



# Red Hat OpenStack Platform 13

## ID 서비스와 통합

Active Directory 또는 Red Hat Identity Management를 외부 인증 백엔드로 사용



## Red Hat OpenStack Platform 13 ID 서비스와 통합

---

Active Directory 또는 Red Hat Identity Management를 외부 인증 백엔드로 사용

OpenStack Team  
rhos-docs@redhat.com

## 법적 공지

Copyright © 2023 Red Hat, Inc.

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

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

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

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

## 초록

Active Directory 또는 Red Hat Identity Management를 외부 인증 백엔드로 사용합니다.

## 차례

보다 포괄적 수용을 위한 오픈 소스 용어 교체 .....	3
<b>1장. ACTIVE DIRECTORY 통합</b> .....	<b>4</b>
1.1. 주요 용어	4
1.2. 가정	4
1.3. 영향 정책	4
1.4. ACTIVE DIRECTORY 도메인 서비스 구성	5
1.5. LDAPS 인증서 설정	6
1.6. ID 서비스 구성	7
1.7. DOMAIN(도메인) 탭에 대한 액세스 권한 부여	17
1.8. 새 프로젝트 생성	18
1.9. 대시보드 로그인 프로세스 변경CHANGES TO THE DASHBOARD LOG IN PROCESS	18
1.10. 명령줄 변경CHANGES TO THE COMMAND LINE	18
1.11. 테스트 AD DS 통합	19
1.12. 관리자가 아닌 사용자에게 대한 RC 파일 생성	19
1.13. 문제 해결	20
<b>2장. IDENTITY MANAGEMENT INTEGRATION</b> .....	<b>22</b>
2.1. IDM 서버 설정	23
2.2. LDAPS 인증서 설정	24
2.3. ID 서비스 구성	25
2.4. DOMAIN(도메인) 탭에 대한 액세스 권한 부여	35
2.5. 새 프로젝트 생성	35
2.6. 관리자가 아닌 사용자에게 대한 RC 파일 생성	37
2.7. 문제 해결	37
<b>3장. NOVAJOIN을 사용하여 IDM과 통합</b> .....	<b>39</b>
3.1. 언더클라우드에 NOVAJOIN 설치 및 설정	39
3.2. 오버클라우드에 NOVAJOIN 설치 및 설정	42
3.3. IDM에서 노드 검증	44
3.4. NOVAJOIN의 DNS 항목 구성	44
<b>4장. DIRECTOR와 함께 도메인별 LDAP 백엔드 사용</b> .....	<b>46</b>
4.1. 구성 옵션 설정	46
4.2. DIRECTOR 배포 설정	46



## 보다 포괄적 수용을 위한 오픈 소스 용어 교체

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

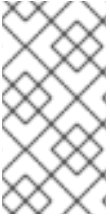
### ID 서비스

ID 서비스(codename *keystone*)는 Red Hat OpenStack Platform 13에 대한 인증 및 권한 부여를 제공합니다.

이 가이드에서는 ID 서비스를 Microsoft Active Directory 도메인 서비스(AD DS), Red Hat IdM(Identity Management) 및 LDAP와 통합하는 방법을 설명합니다.

# 1장. ACTIVE DIRECTORY 통합

이 장에서는 ID 서비스(keystone)를 Active Directory 도메인 서비스와 통합하는 방법을 설명합니다. 이 사용 사례에서 ID 서비스는 특정 AD DS(Active Directory 도메인 서비스) 사용자를 인증하는 동시에 ID 서비스 데이터베이스에 권한 부여 설정 및 중요한 서비스 계정을 유지합니다. 결과적으로 ID 서비스는 사용자 계정 인증을 위해 AD DS에 대한 읽기 전용 액세스 권한을 가지고 인증된 계정에 할당된 권한에 대한 관리를 유지합니다.



## 참고

director를 사용하는 경우 [4장. director와 함께 도메인별LDAP 백엔드 사용](#) 을 참조하십시오. 아래에 참조되는 구성 파일이 Puppet에서 관리되기 때문입니다. 결과적으로 **openstack overcloud deploy** 프로세스를 실행할 때마다 추가하는 사용자 정의 설정을 덮어쓸 수 있습니다.

## 1.1. 주요 용어

- **인증** - 암호를 사용하여 사용자가 원하는 사람인지 확인하는 프로세스입니다.
- **권한 부여** - 인증된 사용자에게 액세스하려는 리소스에 대한 적절한 권한이 있음을 확인합니다.
- **도메인** - 이 용어는 AD DS 도메인과 같지 않으며 대신 사용자, 그룹 및 프로젝트를 파티셔닝하기 위해 ID 서비스에 구성된 추가 네임스페이스를 나타냅니다. 이러한 별도의 도메인은 다른 LDAP 또는 AD DS 환경에서 사용자를 인증하도록 구성할 수 있습니다.

## 1.2. 가정

이 예제 배포에서는 다음과 같은 가정을 수행합니다.

- Active Directory Domain Services 구성 및 운영
- Red Hat OpenStack Platform이 구성 및 작동합니다.
- DNS 이름 확인이 완전히 작동하며 모든 호스트가 적절하게 등록됩니다.
- AD DS 인증 트래픽은 포트 636을 사용하여 LDAPS로 암호화됩니다.

## 1.3. 영향 정책

이러한 단계를 통해 AD DS 사용자는 OpenStack에 인증하고 리소스에 액세스할 수 있습니다. OpenStack 서비스 계정(예: keystone 및 glance) 및 권한 관리(권한, 역할, 프로젝트)은 ID 서비스 데이터베이스에 남아 있습니다. 권한 및 역할은 ID 서비스 관리 도구를 사용하여 AD DS 계정에 할당됩니다.

### 1.3.1. 고가용성 옵션

이 구성은 단일 Active Directory 도메인 컨트롤러의 가용성에 대한 종속성을 생성합니다. ID 서비스를 AD 도메인 컨트롤러에 인증할 수 없는 경우 프로젝트 사용자는 영향을 받습니다. 이러한 위험을 관리하는 데 사용할 수 있는 다양한 옵션을 사용할 수 있습니다. 예를 들어 개별 AD 도메인 컨트롤러가 아닌 DNS 별칭 또는 로드 밸런싱 어플라이언스를 쿼리하도록 ID 서비스를 구성할 수 있습니다. 다른 도메인 컨트롤러를 쿼리하도록 keystone을 구성할 수도 있습니다. 하나를 사용할 수 없게 됩니다.

### 1.3.2. 중단 요구 사항



- AD DS 백엔드를 추가하려면 ID 서비스를 다시 시작해야 합니다.
- keystone v3으로 전환하려면 모든 노드의 Compute 서비스를 다시 시작해야 합니다.
- AD DS에서 계정을 만들 때까지 사용자는 대시보드에 액세스할 수 없습니다. 다운타임을 줄이려면 이 변경 전에 AD DS 계정을 미리 태그하는 것이 좋습니다.

### 1.3.3. 방화벽 구성

방화벽이 AD DS와 OpenStack 간의 트래픽을 필터링하는 경우 다음 포트를 통해 액세스를 허용해야 합니다.

소스	대상	유형	포트
OpenStack 컨트롤러 노드	Active Directory 도메인 컨트롤러	LDAPS	TCP 636

## 1.4. ACTIVE DIRECTORY 도메인 서비스 구성

이 섹션에서는 Active Directory 관리자가 완료해야 하는 작업에 대해 설명합니다.

