그룹의 일부로 나열됩니다.*

6.11. 인증되지 않은 그룹의 클러스터 역할 바인딩

참고

OpenShift Dedicated 4.16 이전에는 인증되지 않은 그룹이 일부 클러스터 역할에 액세스할 수 있었습니다. OpenShift Dedicated 4.16 이전 버전에서 업데이트된 클러스터는 인증되지 않은 그룹에 대해 이 액세스를 유지합니다.

보안상의 이유로 OpenShift Dedicated 4에서는 인증되지 않은 그룹이 클러스터 역할에 대한 기본 액세스 권한을 허용하지 않습니다.

***system:unauthenticated** 를 클러스터 역할에 추가해야 하는 사용 사례가 있습니다.*

클러스터 관리자는 인증되지 않은 사용자를 다음 클러스터 역할에 추가할 수 있습니다.

- ***system:scope-impersonation***
- ***system:webhook***
- ***system:oauth-token-deleter***
- ***self-access-reviewer***

중요

인증되지 않은 액세스를 수정할 때 항상 조직의 보안 표준을 준수하는지 확인하십시오.

7장. 서비스 계정 이해 및 생성

7.1. 서비스 계정 개요

서비스 계정은 구성 요소가 API에 직접 액세스할 수 있는 OpenShift Dedicated 계정입니다. 서비스 계정은 각 프로젝트 내에 존재하는 API 오브젝트입니다. 서비스 계정을 사용하면 일반 사용자의 자격 증명을 공유하지 않고도 API 액세스 권한을 유연하게 제어할 수 있습니다.

OpenShift Dedicated CLI 또는 웹 콘솔을 사용하면 API 토큰이 API에 대한 인증을 받습니다. 일반 사용자의 자격 증명을 사용하지 않고도 API에 액세스할 수 있도록 구성 요소를 서비스 계정과 연결할 수 있습니다.

각 서비스 계정의 사용자 이름은 프로젝트와 이름에서 파생됩니다.

```
system:serviceaccount:<project>:<name>
```

모든 서비스 계정은 다음 두 그룹의 멤버이기도 합니다.

그룹	설명
system:serviceaccounts	시스템의 모든 서비스 계정이 포함됩니다.
system:serviceaccounts:<project>	지정된 프로젝트의 모든 서비스 계정이 포함됩니다.

각 서비스 계정에는 다음 두 가지 시크릿이 자동으로 포함됩니다.

- API 토큰
- OpenShift Container Registry 자격 증명

생성된 API 토큰 및 레지스트리 인증 정보는 만료되지 않지만 시크릿을 삭제하여 무효화할 수 있습니다. 시크릿을 삭제하면 새로운 시크릿이 자동으로 생성되어 대체됩니다.

7.2. 서비스 계정 생성

프로젝트에서 서비스 계정을 생성하고 역할에 바인딩하여 권한을 부여할 수 있습니다.

절차

1. 선택 사항: 현재 프로젝트에서 서비스 계정을 보려면 다음을 수행합니다.

```
$ oc get sa
```

출력 예

```
NAME      SECRETS   AGE
builder   2         2d
default   2         2d
deployer  2         2d
```

2. 현재 프로젝트에서 새 서비스 계정을 생성하려면 다음을 수행합니다.

```
$ oc create sa <service_account_name> 1
```

1 다른 프로젝트에서 서비스 계정을 생성하려면 **-n <project_name>**을 지정합니다.

출력 예

```
serviceaccount "robot" created
```

작은 정보

다음 YAML을 적용하여 서비스 계정을 생성할 수도 있습니다.

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: <service_account_name>
  namespace: <current_project>
```

3. 선택 사항: 서비스 계정의 시크릿을 확인합니다.

```
$ oc describe sa robot
```

출력 예

```
Name:                robot
Namespace:           project1
Labels:              <none>
Annotations:         <none>
Image pull secrets:  robot-dockercfg-qzbhb
Mountable secrets:   robot-dockercfg-qzbhb
Tokens:              robot-token-f4khf
Events:              <none>
```

7.3. 서비스 계정에 역할을 부여하는 예

일반 사용자 계정에 역할을 부여하는 것과 같은 방식으로 서비스 계정에 역할을 부여할 수 있습니다.

- 현재 프로젝트의 서비스 계정을 수정할 수 있습니다. 예를 들어, **top-secret** 개체에서 **robot** 서비스 계정에 **view** 역할을 추가하려면 다음을 수행합니다.

```
$ oc policy add-role-to-user view system:serviceaccount:top-secret:robot
```

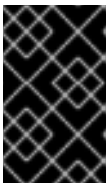
작은 정보

다음 YAML을 적용하여 역할을 추가할 수도 있습니다.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: view
  namespace: top-secret
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: view
subjects:
- kind: ServiceAccount
  name: robot
  namespace: top-secret
```

- *프로젝트의 특정 서비스 계정에 대한 액세스 권한을 부여할 수도 있습니다. 예를 들어 서비스 계정이 속하는 프로젝트에서* ***-z*** *플래그를 사용하고* ***<service_account_name>****을 지정합니다.*

```
$ oc policy add-role-to-user <role_name> -z <service_account_name>
```



중요

프로젝트의 특정 서비스 계정에 대한 액세스 권한을 부여하려면 ***-z*** *플래그를 사용합니다. 이 플래그를 사용하면 오타를 방지하고 지정된 서비스 계정에만 액세스 권한을 부여할 수 있습니다.*

작은 정보

다음 YAML을 적용하여 역할을 추가할 수도 있습니다.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: <rolebinding_name>
  namespace: <current_project_name>
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: <role_name>
subjects:
- kind: ServiceAccount
  name: <service_account_name>
  namespace: <current_project_name>
```

- *다른 네임스페이스를 수정하려면 다음 예제와 같이* ***-n*** *옵션을 사용하여 적용되는 프로젝트 네임스페이스를 나타낼 수 있습니다.*
  - *예를 들어 모든 프로젝트의 모든 서비스 계정에서* ***my-project*** *프로젝트의 리소스를 볼 수 있도록 하려면 다음을 실행합니다.*

```
$ oc policy add-role-to-group view system:serviceaccounts -n my-project
```

작은 정보

다음 YAML을 적용하여 역할을 추가할 수도 있습니다.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: view
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: view
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts
```

- ***managers*** *프로젝트의 모든 서비스 계정에서* ***my-project*** *프로젝트의 리소스를 편집할 수 있도록 하려면 다음을 실행합니다.*

```
$ oc policy add-role-to-group edit system:serviceaccounts:managers -n my-project
```

작은 정보

다음 YAML을 적용하여 역할을 추가할 수도 있습니다.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: edit
  namespace: my-project
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: edit
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:serviceaccounts:managers
```

8장. 애플리케이션에서 서비스 계정 사용

8.1. 서비스 계정 개요

서비스 계정은 구성 요소가 API에 직접 액세스할 수 있는 OpenShift Dedicated 계정입니다. 서비스 계정은 각 프로젝트 내에 존재하는 API 오브젝트입니다. 서비스 계정을 사용하면 일반 사용자의 자격 증명을 공유하지 않고도 API 액세스 권한을 유연하게 제어할 수 있습니다.

OpenShift Dedicated CLI 또는 웹 콘솔을 사용하면 API 토큰이 API에 대한 인증을 받습니다. 일반 사용자의 자격 증명을 사용하지 않고도 API에 액세스할 수 있도록 구성 요소를 서비스 계정과 연결할 수 있습니다.

각 서비스 계정의 사용자 이름은 프로젝트와 이름에서 파생됩니다.

```
system:serviceaccount:<project>:<name>
```

모든 서비스 계정은 다음 두 그룹의 멤버이기도 합니다.

그룹	설명
system:serviceaccounts	시스템의 모든 서비스 계정이 포함됩니다.
system:serviceaccounts:<project>	지정된 프로젝트의 모든 서비스 계정이 포함됩니다.

각 서비스 계정에는 다음 두 가지 시크릿이 자동으로 포함됩니다.

- API 토큰
- OpenShift Container Registry 자격 증명

생성된 API 토큰 및 레지스트리 인증 정보는 만료되지 않지만 시크릿을 삭제하여 무효화할 수 있습니다. 시크릿을 삭제하면 새로운 시크릿이 자동으로 생성되어 대체됩니다.

8.2. 기본 서비스 계정

OpenShift Dedicated 클러스터에는 클러스터 관리를 위한 기본 서비스 계정이 포함되어 있으며 각 프로젝트에 대해 더 많은 서비스 계정을 생성합니다.

8.2.1. 기본 클러스터 서비스 계정

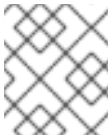
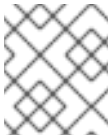
다양한 인프라 컨트롤러가 서비스 계정 자격 증명을 사용하여 실행됩니다. 다음 서비스 계정은 서버 시작 시 OpenShift Dedicated 인프라 프로젝트(**openshift-infra**)에 생성되고 클러스터 전체에서 다음 역할이 지정됩니다.

서비스 계정	설명
replication-controller	**system:replication-controller** 역할이 할당됨
deployment-controller	**system:deployment-controller** 역할이 할당됨

서비스 계정	설명
build-controller	**system:build-controller** 역할이 할당되었습니다. 또한 권한 있는 빌드 pod를 생성하기 위해 권한 있는 보안 컨텍스트 제약 조건에 **build-controller** 서비스 계정이 포함되어 있습니다.

8.2.2. 기본 프로젝트 서비스 계정 및 역할

프로젝트당 3개의 서비스 계정이 자동으로 생성됩니다.

서비스 계정	사용법
builder	빌드 Pod에서 사용합니다. 내부 Docker 레지스트리를 사용하여 프로젝트의 이미지 스트림으로 이미지를 밀어 넣을 수 있는 **system:image-builder** 역할이 제공됩니다.  **참고** **Build** 클러스터 기능이 활성화되지 않은 경우 **builder** 서비스 계정이 생성되지 않습니다.
deployer	배포 Pod에서 사용하며 **system:deployer** 역할을 지정하는 경우 프로젝트에서 복제 컨트롤러 및 Pod를 보고 수정할 수 있습니다.  **참고** **DeploymentConfig** 클러스터 기능이 활성화되지 않은 경우 **배포자** 서비스 계정이 생성되지 않습니다.
default	다른 서비스 계정을 지정하지 않는 한 기타 모든 Pod를 실행하는 데 사용합니다.

*프로젝트의 모든 서비스 계정에는 **system:image-puller** 역할이 부여되어 내부 컨테이너 이미지 레지스트리를 사용하여 프로젝트의 모든 이미지 스트림에서 이미지를 가져올 수 있습니다.*

8.2.3. 자동으로 생성된 이미지 풀 시크릿

기본적으로 OpenShift Dedicated는 각 서비스 계정에 대한 이미지 풀 시크릿을 생성합니다.

참고

OpenShift Dedicated 4.16 이전에는 생성된 각 서비스 계정에 대해 장기 서비스 계정 API 토큰 시크릿도 생성되었습니다. OpenShift Dedicated 4.16부터 이 서비스 계정 API 토큰 시크릿은 더 이상 생성되지 않습니다.

4로 업그레이드한 후 장기 서비스 계정 API 토큰 시크릿은 삭제되지 않으며 계속 작동합니다. 클러스터에서 사용 중인 장기 API 토큰을 감지하거나 필요하지 않은 경우 삭제하는 방법에 대한 자세한 내용은 OpenShift Container Platform의 장기 서비스 계정 API 토큰을 참조하십시오.

이 이미지 풀 시크릿은 OpenShift 이미지 레지스트리를 클러스터의 사용자 인증 및 권한 부여 시스템에 통합하기 위해 필요합니다.

*그러나 **ImageRegistry** 기능을 활성화하지 않거나 Cluster Image Registry Operator 구성에서 통합 OpenShift 이미지 레지스트리를 비활성화하면 각 서비스 계정에 대해 이미지 풀 시크릿이 생성되지 않습니다.*

이전에 활성화된 클러스터에서 통합 OpenShift 이미지 레지스트리가 비활성화되면 이전에 생성된 이미지 가져오기 보안이 자동으로 삭제됩니다.

8.3. 서비스 계정 생성

프로젝트에서 서비스 계정을 생성하고 역할에 바인딩하여 권한을 부여할 수 있습니다.

절차

1. *선택 사항: 현재 프로젝트에서 서비스 계정을 보려면 다음을 수행합니다.*

```
$ oc get sa
```

출력 예

```
NAME      SECRETS   AGE
builder   2         2d
default   2         2d
deployer  2         2d
```

2. *현재 프로젝트에서 새 서비스 계정을 생성하려면 다음을 수행합니다.*

```
$ oc create sa <service_account_name> (1)
```

(1) *다른 프로젝트에서 서비스 계정을 생성하려면 **-n <project_name>**을 지정합니다.*

출력 예

```
serviceaccount "robot" created
```

작은 정보

다음 YAML을 적용하여 서비스 계정을 생성할 수도 있습니다.

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: <service_account_name>
  namespace: <current_project>
```

3. *선택 사항: 서비스 계정의 시크릿을 확인합니다.*

```
$ oc describe sa robot
```

출력 예

```
Name:                robot
Namespace:           project1
Labels:              <none>
Annotations:         <none>
Image pull secrets:  robot-dockercfg-qzbhb
Mountable secrets:   robot-dockercfg-qzbhb
Tokens:              robot-token-f4khf
Events:              <none>
```

9장. 서비스 계정을 OAUTH 클라이언트로 사용

9.1. OAUTH 클라이언트로서의 서비스 계정

서비스 계정을 제한된 형태의 OAuth 클라이언트로 사용할 수 있습니다. 서비스 계정에서는 서비스 계정의 자체 네임스페이스 내에 있는 일부 기본 사용자 정보 및 역할 기반 권한에 액세스할 수 있는 부분적인 범위만 요청할 수 있습니다.

- **user:info**
- **user:check-access**
- **role:<any_role>:<service_account_namespace>**
- **role:<any_role>:<service_account_namespace>:!**

서비스 계정을 OAuth 클라이언트로 사용하는 경우에는 다음과 같습니다.

- **client_id**는 **system:serviceaccount:<service_account_namespace>:<service_account_name>**입니다.
- **client_secret**은 해당 서비스 계정의 API 토큰 중 하나일 수 있습니다. 예를 들면 다음과 같습니다.

```
$ oc sa get-token <service_account_name>
```

- **WWW-Authenticate** 챌린지를 가져오려면 서비스 계정의 **serviceaccounts.openshift.io/oauth-want-challenges** 주석을 **true**로 설정합니다.
- **redirect_uri**는 서비스 계정의 주석과 일치해야 합니다.

9.1.1. 서비스 계정의 URI를 OAuth 클라이언트로 리디렉션

주석 키에는 접두사 **serviceaccounts.openshift.io/oauth-redirecturi.** 또는 **serviceaccounts.openshift.io/oauth-redirectreference.**가 있어야 합니다. 예를 들면 다음과 같습니다.

```
serviceaccounts.openshift.io/oauth-redirecturi.<name>
```

가장 간단한 형식의 주석을 사용하여 유효한 리디렉션 URI를 직접 지정할 수 있습니다. 예를 들면 다음과 같습니다.

```
"serviceaccounts.openshift.io/oauth-redirecturi.first":  "https://example.com"
"serviceaccounts.openshift.io/oauth-redirecturi.second": "https://other.com"
```

위 예에서 **first** 및 **second**라는 접미사는 두 개의 유효한 리디렉션 URI를 구분하는 데 사용됩니다.

복잡한 구성에서는 정적 리디렉션 URI로는 충분하지 않을 수 있습니다. 예를 들면 특정 경로의 모든 Ingress를 유효한 것으로 간주해야 할 수 있습니다. 이 경우 **serviceaccounts.openshift.io/oauth-redirectreference** 접두사를 통한 동적 리디렉션 URI가 작동합니다.

예를 들면 다음과 같습니다.

```
"serviceaccounts.openshift.io/oauth-redirectreference.first": "
{\"kind\":\"OAuthRedirectReference\",\"apiVersion\":\"v1\",\"reference\":
{\"kind\":\"Route\",\"name\":\"jenkins\"}}"
```

-

이 주석의 값은 직렬화된 JSON 데이터를 포함하므로 확장된 형식으로 쉽게 볼 수 있습니다.

```
{
  "kind": "OAuthRedirectReference",
  "apiVersion": "v1",
  "reference": {
    "kind": "Route",
    "name": "jenkins"
  }
}
```

*이제 **OAuthRedirectReference**를 통해 **jenkins**라는 경로를 참조할 수 있음을 확인할 수 있습니다. 따라서 해당 경로의 모든 Ingress가 이제 유효한 것으로 간주됩니다. **OAuthRedirectReference**의 전체 사양은 다음과 같습니다.*

```
{
  "kind": "OAuthRedirectReference",
  "apiVersion": "v1",
  "reference": {
    "kind": ..., (1)
    "name": ..., (2)
    "group": ... (3)
  }
}
```

1. ***kind**는 참조되는 오브젝트의 유형을 나타냅니다. 현재는 **route**만 지원됩니다.*
2. ***name**은 오브젝트의 이름을 나타냅니다. 오브젝트는 서비스 계정과 동일한 네임스페이스에 있어야 합니다.*
3. ***group**은 오브젝트 그룹을 나타냅니다. 경로 그룹이 빈 문자열이므로 이 필드를 비워둡니다.*

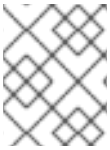
두 주석 접두사를 결합하여 참조 오브젝트에서 제공하는 데이터를 덮어쓸 수 있습니다. 예를 들면 다음과 같습니다.

```
"serviceaccounts.openshift.io/oauth-redirecturi.first":  "custompath"
"serviceaccounts.openshift.io/oauth-redirectreference.first": "
{\"kind\":\"OAuthRedirectReference\",\"apiVersion\":\"v1\",\"reference\":
{\"kind\":\"Route\",\"name\":\"jenkins\"}}"
```

***first** 접미사는 주석을 연결하는 데 사용됩니다. **jenkins** 경로에 **https://example.com**의 Ingress가 있다고 가정하면 이제 **https://example.com/custompath**는 유효한 것으로 간주되지만 **https://example.com**은 유효한 것으로 간주되지 않습니다. 데이터 덮어쓰기를 부분적으로 제공하는 형식은 다음과 같습니다.*

유형	구문
스키마	"https://"
호스트 이름	"//website.com"
포트	"//:8000"

유형	구문
경로	"examplepath"



참고

호스트 이름 덮어쓰기를 지정하면 참조된 오브젝트의 호스트 이름 데이터가 교체되므로 바람직한 동작이 아닙니다.

위 구문의 모든 조합은 다음과 같은 형식을 사용하여 결합할 수 있습니다.

<scheme:>//<hostname><:port>/<path>

유연성을 개선하기 위해 동일한 오브젝트를 두 번 이상 참조할 수 있습니다.

```
"serviceaccounts.openshift.io/oauth-redirecturi.first":  "custompath"
"serviceaccounts.openshift.io/oauth-redirectreference.first": "
{\"kind\":\"OAuthRedirectReference\",\"apiVersion\":\"v1\",\"reference\":
{\"kind\":\"Route\",\"name\":\"jenkins\"}}"
"serviceaccounts.openshift.io/oauth-redirecturi.second":  "//:8000"
"serviceaccounts.openshift.io/oauth-redirectreference.second": "
{\"kind\":\"OAuthRedirectReference\",\"apiVersion\":\"v1\",\"reference\":
{\"kind\":\"Route\",\"name\":\"jenkins\"}}"
```

jenkins*라는 경로에* ***https://example.com*** *Ingress가 있다고 가정하면* ***https://example.com:8000*** *및* ***https://example.com/custompath*** *가 모두 유효한 것으로 간주됩니다.*

정적 및 동적 주석을 동시에 사용하여 원하는 동작을 수행할 수 있습니다.

```
"serviceaccounts.openshift.io/oauth-redirectreference.first": "
{\"kind\":\"OAuthRedirectReference\",\"apiVersion\":\"v1\",\"reference\":
{\"kind\":\"Route\",\"name\":\"jenkins\"}}"
"serviceaccounts.openshift.io/oauth-redirecturi.second": "https://other.com"
```

10장. 범위 지정 토큰

10.1. 범위 지정 토큰 정보

범위가 지정된 토큰을 생성하여 일부 권한을 다른 사용자 또는 서비스 계정에 위임할 수 있습니다. 예를 들어, 프로젝트 관리자가 Pod 생성 권한을 위임하고자 할 수 있습니다.

범위가 지정된 토큰은 지정된 사용자로 확인되지만 범위에 따라 특정 작업으로 제한되는 토큰에 해당합니다. **dedicated-admin** 역할의 사용자만 범위가 지정된 토큰을 생성할 수 있습니다.

범위는 토큰의 범위 집합을 **PolicyRules** 집합으로 변환하여 평가합니다. 그러면 요청이 해당 규칙과 대조됩니다. 추가적인 권한 확인을 위해 "일반"권한 부여자에게 전달하려면 요청 속성이 하나 이상의 범위 규칙과 일치해야 합니다.

10.1.1. 사용자 범위

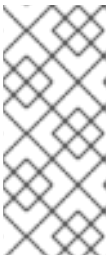
사용자 범위는 지정된 사용자에 대한 정보를 얻는 데 중점을 둡니다. 의도 기반이므로 다음과 같이 규칙이 자동으로 생성됩니다.

- **user:full** - 모든 사용자 권한에 API에 대한 전체 읽기/쓰기 액세스를 허용합니다.
- **user:info** - 사용자 정보(예: 이름, 그룹)에 대한 읽기 전용 액세스를 허용합니다.
- **user:check-access** - **self-localsubjectaccessreviews** 및 **self-subjectaccessreviews**에 대한 액세스를 허용합니다. 해당 항목은 요청 오브젝트에서 빈 사용자 및 그룹을 전달하는 변수입니다.
- **user:list-projects** - 사용자가 액세스할 수 있는 프로젝트를 나열하도록 읽기 전용 액세스를 허용합니다.

10.1.2. 역할 범위

역할 범위를 사용하면 네임스페이스로 필터링되어 지정된 역할과 동일한 수준의 액세스 권한을 가질 수 있습니다.

- **role:<cluster-role name>:<namespace or * for all>** - 클러스터 역할에 따라 지정된 규칙으로 범위가 제한되지만, 지정된 네임스페이스에 한합니다.



참고

경고: 이 경우 액세스 권한이 에스컬레이션되지 않습니다. 시크릿, 역할 바인딩 및 역할과 같은 리소스에 액세스할 수 있는 역할이더라도 이 범위는 해당 리소스에 대한 액세스를 거부합니다. 따라서 예기치 않은 에스컬레이션을 방지할 수 있습니다. 대부분의 사람들은 **edit**와 같은 역할을 에스컬레이션되는 역할로 생각하지 않지만 시크릿에 액세스할 수 있는 경우 에스컬레이션됩니다.

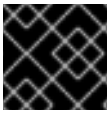
- **role:<cluster-role name>:<namespace or * for all>:!** - 느낌표를 넣어 이 범위에서 액세스 권한을 에스컬레이션할 수 있다는 점을 제외하면 위의 예와 유사합니다.

10.2. 클러스터 역할에 인증되지 않은 그룹 추가

클러스터 관리자는 클러스터 역할 바인딩을 생성하여 OpenShift Dedicated의 다음 클러스터 역할에 인증되지 않은 사용자를 추가할 수 있습니다. 인증되지 않은 사용자는 비공용 클러스터 역할에 액세스할 수 없습니다. 이 작업은 필요한 경우에만 특정 사용 사례에서 수행해야 합니다.

인증되지 않은 사용자를 다음 클러스터 역할에 추가할 수 있습니다.

- ***system:scope-impersonation***
- ***system:webhook***
- ***system:oauth-token-deleter***
- ***self-access-reviewer***



중요

인증되지 않은 액세스를 수정할 때 항상 조직의 보안 표준을 준수하는지 확인하십시오.

사전 요구 사항

- ***cluster-admin*** *역할의 사용자로 클러스터에 액세스할 수 있어야 합니다.*
- *OpenShift CLI(**oc**)가 설치되어 있습니다.*

프로세스

1. ***add-<cluster_role>-unauth.yaml*** *이라는 YAML 파일을 생성하고 다음 콘텐츠를 추가합니다.*

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
 annotations:
   rbac.authorization.kubernetes.io/autoupdate: "true"
 name: <cluster_role>access-unauthenticated
roleRef:
 apiGroup: rbac.authorization.k8s.io
 kind: ClusterRole
 name: <cluster_role>
subjects:
 - apiGroup: rbac.authorization.k8s.io
   kind: Group
   name: system:unauthenticated
```

2. *다음 명령을 실행하여 구성을 적용합니다.*

```
$ oc apply -f add-<cluster_role>.yaml
```

11장. 바인딩된 서비스 계정 토큰 사용

바인딩된 서비스 계정 토큰을 사용하면 AWS IAM 또는 Google Cloud Platform IAM의 OpenShift Dedicated와 같은 클라우드 공급자 IAM(Identity Access Management) 서비스와 통합할 수 있는 기능이 향상됩니다.

11.1. 바인딩된 서비스 계정 토큰 정보

바인딩된 서비스 계정 토큰을 사용하여 지정된 서비스 계정 토큰에 대한 권한 범위를 제한할 수 있습니다. 이 토큰에는 오디언스와 시간이 바인딩되어 있습니다. 이를 통해 IAM 역할에 대한 서비스 계정 인증 및 Pod에 마운트된 임시 인증 정보 생성이 용이해집니다. 볼륨 프로젝션 및 TokenRequest API를 사용하여 바인딩된 서비스 계정 토큰을 요청할 수 있습니다.

11.2. 볼륨 프로젝션을 사용한 바인딩된 서비스 계정 토큰 구성

볼륨 프로젝션을 통해 바인딩된 서비스 계정 토큰을 요청하도록 pod를 구성할 수 있습니다.

사전 요구 사항

- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- 서비스 계정을 생성했습니다. 이 절차에서는 서비스 계정의 이름이 **build-robot**이라고 가정합니다.

프로세스

1. 볼륨 프로젝션을 통해 바인딩된 서비스 계정 토큰을 사용하도록 pod를 구성합니다.
   a. 다음 콘텐츠를 사용하여 **pod-projected-svc-token.yaml**이라는 파일을 생성합니다.

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  securityContext:
    runAsNonRoot: true (1)
    seccompProfile:
      type: RuntimeDefault (2)
  containers:
  - image: nginx
    name: nginx
    volumeMounts:
    - mountPath: /var/run/secrets/tokens
      name: vault-token
    securityContext:
      allowPrivilegeEscalation: false
      capabilities:
        drop: [ALL]
  serviceAccountName: build-robot (3)
  volumes:
  - name: vault-token
    projected:
      sources:
```

```
- serviceAccountToken:
    path: vault-token 4
    expirationSeconds: 7200 5
    audience: vault 6
```

1. 컨테이너가 root로 실행되도록 하여 위험 저하를 최소화합니다.
2. 위험을 줄이기 위해 필수 시스템 호출으로 제한되는 기본 seccomp 프로필을 설정합니다.
3. 기존 서비스 계정에 대한 참조입니다.
4. 토큰을 프로젝션할 파일의 마운트 지점을 기준으로 하는 경로입니다.
5. 필요한 경우 서비스 계정 토큰 만료 시간을 초 단위로 설정합니다. 기본값은 3600초(1시간)이며 이 값은 600초(10분) 이상이어야 합니다. 토큰이 수명의 80% 이상을 경과했거나 24시간 이상된 경우 kubelet에서 토큰을 순환하기 시작합니다.
6. 필요한 경우 의도된 토큰 오디언스를 설정합니다. 토큰 수신자는 수신자 ID가 토큰의 오디언스 클레임과 일치하는지 확인하고 일치하지 않는 경우 토큰을 거부해야 합니다. 오디언스는 기본적으로 API 서버의 식별자입니다.

참고

예기치 않은 실패를 방지하기 위해 OpenShift Dedicated는 **--service-account-extend-token-expiration** 기본값이 **true** 인 초기 토큰 생성에서 1년으로 **expirationSeconds** 값을 덮어씁니다. 이 설정은 변경할 수 없습니다.

b. Pod를 생성합니다.

```
$ oc create -f pod-projected-svc-token.yaml
```

kubelet은 pod를 대신하여 토큰을 요청 및 저장하고, 구성 가능한 파일 경로에 있는 pod에서 토큰을 사용할 수 있도록 설정하고, 토큰이 만료되면 토큰을 갱신합니다.

2. 바인딩된 토큰을 사용하는 애플리케이션에서는 토큰이 회전할 때 다시 로드하는 작업을 처리해야 합니다.
kubelet은 토큰이 수명의 80% 이상을 경과했거나 24시간 이상된 경우, 토큰을 회전합니다.

11.3. POD 외부에서 바인딩된 서비스 계정 토큰 생성

사전 요구 사항

- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- 서비스 계정을 생성했습니다. 이 절차에서는 서비스 계정의 이름이 **build-robot**이라고 가정합니다.

프로세스

- 다음 명령을 실행하여 Pod 외부에서 바인딩된 서비스 계정 토큰을 생성합니다.

```
$ oc create token build-robot
```

-

출력 예

```
eyJhbGciOiJSUzI1NiIsImtpZCI6IkY2M1N4MHRvc2xFNnFSQlA4eG9GYzVPdnN3NkhIV0tRW
mFrUDRNcWx4S0kifQ.eyJhdWQiOlsiaHR0cHM6Ly9pc3N1ZXIyLnRlc3QuY29tIiwiaHR0cHM6L
y9pc3N1ZXIxLnRlc3QuY29tIiwiaHR0cHM6Ly9rdWJlcm5ldGVzLmRlZmF1bHQuc3ZjIl0sImV4c
CI6MTY3OTU0MzgzMCwiaWF0IjoxNjc5NTQwMjMwLCJpc3MiOiJodHRwczovL2lzc3VlcjIudGV
zdC5jb20iLCJrdWJlcm5ldGVzLmlvIjp7Im5hbWVzcGFjZSI6ImRlZmF1bHQiLCJzZXJ2aWNlYW
Njb3VudCI6eyJuYW1lIjoidGVzdC1zYSIsInVpZCI6ImM3ZjA4MjkwLWIzOTUtNGM4NC04NjI4L
TMzMTM1NTVhNWY1OSJ9fSwibmJmIjoxNjc5NTQwMjMwLCJzdWIiOiJzeXN0ZW06c2Vydmlj
ZWFjY291bnQ6ZGVmYXVsdDp0ZXN0LXNhIn0.WyAOPvh1BFMUl3LNhBCrQeaB5wSynbnCf
ojWuNNPSilT4YvFnKibxwREwmzHpV4LO1xOFZHSi6bXBOmG_o-
m0XNDYL3FrGHd65mymiFyluztxa2lgHVxjw5reIV5ZLgNSol3Y8bJqQqmNg3rtQQWRML2kpJB
XdDHNww0E5XOypmffYkfkadli8lN5QQD-
MhsCbiAF8waCYs8bj6V6Y7uUKTcxee8sCjiRMVtXKjQtooERKm-
CH_p57wxCljIBeM89VdaR51NJGued4hVV5lxvVrYZFu89lBEAq4oyQN_d6N1vBWGXQMyoihn
t_fQjn-NfnlJWk-3NSZDIluDJAv7e-MTEk3geDrHVQKNEzDei2-Un64hSzb-
n1g1M0Vn0885wQBQAePC9UlZm8YZlMNk1tq6wIUKQTMv3HPfi5HtBRqVc2eVs0EfMX4-x-
PHhPCasJ6qLJWyj6DvyQ08dP4DW_TWZVGvKlmId0hzwpg59TTcLR0iCklSEJgAVEEd13Aa_
M0-
faD11L3MhUGxw0qxgOsPczdXUsolSISbefs7OKymzFSIkTAn9sDQ8PHMOsuyxsK8vzfrR-
E0z7MAeguZ2kaIY7cZqbN6WFy0caWgx46hrKem9vCKALefElRYbCg3hcBmowBcRTOqaFHL
NnHghhU1LaRpoFzH7OUarqX9SGQ
```

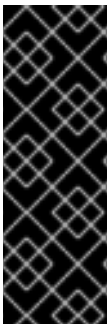
추가 리소스

- *서비스 계정 생성*

12장. 보안 컨텍스트 제약 조건 관리

OpenShift Dedicated에서는 SCC(보안 컨텍스트 제약 조건)를 사용하여 클러스터의 Pod에 대한 권한을 제어할 수 있습니다.

기본 SCC는 설치 중에 그리고 일부 Operator 또는 기타 구성 요소를 설치할 때 생성됩니다. 클러스터 관리자는 OpenShift CLI(**oc**)를 사용하여 고유한 SCC를 생성할 수도 있습니다.



중요

기본 SCC를 수정하지 마십시오. 기본 SCC를 사용자 정의하면 일부 플랫폼 Pod 배포 또는 OpenShift Dedicated가 업그레이드되면 문제가 발생할 수 있습니다. 또한 일부 클러스터 업그레이드 중에 기본 SCC 값이 기본값으로 재설정되므로 해당 SCC에 대한 모든 사용자 정의를 삭제합니다.

기본 SCC를 수정하는 대신 필요에 따라 자체 SCC를 생성하고 수정합니다. 자세한 단계는 보안 컨텍스트 제약 조건 생성 을 참조하십시오.

참고

OpenShift Dedicated 배포에서는 CCO(Customer Cloud Subscription) 모델을 사용하는 클러스터에 대해서만 고유한 SCC를 생성할 수 있습니다. SCC 리소스 생성에는 **cluster-admin** 권한이 필요하므로 Red Hat 클라우드 계정을 사용하는 OpenShift Dedicated 클러스터에 대한 SCC를 생성할 수 없습니다.

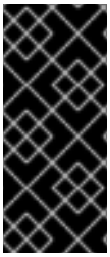
12.1. 보안 컨텍스트 제약 조건 정보

RBAC 리소스에서 사용자 액세스를 제어하는 방식과 유사하게 관리자는 SCC(보안 컨텍스트 제약 조건)를 사용하여 Pod에 대한 권한을 제어할 수 있습니다. 이러한 권한은 Pod에서 수행할 수 있는 작업과 액세스할 수 있는 리소스를 결정합니다. Pod가 시스템에 수용되려면 일련의 조건을 함께 실행해야 하는데, SCC를 사용하여 이러한 조건을 정의할 수 있습니다.

시크릿 컨텍스트 제약 조건을 통해 관리자는 다음을 제어할 수 있습니다.