표 1.1. 구성 단계

Task	세부 정보
서비스 계정을 생성합니다.	서비스 계정의 이름 지정 규칙에 따라 이름을 지정할 수 있습니다(예: <b>svc-ldap</b> ). 일반 도메인 사용자 계정일 수 있습니다. 관리자 권한이 필요하지 않습니다.
사용자 그룹을 생성합니다.	사용자가 OpenStack에 액세스해야 하는 경우 이 그룹의 멤버여야 합니다. 사용자 그룹의 명명 규칙에 따라 이름을 지정할 수 있습니다(예: <b>grp-openstack</b> ). 이 그룹의 멤버도 프로젝트 그룹의 멤버인 경우 대시보드에서 프로젝트에 대한 액세스 권한을 부여할 수 있습니다.
프로젝트 그룹을 생성합니다.	각 OpenStack 프로젝트에는 해당 AD 그룹이 필요합니다. 예를 들어 <b>grp-openstack-demo</b> 및 <b>grp-openstack-admin</b> .
서비스 계정을 구성합니다.	서비스 계정 <b>svc-ldap</b> 는 <b>grp-openstack</b> 그룹의 멤버여야 합니다.
LDAPS 공개 키를 내보냅니다.	공개 키(개인 키가 아님)를 <b>DER 인코딩 x509.cer</b> 파일로 내보냅니다.
키를 OpenStack 관리자에게 보냅니다.	OpenStack 관리자는 이 키를 사용하여 OpenStack과 Active Directory 간의 LDAPS 통신을 암호화합니다.

AD DS 도메인의 NetBIOS 이름을 검색합니다.	OpenStack 관리자는 이 이름을 Keystone 도메인에 사용하므로 환경 간에 일관된 도메인 이름을 지정할 수 있습니다.
-------------------------------	--

예를 들어 다음 절차에서는 Active Directory 도메인 컨트롤러에서 실행되는 PowerShell 명령을 보여줍니다.

1. LDAP 조회 계정을 생성합니다. 이 계정은 ID 서비스에서 AD DS LDAP 서비스를 쿼리하는 데 사용됩니다.

```
PS C:\> New-ADUser -SamAccountName svc-ldap -Name "svc-ldap" -GivenName LDAP -
Surname Lookups -UserPrincipalName svc-ldap@lab.local -Enabled $false -
PasswordNeverExpires $true -Path 'OU=labUsers,DC=lab,DC=local'
```

2. 이 계정의 암호를 설정한 다음 활성화합니다. AD 도메인의 복잡성 요구 사항을 준수하는 암호를 지정하라는 메시지가 표시됩니다.

```
PS C:\> Set-ADAccountPassword svc-ldap -PassThru | Enable-ADAccount
```

3. **grp-openstack** 이라는 OpenStack 사용자를 위한 그룹을 생성합니다.

```
PS C:\> NEW-ADGroup -name "grp-openstack" -groupscope Global -path
"OU=labUsers,DC=lab,DC=local"
```

4. 프로젝트 그룹을 생성합니다.

```
PS C:\> NEW-ADGroup -name "grp-openstack-demo" -groupscope Global -path
"OU=labUsers,DC=lab,DC=local"
PS C:\> NEW-ADGroup -name "grp-openstack-admin" -groupscope Global -path
"OU=labUsers,DC=lab,DC=local"
```

5. **svc-ldap** 사용자를 **grp-openstack** 그룹에 추가합니다.

```
PS C:\> ADD-ADGroupMember "grp-openstack" -members "svc-ldap"
```

6. AD 도메인 컨트롤러에서 **Certificates MMC** 를 사용하여 LDAPS 인증서의 공개 키(개인 키가 아님)를 DER 인코딩 **x509.cer** 파일로 내보냅니다. 이 파일을 OpenStack 관리자에게 보냅니다.

7. AD DS 도메인의 NetBIOS 이름을 검색합니다.

```
PS C:\> Get-ADDomain | select NetBIOSName
NetBIOSName
-----
LAB
```

이 값을 OpenStack 관리자에게 보냅니다.

## 1.5. LDAPS 인증서 설정



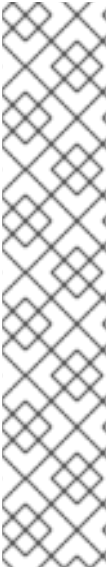
## 참고

LDAP 인증을 위해 여러 도메인을 사용하는 경우 권한이 부여된 프로젝트를 검색할 수 없거나 Peer의 인증서 발급자가 인식되지 않는 것과 같은 다양한 오류가 발생할 수 있습니다. 이는 keystone이 특정 도메인에 대해 잘못된 인증서를 사용하는 경우 발생할 수 있습니다. 이 문제를 해결하려면 모든 LDAPS 공개 키를 단일 .crt 번들에 병합하고 이 파일을 사용하여 모든 keystone 도메인을 구성합니다.

Keystone은 LDAPS 쿼리를 사용하여 사용자 계정의 유효성을 검사합니다. 이 트래픽을 암호화하기 위해 keystone은 **keystone.conf** 에서 정의한 인증서 파일을 사용합니다. 이 절차에서는 Active Directory에서 수신한 공개 키를 .crt 형식으로 변환하고 keystone이 해당 키를 참조할 수 있는 위치로 복사합니다.

1. LDAPS 공개 키를 OpenStack Identity(keystone)를 실행하는 노드에 복사하고 .cer 를 .crt 로 변환합니다. 이 예에서는 **addc.lab.local.cer** 라는 소스 인증서 파일을 사용합니다.

```
# openssl x509 -inform der -in addc.lab.local.cer -out addc.lab.local.crt
# cp addc.lab.local.crt /etc/pki/ca-trust/source/anchors
```



## 참고

필요한 경우 **ldapsearch** 와 같은 진단 명령을 실행해야 하는 경우 RHEL 인증서 저장소에 인증서를 추가해야 합니다.

1. .cer 를 .pem 으로 변환합니다. 이 예에서는 **addc.lab.local.cer** 라는 소스 인증서 파일을 사용합니다.

```
# openssl x509 -inform der -in addc.lab.local.cer -out addc.lab.local.pem
```

2. OpenStack 컨트롤러에 .pem 을 설치합니다. 예를 들어 Red Hat Enterprise Linux 의 경우:

```
# cp addc.lab.local.pem /etc/pki/ca-trust/source/anchors/
# update-ca-trust
```

## 1.6. ID 서비스 구성

이러한 단계는 AD DS와의 통합을 위해 ID 서비스(keystone)를 준비합니다.



## 참고

director를 사용하는 경우 아래에 참조되는 구성 파일은 Puppet에서 관리합니다. 결과적으로 **openstack overcloud deploy** 프로세스를 실행할 때마다 추가하는 사용자 정의 설정을 덮어쓸 수 있습니다. director 기반 배포에 이러한 설정을 적용하려면 [4장. director와 함께 도메인별 LDAP 백엔드 사용](#) 을 참조하십시오.

### 1.6.1. 컨트롤러 구성



## 참고

구성 파일을 업데이트하려면 특정 OpenStack 서비스가 컨테이너 내에서 실행되고, 이 서비스는 keystone, nova, cinder에 적용됩니다. 따라서 다음을 고려해야 할 특정 관리 사례가 있습니다.

- 물리적 노드의 호스트 운영 체제(예: `/etc/cinder/cinder.conf`)에 있는 구성 파일을 업데이트하지 마십시오. 컨테이너화된 서비스에서 이 파일을 참조하지 않기 때문입니다.
- 컨테이너 내에서 실행 중인 구성 파일을 업데이트하지 마십시오. 컨테이너를 다시 시작하면 변경 사항이 손실되기 때문입니다.  
대신 컨테이너화된 서비스에 변경 사항을 추가해야 하는 경우 컨테이너를 생성하는 데 사용되는 구성 파일을 업데이트해야 합니다. 이러한 값은 `/var/lib/config-data/puppet-generated/`에 저장됩니다.