- Pod에서 **allowPrivilegedContainer** 플래그를 사용하여 권한 있는 컨테이너를 실행할 수 있는지 여부
- Pod가 **allowPrivilegeEscalation** 플래그로 제한되는지 여부
- 컨테이너에서 요청할 수 있는 기능
- 호스트 디렉터리를 볼륨으로 사용
- 컨테이너의 SELinux 컨텍스트
- 컨테이너 사용자 ID
- 호스트 네임스페이스 및 네트워킹 사용
- Pod 볼륨을 보유한 **FSGroup**의 할당
- 허용되는 추가 그룹의 구성
- 컨테이너에 루트 파일 시스템에 대한 쓰기 액세스 권한이 필요한지 여부

- *볼륨 유형 사용*
- *허용 가능한 **seccomp** 프로필 구성*

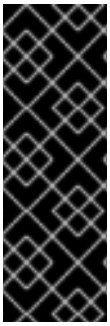


중요

*OpenShift Dedicated의 네임스페이스에 **openshift.io/run-level** 레이블을 설정하지 마십시오. 이 레이블은 내부 OpenShift Dedicated 구성 요소에서 Kubernetes API 서버 및 OpenShift API 서버와 같은 주요 API 그룹의 시작을 관리하는 데 사용됩니다. **openshift.io/run-level** 레이블이 설정된 경우 해당 네임스페이스의 Pod에 SCC가 적용되지 않으므로 해당 네임스페이스에서 실행되는 모든 워크로드에 높은 권한이 부여됩니다.*

12.1.1. 기본 보안 컨텍스트 제약 조건

아래 표에 설명된 대로 클러스터에는 몇 가지 기본 SCC(보안 컨텍스트 제약 조건)가 포함되어 있습니다. Operator 또는 기타 구성 요소를 OpenShift Dedicated에 설치할 때 추가 SCC가 설치될 수 있습니다.



중요

기본 SCC를 수정하지 마십시오. 기본 SCC를 사용자 정의하면 일부 플랫폼 Pod 배포 또는 OpenShift Dedicated가 업그레이드되면 문제가 발생할 수 있습니다. 또한 일부 클러스터 업그레이드 중에 기본 SCC 값이 기본값으로 재설정되므로 해당 SCC에 대한 모든 사용자 정의를 삭제합니다.

기본 SCC를 수정하는 대신 필요에 따라 자체 SCC를 생성하고 수정합니다. 자세한 단계는 보안 컨텍스트 제약 조건 생성 을 참조하십시오.

표 12.1. 기본 보안 컨텍스트 제약 조건

보안 컨텍스트 제약 조건	설명
anyuid	**restricted** SCC의 모든 기능을 제공하지만 사용자는 모든 UID 및 GID로 실행할 수 있습니다.
nonroot	**restricted** SCC의 모든 기능을 제공하지만 사용자는 루트 이외의 UID로 실행할 수 있습니다. 사용자는 UID를 지정하거나 컨테이너 런타임의 매니페스트에 지정해야 합니다.
nonroot-v2	**nonroot** SCC와 유사하지만 다음과 같은 차이점이 있습니다. • **ALL** 기능은 컨테이너에서 삭제됩니다. • **NET_BIND_SERVICE** 기능은 명시적으로 추가할 수 있습니다. • **seccompProfile** 은 기본적으로 **runtime/default** 로 설정됩니다. • 보안 컨텍스트에서 **allowPrivilegeEscalation**을 설정하지 않거나 **false**로 설정해야 합니다.

보안 컨텍스트 제약 조건	설명
restricted	모든 호스트 기능에 대한 액세스를 거부하고 네임스페이스에 할당된 UID 및 SELinux 컨텍스트로 Pod를 실행해야 합니다. **제한된** SCC의 경우 다음을 수행합니다. ● Pod가 권한에 따라 실행되지 않도록 합니다. ● Pod에서 호스트 디렉터리 볼륨을 마운트할 수 없는지 확인합니다. ● 미리 할당된 UID 범위에서 Pod를 사용자로 실행해야 합니다. ● Pod를 사전 할당된 MCS 라벨로 실행해야 합니다. ● Pod를 사전 할당된 FSGroup으로 실행해야 합니다. ● Pod에서 추가 그룹을 사용하도록 허용 OpenShift Dedicated 4.10 또는 이전 버전에서 업그레이드된 클러스터에서는 인증된 사용자가 이 SCC를 사용할 수 있습니다. 액세스 권한이 명시적으로 부여되지 않는 한 새로운 OpenShift Dedicated 4.11 이상 설치 사용자는 **제한된** SCC를 더 이상 사용할 수 없습니다.
restricted-v2	**restricted** SCC와 유사하지만 다음과 같은 차이점이 있습니다. ● **ALL** 기능은 컨테이너에서 삭제됩니다. ● **NET_BIND_SERVICE** 기능은 명시적으로 추가할 수 있습니다. ● **seccompProfile** 은 기본적으로 **runtime/default** 로 설정됩니다. ● 보안 컨텍스트에서 **allowPrivilegeEscalation**을 설정하지 않거나 **false**로 설정해야 합니다. 이는 새 설치에서 제공하는 가장 제한적인 SCC이며 인증된 사용자에게 기본적으로 사용됩니다.  **참고** **restricted-v2** SCC는 기본적으로 시스템에 포함된 SCC의 가장 제한적인 SCC입니다. 그러나 더 제한적인 사용자 지정 SCC를 생성할 수 있습니다. 예를 들어 **readOnlyRootFilesystem** 을 **true** 로 제한하는 SCC를 생성할 수 있습니다.

12.1.2. 보안 컨텍스트 제약 조건 설정

SCC(보안 컨텍스트 제약 조건)는pod에서 액세스할 수 있는 보안 기능을 제어하는 설정 및 전략으로 구성됩니다. 이 설정은 세 가지 범주로 분류됩니다.

카테고리	설명
부울로 제어	이 유형의 필드는 기본적으로 가장 제한적인 값으로 설정됩니다. 예를 들어, **AllowPrivilegedContainer**는 값이 지정되지 않은 경우 항상**false**로 설정됩니다.

카테고리	설명
허용 가능한 설정으로 제어	이 유형의 필드는 해당 값이 허용되는지 확인하기 위해 설정과 대조됩니다.
전략으로 제어	가치를 생성하는 전략이 있는 항목에서는 다음을 제공합니다. ● 가치를 생성하는 메커니즘 ● 지정된 값이 허용된 값 집합에 속하도록 하는 메커니즘

CRI-O에는Pod의 각 컨테이너에 허용되는 다음 기본 기능 목록이 있습니다.

- ***CHOWN***
- ***DAC_OVERRIDE***
- ***FSETID***
- ***FOWNER***
- ***SETGID***
- ***SETUID***
- ***SETPCAP***
- ***NET_BIND_SERVICE***
- ***KILL***

컨테이너는 이 기본 목록의 기능을 사용하지만Pod 매니페스트 작성자는 추가 기능을 요청하거나 일부 기본 동작을 제거하여 목록을 변경할 수 있습니다. ***allowedCapabilities***,***defaultAddCapabilities***, ***requiredDropCapabilities*** *매개변수를 사용하여Pod에서 이러한 요청을 제어합니다. 이러한 매개변수를 사용하면 요청할 수 있는 기능, 각 컨테이너에 추가해야 하는 기능 및 각 컨테이너에서 금지 또는 삭제해야 하는 기능을 지정할 수 있습니다.*

참고

requiredDropCapabilities *매개변수를* ***ALL*** *로 설정하여 컨테이너에서 모든 기능을 삭제할 수 있습니다.* ***restricted-v2*** *SCC에서 수행하는 작업입니다.*

12.1.3. 보안 컨텍스트 제약 조건 전략

RunAsUser

- ***MustRunAs*** *-* ***runAsUser****를 구성해야 합니다. 구성된* ***runAsUser****를 기본값으로 사용합니다. 구성된* ***runAsUser****에 대해 검증합니다.*

  MustRunAs 스니펫 예

  ```
  ...
  ```

```
runAsUser:
  type: MustRunAs
  uid: <id>
...
```

- ***MustRunAsRange*** - *사전 할당된 값을 사용하지 않는 경우 최솟값과 최댓값을 정의해야 합니다. 최솟값을 기본값으로 사용합니다. 전체 허용 범위에 대해 검증합니다.*

  MustRunAsRange 스니펫 예

```
...
runAsUser:
  type: MustRunAsRange
  uidRangeMax: <maxvalue>
  uidRangeMin: <minvalue>
...
```

- ***MustRunAsNonRoot*** - *Pod를 0이 아닌 **runAsUser**를 사용하여 제출하거나 Pod의 이미지에 **USER** 지시문이 정의되어 있어야 합니다. 기본값이 제공되지 않습니다.*

  MustRunAsNonRoot 스니펫 예

```
...
runAsUser:
  type: MustRunAsNonRoot
...
```

- ***RunAsAny*** - *기본값이 제공되지 않습니다. 모든 **runAsUser**를 지정할 수 있습니다.*

  RunAsAny 스니펫 예

```
...
runAsUser:
  type: RunAsAny
...
```

SELinuxContext

- ***MustRunAs*** - *사전 할당된 값을 사용하지 않는 경우 **seLinuxOptions**를 구성해야 합니다. **seLinuxOptions**를 기본값으로 사용합니다. **seLinuxOptions**에 대해 검증합니다.*
- ***RunAsAny*** - *기본값이 제공되지 않습니다. 모든 **seLinuxOptions**를 지정할 수 있습니다.*

SupplementalGroups

- ***MustRunAs*** - *사전 할당된 값을 사용하지 않는 경우 범위를 하나 이상 지정해야 합니다. 첫 번째 범위의 최솟값을 기본값으로 사용합니다. 모든 범위에 대해 검증합니다.*
- ***RunAsAny*** - *기본값이 제공되지 않습니다. 임의의 **supplementalGroups**을 지정할 수 있습니다.*

FSGroup

- ***MustRunAs*** - 사전 할당된 값을 사용하지 않는 경우 범위를 하나 이상 지정해야 합니다. 첫 번째 범위의 최솟값을 기본값으로 사용합니다. 첫 번째 범위의 첫 번째 ID에 대해 검증합니다.
- ***RunAsAny*** - 기본값이 제공되지 않습니다. ***fsGroup*** ID를 지정할 수 있습니다.

12.1.4. CCS 클러스터의 볼륨 제어

SCC의 ***volumes*** 필드를 설정하여 CCO(Customer Cloud Subscription) 클러스터를 사용하여 OpenShift Dedicated에 대한 특정 볼륨 유형의 사용을 제어할 수 있습니다.

이 필드에 허용되는 값은 볼륨 생성 시 정의되는 볼륨 소스에 해당합니다.

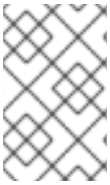
- ***awsElasticBlockStore***
- ***azureDisk***
- ***azureFile***
- ***cephFS***
- ***cinder***
- ***configMap***
- ***csi***
- ***downwardAPI***
- ***emptyDir***
- ***fc***
- ***flexVolume***
- ***flocker***
- ***gcePersistentDisk***
- ***ephemeral***
- ***gitRepo***
- ***glusterfs***
- ***hostPath***
- ***iscsi***
- ***nfs***
- ***persistentVolumeClaim***
- ***photonPersistentDisk***
- ***portworxVolume***
- ***projected***

- ***quobyte***
- ***rbd***
- ***scaleIO***
- ***secret***
- ***storageos***
- ***vsphereVolume***
- ***** (모든 볼륨 유형의 사용을 허용하는 특수 값)
- ***none*** (모든 볼륨 유형의 사용을 허용하지 않는 특수 값입니다. 이전 버전과의 호환성을 위해서만 존재합니다.)

새 SCC에 허용되는 볼륨의 최소 권장 집합은 ***configMap***, ***downwardAPI***, ***emptyDir***, ***persistentVolumeClaim***, ***secret***, ***projected*** 입니다.

참고

OpenShift Dedicated의 각 릴리스에 새 유형이 추가되므로 허용되는 볼륨 유형 목록은 포괄적이지 않습니다.



참고

이전 버전과의 호환성을 위해 ***allowHostDirVolumePlugin***을 사용하면 ***volumes*** 필드의 설정을 덮어씁니다. 예를 들어, ***allowHostDirVolumePlugin***이 false로 설정되어 있지만 ***volumes*** 필드에서 허용되는 경우 ***volumes***에서 ***hostPath*** 값이 제거됩니다.

12.1.5. 승인 제어

SCC를 통한 허용 제어를 사용하면 사용자에게 부여된 기능을 기반으로 리소스 생성을 제어할 수 있습니다.

SCC 측면에서는 허용 컨트롤러가 컨텍스트에서 사용 가능한 사용자 정보를 검사하여 적절한 SCC 집합을 검색할 수 있음을 의미합니다. 이 경우 Pod에 운영 환경에 대한 요청을 하거나 Pod에 적용할 일련의 제약 조건을 생성할 수 있는 권한이 부여됩니다.

허용 작업에서 Pod를 승인하는 데 사용하는 SCC 집합은 사용자 ID 및 사용자가 속하는 그룹에 따라 결정됩니다. 또한 Pod에서 서비스 계정을 지정하는 경우, 허용된 SCC 집합에 서비스 계정에 액세스할 수 있는 모든 제약 조건이 포함됩니다.

참고

배포와 같은 워크로드 리소스를 생성할 때 서비스 계정만 SCC를 찾고 Pod가 생성될 때 Pod를 허용하는 데 사용됩니다.

허용 작업에서는 다음 방법을 사용하여 Pod에 대한 최종 보안 컨텍스트를 생성합니다.

1. 사용 가능한 모든 SCC를 검색합니다.
2. 요청에 지정되지 않은 보안 컨텍스트 설정에 대한 필드 값을 생성합니다.

3. *사용 가능한 제약 조건에 대해 최종 설정을 검증합니다.*

일치하는 제약 조건 집합이 있는 경우 Pod가 승인됩니다. 요청을 SCC와 일치시킬 수 없는 경우 Pod가 거부됩니다.

Pod는 SCC에 대해 모든 필드를 검증해야 합니다. 다음은 검증이 필요한 필드 중 단2개의 예입니다.

참고

이러한 예제는 사전 할당된 값을 사용하는 전략 컨텍스트에 있습니다.

MustRunAs의 FSGroup SCC 전략

*Pod에서 **fsGroup** ID를 정의하는 경우 해당 ID는 기본 **fsGroup** ID와 같아야 합니다. 그렇지 않으면 해당 SCC에서 Pod를 검증하지 않고 다음 SCC를 평가합니다.*

***SecurityContextConstraints.fsGroup** 필드에 값 **RunAsAny**가 있고 Pod 사양에서 **Pod.spec.securityContext.fsGroup**을 생략하는 경우, 이 필드는 유효한 것으로 간주됩니다. 유효성을 확인하는 동안 다른 SCC 설정에서 다른 Pod 필드를 거부하여 Pod가 실패할 수 있습니다.*

MustRunAs의 SupplementalGroups SCC 전략

*Pod 사양에서 하나 이상의 **supplementalGroups** ID를 정의하는 경우, Pod의 ID는 네임스페이스의 **openshift.io/sa.scc.supplemental-groups** 주석에 있는 ID 중 하나와 같아야 합니다. 그렇지 않으면 해당 SCC에서 Pod를 검증하지 않고 다음 SCC를 평가합니다.*

***SecurityContextConstraints.supplementalGroups** 필드에 값 **RunAsAny**가 있고 Pod 사양에서 **Pod.spec.securityContext.supplementalGroups**를 생략하는 경우, 이 필드는 유효한 것으로 간주됩니다. 유효성을 확인하는 동안 다른 SCC 설정에서 다른 Pod 필드를 거부하여 Pod가 실패할 수 있습니다.*

12.1.6. 보안 컨텍스트 제약 조건 우선순위 지정

SCC(보안 컨텍스트 제약 조건)에는 승인 컨트롤러의 요청을 확인할 때 순서에 영향을 주는 우선순위 필드가 있습니다.

*우선순위 **0**은 가능한 가장 낮은 우선 순위입니다. nil 우선순위는 **0** 또는 가장 낮은 우선순위로 간주됩니다. 정렬 시 우선순위가 높은 SCC가 집합의 앞쪽으로 이동합니다.*

사용 가능한 전체 SCC 집합이 결정되면 SCC가 다음과 같은 방식으로 정렬됩니다.

1. *우선순위가 가장 높은 SCC가 먼저 정렬됩니다.*
2. *우선순위가 동일한 경우 SCC는 가장 제한적인 것에서 가장 제한적이지 않은 것으로 정렬됩니다.*
3. *우선순위와 제한이 모두 동일한 경우 SCC는 이름별로 정렬됩니다.*

*기본적으로 클러스터 관리자에게 부여되는 **anyuid** SCC는 SCC 집합에서 우선순위가 부여됩니다. 이를 통해 클러스터 관리자는 Pod의 **SecurityContext**에 **RunAsUser**를 지정하여 모든 사용자로 Pod를 실행할 수 있습니다.*

12.2. 미리 할당된 보안 컨텍스트 제약 조건 값 정보

허용 컨트롤러는 SCC(보안 컨텍스트 제약 조건)의 특정 조건을 인식하여 네임스페이스에서 미리 할당된 값을 조회하고 pod를 처리하기 전에 SCC를 채울 수 있습니다. 각 SCC 전략은 실행 중인 pod에 정의된 다양한 ID에 대한 최종 값을 만들기 위해 pod 사양 값으로 집계된 각 정책에 대해, 허용된 경우 사전 할당된

값을 사용하여 다른 전략과 독립적으로 평가됩니다.

다음 SCC를 사용하면 Pod 사양에 범위가 정의되지 않은 경우 허용 컨트롤러에서 사전 할당된 값을 찾습니다.

1. *최소 또는 최대 집합이 없는* ***MustRunAsRange****의* ***RunAsUser*** *전략. 허용 작업에서는 범위 필드를 채우기 위해* ***openshift.io/sa.scc.uid-range*** *주석을 찾습니다.*

2. *수준이 설정되지 않은* ***MustRunAs****의* ***SELinuxContext*** *전략. 허용 작업에서는 수준을 채울* ***openshift.io/sa.scc.mcs*** *주석을 찾습니다.*

3. ***MustRunAs****의* ***FSGroup*** *전략. 허용 작업에서는* ***openshift.io/sa.scc.supplemental-group*** *주석을 찾습니다.*

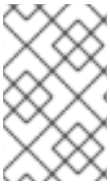
4. ***MustRunAs****의* ***SupplementalGroups*** *전략. 허용 작업에서는* ***openshift.io/sa.scc.supplemental-group*** *주석을 찾습니다.*

생성 단계 중 보안 컨텍스트 공급자는 Pod에서 구체적으로 설정되지 않은 모든 매개변수 값으로 기본값을 사용합니다. 기본값은 선택한 전략에 따라 다릅니다.

1. ***RunAsAny*** *및* ***MustRunAsNonRoot*** *전략에서는 기본값을 제공하지 않습니다. Pod에 그룹 ID와 같은 매개변수 값이 필요한 경우, Pod 사양에 값을 정의해야 합니다.*

2. ***MustRunAs*** *(단일 값) 전략에서는 항상 사용되는 기본값을 제공합니다. 예를 들어, 그룹 ID의 경우 Pod 사양에서 고유한 ID 값을 정의하더라도 네임스페이스의 기본 매개변수 값도 Pod의 그룹에 나타납니다.*

3. ***MustRunAsRange*** *및* ***MustRunAs*** *(범위 기반) 전략에서는 최소 범위 값을 제공합니다. 단일 값* ***MustRunAs*** *전략과 마찬가지로 네임스페이스의 기본 매개 변수 값이 실행 중인 Pod에 나타납니다. 범위 기반 전략을 여러 범위로 구성할 수 있는 경우, 처음 구성된 최소 범위 값을 제공합니다.*



참고

openshift.io/sa.scc.supplemental-groups *주석이 네임스페이스에 존재하지 않는 경우,* ***FSGroup*** *및* ***SupplementalGroups*** *전략이* ***openshift.io/sa.scc.uid-range*** *주석으로 변경됩니다. 둘 다 존재하지 않으면 SCC가 생성되지 않습니다.*

참고

기본적으로 주석 기반 ***FSGroup*** *전략은 주석의 최솟값에 따라 단일 범위로 자체 구성됩니다. 예를 들어, 주석이* ***1/3****이면* ***FSGroup*** *전략은 최솟값이자 최댓값인* ***1****로 구성됩니다.* ***FSGroup*** *필드에 더 많은 그룹을 허용하려면 주석을 사용하지 않는 사용자 정의 SCC를 구성하면 됩니다.*

참고

openshift.io/sa.scc.supplemental-groups *주석에는* ***<start>/<length*** *또는* ***<start>-<end>*** *형식의 쉼표로 구분된 블록 목록을 사용할 수 있습니다.* ***openshift.io/sa.scc.uid-range*** *주석에는 단일 블록만 사용할 수 있습니다.*

12.3. 보안 컨텍스트 제약 조건의 예

다음 예에서는 SCC(보안 컨텍스트 제약 조건)의 형식 및 주석을 보여줍니다.

주석이 달린 privileged SCC

```
allowHostDirVolumePlugin: true
allowHostIPC: true
allowHostNetwork: true
allowHostPID: true
allowHostPorts: true
allowPrivilegedContainer: true
allowedCapabilities: (1)
- '*'
apiVersion: security.openshift.io/v1
defaultAddCapabilities: [] (2)
fsGroup: (3)
  type: RunAsAny
groups: (4)
- system:cluster-admins
- system:nodes
kind: SecurityContextConstraints
metadata:
  annotations:
    kubernetes.io/description: 'privileged allows access to all privileged and host
      features and the ability to run as any user, any group, any fsGroup, and with
      any SELinux context.  WARNING: this is the most relaxed SCC and should be used
      only for cluster administration. Grant with caution.'
  creationTimestamp: null
  name: privileged
priority: null
readOnlyRootFilesystem: false
requiredDropCapabilities: (5)
- KILL
- MKNOD
- SETUID
- SETGID
runAsUser: (6)
  type: RunAsAny
seLinuxContext: (7)
  type: RunAsAny
seccompProfiles:
- '*'
supplementalGroups: (8)
  type: RunAsAny
users: (9)
- system:serviceaccount:default:registry
- system:serviceaccount:default:router
- system:serviceaccount:openshift-infra:build-controller
volumes: (10)
- '*'
```

(1) *pod에서 요청할 수 있는 기능 목록입니다. 빈 목록은 기능을 요청할 수 없음을 나타내고, 특수 기호 *는 모든 기능을 요청할 수 있음을 나타냅니다.*

(2) *임의의 pod에 추가된 추가 기능 목록입니다.*

(3) *보안 컨텍스트에 허용되는 값을 지시하는 **FSGroup** 전략입니다.*

(4) *이 SCC에 액세스할 수 있는 그룹입니다.*

5 *Pod에서 삭제할 기능 목록입니다. 또는 **ALL** 을 지정하여 모든 기능을 삭제합니다.*

6 *보안 컨텍스트에 허용되는 값을 지시하는 **runAsUser** 전략 유형입니다.*

7 *보안 컨텍스트에 허용되는 값을 지시하는 **seLinuxContext** 전략 유형입니다.*

8 ***supplementalGroups** 전략: 보안 컨텍스트에 허용되는 추가 그룹을 지시합니다.*

9 *이 SCC에 액세스할 수 있는 사용자입니다.*

10 *보안 컨텍스트에 허용되는 볼륨 유형입니다. 예에서는 * 를 사용하면 모든 볼륨 유형을 사용할 수 있습니다.*

*SCC의 **users** 및 **groups** 필드는 SCC에 액세스할 수 있는 사용자를 제어합니다. 기본적으로 클러스터 관리자, 노드 및 빌드 컨트롤러에는 권한 있는 SCC에 대한 액세스 권한이 부여됩니다. 인증된 모든 사용자에게는 **restricted-v2** SCC에 대한 액세스 권한이 부여됩니다.*

명시적인 runAsUser 설정이 없는 경우

```
apiVersion: v1
kind: Pod
metadata:
  name: security-context-demo
spec:
  securityContext: 1
  containers:
  - name: sec-ctx-demo
    image: gcr.io/google-samples/node-hello:1.0
```

1 *컨테이너 또는 pod에서 실행해야 하는 사용자 ID를 요청하지 않는 경우, 유효 UID는 이 pod를 내보내는 SCC에 따라 다릅니다. **restricted-v2** SCC는 기본적으로 인증된 모든 사용자에게 부여되므로 모든 사용자 및 서비스 계정에서 사용할 수 있으며 대부분의 경우 사용됩니다. **restricted-v2** SCC는 **securityContext.runAsUser** 필드의 가능한 값을 제한하고 기본값을 설정하는 데 **MustRunAsRange** 전략을 사용합니다. 허용 플러그인은 이 범위를 제공하지 않기 때문에 현재 프로젝트에서 **openshift.io/sa.scc.uid-range** 주석을 찾아 범위 필드를 채웁니다. 결국 컨테이너에는 모든 프로젝트의 범위가 다르기 때문에 예측하기 어려운 범위의 첫 번째 값과 동일한 **runAsUser**가 있게 됩니다.*

명시적인 runAsUser 설정

```
apiVersion: v1
kind: Pod
metadata:
  name: security-context-demo
spec:
  securityContext:
    runAsUser: 1000 1
  containers:
    - name: sec-ctx-demo
      image: gcr.io/google-samples/node-hello:1.0
```

1 특정 사용자 ID를 요청하는 컨테이너 또는 포드는 서비스 계정 또는 사용자에게 이러한 사용자 ID를 허용하는 SCC에 대한 액세스 권한이 부여된 경우에만 OpenShift Dedicated에서 승인됩니다. SCC를 사용하면 임의의 ID, 특정 범위에 속하는 ID 또는 요청과 관련된 정확한 사용자 ID를 허용할 수 있습니다.

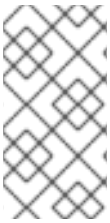
이 구성은 SELinux, fsGroup 및 추가 그룹에 유효합니다.

12.4. CCS 클러스터에 대한 보안 컨텍스트 제약 조건 생성

기본 SCC(보안 컨텍스트 제약 조건)가 애플리케이션 워크로드 요구 사항을 충족하지 않는 경우 OpenShift CLI(**oc**)를 사용하여 사용자 정의 SCC를 생성할 수 있습니다.

중요

자체 SCC를 생성하고 수정하는 것은 클러스터에 불안정할 수 있는 고급 작업입니다. 자체 SCC 사용에 대한 질문이 있는 경우 Red Hat 지원에 문의하십시오. Red Hat 지원에 문의하는 방법에 대한 자세한 내용은 지원 요청을 참조하십시오.



참고

OpenShift Dedicated 배포에서는 CCO(Customer Cloud Subscription) 모델을 사용하는 클러스터에 대해서만 고유한 SCC를 생성할 수 있습니다. SCC 리소스 생성에는 **cluster-admin** 권한이 필요하므로 Red Hat 클라우드 계정을 사용하는 OpenShift Dedicated 클러스터에 대한 SCC를 생성할 수 없습니다.

사전 요구 사항

- OpenShift CLI(**oc**)를 설치합니다.
- **cluster-admin** 역할의 사용자로 클러스터에 로그인합니다.

프로세스

1. **scc-admin.yaml** 이라는 YAML 파일에 SCC를 정의합니다.

```
kind: SecurityContextConstraints
apiVersion: security.openshift.io/v1
metadata:
  name: scc-admin
allowPrivilegedContainer: true
runAsUser:
  type: RunAsAny
seLinuxContext:
  type: RunAsAny
fsGroup:
  type: RunAsAny
supplementalGroups:
  type: RunAsAny
users:
- my-admin-user
groups:
- my-admin-group
```

필요한 경우 **requiredDropCapabilities** 필드를 원하는 값으로 설정하여 SCC의 특정 기능을 삭제

할 수 있습니다. 지정된 기능은 컨테이너에서 삭제됩니다. 모든 기능을 삭제하려면 **ALL** 을 지정합니다. 예를 들어 **KILL,MKNOD**, **SYS_CHROOT** 기능을 삭제하는 SCC를 생성하려면 SCC 오브젝트에 다음을 추가합니다.

```
requiredDropCapabilities:
- KILL
- MKNOD
- SYS_CHROOT
```

참고

allowedCapabilities 및 **requiredDropCapabilities** 모두에서 기능을 나열할 수 없습니다.

CRI-O는 Docker 문서 에 있는 동일한 기능 값 목록을 지원합니다.

2. 파일에서 전달하여 SCC 생성:

```
$ oc create -f scc-admin.yaml
```

출력 예

```
securitycontextconstraints "scc-admin" created
```

검증

- SCC가 생성되었는지 확인합니다.

```
$ oc get scc scc-admin
```

출력 예

```
NAME      PRIV    CAPS    SELINUX    RUNASUSER    FSGROUP    SUPGROUP    PRIORITY    READONLYROOTFS    VOLUMES
scc-admin    true    []    RunAsAny    RunAsAny    RunAsAny    RunAsAny    <none>    false    [awsElasticBlockStore azureDisk azureFile cephFS cinder configMap downwardAPI emptyDir fc flexVolume flocker gcePersistentDisk gitRepo glusterfs iscsi nfs persistentVolumeClaim photonPersistentDisk quobyte rbd secret vsphere]
```

12.5. 특정 SCC가 필요하도록 워크로드 구성

특정 SCC(보안 컨텍스트 제약 조건)가 필요하도록 워크로드를 구성할 수 있습니다. 이는 특정 SCC를 워크로드에 고정하거나 필요한 SCC가 클러스터의 다른 SCC에서 선점되지 않도록 하려는 경우에 유용합니다.

특정 SCC가 필요한 경우 워크로드에 **openshift.io/required-scc** 주석을 설정합니다. 배포 또는 데몬 세트와 같은 Pod 매니페스트 템플릿을 설정할 수 있는 모든 리소스에 이 주석을 설정할 수 있습니다.

SCC는 클러스터에 있어야 하며 워크로드에 적용해야 합니다. 그러지 않으면 Pod 허용이 실패합니다. Pod를 생성하는 사용자가 Pod를 생성하는 사용자에게 Pod의 네임스페이스의 SCC에 대한 **use** 권한이 있는 경우 SCC는 워크로드에 적용되는 것으로 간주됩니다.

주의

*실시간 Pod 매니페스트에서 **openshift.io/required-scc** 주석을 변경하지 마십시오. 이렇게 하면 Pod 승인이 실패합니다. 필요한 SCC를 변경하려면 기본 pod 템플릿의 주석을 업데이트하여 Pod를 삭제하고 다시 생성합니다.*

사전 요구 사항

- *SCC가 클러스터에 있어야 합니다.*

프로세스

1. *배포에 대한 YAML 파일을 생성하고 **openshift.io/required-scc** 주석을 설정하여 필요한 SCC를 지정합니다.*

   deployment.yaml의 예

   ```
   apiVersion: config.openshift.io/v1
   kind: Deployment
   apiVersion: apps/v1
   spec:
   # ...
     template:
       metadata:
         annotations:
           openshift.io/required-scc: "my-scc" 1
   # ...
   ```

   1 *사용할 SCC 이름을 지정합니다.*

2. *다음 명령을 실행하여 리소스를 생성합니다.*

   ```
   $ oc create -f deployment.yaml
   ```

검증

- *배포에서 지정된 SCC를 사용했는지 확인합니다.*

  a. *다음 명령을 실행하여 Pod의 **openshift.io/scc** 주석 값을 확인합니다.*

     ```
     $ oc get pod <pod_name> -o jsonpath='{.metadata.annotations.openshift\.io\/scc}{"\n"}' 1
     ```

     1 ***<pod_name>**을 배포 Pod 이름으로 바꿉니다.*

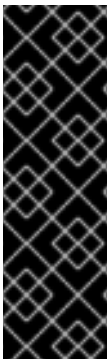
  b. *출력을 검사하고 표시된 SCC가 배포에 정의된 SCC와 일치하는지 확인합니다.*

     출력 예

```
my-scc
```

12.6. 보안 컨텍스트 제약 조건에 대한 역할 기반 액세스

SBA를 RBAC에서 처리하는 리소스로 지정할 수 있습니다. 그러면 SCC에 대한 액세스 권한의 범위를 특정 프로젝트 또는 전체 클러스터로 지정할 수 있습니다. SCC에 사용자, 그룹 또는 서비스 계정을 직접 할당하면 클러스터 전체 범위가 유지됩니다.



중요

기본 프로젝트에서 워크로드를 실행하거나 기본 프로젝트에 대한 액세스를 공유하지 마세요. 기본 프로젝트는 핵심 클러스터 구성 요소를 실행하기 위해 예약되어 있습니다.

다음 기본 프로젝트는 높은 권한이 있는 것으로 간주됩니다. **default**, **kube-public**, **kube-system**, **openshift**, **openshift-infra**, **openshift-node** 및 **openshift.io/run-level** 레이블이 **0** 또는 **1** 로 설정된 기타 시스템 생성 프로젝트입니다. Pod 보안 승인, 보안 컨텍스트 제약 조건, 클러스터 리소스 할당량 및 이미지 참조 확인과 같은 승인 플러그인에 의존하는 기능은 높은 권한 있는 프로젝트에서 작동하지 않습니다.

역할에 대한 SCC 액세스 권한을 포함하려면 역할을 생성할 때 **scc** 리소스를 지정하십시오.

```
$ oc create role <role-name> --verb=use --resource=scc --resource-name=<scc-name> -n
<namespace>
```

그러면 다음과 같은 역할 정의가 생성됩니다.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
...
  name: role-name 1
  namespace: namespace 2
...
rules:
- apiGroups:
  - security.openshift.io 3
  resourceNames:
  - scc-name 4
  resources:
  - securitycontextconstraints 5
  verbs: 6
  - use
```

1. 역할의 이름입니다.
2. 정의된 역할의 네임스페이스입니다. 지정하지 않는 경우 기본값은 **default** 입니다.
3. **SecurityContextConstraints** 리소스를 포함하는 API 그룹입니다. **scc** 가 리소스로 지정되면 자동으로 정의됩니다.
4. 액세스할 SCC 이름의 예입니다.
5. 사용자가 **resourceNames** 필드에 SCC 이름을 지정할 수 있는 리소스 그룹의 이름입니다.

6 역할에 적용할 동사 목록입니다.

이러한 규칙이 있는 로컬 또는 클러스터 역할을 통해 RoleBinding 또는 ClusterRoleBinding으로 바인딩된 주체는 **scc-name**이라는 사용자 정의 SCC를 사용할 수 있습니다.

참고

RBAC는 에스컬레이션되지 않도록 설계되었으므로 프로젝트 관리자도 SCC에 대한 액세스 권한을 부여할 수 없습니다. **restricted-v2** SCC를 포함하여 기본적으로 SCC 리소스에 동사 **use** 를 사용할 수 없습니다.

12.7. 보안 컨텍스트 제약 조건 명령 참조

OpenShift CLI(**oc**)를 사용하여 인스턴스의 SCC(보안 컨텍스트 제약 조건)를 일반 API 오브젝트로 관리할 수 있습니다.

12.7.1. 보안 컨텍스트 제약 조건 나열

현재 SCC 목록을 가져오려면 다음을 실행합니다.