예를 들면 다음과 같습니다.

- keystone: `/var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf`
- cinder: `/var/lib/config-data/puppet-generated/cinder/etc/cinder/cinder.conf`
- nova: `/var/lib/config-data/puppet-generated/nova/etc/nova/nova.conf`  
그런 다음 컨테이너를 다시 시작하면 변경 사항이 적용됩니다. 예: `sudo docker restart keystone`

keystone 서비스를 실행하는 각 OpenStack 노드에서 다음 절차를 수행합니다.

1. SELinux를 구성합니다.

```
# setsebool -P authlogin_nsswitch_use_ldap=on
```

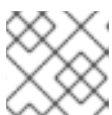
출력에는 다음과 유사한 메시지가 포함될 수 있습니다. 그들은 무시될 수 있습니다:

```
Full path required for exclude: net:[4026532245].
```

2. 도메인 디렉토리를 생성합니다.

```
# mkdir /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/  
# chown 42425:42425 /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/
```

3. 여러 백엔드를 사용하도록 keystone을 구성합니다.



## 참고

`yum install crudini` 를 사용하여 `crudini`를 설치해야 할 수 있습니다.

```
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf  
identity domain_specific_drivers_enabled true  
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf  
identity domain_config_dir /etc/keystone/domains  
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf  
assignment driver sql
```



### 참고

director를 사용하는 경우 `/var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf` 는 Puppet에서 관리합니다. 결과적으로 **openstack overcloud deploy** 프로세스를 실행할 때마다 추가하는 사용자 정의 설정을 덮어쓸 수 있습니다. 따라서 매번 이 구성을 수동으로 다시 추가해야 할 수 있습니다. director 기반 배포의 경우 4장. [director와 함께 도메인별 LDAP 백엔드 사용](#) 에서 참조하십시오.

4. 대시보드에서 여러 도메인을 활성화합니다. `/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings` 에 다음 행을 추가합니다.

```
OPENSTACK_API_VERSIONS = {
    "identity": 3
}
OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = 'Default'
```



### 참고

director를 사용하는 경우 `/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings` 가 Puppet에서 관리합니다. 결과적으로 **openstack overcloud deploy** 프로세스를 실행할 때마다 추가하는 사용자 정의 설정을 덮어쓸 수 있습니다. 따라서 매번 이 구성을 수동으로 다시 추가해야 할 수 있습니다.

horizon 컨테이너를 다시 시작하여 설정을 적용합니다.

```
$ sudo docker restart horizon
```

5. 추가 백엔드를 구성합니다.  
이 예제에서 **LAB** 은 ID 서비스 도메인으로 사용할 NetBIOS 이름입니다.

- a. AD DS 통합을 위한 keystone 도메인을 만듭니다.

이전에 검색한 NetBIOS 이름 값을 도메인 이름으로 사용합니다. 이 방법을 사용하면 로그인 프로세스 중에 사용자에게 일관된 도메인 이름을 표시할 수 있습니다. 예를 들어 NetBIOS 이름이 **LAB** 인 경우:

```
$ openstack domain create LAB
```



### 참고

이 명령을 사용할 수 없는 경우 # 소스 **overcloudrc-v3**를 실행하여 명령줄 세션에 **keystone v3** 을 활성화했는지 확인합니다.

- b. 구성 파일을 생성합니다.

AD DS 백엔드를 추가하려면 `/var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf` 라는 새 파일에 LDAP 설정을 입력합니다. **AD DS** 배포에 맞게 아래 샘플 설정을 편집해야 합니다.

```
[ldap]
url          = ldaps://addc.lab.local:636
user         = CN=svc-ldap,OU=labUsers,DC=lab,DC=local
password     = RedactedComplexPassword
suffix       = DC=lab,DC=local
user_tree_dn = OU=labUsers,DC=lab,DC=local
user_objectclass = person
user_filter  = (|(memberOf=cn=grp-openstack,OU=labUsers,DC=lab,DC=local)
(memberOf=cn=grp-openstack-admin,OU=labUsers,DC=lab,DC=local)
(memberOf=memberOf=cn=grp-openstack-demo,OU=labUsers,DC=lab,DC=local))
user_id_attribute = sAMAccountName
user_name_attribute = sAMAccountName
user_mail_attribute = mail
user_pass_attribute =
user_enabled_attribute = userAccountControl
user_enabled_mask = 2
user_enabled_default = 512
user_attribute_ignore = password,tenant_id,tenants
group_objectclass = group
group_tree_dn = OU=labUsers,DC=lab,DC=local
group_filter = (CN=grp-openstack*)
group_id_attribute = cn
group_name_attribute = name
use_tls = False
tls_cacertfile = /etc/pki/ca-trust/source/anchors/anchorsaddc.lab.local.pem

query_scope = sub
chase_referrals = false

[identity]
driver = ldap
```

각 설정에 대한 설명:

설정	설명
url	인증에 사용할 AD 도메인 컨트롤러입니다. LDAPS 포트 <b>636</b> 을 사용합니다.
user	LDAP 쿼리에 사용할 AD 계정의 <i>Distinguished</i> 이름입니다. 예를 들어 <b>Get-ADuser svc-ldap   DistinguishedName</b> 을 사용하여 AD에서 <b>svc-ldap</b> 계정의 <b>Distinguished Name</b> 값을 찾을 수 있습니다.
암호	위에서 사용된 AD 계정의 일반 텍스트 암호입니다.

설정	설명
<b>suffix</b>	AD 도메인의 <i>Distinguished</i> 이름입니다. <b>Get-ADDomain   select DistinguishedName</b> 을 사용하여 이 값을 찾을 수 있습니다.
<b>user_tree_dn</b>	OpenStack 계정이 포함된 조직 <i>단위</i> (OU)입니다.
<b>user_objectclass</b>	LDAP 사용자의 유형을 정의합니다. AD의 경우 <b>person</b> 유형을 사용합니다.
<b>user_filter</b>	ID 서비스에 표시되는 사용자를 필터링합니다. 결과적으로 <b>grp-openstack</b> 그룹의 멤버만 ID 서비스에 정의된 권한을 가질 수 있습니다. 이 값은 그룹의 전체 <i>Distinguished Name</i> : <b>Get-ADGroup grp-openstack   DistinguishedName</b> 을 선택합니다.
<b>user_id_attribute</b>	사용자 ID에 사용할 AD 값을 매핑합니다.
<b>user_name_attribute</b>	<i>이름</i> 에 사용할 AD 값을 매핑합니다.
<b>user_mail_attribute</b>	사용자 이메일 주소에 사용할 AD 값을 매핑합니다.
<b>user_pass_attribute</b>	이 값을 비워 둡니다.
<b>user_enabled_attribute</b>	계정 활성화 여부를 확인하는 AD 설정입니다.
<b>user_enabled_mask</b>	계정이 활성화되어 있는지 여부를 확인하는 값을 정의합니다. 부울이 반환되지 않는 경우 사용되지 않습니다.
<b>user_enabled_default</b>	계정이 활성화되었음을 나타내는 AD 값입니다.
<b>user_attribute_ignore</b>	ID 서비스에서 무시해야 하는 사용자 속성을 정의합니다.
<b>group_objectclass</b>	<i>그룹</i> 에 사용할 AD 값을 매핑합니다.
<b>group_tree_dn</b>	사용자 그룹을 포함하는 조직 <i>단위</i> (OU)입니다.
<b>group_filter</b>	ID 서비스에 표시된 그룹을 필터링합니다.
<b>group_id_attribute</b>	그룹 ID에 사용할 AD 값을 매핑합니다.
<b>group_name_attribute</b>	그룹 이름에 사용할 AD 값을 매핑합니다.