```
$ oc get scc
```

출력 예

```
NAME                              PRIV    CAPS                    SELINUX     RUNASUSER          FSGROUP
SUPGROUP    PRIORITY     READONLYROOTFS   VOLUMES
anyuid                            false   <no value>              MustRunAs   RunAsAny           RunAsAny
RunAsAny    10           false
["configMap","downwardAPI","emptyDir","persistentVolumeClaim","projected","secret"]
hostaccess                        false   <no value>              MustRunAs   MustRunAsRange     MustRunAs
RunAsAny    <no value>   false
["configMap","downwardAPI","emptyDir","hostPath","persistentVolumeClaim","projected","secret"]
hostmount-anyuid                  false   <no value>              MustRunAs   RunAsAny           RunAsAny
RunAsAny    <no value>   false
["configMap","downwardAPI","emptyDir","hostPath","nfs","persistentVolumeClaim","projected","secret"]

hostnetwork                       false   <no value>              MustRunAs   MustRunAsRange     MustRunAs
MustRunAs   <no value>   false
["configMap","downwardAPI","emptyDir","persistentVolumeClaim","projected","secret"]
hostnetwork-v2                    false   ["NET_BIND_SERVICE"]    MustRunAs   MustRunAsRange
MustRunAs   MustRunAs    <no value>   false
["configMap","downwardAPI","emptyDir","persistentVolumeClaim","projected","secret"]
node-exporter                     true    <no value>              RunAsAny    RunAsAny           RunAsAny
RunAsAny    <no value>   false        ["*"]
nonroot                           false   <no value>              MustRunAs   MustRunAsNonRoot   RunAsAny
RunAsAny    <no value>   false
["configMap","downwardAPI","emptyDir","persistentVolumeClaim","projected","secret"]
nonroot-v2                        false   ["NET_BIND_SERVICE"]    MustRunAs   MustRunAsNonRoot
RunAsAny    RunAsAny     <no value>   false
["configMap","downwardAPI","emptyDir","persistentVolumeClaim","projected","secret"]
privileged                        true    ["*"]                   RunAsAny    RunAsAny           RunAsAny    RunAsAny
<no value>   false        ["*"]
restricted                        false   <no value>              MustRunAs   MustRunAsRange     MustRunAs
```

```
RunAsAny    <no value>   false
["configMap","downwardAPI","emptyDir","persistentVolumeClaim","projected","secret"]
restricted-v2                false   ["NET_BIND_SERVICE"]   MustRunAs   MustRunAsRange
MustRunAs   RunAsAny    <no value>   false
["configMap","downwardAPI","emptyDir","persistentVolumeClaim","projected","secret"]
```

12.7.2. 보안 컨텍스트 제약 조건 검사

SCC가 적용되는 사용자, 서비스 계정 및 그룹을 포함하여 특정 SCC에 대한 정보를 볼 수 있습니다.

예를 들어 **restricted** SCC를 검사하려면 다음을 실행합니다.

```
$ oc describe scc restricted
```

출력 예

```
Name:                           restricted
Priority:                       <none>
Access:
  Users:                        <none> (1)
  Groups:                       <none> (2)
Settings:
  Allow Privileged:             false
  Allow Privilege Escalation:       true
  Default Add Capabilities:         <none>
  Required Drop Capabilities:       KILL,MKNOD,SETUID,SETGID
  Allowed Capabilities:             <none>
  Allowed Seccomp Profiles:         <none>
  Allowed Volume Types:
configMap,downwardAPI,emptyDir,persistentVolumeClaim,projected,secret
  Allowed Flexvolumes:              <all>
  Allowed Unsafe Sysctls:           <none>
  Forbidden Sysctls:                <none>
  Allow Host Network:               false
  Allow Host Ports:                 false
  Allow Host PID:                   false
  Allow Host IPC:                   false
  Read Only Root Filesystem:        false
  Run As User Strategy: MustRunAsRange
    UID:                        <none>
    UID Range Min:                  <none>
    UID Range Max:                  <none>
  SELinux Context Strategy: MustRunAs
    User:                       <none>
    Role:                       <none>
    Type:                       <none>
    Level:                      <none>
  FSGroup Strategy: MustRunAs
    Ranges:                     <none>
  Supplemental Groups Strategy: RunAsAny
    Ranges:                     <none>
```

(1) SCC가 적용되는 사용자 및 서비스 계정을 나열합니다.

2 SCC가 적용되는 그룹을 나열합니다.

12.8. 추가 리소스

- 지원 요청

13장. POD 보안 허용 이해 및 관리

Pod 보안 승인은 Kubernetes Pod 보안 표준을 구현한 것입니다. Pod 보안 승인을 사용하여 Pod의 동작을 제한합니다.

13.1. POD 보안 승인 정보

OpenShift Dedicated에는 Kubernetes Pod 보안 승인이 포함되어 있습니다. 전역적으로 정의되거나 네임스페이스 수준에서 정의되지 않은 Pod 보안 승인은 클러스터에 허용되지 않으며 실행할 수 없습니다.

전역적으로 **privileged** 있는 프로필이 적용되며, **restricted** 프로필은 경고 및 감사에 사용됩니다.

네임스페이스 수준에서 Pod 보안 승인 설정을 구성할 수도 있습니다.

중요

기본 프로젝트에서 워크로드를 실행하거나 기본 프로젝트에 대한 액세스를 공유하지 마세요. 기본 프로젝트는 핵심 클러스터 구성 요소를 실행하기 위해 예약되어 있습니다.

다음 기본 프로젝트는 높은 권한이 있는 것으로 간주됩니다. **default**, **kube-public**, **kube-system**, **openshift**, **openshift-infra**, **openshift-node** 및 **openshift.io/run-level** 레이블이 **0** 또는 **1** 로 설정된 기타 시스템 생성 프로젝트입니다. Pod 보안 승인, 보안 컨텍스트 제약 조건, 클러스터 리소스 할당량 및 이미지 참조 확인과 같은 승인 플러그인에 의존하는 기능은 높은 권한 있는 프로젝트에서 작동하지 않습니다.

13.1.1. Pod 보안 승인 모드

네임스페이스에 대해 다음 Pod 보안 승인 모드를 구성할 수 있습니다.

표 13.1. Pod 보안 승인 모드

모드	레이블	설명
enforce	**pod-security.kubernetes.io/enforce**	설정된 프로필을 준수하지 않는 경우 허용에서 Pod를 거부합니다.
audit	**pod-security.kubernetes.io/audit**	Pod가 설정된 프로필을 준수하지 않는 경우 감사 이벤트 로그
warn	**pod-security.kubernetes.io/warn**	Pod가 설정된 프로필을 준수하지 않는 경우 경고 표시

13.1.2. Pod 보안 승인 프로필

각 Pod 보안 승인 모드를 다음 프로필 중 하나로 설정할 수 있습니다.

표 13.2. Pod 보안 승인 프로필

프로필	설명
privileged	최소 제한 정책; 알려진 권한 에스컬레이션 허용
baseline	최소한의 제한 정책; 알려진 권한 에스컬레이션을 방지
restricted	가장 제한적인 정책; 현재 Pod 강화 모범 사례를 따릅니다.

13.1.3. 권한이 있는 네임스페이스

다음 시스템 네임스페이스는 항상 **privileged** 있는 Pod 보안 승인 프로필로 설정됩니다.

- ***default***
- ***kube-public***
- ***kube-system***

이러한 권한 있는 네임스페이스의 Pod 보안 프로필을 변경할 수 없습니다.

13.1.4. Pod 보안 승인 및 보안 컨텍스트 제약 조건

Pod 보안 승인 표준 및 보안 컨텍스트 제약 조건은 두 개의 독립 컨트롤러에 의해 조정 및 적용됩니다. 두 컨트롤러는 보안 정책을 적용하기 위해 다음 프로세스를 사용하여 독립적으로 작동합니다.

1. 보안 컨텍스트 제약 조건 컨트롤러는 Pod의 할당된 SCC에 따라 일부 보안 컨텍스트 필드를 변경할 수 있습니다. 예를 들어 seccomp 프로필이 비어 있거나 설정되지 않은 경우 Pod의 할당된 SCC에서 **seccompProfiles** 필드를 **runtime/default** 로 적용하는 경우 컨트롤러는 기본 유형을 **RuntimeDefault** 로 설정합니다.
2. 보안 컨텍스트 제약 조건 컨트롤러는 Pod의 보안 컨텍스트를 일치하는 SCC에 대해 검증합니다.
3. Pod 보안 승인 컨트롤러는 네임스페이스에 할당된 Pod 보안 표준에 대해 Pod의 보안 컨텍스트를 검증합니다.

13.2. POD 보안 승인 동기화 정보

글로벌 Pod 보안 승인 제어 구성 외에도 컨트롤러는 지정된 네임스페이스에 있는 서비스 계정의 SCC 권한에 따라 Pod 보안 승인 제어 **warn** 및 **audit** 레이블을 네임스페이스에 적용합니다.

컨트롤러는 각 네임스페이스에서 보안 컨텍스트 제약 조건을 사용하도록 **ServiceAccount** 오브젝트 권한을 검사합니다. SCC(보안 컨텍스트 제약 조건)는 필드 값을 기반으로 Pod 보안 프로필에 매핑됩니다. 컨트롤러는 이러한 변환된 프로필을 사용합니다. Pod가 생성될 때 경고 및 로깅 감사 이벤트를 표시하지 않도록 Pod 보안 승인 **warn** 및 **audit** 레이블은 네임스페이스에서 가장 권한이 있는 Pod 보안 프로필로 설정됩니다.

네임스페이스 레이블 지정은 네임스페이스 로컬 서비스 계정 권한을 기반으로 합니다.

Pod를 직접 적용하면 Pod를 실행하는 사용자의 SCC 권한을 사용할 수 있습니다. 그러나 사용자 권한은 자동 레이블 지정 중에 고려되지 않습니다.

13.2.1. Pod 보안 승인 동기화 네임스페이스 제외

*시스템에서 생성한 네임스페이스 및 **openshift-*** 접두사가 지정된 네임스페이스에서 Pod 보안 승인 동기화가 영구적으로 비활성화됩니다.*

클러스터 페이로드의 일부로 정의된 네임스페이스에는 Pod 보안 승인 동기화가 영구적으로 비활성화됩니다. 다음 네임스페이스는 영구적으로 비활성화되어 있습니다.

- ***default***
- ***kube-node-lease***
- ***kube-system***
- ***kube-public***
- ***openshift***
- ***openshift-**접두사가 있는 모든 system-created 네임스페이스*

13.3. POD 보안 승인 동기화 제어

대부분의 네임스페이스에 대해 자동 Pod 보안 승인 동기화를 활성화하거나 비활성화할 수 있습니다.

중요

시스템에서 생성한 네임스페이스에서 Pod 보안 승인 동기화를 활성화할 수 없습니다. 자세한 내용은 Pod 보안 승인 동기화 네임스페이스 제외를 참조하십시오.

프로세스

- *구성할 각 네임스페이스에 대해 **security.openshift.io/scc.podSecurityLabelSync** 라벨 값을 설정합니다.*
  - *네임스페이스에서 Pod 보안 승인 레이블 동기화를 비활성화하려면 **security.openshift.io/scc.podSecurityLabelSync** 레이블 값을 **false** 로 설정합니다. 다음 명령을 실행합니다.*

    ```
    $ oc label namespace <namespace>
    security.openshift.io/scc.podSecurityLabelSync=false
    ```

  - *네임스페이스에서 Pod 보안 승인 레이블 동기화를 활성화하려면 **security.openshift.io/scc.podSecurityLabelSync** 레이블 값을 **true** 로 설정합니다. 다음 명령을 실행합니다.*

    ```
    $ oc label namespace <namespace>
    security.openshift.io/scc.podSecurityLabelSync=true
    ```

추가 리소스

- *Pod 보안 승인 동기화 네임스페이스 제외*

13.4. 네임스페이스에 대한 POD 보안 승인 구성

네임스페이스 수준에서 Pod 보안 승인 설정을 구성할 수 있습니다. 네임스페이스의 각 Pod 보안 승인 모드에 대해 사용할 Pod 보안 승인 프로필을 설정할 수 있습니다.

프로세스

- 네임스페이스에 설정할 각 Pod 보안 승인 모드에 대해 다음 명령을 실행합니다.

```
$ oc label namespace <namespace> \ 1
    pod-security.kubernetes.io/<mode>=<profile> \ 2
    --overwrite
```

1 구성할 네임스페이스로 **<namespace>**를 설정합니다.

2 **<mode>**를 **enforce**, **warn**, 또는 **audit**으로 설정합니다. **<profile>**을 **restricted**,**baseline** 또는 **privileged** 로 설정합니다.

13.5. POD 보안 승인 경고 정보

Pod 보안 승인 컨트롤러의 감사 수준에 Pod 거부가 있다고 보고하면 **PodSecurityViolation** 경고가 트리거됩니다. 이 경고는 하루 동안 지속됩니다.

Kubernetes API 서버 감사 로그를 보고 트리거된 경고를 조사합니다. 예를 들어 글로벌 적용이 **restricted** Pod 보안 수준으로 설정된 경우 워크로드가 승인에 실패할 가능성이 높습니다.

Pod 보안 승인 위반 감사 이벤트를 식별하는 데 도움이 되는 내용은 Kubernetes 문서의 감사 주석을 참조하십시오.

13.6. 추가 리소스

- 감사 로그 보기
- 보안 컨텍스트 제약 조건 관리

14 장. LDAP 그룹 동기화

dedicated-admin 역할의 관리자는 그룹을 사용하여 사용자를 관리하고 권한을 변경하며 협업을 개선할 수 있습니다. 조직에서 이미 사용자 그룹을 생성하여 LDAP 서버에 저장했을 수 있습니다. OpenShift Dedicated는 이러한 LDAP 레코드를 내부 OpenShift Dedicated 레코드와 동기화하여 한 곳에서 그룹을 관리할 수 있습니다. OpenShift Dedicated는 현재 그룹 멤버십을 정의하는 데 세 가지 공통 스키마(RFC 2307, Active Directory, 보강된 Active Directory)를 사용하여 LDAP 서버와 그룹 동기화를 지원합니다.

LDAP 구성에 대한 자세한 내용은 LDAP ID 공급자 구성을 참조하십시오.

참고

그룹을 동기화하려면 **dedicated-admin** 권한이 있어야 합니다.

14.1. LDAP 동기화 구성 정보

LDAP 동기화를 실행하려면 동기화 구성 파일이 필요합니다. 이 파일에는 다음과 같은 LDAP 클라이언트 구성 세부 사항이 있습니다.

- LDAP 서버 연결에 필요한 구성
- LDAP 서버에서 사용되는 스키마에 종속된 동기화 구성 옵션
- OpenShift Dedicated 그룹 이름을 LDAP 서버의 그룹에 매핑하는 관리자 정의 이름 매핑 목록입니다.

구성 파일의 형식은 사용 중인 스키마(RFC 2307, Active Directory 또는 보강된 Active Directory)에 따라 다릅니다.

LDAP 클라이언트 구성

구성의 LDAP 클라이언트 구성 섹션에서는 LDAP 서버에 대한 연결을 정의합니다.

구성의 LDAP 클라이언트 구성 섹션에서는 LDAP 서버에 대한 연결을 정의합니다.

LDAP 클라이언트 구성

```
url: ldap://10.0.0.0:389 (1)
bindDN: cn=admin,dc=example,dc=com (2)
bindPassword: <password> (3)
insecure: false (4)
ca: my-ldap-ca-bundle.crt (5)
```

(1) 연결 프로토콜, 데이터베이스를 호스팅하는 LDAP 서버의 IP 주소, 연결할 포트로 **scheme://host:port** 형식으로 되어 있습니다.

(2) 바인딩 DN으로 사용할 선택적 고유 이름(DN) 입니다. OpenShift Dedicated는 동기화 작업을 위한 항목을 검색하는 데 높은 권한이 필요한 경우 이 기능을 사용합니다.

(3) 바인딩에 사용할 선택적 암호입니다. OpenShift Dedicated는 동기화 작업을 위한 항목을 검색하는 데 높은 권한이 필요한 경우 이 기능을 사용합니다. 이 값은 환경 변수, 외부 파일 또는 암호화된 파일로 제공될 수도 있습니다.

(4)

*false*인 경우 보안 LDAP(*ldaps://*) URL이 TLS를 사용하여 연결되고 안전하지 않은 LDAP(*ldap://*) URL은 TLS로 업그레이드됩니다. *true*인 경우 서버에 TLS 연결이 이루어지지 않으며 *ldaps://* URL

5 구성된 URL에 대한 서버 인증서의 유효성을 확인하는 데 사용할 인증서 번들입니다. 비어 있는 경우 OpenShift Dedicated는 시스템에서 신뢰하는 루트를 사용합니다. *insecure*가 *false*로 설정된 경우에만 적용됩니다.

LDAP 쿼리 정의

동기화 구성은 동기화에 필요한 항목에 대한 LDAP 쿼리 정의로 구성됩니다. LDAP 쿼리의 특정 정의는 LDAP 서버에 멤버십 정보를 저장하는 데 사용하는 스키마에 따라 다릅니다.

LDAP 쿼리 정의