설정	설명
<b>use_tls</b>	TLS를 사용할지 여부를 정의합니다. STARTTLS 대신 LDAPS로 암호화하는 경우 이를 비활성화해야 합니다.
<b>tls_cacertfile</b>	.crt 인증서 파일의 경로를 지정합니다.
<b>query_scope</b>	<b>grp-openstack</b> 그룹의 멤버인 사용자를 찾을 때 중첩된 하위 OU 내에서도 검색하도록 ID 서비스를 구성합니다.
<b>chase_referrals</b>	<b>false</b> 로 설정하면 <b>python-ldap</b> 가 익명 액세스의 모든 추천을 반영하지 않도록 합니다.

- 구성 파일의 소유권을 **keystone** 사용자로 변경합니다.

```
# chown 42425:42425 /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf
```

- keystone** 서비스를 다시 시작하여 변경 사항을 적용합니다.

```
# sudo docker restart keystone
```

- admin** 사용자에게 도메인에 대한 액세스 권한을 부여합니다.



참고

이 경우 **OpenStack admin** 계정에 실제 **AD DS** 도메인에 대한 권한이 부여되지 않습니다. 이 경우 **domain** 이라는 용어는 **OpenStack**의 **keystone** 도메인 사용을 나타냅니다.

- LAB** 도메인의 ID 를 가져옵니다.

```
# openstack domain show LAB
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
```



```
| id | 6800b0496429431ab1c4efbb3fe810d4 |
| name | LAB |
+-----+-----+
```

b.

**admin** 사용자의 ID 값을 가져옵니다.

```
# openstack user list --domain default | grep admin
| 3d75388d351846c6a880e53b2508172a | admin |
```

c.

**admin** 역할의 ID 값을 가져옵니다.

```
# openstack role list
+-----+-----+
| ID | Name |
+-----+-----+
| 544d48aaffde48f1b3c31a52c35f01f9 | SwiftOperator |
| 6d005d783bf0436e882c55c62457d33d | ResellerAdmin |
| 785c70b150ee4c778fe4de088070b4cf | admin |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_ |
+-----+-----+
```

d.

반환된 도메인 및 **admin** ID를 사용하여 **admin** 사용자를 **keystone LAB** 도메인의 **admin** 역할에 추가하는 명령을 구성합니다.

```
# openstack role add --domain 6800b0496429431ab1c4efbb3fe810d4 --user
3d75388d351846c6a880e53b2508172a 785c70b150ee4c778fe4de088070b4cf
```

e.

명령에 **NetBIOS** 이름을 추가하여 **AD DS** 도메인의 사용자 목록을 확인합니다.



참고

재부팅 또는 서비스를 다시 시작한 후 **LDAP**를 쿼리할 수 있는 시간이 걸릴 수 있습니다.

```
# openstack user list --domain LAB
```

f.

로컬 **ID** 서비스 데이터베이스에서 서비스 계정을 확인합니다.

```
# openstack user list --domain default
```

## 1.6.2. Active Directory 그룹 구성원이 프로젝트에 액세스할 수 있도록 허용

인증된 사용자가 **OpenStack** 리소스에 액세스할 수 있도록 하려면 특정 **Active Directory** 그룹에 프로젝트에 대한 액세스 권한을 부여하도록 권한을 부여하는 것이 좋습니다. 이렇게 하면 **OpenStack** 관리자가 각 사용자를 프로젝트의 역할에 할당할 필요가 없습니다. 대신 **Active Directory** 그룹에 프로젝트에 역할이 부여됩니다. 따라서 이러한 **Active Directory** 그룹의 멤버인 **Active Directory** 사용자는 사전 결정된 프로젝트에 액세스할 수 있습니다.



### 참고

개별 **Active Directory** 사용자의 권한을 수동으로 관리하려면 [1.6.3절. “Active Directory 사용자가 프로젝트에 액세스할 수 있도록 허용”](#) 을 참조하십시오.

이 섹션에서는 **Active Directory** 관리자가 이미 다음 단계를 완료했다고 가정합니다.

- **Active Directory**에서 **grp-openstack-admin** 이라는 그룹을 만듭니다.
- **Active Directory**에 **grp-openstack-demo** 라는 그룹을 만듭니다.
- 필요에 따라 위의 그룹 중 하나에 **Active Directory** 사용자를 추가합니다.
- **Active Directory** 사용자를 **grp-openstack** 그룹에 추가합니다.
- 지정된 프로젝트를 고려하십시오. 이 예에서는 **openstack project create --domain default --description "Demo Project"** 데모를 사용하여 생성된 데모 프로젝트를 사용합니다.

이러한 단계는 **AD** 그룹에 역할을 할당합니다. 그러면 그룹 멤버가 **OpenStack** 리소스에 액세스할 수 있는 권한이 있습니다.

1. **AD** 그룹 목록을 검색합니다.

```
# openstack group list --domain LAB
+-----+-----+
| ID                                     | Name           |
+-----+-----+
| 185277be62ae17e498a69f98a59b66934fb1d6b7f745f14f5f68953a665b8851 | grp-
```

```

openstack |
| a8d17f19f464c4548c18b97e4aa331820f9d3be52654aa8094e698a9182cbb88 | grp-
openstack-admin |
| d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8 | grp-
openstack-demo |
+-----+-----+

```

2.

역할 목록을 검색합니다.

```

# openstack role list
+-----+-----+
| ID                | Name          |
+-----+-----+
| 0969957bce5e4f678ca6cef00e1abf8a | ResellerAdmin |
| 1fcb3c9b50aa46ee8196aaaecc2b76b7 | admin          |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_      |
| d3570730eb4b4780a7fed97eba197e1b | SwiftOperator |
+-----+-----+

```

3.

이러한 역할 중 하나 이상에 추가하여 **Active Directory** 그룹에 프로젝트에 대한 액세스 권한을 부여합니다. 예를 들어 **grp-openstack-demo** 그룹의 사용자가 **demo** 프로젝트의 일반 사용자가 되도록 하려면 **\_member\_** 역할에 그룹을 추가해야 합니다.

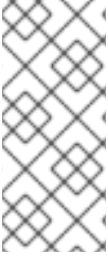
```

# openstack role add --project demo --group
d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8 _member_

```

결과적으로 **grp-openstack-demo**의 멤버는 **AD DS** 사용자 이름과 암호를 입력하여 대시보드에 로그인할 수 있으며 **Domain(도메인)** 필드에 **LAB** 을 입력합니다.

The image shows a login interface with a dark background. It contains three input fields: 'Domain' with the text 'LAB', 'User Name' with the text 'user1', and 'Password' with a masked password represented by dots. A blue 'Connect' button is located at the bottom right of the form.



## 참고

사용자에게 **Error: Unable to retrieve container list** 가 표시되면 컨테이너 목록을 검색할 수 있으며 컨테이너를 관리할 수 있을 것으로 예상되는 경우 **SwiftOperator** 역할에 추가해야 합니다.

### 1.6.3. Active Directory 사용자가 프로젝트에 액세스할 수 있도록 허용

**grp-openstack AD** 그룹의 멤버인 **AD DS** 사용자는 대시보드에서 **프로젝트에** 로그인할 수 있는 권한을 부여할 수 있습니다.

1.

**AD** 사용자 목록을 검색합니다.

```
# openstack user list --domain LAB
+-----+-----+
| ID                                     | Name          |
+-----+-----+
| 1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e | user1         |
| 12c062fadddc5f8b065434d9ff6fce03eb9259537c93b411224588686e9a38bf1 | user2         |
| afaf48031eb54c3e44e4cb0353f5b612084033ff70f63c22873d181fdae2e73c | user3         |
| e47fc21dcf0d9716d2663766023e2d8dc15a6d9b01453854a898cabb2396826e | user4         |
|                                     |               |
+-----+-----+
```

2.

역할 목록을 검색합니다.

```
# openstack role list
+-----+-----+
| ID                                     | Name          |
+-----+-----+
| 544d48aaffde48f1b3c31a52c35f01f9 | SwiftOperator |
| 6d005d783bf0436e882c55c62457d33d | ResellerAdmin |
| 785c70b150ee4c778fe4de088070b4cf | admin         |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_     |
+-----+-----+
```

3.

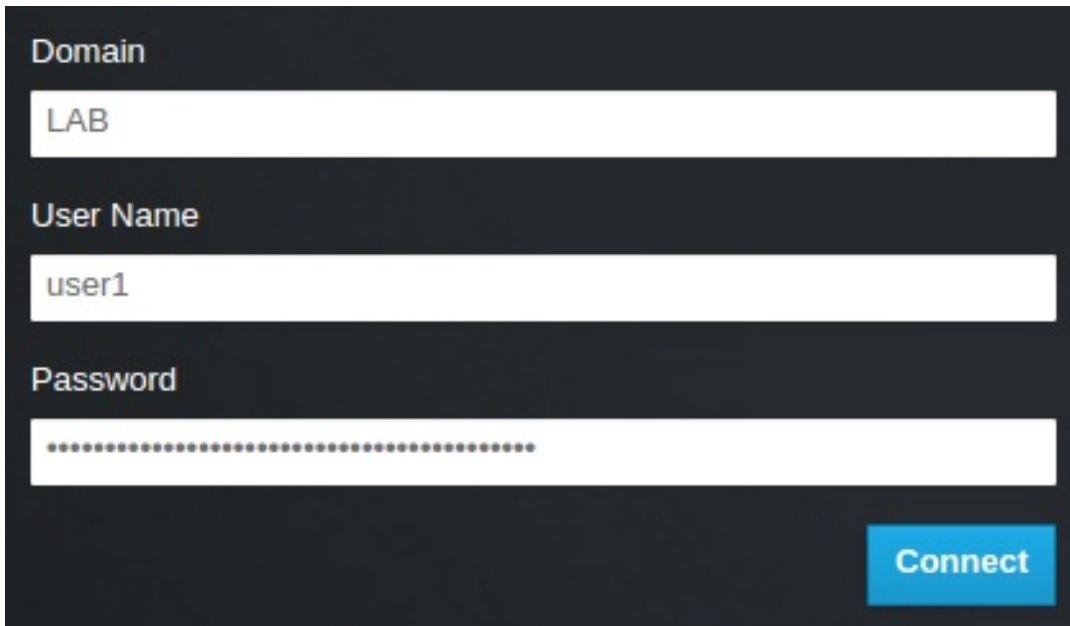
이러한 역할 중 하나 이상에 추가하여 사용자에게 프로젝트에 대한 액세스 권한을 부여합니다. 예를 들어 **user1** 이 **demo** 프로젝트의 일반 사용자가 되도록 하려면 **member** 역할에 추가합니다.

```
# openstack role add --project demo --user
1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e _member_
```

또는 **user1** 이 **demo** 프로젝트의 관리자인 경우 **admin** 역할에 추가합니다.

```
# openstack role add --project demo --user
1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e admin
```

결과적으로 **user1** 은 도메인 필드에 **LAB** 을 입력하는 경우 **AD DS** 사용자 이름과 암호를 입력하여 대시보드에 로그인할 수 있습니다.



The screenshot shows a dark-themed login interface. It has three input fields: 'Domain' containing 'LAB', 'User Name' containing 'user1', and 'Password' which is masked with a series of dots. A blue 'Connect' button is located at the bottom right of the form.



#### 참고

사용자에게 **Error: Unable to retrieve container list** 가 표시되면 컨테이너 목록을 검색할 수 있으며 컨테이너를 관리할 수 있을 것으로 예상되는 경우 **SwiftOperator** 역할에 추가해야 합니다.

### 1.7. DOMAIN(도메인) 탭에 대한 액세스 권한 부여

**admin** 사용자가 **Domain** 탭을 볼 수 있도록 하려면 기본 도메인에서 **admin** 역할을 할당해야 합니다.

1. 관리자 의 **UUID**를 찾습니다.

```
$ openstack user list | grep admin
| a6a8adb6356f4a879f079485dad1321b | admin |
```

2. **default** 도메인의 **admin** 역할을 **admin** 사용자에게 추가합니다.

```
$ openstack role add --domain default --user a6a8adb6356f4a879f079485dad1321b admin
```

결과적으로 **admin** 사용자는 **Domain** 탭을 볼 수 있습니다.

## 1.8. 새 프로젝트 생성

이러한 통합 단계를 완료한 후에는 새 프로젝트를 생성할 때 **Default** 도메인에 생성할지 또는 방금 만든 **keystone** 도메인에 생성할지 결정해야 합니다. 이러한 결정은 워크 플로우와 사용자 계정을 관리하는 방법을 고려해 볼 수 있습니다. **Default** 도메인은 서비스 계정 및 **admin** 프로젝트를 관리하는 데 사용되는 내부 도메인으로 간주할 수 있습니다. 분리 목적을 위해 **AD** 지원 사용자를 별도의 **keystone** 도메인에 보관할 수 있습니다.

## 1.9. 대시보드 로그인 프로세스 변경 CHANGES TO THE DASHBOARD LOG IN PROCESS

**ID** 서비스에서 여러 도메인을 구성하면 대시보드 로그인 페이지에서 새 도메인 필드가 활성화됩니다. 사용자는 로그인 자격 증명과 일치하는 도메인을 입력해야 합니다. 이 필드는 **keystone**에 있는 도메인 중 하나로 수동으로 입력해야 합니다. **openstack** 명령을 사용하여 사용 가능한 항목을 나열합니다.

이 예에서는 **AD DS** 계정이 **LAB** 도메인을 지정해야 합니다. **admin** 등의 기본 제공 **keystone** 계정은 **Default** 를 도메인으로 지정해야 합니다.

```
# openstack domain list
+-----+-----+-----+-----+
| ID                | Name  | Enabled | Description                                     |
+-----+-----+-----+-----+
| 6800b0496429431ab1c4efbb3fe810d4 | LAB   | True    |                                               |
| default           | Default | True    | Owns users and tenants (i.e. projects) available on Identity API v2. |
+-----+-----+-----+-----+
```

## 1.10. 명령줄 변경 CHANGES TO THE COMMAND LINE

특정 명령의 경우 해당 도메인을 지정해야 할 수 있습니다. 예를 들어 이 명령에서 **--domain LAB** 을 추가하면 **LAB** 도메인(**grp-openstack** 그룹의 멤버)에서 사용자를 반환합니다.

```
# openstack user list --domain LAB
```

**--domain Default** 는 기본 제공 **keystone** 계정을 반환합니다.

```
# openstack user list --domain Default
```

### 1.11. 테스트 AD DS 통합

이 절차에서는 대시보드 기능에 대한 사용자 액세스를 테스트하여 **AD DS** 통합을 검증합니다.

1. **AD**에서 **test** 사용자를 만들고 해당 사용자를 **grp-openstack AD DS** 그룹에 추가합니다.
2. 사용자를 **demo** 테넌트의 **\_member\_** 역할에 추가합니다.
3. **AD test** 사용자의 자격 증명을 사용하여 대시보드에 로그인합니다.
4. 각 탭을 클릭하여 오류 메시지 없이 성공적으로 표시되는지 확인합니다.
5. 대시보드를 사용하여 테스트 인스턴스를 빌드합니다.