```
baseDN: ou=users,dc=example,dc=com 1
scope: sub 2
derefAliases: never 3
timeout: 0 4
filter: (objectClass=person) 5
pageSize: 0 6
```

1 모든 검색이 시작되는 디렉터리 분기의 DN(고유 이름)입니다. 디렉터리 트리의 최상위를 지정해야 하지만 디렉터리의 하위 트리를 지정할 수도 있습니다.

2 검색 범위입니다. 유효한 값은 *base*, *one* 또는 *sub*입니다. 값이 정의되지 않은 경우 *sub* 범위로 가정됩니다. 범위 옵션에 대한 설명은 아래 표를 참조하십시오.

3 LDAP 트리의 별칭에 관한 검색 동작입니다. 유효한 값은 *never*, *search*, *base* 또는 *always*입니다. 값이 정의되지 않은 경우 기본값은 *always* 역참조 별칭으로 설정됩니다. 역참조 동작에 대한 설명은 아래 표를 참조하십시오.

4 클라이언트에서 검색할 수 있는 시간 제한(초)입니다. 값이 *0*이면 클라이언트 쪽 제한이 적용되지 않습니다.

5 유효한 LDAP 검색 필터입니다. 정의되지 않은 경우 기본값은 *(objectClass=*)*입니다.

6 서버 응답 페이지의 최대 크기 옵션으로, LDAP 항목으로 측정합니다. *0*으로 설정하면 응답 페이지에 크기 제한이 없습니다. 클라이언트 또는 서버에서 기본적으로 허용하는 것보다 쿼리에서 더 많은 항목을 반환하는 경우, 페이징 크기를 설정해야 합니다.

표 14.1. LDAP 검색 범위 옵션

LDAP 검색 범위	설명
base	쿼리의 기본 DN에 의해 지정된 오브젝트만 고려합니다.
one	쿼리의 기본 DN과 동일한 수준의 트리에 있는 모든 오브젝트를 고려합니다.
sub	쿼리에 대해 제공된 기본 DN을 기반으로 하는 전체 하위 트리를 고려합니다.

표 14.2. LDAP 역참조 동작

역참조 행동	설명
never	LDAP 트리에 있는 별칭을 역참조하지 않도록 합니다.
search	검색하는 동안 역참조 별칭만 발견되었습니다.
base	기본 오브젝트를 찾는 동안 별칭만 역참조합니다.
always	항상 LDAP 트리에 있는 모든 별칭을 역참조합니다.

사용자 정의 이름 매핑

사용자 정의 이름 매핑은 OpenShift Dedicated 그룹의 이름을 LDAP 서버에서 그룹을 찾는 고유 식별자에 명시적으로 매핑합니다. 매핑에서는 일반 YAML 구문을 사용합니다. 사용자 정의 매핑은 LDAP 서버의 모든 그룹에 대한 항목을 포함하거나 그룹의 하위 집합만 포함할 수 있습니다. LDAP 서버에 사용자 정의 이름 매핑이 없는 그룹이 있는 경우 동기화 중 기본 동작은 OpenShift Dedicated 그룹 이름으로 지정된 속성을 사용하는 것입니다.

사용자 정의 이름 매핑

```
groupUIDNameMapping:
  "cn=group1,ou=groups,dc=example,dc=com": firstgroup
  "cn=group2,ou=groups,dc=example,dc=com": secondgroup
  "cn=group3,ou=groups,dc=example,dc=com": thirdgroup
```

14.1.1. RFC 2307 구성 파일 정보

RFC 2307 스키마를 사용하려면 사용자 및 그룹 항목에 대한 LDAP 쿼리 정의와 내부 OpenShift Dedicated 레코드에서 해당 항목을 표시하는 속성을 제공해야 합니다.

명확하게 하기 위해 OpenShift Dedicated에서 생성하는 그룹은 사용자 또는 관리자용 필드에 대해 가능한 경우 고유 이름 이외의 속성을 사용해야 합니다. 예를 들어 이메일로 OpenShift Dedicated 그룹의 사용자를 식별하고 그룹 이름을 공통 이름으로 사용합니다. 다음 구성 파일에서는 이러한 관계를 생성합니다.

참고

사용자 정의 이름 매핑을 사용하는 경우에는 구성 파일이 다릅니다.

RFC 2307 스키마를 사용하는 LDAP 동기화 구성: rfc2307_config.yaml

```
kind: LDAPSyncConfig
apiVersion: v1
url: ldap://LDAP_SERVICE_IP:389 (1)
insecure: false (2)
rfc2307:
    groupsQuery:
        baseDN: "ou=groups,dc=example,dc=com"
        scope: sub
        derefAliases: never
        pageSize: 0
    groupUIDAttribute: dn (3)
```

```
    groupNameAttributes: [ cn ] 4
    groupMembershipAttributes: [ member ] 5
    usersQuery:
      baseDN: "ou=users,dc=example,dc=com"
      scope: sub
      derefAliases: never
      pageSize: 0
    userUIDAttribute: dn 6
    userNameAttributes: [ mail ] 7
    tolerateMemberNotFoundErrors: false
    tolerateMemberOutOfScopeErrors: false
```

1. 이 그룹의 레코드가 저장된 LDAP 서버의 IP 주소 및 호스트입니다.
2. **false**인 경우 보안 LDAP(**ldaps://**) URL이 TLS를 사용하여 연결되고 안전하지 않은 LDAP(**ldap://**) URL은 TLS로 업그레이드됩니다. **true**인 경우 서버에 TLS 연결이 이루어지지 않으며 **ldaps://** URL 스키마를 사용할 수 없습니다.
3. LDAP 서버에서 그룹을 고유하게 식별하는 속성입니다. **groupUIDAttribute**로 DN을 사용할 때는 **groupsQuery** 필터를 지정할 수 없습니다. 세분화된 필터링의 경우 허용 목록/차단 목록 메서드를 사용하십시오.
4. 그룹 이름으로 사용할 속성입니다.
5. 멤버십 정보를 저장하는 그룹의 속성입니다.
6. LDAP 서버에서 사용자를 고유하게 식별하는 속성입니다. userUIDAttribute로 DN을 사용할 때는 **usersQuery** 필터를 지정할 수 없습니다. 세분화된 필터링의 경우 허용 목록/차단 목록 메서드를 사용하십시오.
7. OpenShift Dedicated 그룹 레코드에서 사용자 이름으로 사용할 속성입니다.

14.1.2. Active Directory 구성 파일 정보

Active Directory 스키마를 사용하려면 사용자 항목에 대한 LDAP 쿼리 정의와 내부 OpenShift Dedicated 그룹 레코드에서 해당 항목을 나타내는 속성을 제공해야 합니다.

명확하게 하기 위해 OpenShift Dedicated에서 생성하는 그룹은 사용자 또는 관리자용 필드에 대해 가능한 경우 고유 이름 이외의 속성을 사용해야 합니다. 예를 들어, OpenShift Dedicated 그룹의 사용자를 이메일로 식별하지만 LDAP 서버의 그룹 이름으로 그룹 이름을 정의합니다. 다음 구성 파일에서는 이러한 관계를 생성합니다.

Active Directory 스키마를 사용하는 LDAP 동기화 구성: active_directory_config.yaml

```
kind: LDAPSyncConfig
apiVersion: v1
url: ldap://LDAP_SERVICE_IP:389
activeDirectory:
    usersQuery:
      baseDN: "ou=users,dc=example,dc=com"
      scope: sub
      derefAliases: never
      filter: (objectclass=person)
```

```
    pageSize: 0
  userNameAttributes: [ mail ] (1)
  groupMembershipAttributes: [ memberOf ] (2)
```

1. OpenShift Dedicated 그룹 레코드에서 사용자 이름으로 사용할 속성입니다.
2. 멤버십 정보를 저장하는 사용자 속성입니다.

14.1.3. 보강된 Active Directory 구성 파일 정보

보강된 Active Directory 스키마를 사용하려면 사용자 항목 및 그룹 항목에 대한 LDAP 쿼리 정의와 내부 OpenShift Dedicated 그룹 레코드에서 해당 항목을 표시하는 속성을 제공해야 합니다.

명확하게 하기 위해 OpenShift Dedicated에서 생성하는 그룹은 사용자 또는 관리자용 필드에 대해 가능한 경우 고유 이름 이외의 속성을 사용해야 합니다. 예를 들어 이메일로 OpenShift Dedicated 그룹의 사용자를 식별하고 그룹 이름을 공통 이름으로 사용합니다. 다음 구성 파일에서는 이러한 관계를 생성합니다.

보강된 Active Directory 스키마를 사용하는 LDAP 동기화 구성: augmented_active_directory_config.yaml

```
kind: LDAPSyncConfig
apiVersion: v1
url: ldap://LDAP_SERVICE_IP:389
augmentedActiveDirectory:
  groupsQuery:
    baseDN: "ou=groups,dc=example,dc=com"
    scope: sub
    derefAliases: never
    pageSize: 0
  groupUIDAttribute: dn (1)
  groupNameAttributes: [ cn ] (2)
  usersQuery:
    baseDN: "ou=users,dc=example,dc=com"
    scope: sub
    derefAliases: never
    filter: (objectclass=person)
    pageSize: 0
  userNameAttributes: [ mail ] (3)
  groupMembershipAttributes: [ memberOf ] (4)
```

1. LDAP 서버에서 그룹을 고유하게 식별하는 속성입니다. groupUIDAttribute로 DN을 사용할 때는 **groupsQuery** 필터를 지정할 수 없습니다. 세분화된 필터링의 경우 허용 목록/차단 목록 메서드를 사용하십시오.
2. 그룹 이름으로 사용할 속성입니다.
3. OpenShift Dedicated 그룹 레코드에서 사용자 이름으로 사용할 속성입니다.
4. 멤버십 정보를 저장하는 사용자 속성입니다.

14.2. LDAP 동기화 실행

동기화 구성 파일을 생성하면 동기화를 시작할 수 있습니다. 관리자는 OpenShift Dedicated를 사용하여 동일한 서버에서 다양한 동기화 유형을 수행할 수 있습니다.

14.2.1. OpenShift Dedicated와 LDAP 서버 동기화

LDAP 서버의 모든 그룹을 OpenShift Dedicated와 동기화할 수 있습니다.

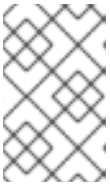
사전 요구 사항

- *동기화 구성 파일을 생성합니다.*
- ***dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.*

프로세스

- *LDAP 서버의 모든 그룹을 OpenShift Dedicated와 동기화하려면 다음을 수행합니다.*

```
$ oc adm groups sync --sync-config=config.yaml --confirm
```



참고

*기본적으로 모든 그룹 동기화 작업은 시험 실행이므로 OpenShift Dedicated 그룹 레코드를 변경하려면 **oc adm groups sync** 명령에 **--confirm** 플래그를 설정해야 합니다.*

14.2.2. LDAP 서버와 OpenShift Dedicated 그룹 동기화

구성 파일에 지정된 LDAP 서버의 그룹에 해당하는 OpenShift Dedicated에 이미 있는 모든 그룹을 동기화할 수 있습니다.

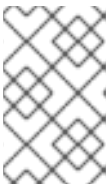
사전 요구 사항

- *동기화 구성 파일을 생성합니다.*
- ***dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.*

프로세스

- *LDAP 서버와 OpenShift Dedicated 그룹을 동기화하려면 다음을 수행합니다.*

```
$ oc adm groups sync --type=openshift --sync-config=config.yaml --confirm
```

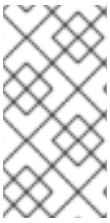


참고

*기본적으로 모든 그룹 동기화 작업은 시험 실행이므로 OpenShift Dedicated 그룹 레코드를 변경하려면 **oc adm groups sync** 명령에 **--confirm** 플래그를 설정해야 합니다.*

14.2.3. OpenShift Dedicated와 LDAP 서버의 하위 그룹 동기화

허용 목록 파일, 블랙리스트 파일 또는 둘 다를 사용하여 OpenShift Dedicated와 LDAP 그룹의 하위 집합을 동기화할 수 있습니다.



참고

차단 목록 파일, 허용 목록 파일 또는 허용 목록 리터럴을 조합하여 사용할 수 있습니다. 허용 목록 및 차단 목록 파일에는 한 줄에 하나의 고유한 그룹 식별자를 포함해야 하며, 명령 자체에 허용 목록 리터럴을 직접 포함할 수 있습니다. 이러한 지침은 OpenShift Dedicated 에 이미 존재하는 그룹뿐만 아니라 LDAP 서버에 있는 그룹에 적용됩니다.

사전 요구 사항

- *동기화 구성 파일을 생성합니다.*
- ***dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.*

프로세스

- *LDAP 그룹의 서브 세트를 OpenShift Dedicated와 동기화하려면 다음 명령을 사용합니다.*

```
$ oc adm groups sync --whitelist=<whitelist_file> \
            --sync-config=config.yaml      \
            --confirm
```

```
$ oc adm groups sync --blacklist=<blacklist_file> \
            --sync-config=config.yaml      \
            --confirm
```

```
$ oc adm groups sync <group_unique_identifier>    \
            --sync-config=config.yaml      \
            --confirm
```

```
$ oc adm groups sync <group_unique_identifier>  \
            --whitelist=<whitelist_file> \
            --blacklist=<blacklist_file> \
            --sync-config=config.yaml    \
            --confirm
```

```
$ oc adm groups sync --type=openshift           \
            --whitelist=<whitelist_file> \
            --sync-config=config.yaml    \
            --confirm
```

참고

*기본적으로 모든 그룹 동기화 작업은 시험 실행이므로 OpenShift Dedicated 그룹 레코드를 변경하려면 **oc adm groups sync** 명령에 **--confirm** 플래그를 설정해야 합니다.*

14.3. 그룹 정리 작업 실행

또한 생성한 LDAP 서버의 레코드가 더 이상 존재하지 않는 경우 관리자는 OpenShift Dedicated 레코드에서 그룹을 제거하도록 선택할 수도 있습니다. 정리 작업에는 동기화 작업에 사용한 것과 동일한 동기화 구성 파일과 허용 목록 또는 차단 목록을 사용할 수 있습니다.

예를 들면 다음과 같습니다.

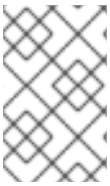
```
$ oc adm prune groups --sync-config=/path/to/ldap-sync-config.yaml --confirm
```

```
$ oc adm prune groups --whitelist=/path/to/whitelist.txt --sync-config=/path/to/ldap-sync-config.yaml --confirm
```

```
$ oc adm prune groups --blacklist=/path/to/blacklist.txt --sync-config=/path/to/ldap-sync-config.yaml --confirm
```

14.4. LDAP 그룹 동기화의 예

이 섹션에는 RFC 2307, Active Directory, 보강된 Active Directory 스키마에 대한 예가 포함되어 있습니다.



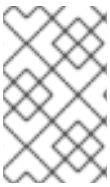
참고

이러한 예에서는 모든 사용자를 해당 그룹의 직접 멤버로 가정합니다. 특히, 어떠한 그룹도 다른 그룹의 멤버가 될 수 없습니다. 중첩 그룹을 동기화하는 방법에 대한 정보는 중첩 멤버십 동기화 예를 참조하십시오.

14.4.1. RFC 2307 스키마를 사용한 그룹 동기화

*RFC 2307 스키마의 경우 다음 예제에서는 **Jane**과 **Jim**이라는 두 멤버가 있는 그룹 **admins**를 동기화합니다. 예제에서는 다음을 설명합니다.*

- *그룹 및 사용자를 LDAP 서버에 추가하는 방법*
- *OpenShift Dedicated의 결과 그룹 레코드는 동기화 후입니다.*



참고

이러한 예에서는 모든 사용자를 해당 그룹의 직접 멤버로 가정합니다. 특히, 어떠한 그룹도 다른 그룹의 멤버가 될 수 없습니다. 중첩 그룹을 동기화하는 방법에 대한 정보는 중첩 멤버십 동기화 예를 참조하십시오.

*RFC 2307 스키마에서는 사용자(Jane 및 Jim)와 그룹 모두 LDAP 서버에 최상위 항목으로 존재하며, 그룹 멤버십은 그룹 속성에 저장됩니다. 다음의 **ldif** 조각에서는 이 스키마의 사용자 및 그룹을 정의합니다.*

RFC 2307 스키마를 사용하는 LDAP 항목: rfc2307.ldif

```
dn: ou=users,dc=example,dc=com
objectClass: organizationalUnit
ou: users
dn: cn=Jane,ou=users,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Jane
sn: Smith
displayName: Jane Smith
mail: jane.smith@example.com
dn: cn=Jim,ou=users,dc=example,dc=com
```

```
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Jim
sn: Adams
displayName: Jim Adams
mail: jim.adams@example.com
dn: ou=groups,dc=example,dc=com
objectClass: organizationalUnit
ou: groups
dn: cn=admins,ou=groups,dc=example,dc=com (1)
objectClass: groupOfNames
cn: admins
owner: cn=admin,dc=example,dc=com
description: System Administrators
member: cn=Jane,ou=users,dc=example,dc=com (2)
member: cn=Jim,ou=users,dc=example,dc=com
```

(1) 그룹은 LDAP 서버의 최상위 항목입니다.

(2) 그룹 멤버와 함께 그룹 속성이 식별을 위한 참조 정보로 나열됩니다.

사전 요구 사항

- 구성 파일을 생성합니다.
- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.

프로세스

- **rfc2307_config.yaml** 파일과의 동기화를 실행합니다.