#### 참고

이러한 단계에 문제가 발생하면 기본 제공 관리자 계정으로 **3-5** 단계를 수행하십시오. 성공하면 **OpenStack**이 여전히 예상대로 작동하고 **AD databind ID** 통합 설정 내의 문제가 있음을 보여줍니다. **1.13절. “문제 해결”** 을 참조하십시오.

### 1.12. 관리자가 아닌 사용자에 대한 RC 파일 생성

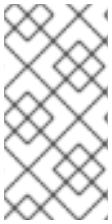
관리자가 아닌 사용자의 **RC** 파일을 만들어야 할 수 있습니다. 예를 들면 다음과 같습니다.

```
$ cat overcloudrc-v3-user1
# Clear any old environment that may conflict.
for key in $( set | awk '{FS="="} /^OS_/ {print $1}' ); do unset $key ; done
export OS_USERNAME=user1
export NOVA_VERSION=1.1
export OS_PROJECT_NAME=demo
export OS_PASSWORD=RedactedComplexPassword
export OS_NO_CACHE=True
```

```
export COMPUTE_API_VERSION=1.1
export no_proxy=,10.0.0.5,192.168.2.11
export OS_CLOUDNAME=overcloud
export OS_AUTH_URL=https://10.0.0.5:5000/v3
export OS_AUTH_TYPE=password
export PYTHONWARNINGS="ignore:Certificate has no, ignore:A true
SSLContext object is not available"
export OS_IDENTITY_API_VERSION=3
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=LAB
```

### 1.13. 문제 해결

#### 1.13.1. LDAP 연결 테스트



##### 참고

이 명령은 호스트 운영 체제에서 필요한 인증서를 찾아야 합니다. 자세한 내용은 **LDAPS 인증서 구성** 섹션을 참조하십시오.

**ldapsearch** 를 사용하여 **Active Directory** 도메인 컨트롤러에 대한 테스트 쿼리를 원격으로 수행합니다. 여기에서 성공하면 네트워크 연결이 작동하고 **AD DS** 서비스가 작동 중임을 나타냅니다. 이 예에서 테스트 쿼리는 포트 **636** 에서 서버 **addc.lab.local** 에 대해 수행됩니다.

```
# ldapsearch -Z -x -H ldaps://addc.lab.local:636 -D "svc-ldap@lab.local" -W -b
"OU=labUsers,DC=lab,DC=local" -s sub "(cn=*)" cn
```



##### 참고

**ldapsearch** 는 **openldap-clients** 패키지의 일부입니다. **# yum install openldap-clients** 를 사용하여 이 패키지를 설치할 수 있습니다.

#### 1.13.2. 인증서 신뢰 구성 테스트

**Peer**의 인증서 발급자가 인식되지 않는 경우 **ldapsearch** 로 테스트하는 동안 **TLS\_CACERTDIR** 경로가 올바르게 설정되었는지 확인합니다. 예를 들면 다음과 같습니다.

- 

**/etc/openldap/ldap.conf**

```
TLS_CACERTDIR /etc/openldap/certs
```



## 참고

임시 해결 방법으로 인증서 유효성 검사를 비활성화하는 것이 좋습니다.

이 설정을 영구적으로 구성해서는 안 됩니다.

- `/etc/openldap/ldap.conf`

```
TLS_REQCERT allow
```

이 값을 설정한 후 `ldapsearch` 쿼리가 작동하는 경우 인증서 신뢰가 올바르게 구성되었는지 확인해야 할 수 있습니다.

## 1.13.3. 포트 액세스 테스트

`nc` 를 사용하여 **LDAPS** 포트 **636** 에 원격으로 액세스할 수 있는지 확인합니다. 이 예에서는 서버 `addc.lab.local` 에 대해 프로브가 수행됩니다. `ctrl-c`를 눌러 프롬프트를 종료합니다.

```
# nc -v addc.lab.local 636
Ncat: Version 6.40 ( http://nmap.org/ncat )
Ncat: Connected to 192.168.200.10:636.
^C
```

연결을 설정하지 않으면 방화벽 구성 문제가 표시될 수 있습니다.

## 2장. IDENTITY MANAGEMENT INTEGRATION

**Red Hat IdM(Identity Manager)**과 **OpenStack ID** 통합을 계획할 때 두 서비스가 모두 구성되고 작동하는지 확인하고 사용자 관리 및 방화벽 설정에 대한 통합의 영향을 검토합니다. **Red Hat Identity Manager IdM**은 로드 밸런싱을 수행하기 위해 **SRV** 레코드에 따라 다릅니다. **IdM** 앞에 로드 밸런서를 배치해서는 안 됩니다.

### 사전 요구 사항

- **Red Hat Identity Management** 구성 및 운영.
- **Red Hat OpenStack Platform**이 구성 및 작동합니다.
- **DNS** 이름 확인이 완전히 작동하며 모든 호스트가 적절하게 등록됩니다.

### 권한 및 역할

이 통합을 통해 **IdM** 사용자는 **OpenStack**에 인증하고 리소스에 액세스할 수 있습니다. **OpenStack** 서비스 계정(예: **keystone** 및 **glance**) 및 권한 관리(권한 및 역할)은 **ID** 서비스 데이터베이스에 유지됩니다. 권한 및 역할은 **ID** 서비스 관리 툴을 사용하여 **IdM** 계정에 할당됩니다.

### 고가용성 옵션

이 구성은 단일 **IdM** 서버의 가용성에 종속됩니다. **ID** 서비스를 **IdM** 서버에 인증할 수 없는 경우 프로젝트 사용자가 영향을 받습니다. **IdM** 앞에 로드 밸런서를 배치하는 것은 권장되지 않지만 다른 **IdM** 서버를 쿼리하도록 **keystone**을 구성할 수 있습니다.

### 중단 요구 사항

- **IdM** 백엔드를 추가하려면 **ID** 서비스를 다시 시작해야 합니다.
- 사용자는 **IdM**에서 계정을 생성할 때까지 대시보드에 액세스할 수 없습니다. 다운타임을 줄려면 이 변경 전에 **IdM** 계정을 미리 태그하는 것이 좋습니다.

### 방화벽 구성

**IdM**과 **OpenStack** 간의 통신은 다음과 같습니다.

- 사용자 인증

- 2시간마다 컨트롤러에서 **CRL**(인증서 해지 목록)의 **IdM** 검색
- 만료 시 새 인증서에 대한 **certmonger** 요청



## 참고

초기 요청이 실패하는 경우 주기적인 **certmonger** 작업은 새 인증서를 계속 요청합니다.

방화벽이 **IdM**과 **OpenStack** 간의 트래픽을 필터링하는 경우 다음 포트를 통해 액세스를 허용해야 합니다.

+

소스	대상	유형	포트
OpenStack 컨트롤러 노드	Red Hat Identity Management	LDAPS	TCP 636

## 2.1. IDENTITY 서버 설정

**IdM** 서버에서 다음 명령을 실행합니다.

1. **LDAP** 조회 계정을 생성합니다. 이 계정은 **ID** 서비스에서 **IdM LDAP** 서비스를 쿼리하는 데 사용됩니다.

```
# kinit admin
# ipa user-add
First name: OpenStack
Last name: LDAP
User [administrator]: svc-ldap
```



## 참고

생성된 후 이 계정의 암호 만료 설정을 검토합니다.

2.

**grp-openstack** 이라는 **OpenStack** 사용자를 위한 그룹을 생성합니다. 이 그룹의 멤버만 **OpenStack ID**에 할당된 권한이 있을 수 있습니다.

```
# ipa group-add --desc="OpenStack Users" grp-openstack
```

3.