```
$ oc adm groups sync --sync-config=rfc2307_config.yaml --confirm
```

OpenShift Dedicated는 위의 동기화 작업의 결과로 다음 그룹 레코드를 생성합니다.

rfc2307_config.yaml 파일을 사용하여 생성된 OpenShift Dedicated 그룹

```
apiVersion: user.openshift.io/v1
kind: Group
metadata:
  annotations:
    openshift.io/ldap.sync-time: 2015-10-13T10:08:38-0400 (1)
    openshift.io/ldap.uid: cn=admins,ou=groups,dc=example,dc=com (2)
    openshift.io/ldap.url: LDAP_SERVER_IP:389 (3)
  creationTimestamp:
  name: admins (4)
users: (5)
- jane.smith@example.com
- jim.adams@example.com
```

(1) 이 OpenShift Dedicated 그룹이 LDAP 서버와 마지막으로 동기화된 경우 ISO 6801 형식으로 되어 있습니다.

2 LDAP 서버에서 그룹의 고유 식별자입니다.

3 이 그룹의 레코드가 저장된 LDAP 서버의 IP 주소 및 호스트입니다.

4 동기화 파일에서 지정한 그룹의 이름입니다.

5 동기화 파일에서 지정한 대로 이름이 지정된 그룹 멤버에 해당하는 사용자입니다.

14.4.2. RFC2307 스키마와 사용자 정의 이름 매핑을 사용한 그룹 동기화

사용자 정의 이름 매핑과 그룹을 동기화하면 구성 파일이 이러한 매핑을 포함하도록 아래와 같이 변경됩니다.

사용자 정의 이름 매핑과 함께 RFC 2307 스키마를 사용하는 LDAP 동기화 구성: rfc2307_config_user_defined.yaml

```
kind: LDAPSyncConfig
apiVersion: v1
groupUIDNameMapping:
  "cn=admins,ou=groups,dc=example,dc=com": Administrators 1
rfc2307:
    groupsQuery:
        baseDN: "ou=groups,dc=example,dc=com"
        scope: sub
        derefAliases: never
        pageSize: 0
    groupUIDAttribute: dn 2
    groupNameAttributes: [ cn ] 3
    groupMembershipAttributes: [ member ]
    usersQuery:
        baseDN: "ou=users,dc=example,dc=com"
        scope: sub
        derefAliases: never
        pageSize: 0
    userUIDAttribute: dn 4
    userNameAttributes: [ mail ]
    tolerateMemberNotFoundErrors: false
    tolerateMemberOutOfScopeErrors: false
```

1 사용자 정의 이름 매핑입니다.

2 사용자 정의 이름 매핑에서 키에 사용되는 고유 식별자 속성입니다. groupUIDAttribute로 DN을 사용할 때는 **groupsQuery** 필터를 지정할 수 없습니다. 세분화된 필터링의 경우 허용 목록/차단 목록 메서드를 사용하십시오.

3 사용자 정의 이름 매핑에 고유 식별자가 없는 경우 OpenShift Dedicated 그룹의 이름을 지정하는 속성입니다.

4 LDAP 서버에서 사용자를 고유하게 식별하는 속성입니다. userUIDAttribute로 DN을 사용할 때는 **usersQuery** 필터를 지정할 수 없습니다. 세분화된 필터링의 경우 허용 목록/차단 목록 메서드를 사용하십시오.

사전 요구 사항

- *구성 파일을 생성합니다.*
- ***dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.*

프로세스

- ***rfc2307_config_user_defined.yaml** 파일과의 동기화를 실행합니다.*

```
$ oc adm groups sync --sync-config=rfc2307_config_user_defined.yaml --confirm
```

OpenShift Dedicated는 위의 동기화 작업의 결과로 다음 그룹 레코드를 생성합니다.

rfc2307_config_user_defined.yaml 파일을 사용하여 생성된 OpenShift Dedicated 그룹

```
apiVersion: user.openshift.io/v1
kind: Group
metadata:
  annotations:
    openshift.io/ldap.sync-time: 2015-10-13T10:08:38-0400
    openshift.io/ldap.uid: cn=admins,ou=groups,dc=example,dc=com
    openshift.io/ldap.url: LDAP_SERVER_IP:389
  creationTimestamp:
  name: Administrators (1)
users:
- jane.smith@example.com
- jim.adams@example.com
```

(1) *사용자 정의 이름 매핑에 의해 지정된 그룹의 이름입니다.*

14.4.3. RFC 2307과 사용자 정의 오류 허용 오차를 사용한 그룹 동기화

기본적으로 동기화 중인 그룹에 멤버 쿼리에 정의된 범위를 벗어나는 항목이 있는 멤버가 포함된 경우, 그룹 동기화에 오류가 발생하여 실패합니다.

```
Error determining LDAP group membership for "<group>": membership lookup for user "<user>" in group "<group>" failed because of "search for entry with dn="<user-dn>" would search outside of the base dn specified (dn="<base-dn>")".
```

*이러한 오류는 대부분 **usersQuery** 필드의 **baseDN**이 잘못 구성되었음을 나타냅니다. 그러나 **baseDN**에에 의도적으로 일부 그룹 멤버가 포함되어 있지 않은 경우, **tolerateMemberOutOfScopeErrors:true**를 설정하면 그룹을 계속 동기화할 수 있습니다. 범위를 벗어난 멤버는 무시됩니다.*

마찬가지로 그룹 동기화 프로세스에서 그룹 멤버를 찾지 못하면 오류와 함께 실패합니다.

```
Error determining LDAP group membership for "<group>": membership lookup for user "<user>" in group "<group>" failed because of "search for entry with base dn="<user-dn>" refers to a non-existent entry".
Error determining LDAP group membership for "<group>": membership lookup for user "<user>" in group "<group>" failed because of "search for entry with base dn="<user-dn>" and filter "<filter>" did not return any results".
```

*이러한 오류는 대부분 **usersQuery** 필드가 잘못 구성되었음을 나타냅니다. 그러나 그룹에 누락된 것으로 알려진 멤버 항목이 포함된 경우, **tolerateMemberNotFoundErrors:true**로 설정하면 그룹을 계속 동기화할 수 있습니다. 문제가 있는 멤버는 무시됩니다.*

주의

LDAP 그룹 동기화에 오류 허용을 사용하면 동기화 프로세스에서 문제가 있는 멤버 항목을 무시합니다. LDAP 그룹 동기화가 올바르게 구성되지 않으면 동기화된 OpenShift Dedicated 그룹에 멤버가 누락될 수 있습니다.

그룹 멤버십에 문제가 있는 RFC 2307 스키마를 사용하는 LDAP 항목: rfc2307_problematic_users.ldif

```
dn: ou=users,dc=example,dc=com
objectClass: organizationalUnit
ou: users
dn: cn=Jane,ou=users,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Jane
sn: Smith
displayName: Jane Smith
mail: jane.smith@example.com
dn: cn=Jim,ou=users,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Jim
sn: Adams
displayName: Jim Adams
mail: jim.adams@example.com
dn: ou=groups,dc=example,dc=com
objectClass: organizationalUnit
ou: groups
dn: cn=admins,ou=groups,dc=example,dc=com
objectClass: groupOfNames
cn: admins
owner: cn=admin,dc=example,dc=com
description: System Administrators
member: cn=Jane,ou=users,dc=example,dc=com
member: cn=Jim,ou=users,dc=example,dc=com
member: cn=INVALID,ou=users,dc=example,dc=com (1)
member: cn=Jim,ou=OUTOFSCOPE,dc=example,dc=com (2)
```

1. *LDAP 서버에 존재하지 않는 멤버입니다.*
2. *동기화 작업에 대한 사용자 쿼리에 있을 수 있지만 **baseDN**에 없는 멤버입니다.*

위 예제의 오류를 허용하려면 동기화 구성 파일에 다음을 추가해야 합니다.

오류를 허용하는 RFC 2307 스키마를 사용하는 LDAP 동기화 구성: rfc2307_config_tolerating.yaml

```
kind: LDAPSyncConfig
apiVersion: v1
url: ldap://LDAP_SERVICE_IP:389
rfc2307:
    groupsQuery:
        baseDN: "ou=groups,dc=example,dc=com"
        scope: sub
        derefAliases: never
    groupUIDAttribute: dn
    groupNameAttributes: [ cn ]
    groupMembershipAttributes: [ member ]
    usersQuery:
        baseDN: "ou=users,dc=example,dc=com"
        scope: sub
        derefAliases: never
    userUIDAttribute: dn (1)
    userNameAttributes: [ mail ]
    tolerateMemberNotFoundErrors: true (2)
    tolerateMemberOutOfScopeErrors: true (3)
```

(1) LDAP 서버에서 사용자를 고유하게 식별하는 속성입니다. userUIDAttribute로 DN을 사용할 때는 **usersQuery** 필터를 지정할 수 없습니다. 세분화된 필터링의 경우 허용 목록/차단 목록 메서드를 사용하십시오.

(2) **true**인 경우 동기화 작업에서는 일부 멤버가 없는 그룹을 허용하고 LDAP 항목이 없는 멤버를 무시합니다. 그룹 멤버를 찾을 수 없는 경우 동기화 작업의 기본 동작은 실패하는 것입니다.

(3) **true**인 경우 동기화 작업에서는 일부 멤버가 **usersQuery** 기본 DN에 지정된 사용자 범위를 벗어나는 그룹을 허용하고, 멤버 조회 범위를 벗어나는 멤버를 무시합니다. 그룹 멤버가 범위를 벗어나는 경우 동기화 작업의 기본 동작은 실패하는 것입니다.

사전 요구 사항

- 구성 파일을 생성합니다.
- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.

프로세스

- **rfc2307_config_tolerating.yaml** 파일과의 동기화를 실행합니다.

```
$ oc adm groups sync --sync-config=rfc2307_config_tolerating.yaml --confirm
```

OpenShift Dedicated는 위의 동기화 작업의 결과로 다음 그룹 레코드를 생성합니다.

rfc2307_config.yaml 파일을 사용하여 생성된 OpenShift Dedicated 그룹

```
apiVersion: user.openshift.io/v1
```

```
kind: Group
metadata:
  annotations:
    openshift.io/ldap.sync-time: 2015-10-13T10:08:38-0400
    openshift.io/ldap.uid: cn=admins,ou=groups,dc=example,dc=com
    openshift.io/ldap.url: LDAP_SERVER_IP:389
  creationTimestamp:
  name: admins
users: (1)
- jane.smith@example.com
- jim.adams@example.com
```

(1) 동기화 파일에서 지정한 대로 그룹 멤버에 해당하는 사용자입니다. 조회에 허용되는 오류가 발생한 멤버가 없습니다.

14.4.4. Active Directory 스키마를 사용하여 그룹 동기화

Active Directory 스키마에서는 두 사용자(Jane 및 Jim) 모두 LDAP 서버에 최상위 항목으로 존재하며, 그룹 멤버십은 사용자 속성에 저장됩니다. 다음의 **ldif** 조각에서는 이 스키마의 사용자 및 그룹을 정의합니다.

Active Directory 스키마를 사용하는 LDAP 항목: active_directory.ldif

```
dn: ou=users,dc=example,dc=com
objectClass: organizationalUnit
ou: users

dn: cn=Jane,ou=users,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: testPerson
cn: Jane
sn: Smith
displayName: Jane Smith
mail: jane.smith@example.com
memberOf: admins (1)

dn: cn=Jim,ou=users,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: testPerson
cn: Jim
sn: Adams
displayName: Jim Adams
mail: jim.adams@example.com
memberOf: admins
```

(1) 사용자의 그룹 멤버십은 사용자의 속성으로 나열되며, 그룹은 서버에 항목으로 존재하지 않습니다. **memberOf** 속성이 사용자의 리터럴 속성이 아닐 수도 있습니다. 일부 LDAP 서버에서는 검색 중 생성되어 클라이언트에 반환되지만 데이터베이스에는 커밋되지 않습니다.

사전 요구 사항

- *구성 파일을 생성합니다.*
- ***dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.*

프로세스

- ***active_directory_config.yaml** 파일과의 동기화를 실행합니다.*

```
$ oc adm groups sync --sync-config=active_directory_config.yaml --confirm
```

OpenShift Dedicated는 위의 동기화 작업의 결과로 다음 그룹 레코드를 생성합니다.

active_directory_config.yaml 파일을 사용하여 생성된 OpenShift Dedicated 그룹

```
apiVersion: user.openshift.io/v1
kind: Group
metadata:
  annotations:
    openshift.io/ldap.sync-time: 2015-10-13T10:08:38-0400 (1)
    openshift.io/ldap.uid: admins (2)
    openshift.io/ldap.url: LDAP_SERVER_IP:389 (3)
  creationTimestamp:
  name: admins (4)
users: (5)
- jane.smith@example.com
- jim.adams@example.com
```

(1) *이 OpenShift Dedicated 그룹이 LDAP 서버와 마지막으로 동기화된 경우 ISO 6801 형식으로 되어 있습니다.*

(2) *LDAP 서버에서 그룹의 고유 식별자입니다.*

(3) *이 그룹의 레코드가 저장된 LDAP 서버의 IP 주소 및 호스트입니다.*

(4) *LDAP 서버에 나열된 그룹의 이름입니다.*

(5) *동기화 파일에서 지정한 대로 이름이 지정된 그룹 멤버에 해당하는 사용자입니다.*

14.4.5. 보강된 Active Directory 스키마를 사용하여 그룹 동기화

*보강된 Active Directory 스키마에서는 사용자(Jane 및 Jim)와 그룹 모두 LDAP 서버에 최상위 항목으로 존재하며, 그룹 멤버십은 사용자 속성에 저장됩니다. 다음의 **ldif** 조각에서는 이 스키마의 사용자 및 그룹을 정의합니다.*

보강된 Active Directory 스키마를 사용하는 LDAP 항목: augmented_active_directory.ldif

```
dn: ou=users,dc=example,dc=com
objectClass: organizationalUnit
ou: users

dn: cn=Jane,ou=users,dc=example,dc=com
```

```
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: testPerson
cn: Jane
sn: Smith
displayName: Jane Smith
mail: jane.smith@example.com
memberOf: cn=admins,ou=groups,dc=example,dc=com (1)

dn: cn=Jim,ou=users,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: testPerson
cn: Jim
sn: Adams
displayName: Jim Adams
mail: jim.adams@example.com
memberOf: cn=admins,ou=groups,dc=example,dc=com

dn: ou=groups,dc=example,dc=com
objectClass: organizationalUnit
ou: groups

dn: cn=admins,ou=groups,dc=example,dc=com (2)
objectClass: groupOfNames
cn: admins
owner: cn=admin,dc=example,dc=com
description: System Administrators
member: cn=Jane,ou=users,dc=example,dc=com
member: cn=Jim,ou=users,dc=example,dc=com
```

(1) 사용자의 그룹 멤버십이 사용자 속성으로 나열됩니다.

(2) 그룹은 LDAP 서버의 최상위 항목입니다.

사전 요구 사항

- 구성 파일을 생성합니다.
- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.

프로세스

- **augmented_active_directory_config.yaml** 파일과의 동기화를 실행합니다.

  ```
  $ oc adm groups sync --sync-config=augmented_active_directory_config.yaml --confirm
  ```

  OpenShift Dedicated는 위의 동기화 작업의 결과로 다음 그룹 레코드를 생성합니다.

  augmented_active_directory_config.yaml 파일을 사용하여 생성된 OpenShift Dedicated 그룹

```
apiVersion: user.openshift.io/v1
kind: Group
metadata:
  annotations:
    openshift.io/ldap.sync-time: 2015-10-13T10:08:38-0400 (1)
    openshift.io/ldap.uid: cn=admins,ou=groups,dc=example,dc=com (2)
    openshift.io/ldap.url: LDAP_SERVER_IP:389 (3)
  creationTimestamp:
  name: admins (4)
users: (5)
- jane.smith@example.com
- jim.adams@example.com
```

(1) 이 OpenShift Dedicated 그룹이 LDAP 서버와 마지막으로 동기화된 경우 ISO 6801 형식으로 되어 있습니다.

(2) LDAP 서버에서 그룹의 고유 식별자입니다.

(3) 이 그룹의 레코드가 저장된 LDAP 서버의 IP 주소 및 호스트입니다.

(4) 동기화 파일에서 지정한 그룹의 이름입니다.

(5) 동기화 파일에서 지정한 대로 이름이 지정된 그룹 멤버에 해당하는 사용자입니다.

14.4.5.1. LDAP 중첩 멤버십 동기화의 예

OpenShift Dedicated의 그룹은 중첩되지 않습니다. 데이터를 사용하려면 LDAP 서버에서 그룹 멤버십을 평면화해야 합니다. Microsoft의 Active Directory 서버에서는 OID **1.2.840.113556.1.4.1941**이 있는 **LDAP_MATCHING_RULE_IN_CHAIN** 규칙을 통해 이 기능을 지원합니다. 또한 이 일치 규칙을 사용하는 경우에는 명시적으로 허용된 그룹만 동기화할 수 있습니다.

이 섹션에는 한 명의 사용자 **Jane**과 하나의 그룹 **otheradmins**가 멤버인 **admins**라는 그룹을 동기화하는 보강된 Active Directory 스키마에 대한 예가 있습니다. **otheradmins** 그룹에는 **Jim**이라는 한 명의 사용자 멤버가 있습니다. 이 예제에서는 다음을 설명합니다.

- 그룹 및 사용자를 LDAP 서버에 추가하는 방법
- LDAP 동기화 구성 파일의 내용
- OpenShift Dedicated의 결과 그룹 레코드는 동기화 후입니다.

보강된 Active Directory 스키마에서는 사용자(**Jane** 및 **Jim**)와 그룹 모두 LDAP 서버에 최상위 항목으로 존재하며, 그룹 멤버십은 사용자 또는 그룹 속성에 저장됩니다. 다음의 **ldif** 조각에서는 이 스키마의 사용자 및 그룹을 정의합니다.

보강된 Active Directory 스키마를 중첩 멤버와 함께 사용하는 LDAP 항목: augmented_active_directory_nested.ldif

```
dn: ou=users,dc=example,dc=com
objectClass: organizationalUnit
ou: users

dn: cn=Jane,ou=users,dc=example,dc=com
objectClass: person
```

```
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: testPerson
cn: Jane
sn: Smith
displayName: Jane Smith
mail: jane.smith@example.com
memberOf: cn=admins,ou=groups,dc=example,dc=com (1)

dn: cn=Jim,ou=users,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: testPerson
cn: Jim
sn: Adams
displayName: Jim Adams
mail: jim.adams@example.com
memberOf: cn=otheradmins,ou=groups,dc=example,dc=com (2)

dn: ou=groups,dc=example,dc=com
objectClass: organizationalUnit
ou: groups

dn: cn=admins,ou=groups,dc=example,dc=com (3)
objectClass: group
cn: admins
owner: cn=admin,dc=example,dc=com
description: System Administrators
member: cn=Jane,ou=users,dc=example,dc=com
member: cn=otheradmins,ou=groups,dc=example,dc=com

dn: cn=otheradmins,ou=groups,dc=example,dc=com (4)
objectClass: group
cn: otheradmins
owner: cn=admin,dc=example,dc=com
description: Other System Administrators
memberOf: cn=admins,ou=groups,dc=example,dc=com (5) (6)
member: cn=Jim,ou=users,dc=example,dc=com
```

(1) (2) (5) *사용자 및 그룹의 멤버십은 오브젝트 속성으로 나열됩니다.*

(3) (4) *그룹은 LDAP 서버의 최상위 항목입니다.*

(6) ***otheradmins*** *그룹은* ***admins*** *그룹의 멤버입니다.*

중첩 그룹을 Active Directory와 동기화할 때 사용자 항목 및 그룹 항목에 대한 LDAP 쿼리 정의와 내부 OpenShift Dedicated 그룹 레코드에서 해당 그룹을 표시하는 속성을 제공해야 합니다. 또한 이 구성에는 특정 변경이 필요합니다.

- ***oc adm groups sync*** *명령에서 그룹을 명시적으로 허용해야 합니다.*
- ***LDAP_MATCHING_RULE_IN_CHAIN*** *규칙을 준수하려면 사용자의* ***groupMembershipAttributes*** *에* ***"memberOf:1.2.840.113556.1.4.1941:"*** *이 포함되어야 합니다.*

- ***groupUIDAttribute***를 ***dn***으로 설정해야 합니다.
- ***groupsQuery***의 경우
  - ***filter***를 설정하지 않아야 합니다.
  - 유효한 ***derefAliases***를 설정해야 합니다.
  - 해당 값이 무시되므로 ***baseDN***을 설정하지 않아야 합니다.
  - 해당 값이 무시되므로 ***scope***를 설정하지 않아야 합니다.

명확하게 하기 위해 OpenShift Dedicated에서 생성하는 그룹은 사용자 또는 관리자용 필드에 대해 가능한 경우 고유 이름 이외의 속성을 사용해야 합니다. 예를 들어 이메일로 OpenShift Dedicated 그룹의 사용자를 식별하고 그룹 이름을 공통 이름으로 사용합니다. 다음 구성 파일에서는 이러한 관계를 생성합니다.

중첩 멤버와 함께 보강된 Active Directory 스키마를 사용하는 LDAP 동기화 구성: augmented_active_directory_config_nested.yaml

```
kind: LDAPSyncConfig
apiVersion: v1
url: ldap://LDAP_SERVICE_IP:389
augmentedActiveDirectory:
  groupsQuery: (1)
    derefAliases: never
    pageSize: 0
  groupUIDAttribute: dn (2)
  groupNameAttributes: [ cn ] (3)
  usersQuery:
    baseDN: "ou=users,dc=example,dc=com"
    scope: sub
    derefAliases: never
    filter: (objectclass=person)
    pageSize: 0
  userNameAttributes: [ mail ] (4)
  groupMembershipAttributes: [ "memberOf:1.2.840.113556.1.4.1941:" ] (5)
```

(1) ***groupsQuery*** 필터를 지정할 수 없습니다. ***groupsQuery*** 기본 DN 및 범위 값은 무시됩니다. ***groupsQuery***에서 유효한 ***derefAliases***를 설정해야 합니다.

(2) LDAP 서버에서 그룹을 고유하게 식별하는 속성입니다. ***dn***으로 설정해야 합니다.

(3) 그룹 이름으로 사용할 속성입니다.

(4) OpenShift Dedicated 그룹 레코드에서 사용자 이름으로 사용할 속성입니다. ***mail*** 또는 ***sAMAccountName***은 대부분의 설치에서 기본 선택 사항입니다.

(5) 멤버십 정보를 저장하는 사용자 속성입니다. ***LDAP_MATCHING_RULE_IN_CHAIN*** 사용에 유의하십시오.

사전 요구 사항

- 구성 파일을 생성합니다.

- ***dedicated-admin*** *역할의 사용자로 클러스터에 액세스할 수 있습니다.*

프로세스

- ***augmented_active_directory_config_nested.yaml*** *파일과의 동기화를 실행합니다.*

```
$ oc adm groups sync \
   'cn=admins,ou=groups,dc=example,dc=com' \
   --sync-config=augmented_active_directory_config_nested.yaml \
   --confirm
```

참고

cn=admins,ou=groups,dc=example,dc=com *그룹을 명시적으로 허용해야 합니다.*

OpenShift Dedicated는 위의 동기화 작업의 결과로 다음 그룹 레코드를 생성합니다.

augmented_active_directory_config_nested.yaml 파일을 사용하여 생성된 OpenShift Dedicated 그룹

```
apiVersion: user.openshift.io/v1
kind: Group
metadata:
  annotations:
    openshift.io/ldap.sync-time: 2015-10-13T10:08:38-0400 (1)
    openshift.io/ldap.uid: cn=admins,ou=groups,dc=example,dc=com (2)
    openshift.io/ldap.url: LDAP_SERVER_IP:389 (3)
  creationTimestamp:
  name: admins (4)
users: (5)
- jane.smith@example.com
- jim.adams@example.com
```

(1) *이 OpenShift Dedicated 그룹이 LDAP 서버와 마지막으로 동기화된 경우 ISO 6801 형식으로 되어 있습니다.*

(2) *LDAP 서버에서 그룹의 고유 식별자입니다.*

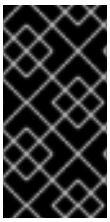
(3) *이 그룹의 레코드가 저장된 LDAP 서버의 IP 주소 및 호스트입니다.*

(4) *동기화 파일에서 지정한 그룹의 이름입니다.*

(5) *동기화 파일에서 지정한 대로 이름이 지정된 그룹 멤버에 해당하는 사용자입니다. 그룹 멤버십이 Microsoft Active Directory Server에 의해 평면화되었으므로 중첩 그룹의 멤버가 포함됩니다.*

14.5. LDAP 동기화 구성 사양

구성 파일의 오브젝트 사양은 다음과 같습니다. 필드는 스키마 오브젝트에 따라 달라집니다. 예를 들어, v1.ActiveDirectoryConfig에는 ***groupsQuery*** *필드가 없지만 v1.RFC2307Config 및 v1.AugmentedActiveDirectoryConfig에는 있습니다.*



중요

*바이너리 속성은 지원되지 않습니다. LDAP 서버에서 제공하는 모든 속성 데이터는 UTF-8 인코딩 문자열 형식이어야 합니다. 예를 들면 **objectGUID**와 같은 바이너리 속성을 ID 속성으로 사용하지 마십시오. 그 대신 **sAMAccountName** 또는 **userPrincipalName**과 같은 문자열 속성을 사용해야 합니다.*

14.5.1. v1.LDAPSyncConfig

***LDAPSyncConfig**에는 LDAP 그룹 동기화를 정의하는 데 필요한 구성 옵션이 있습니다.*

이름	설명	스키마
kind	이 오브젝트가 나타내는 REST 리소스에 해당하는 문자열 값입니다. 서버는 클라이언트에서 요청을 제출한 끝점에서 이를 유추할 수 있습니다. CamelCase로 업데이트할 수 없습니다. 자세한 내용은 https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#types-kinds를 참조하십시오.	문자열
apiVersion	버전이 지정된 이 오브젝트 표현의 스키마를 정의합니다. 서버는 인식된 스키마를 최신 내부 값으로 변환해야 하며, 인식되지 않는 값을 거부할 수 있습니다. 자세한 내용은 https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#resources를 참조하십시오.	문자열
url	호스트는 연결할 LDAP 서버의 스키마, 호스트 및 포트입니다 (**scheme://host:port**).	문자열
bindDN	LDAP 서버에 바인딩할 선택적 DN입니다.	문자열
bindPassword	검색 단계에서 바인딩할 선택적 암호입니다.	v1.StringSource

이름	설명	스키마
insecure	**true**인 경우 연결에서 TLS를 사용하지 않아야 함을 나타냅니다. **false**인 경우 **ldaps://** URL은 TLS를 사용하여 연결되고, **ldap://** URL은 https://tools.ietf.org/html/rfc2830에 지정된 StartTLS를 사용하여 TLS 연결로 업그레이드됩니다. **insecure**를 **true**로 설정하는 경우 **ldaps://** URL 스키마를 사용할 수 없습니다.	부울
ca	서버에 요청할 때 사용하는 신뢰할 수 있는 인증 기관 번들 옵션입니다. 비어있는 경우 기본 시스템 루트가 사용됩니다.	문자열
groupUIDNameMapping	LDAP 그룹 UID를 OpenShift Dedicated 그룹 이름에 직접 매핑하는 옵션입니다.	object
rfc2307	RFC2307과 유사한 방식으로 설정된 LDAP 서버에서 데이터를 추출하는 구성이 있습니다. 최상위 그룹 및 사용자 항목이 있고 멤버를 나열하는 그룹 항목 다중값 속성에 따라 그룹 멤버십이 결정됩니다.	v1.RFC2307Config
activeDirectory	Active Directory 방식과 유사한 방식으로 설정된 LDAP 서버에서 데이터를 추출하는 구성이 있습니다. 최상위 사용자 항목이 있고 멤버가 속한 그룹을 나열하는 멤버 다중값 속성에 따라 그룹 멤버십이 결정됩니다.	v1.ActiveDirectoryConfig
augmentedActiveDirectory	위에서 설명한 Active Directory 방식과 유사한 방식으로 설정된 LDAP 서버에서 데이터를 추출하는 구성이 있습니다. 단, 최상위 그룹 항목이 존재하고 여기에 그룹 멤버십이 아닌 메타데이터를 보관합니다.	v1.AugmentedActiveDirectoryConfig

14.5.2. v1.StringSource

***StringSource**를 사용하면 문자열을 인라인으로 지정하거나 환경 변수 또는 파일을 통해 외부에서 지정할 수 있습니다. 문자열 값만 포함하는 경우 간단한 JSON 문자열로 마샬링됩니다.*

이름	설명	스키마
value	일반 텍스트 값 또는 **keyFile**이 지정된 경우 암호화된 값을 지정합니다.	문자열
env	일반 텍스트 값을 포함하는 환경 변수 또는 **keyFile**이 지정된 경우 암호화된 값을 지정합니다.	문자열
file	일반 텍스트 값이 포함된 파일 또는 **keyFile**이 지정된 경우 암호화된 값을 참조합니다.	문자열
keyFile	값을 해독하는 데 사용할 키가 들어 있는 파일을 참조합니다.	문자열

14.5.3. v1.LDAPQuery

***LDAPQuery**에는 LDAP 쿼리를 빌드하는 데 필요한 옵션이 있습니다.*

이름	설명	스키마
baseDN	모든 검색을 시작하는 디렉터리 분기의 DN입니다.	문자열
scope	검색 범위 옵션입니다. **base**(기본 오브젝트만), **one**(기본 수준의 모든 오브젝트), **sub**(전체 하위 트리)를 사용할 수 있습니다. 설정하지 않는 경우 기본값은 **sub**입니다.	문자열
derefAliases	별칭과 관련된 검색의 선택적 동작입니다. **never**(별칭을 역참조하지 않음), **search**(검색에서만 역참조), **base**(기본 오브젝트를 찾을 때만 역참조), **always**(항상 역참조)가 될 수 있습니다. 설정하지 않는 경우 기본값은 **always**입니다.	문자열
timeout	응답 대기를 종료하기 전에 서버에 대한 요청을 처리 중 상태로 유지할 수 있는 시간(초)이 있습니다. 이 값이 **0**이면 클라이언트 쪽 제한이 적용되지 않습니다.	정수
filter	기본 DN이 있는 LDAP 서버에서 관련 항목을 모두 검색하는 유효한 LDAP 검색 필터입니다.	문자열

이름	설명	스키마
pageSize	LDAP 항목으로 측정한 최대 기본 페이지 크기입니다. 페이지 크기가 **0**이면 페이징이 수행되지 않습니다.	정수

14.5.4. v1.RFC2307Config

***RFC2307Config**에는 RFC2307 스키마를 사용하여 LDAP 그룹 동기화에서 LDAP 서버와 상호 작용하는 방식을 정의하는 데 필요한 구성 옵션이 있습니다.*

이름	설명	스키마
groupsQuery	그룹 항목을 반환하는 LDAP 쿼리용 템플릿이 있습니다.	v1.LDAPQuery
groupUIDAttribute	고유 식별자로 해석되는 LDAP 그룹 항목의 속성을 정의합니다 (**IdapGroupUID**).	문자열
groupNameAttributes	OpenShift Dedicated 그룹에 사용할 이름으로 해석되는 LDAP 그룹 항목의 속성을 정의합니다.	문자열 배열
groupMembershipAttributes	멤버로 해석되는 LDAP 그룹 항목의 속성을 정의합니다. 해당 속성에 포함된 값을 **UserUIDAttribute**로 쿼리할 수 있어야 합니다.	문자열 배열
usersQuery	사용자 항목을 반환하는 LDAP 쿼리용 템플릿이 있습니다.	v1.LDAPQuery
userUIDAttribute	고유 식별자로 해석되는 LDAP 사용자 항목의 속성을 정의합니다. **GroupMembershipAttributes**에 있는 값과 일치해야 합니다.	문자열

이름	설명	스키마
userNameAttributes	OpenShift Dedicated 사용자 이름으로 사용할 LDAP 사용자 항목의 속성을 순서대로 정의합니다. 값이 비어 있지 않은 첫 번째 속성이 사용됩니다. 이 값은 **LDAPPasswordIdentityProvider**의 **PreferredUsername** 설정과 일치해야 합니다. OpenShift Dedicated 그룹 레코드에서 사용자 이름으로 사용할 속성입니다. **mail** 또는 **sAMAccountName**은 대부분의 설치에서 기본 선택 사항입니다.	문자열 배열
tolerateMemberNotFoundErrors	누락된 사용자 항목이 있는 경우 LDAP 동기화 작업의 동작을 결정합니다. **true**인 경우 찾지 못한 사용자에 대한 LDAP 쿼리가 허용되며, 오류만 기록됩니다. **false** 인 경우 사용자 쿼리에서 아무것도 찾지 못하면 LDAP 동기화 작업이 실패합니다. 기본값은 **false**입니다. 이 플래그를 **true**로 설정하여 LDAP 동기화 작업을 잘못 구성하면 그룹 멤버십이 제거될 수 있으므로 이 플래그를 주의해서 사용하십시오.	부울
tolerateMemberOutOfScopeErrors	범위를 벗어난 사용자 항목이 있는 경우 LDAP 동기화 작업의 동작을 결정합니다. **true**인 경우 모든 사용자 쿼리에 지정된 기본 DN을 벗어나는 사용자의 LDAP 쿼리가 허용되며 오류만 기록됩니다. **false**인 경우 사용자 쿼리가 모든 사용자 쿼리에서 지정하는 기본 DN 외부에서 검색하면 LDAP 동기화 작업이 실패합니다. 이 플래그를 **true**로 설정하여 LDAP 동기화 작업을 잘못 구성하면 그룹에서 사용자가 누락될 수 있으므로 이 플래그를 주의해서 사용하는 것이 좋습니다.	부울

14.5.5. v1.ActiveDirectoryConfig

ActiveDirectoryConfig*에는 Active Directory 스키마를 사용하여 LDAP 그룹 동기화에서 LDAP 서버와 상호 작용하는 방식을 정의하는 데 필요한 구성 옵션이 있습니다.*

이름	설명	스키마
usersQuery	사용자 항목을 반환하는 LDAP 쿼리용 템플릿이 있습니다.	v1.LDAPQuery
userNameAttributes	OpenShift Dedicated 사용자 이름으로 해석되는 LDAP 사용자 항목의 속성을 정의합니다. OpenShift Dedicated 그룹 레코드에서 사용자 이름으로 사용할 속성입니다. **mail** 또는 **sAMAccountName**은 대부분의 설치에서 기본 선택 사항입니다.	문자열 배열
groupMembershipAttributes	멤버가 속한 그룹으로 해석되는 LDAP 사용자 항목의 속성을 정의합니다.	문자열 배열

14.5.6. v1.AugmentedActiveDirectoryConfig

AugmentedActiveDirectoryConfig*에는 보강된 Active Directory 스키마를 사용하여 LDAP 그룹 동기화에서 LDAP 서버와 상호 작용하는 방식을 정의하는 데 필요한 구성 옵션이 있습니다.*

이름	설명	스키마
usersQuery	사용자 항목을 반환하는 LDAP 쿼리용 템플릿이 있습니다.	v1.LDAPQuery
userNameAttributes	OpenShift Dedicated 사용자 이름으로 해석되는 LDAP 사용자 항목의 속성을 정의합니다. OpenShift Dedicated 그룹 레코드에서 사용자 이름으로 사용할 속성입니다. **mail** 또는 **sAMAccountName**은 대부분의 설치에서 기본 선택 사항입니다.	문자열 배열
groupMembershipAttributes	멤버가 속한 그룹으로 해석되는 LDAP 사용자 항목의 속성을 정의합니다.	문자열 배열
groupsQuery	그룹 항목을 반환하는 LDAP 쿼리용 템플릿이 있습니다.	v1.LDAPQuery
groupUIDAttribute	고유 식별자로 해석되는 LDAP 그룹 항목의 속성을 정의합니다 (**IdapGroupUID**).	문자열

이름	설명	스키마
groupNameAttributes	OpenShift Dedicated 그룹에 사용할 이름으로 해석되는 LDAP 그룹 항목의 속성을 정의합니다.	문자열 배열