**svc-ldap** 계정 암호를 설정하고 **grp-openstack** 그룹에 추가합니다.

```
# ipa passwd svc-ldap
# ipa group-add-member --users=svc-ldap grp-openstack
```

4.

**svc-ldap** 사용자로 로그인하고 메시지가 표시되면 암호 변경을 수행합니다.

```
# kinit svc-ldap
```

## 2.2. LDAPS 인증서 설정



### 참고

**LDAP** 인증을 위해 여러 도메인을 사용하는 경우 권한이 부여된 프로젝트를 검색할 수 없거나 **Peer**의 인증서 발급자가 인식되지 않는 것과 같은 다양한 오류가 발생할 수 있습니다. 이는 **keystone**이 특정 도메인에 대해 잘못된 인증서를 사용하는 경우 발생할 수 있습니다. 이 문제를 해결하려면 모든 **LDAPS** 공개 키를 단일 **.crt** 번들에 병합하고 이 파일을 사용하도록 모든 **keystone** 도메인을 구성합니다.

1.

**IdM** 환경에서 **LDAPS** 인증서를 찾습니다. 이 파일은 **/etc/openldap/ldap.conf** 를 사용하여 찾을 수 있습니다.

```
TLS_CACERT /etc/ipa/ca.crt
```

2.

**keystone** 서비스를 실행하는 **OpenStack** 노드에 파일을 복사합니다. 예를 들어, 이 명령은 **scp** 를 사용하여 **ca.crt** 를 **node.lab.local** 이라는 노드에 복사합니다.

```
# scp /etc/ipa/ca.crt root@node.lab.local:/root/
```

3.

**OpenStack** 노드에서 **.crt** 를 **.pem** 으로 변환합니다.

```
# openssl x509 -in ca.crt -out ca.pem -outform PEM
```

4.

**.crt**를 인증서 디렉터리에 복사합니다. 다음은 **keystone** 서비스가 인증서에 액세스하는 데 사용할 위치입니다.

```
# cp ca.crt /etc/pki/ca-trust/source/anchors
```

참고

필요한 경우 **ldapsearch** 와 같은 진단 명령을 실행해야 하는 경우 **RHEL** 인증서 저장소에 인증서를 추가해야 합니다. 예를 들면 다음과 같습니다.

```
# cp ca.pem /etc/pki/ca-trust/source/anchors/  
# update-ca-trust
```

## 2.3. ID 서비스 구성

이 단계에서는 **IdM**과의 통합을 위해 **ID** 서비스를 준비합니다.

참고

**director**를 사용하는 경우 아래에 참조되는 구성 파일은 **Puppet**에서 관리합니다. 결과적으로 **openstack overcloud deploy** 프로세스를 실행할 때마다 추가하는 사용자 정의 설정을 덮어쓸 수 있습니다. **director** 기반 배포에 이러한 설정을 적용하려면 [4장. director와 함께 도메인별 LDAP 백엔드 사용](#) 을 참조하십시오.

### 2.3.1. 컨트롤러 구성

## 참고

구성 파일을 업데이트하려면 특정 **OpenStack** 서비스가 컨테이너 내에서 실행되고, 이 서비스는 **keystone**, **nova**, **cinder**에 적용됩니다. 따라서 다음을 고려해야 할 특정 관리 사례가 있습니다.

- 물리적 노드의 호스트 운영 체제(예: `/etc/cinder/cinder.conf`)에 있는 구성 파일을 업데이트하지 마십시오. 컨테이너화된 서비스에서 이 파일을 참조하지 않기 때문입니다.
- 컨테이너 내에서 실행 중인 구성 파일을 업데이트하지 마십시오. 컨테이너를 다시 시작하면 변경 사항이 손실되기 때문입니다.

대신 컨테이너화된 서비스에 변경 사항을 추가해야 하는 경우 컨테이너를 생성하는 데 사용되는 구성 파일을 업데이트해야 합니다. 이러한 값은 `/var/lib/config-data/puppet-generated/`에 저장됩니다.

예를 들면 다음과 같습니다.

- **keystone:** `/var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf`
- **cinder:** `/var/lib/config-data/puppet-generated/cinder/etc/cinder/cinder.conf`
- **nova:** `/var/lib/config-data/puppet-generated/nova/etc/nova/nova.conf`

그런 다음 컨테이너를 다시 시작하면 변경 사항이 적용됩니다. 예: `sudo docker restart keystone`

**keystone** 서비스를 실행하는 컨트롤러에서 다음 절차를 수행합니다.

1. **SELinux**를 구성합니다.

```
# setsebool -P authlogin_nsswitch_use_ldap=on
```

출력에는 다음과 유사한 메시지가 포함될 수 있습니다. 그들은 무시될 수 있습니다:

```
Full path required for exclude: net:[4026532245].
```

2.

도메인 디렉토리를 생성합니다.

```
# mkdir /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/
# chown 42425:42425 /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/
```

3.

여러 백엔드를 사용하도록 ID 서비스를 구성합니다.



참고

**yum install crudini** 를 사용하여 **crudini**를 설치해야 할 수 있습니다.

```
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf
identity domain_specific_drivers_enabled true
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf
identity domain_config_dir /etc/keystone/domains
# crudini --set /var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf
assignment driver sql
```



참고

**director**를 사용하는 경우 **/var/lib/config-data/puppet-generated/keystone/etc/keystone/keystone.conf** 는 **Puppet**에서 관리합니다. 결과적으로 **openstack overcloud deploy** 프로세스를 실행할 때마다 추가하는 사용자 정의 설정을 덮어쓸 수 있습니다. 따라서 매번 이 구성을 수동으로 다시 추가해야 할 수 있습니다. **director** 기반 배포의 경우 [4장. director와 함께 도메인별 LDAP 백엔드 사용](#)에서 참조하십시오.

4.

대시보드에서 여러 도메인을 활성화합니다. **/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local\_settings** 에 다음 행을 추가합니다.

```
OPENSTACK_API_VERSIONS = {
    "identity": 3
}
OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = 'Default'
```



## 참고

**director**를 사용하는 경우 `/var/lib/config-data/puppet-generated/horizon/etc/openstack-dashboard/local_settings` 가 **Puppet**에서 관리합니다. 결과적으로 **openstack overcloud deploy** 프로세스를 실행할 때마다 추가하는 사용자 정의 설정을 덮어쓸 수 있습니다. 따라서 매번 이 구성을 수동으로 다시 추가해야 할 수 있습니다.

**horizon** 컨테이너를 다시 시작하여 설정을 적용합니다.

```
$ sudo docker restart horizon
```

5.

추가 백엔드를 구성합니다.

a.

**IdM** 통합을 위한 **keystone** 도메인을 생성합니다. 새 **keystone** 도메인에 사용할 이름을 결정한 다음 도메인을 생성합니다. 예를 들어 이 명령은 **LAB**:이라는 **keystone** 도메인을 생성합니다.

```
$ openstack domain create LAB
```

b.

구성 파일을 생성합니다.

**IdM** 백엔드를 추가하려면 `/var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf` 라는 새 파일에 **LDAP** 설정을 입력합니다(랩이 이전에 생성된 도메인 이름임). **IdM** 배포에 맞게 아래 샘플 설정을 편집해야 합니다.

```
[ldap]
url = ldaps://idm.lab.local
user = uid=svc-ldap,cn=users,cn=accounts,dc=lab,dc=local
user_filter = (memberOf=cn=grp-openstack,cn=groups,cn=accounts,dc=lab,dc=local)
password = RedactedComplexPassword
user_tree_dn = cn=users,cn=accounts,dc=lab,dc=local
user_objectclass = inetUser
user_id_attribute = uid
user_name_attribute = uid
user_mail_attribute = mail
user_pass_attribute =
group_tree_dn = cn=groups,cn=accounts,dc=lab,dc=local
group_objectclass = groupOfNames
group_id_attribute = cn
group_name_attribute = cn
group_member_attribute = member
group_desc_attribute = description
```



```

use_tls          = False
query_scope      = sub
chase_referrals  = false
tls_cacertfile  = /etc/pki/ca-trust/source/anchors/anchorsca.crt

```

```

[identity]
driver = ldap

```

각 설정에 대한 설명:

설정	설명
<b>url</b>	인증에 사용할 IdM 서버입니다. LDAPS 포트 <b>636</b> 을 사용합니다.
<b>user</b>	LDAP 쿼리에 사용할 IdM 계정입니다.
<b>암호</b>	위에서 사용된 IdM 계정의 일반 텍스트 암호입니다.
<b>user_filter</b>	ID 서비스에 표시되는 사용자를 필터링합니다. 결과적으로 <b>grp-openstack</b> 그룹의 멤버만 ID 서비스에 정의된 권한을 가질 수 있습니다.
<b>user_tree_dn</b>	IdM의 OpenStack 계정 경로입니다.
<b>user_objectclass</b>	LDAP 사용자의 유형을 정의합니다. IdM의 경우 <b>inetUser</b> 유형을 사용합니다.
<b>user_id_attribute</b>	사용자 ID에 사용할 IdM 값을 매핑합니다.
<b>user_name_attribute</b>	<i>이름</i> 에 사용할 IdM 값을 매핑합니다.
<b>user_mail_attribute</b>	사용자 이메일 주소에 사용할 IdM 값을 매핑합니다.
<b>user_pass_attribute</b>	이 값을 비워 둡니다.



#### 참고

**IdM 그룹과의 통합은 중첩 그룹이 아닌 직접 멤버만 반환합니다. 결과적으로 LDAP\_MATCHING\_RULE\_IN\_CHAIN 또는:1.2.840.113556.1.4.1941:을 사용하는 쿼리는 현재 IdM에서 작동하지 않습니다.**

- 6. 구성 파일의 소유권을 **keystone** 사용자로 변경합니다.

```
# chown 42425:42425 /var/lib/config-data/puppet-generated/keystone/etc/keystone/domains/keystone.LAB.conf
```

- 7. **admin** 사용자에게 도메인에 대한 액세스 권한을 부여합니다.



참고

이 경우 **OpenStack admin** 계정에 **IdM**의 권한이 부여되지 않습니다. 이 경우 **domain**이라는 용어는 **OpenStack**의 **keystone** 도메인 사용을 나타냅니다.

- a. **LAB** 도메인의 ID 를 가져옵니다.

```
$ openstack domain show LAB
+-----+-----+
| Field | Value          |
+-----+-----+
| enabled | True           |
| id      | 6800b0496429431ab1c4efbb3fe810d4 |
| name    | LAB            |
+-----+-----+
```

- b. **admin** 사용자의 ID 값을 가져옵니다.

```
$ openstack user list --domain default | grep admin
| 3d75388d351846c6a880e53b2508172a | admin |
```

- c. **admin** 역할의 ID 값을 가져옵니다.

```
# openstack role list
+-----+-----+
| ID              | Name          |
+-----+-----+
| 544d48aaffde48f1b3c31a52c35f01f9 | SwiftOperator |
| 6d005d783bf0436e882c55c62457d33d | ResellerAdmin |
| 785c70b150ee4c778fe4de088070b4cf | admin         |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_     |
+-----+-----+
```

- d.

반환된 도메인 및 **admin ID**를 사용하여 **admin** 사용자를 **keystone LAB** 도메인의 **admin** 역할에 추가하는 명령을 구성합니다.

```
$ openstack role add --domain 6800b0496429431ab1c4efbb3fe810d4 --user
3d75388d351846c6a880e53b2508172a 785c70b150ee4c778fe4de088070b4cf
```

8.

**keystone** 서비스를 다시 시작하여 변경 사항을 적용합니다.

```
$ sudo docker restart keystone
```

9.

명령에 **keystone** 도메인 이름을 추가하여 **IdM** 도메인의 사용자 목록을 확인합니다.

```
$ openstack user list --domain LAB
```

10.

로컬 **keystone** 데이터베이스에서 서비스 계정을 확인합니다.

```
$ openstack user list --domain default
```

### 2.3.2. IdM 그룹 구성원이 프로젝트에 액세스하도록 허용

인증된 사용자가 **OpenStack** 리소스에 액세스할 수 있도록 하려면 특정 **IdM** 그룹에 프로젝트에 대한 액세스 권한을 부여하도록 권한을 부여하는 것이 좋습니다. 이렇게 하면 **OpenStack** 관리자가 각 사용자를 프로젝트의 역할에 할당할 필요가 없습니다. 대신, **IdM** 그룹에 프로젝트에 역할이 부여됩니다. 결과적으로 이러한 **IdM** 그룹의 멤버인 **IdM** 사용자는 사전 결정된 프로젝트에 액세스할 수 있습니다.



#### 참고

개별 **IdM** 사용자의 권한을 수동으로 관리하려면 [2.3.3절. “IdM 사용자가 프로젝트에 액세스할 수 있도록 허용”](#)을 참조하십시오.

이 섹션에서는 **IdM** 관리자가 다음 단계를 이미 완료했다고 가정합니다.

- **IdM**에 **grp-openstack-admin** 이라는 그룹을 생성합니다.
- **IdM**에 **grp-openstack-demo** 라는 그룹을 생성합니다.

- 필요에 따라 위의 그룹 중 하나에 **IdM** 사용자를 추가합니다.
- **grp-openstack** 그룹에 **IdM** 사용자를 추가합니다.
- 지정된 프로젝트를 고려하십시오. 이 예에서는 **openstack project create --domain default --description "Demo Project"** 데모를 사용하여 생성된 데모 프로젝트를 사용합니다.

이 단계에서는 **IdM** 그룹에 역할을 할당합니다. 그러면 그룹 멤버가 **OpenStack** 리소스에 액세스할 수 있는 권한이 있습니다.

1. **IdM** 그룹 목록을 검색합니다.

```
$ openstack group list --domain LAB
+-----+-----+
| ID                               | Name           |
+-----+-----+
| 185277be62ae17e498a69f98a59b66934fb1d6b7f745f14f5f68953a665b8851 | grp-
openstack |
| a8d17f19f464c4548c18b97e4aa331820f9d3be52654aa8094e698a9182cbb88 | grp-
openstack-admin |
| d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8 | grp-
openstack-demo |
+-----+-----+
```

2. 역할 목록을 검색합니다.

```
$ openstack role list
+-----+-----+
| ID                               | Name           |
+-----+-----+
| 0969957bce5e4f678ca6cef00e1abf8a | ResellerAdmin |
| 1fcb3c9b50aa46ee8196aaaecc2b76b7 | admin          |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_      |
| d3570730eb4b4780a7fed97eba197e1b | SwiftOperator |
+-----+-----+
```

3. 이러한 역할 중 하나 이상에 추가하여 **IdM** 그룹에 프로젝트에 대한 액세스 권한을 부여합니다. 예를 들어 **grp-openstack-demo** 그룹의 사용자가 **demo** 프로젝트의 일반 사용자가 되도록 하려면 **\_member\_** 역할에 그룹을 추가해야 합니다.

```
$ openstack role add --project demo --group
d971bb3bd5e64a454cbd0cc7af4c0773e78d61b5f81321809f8323216938cae8 _member_
```

결과적으로 **grp-openstack-demo** 의 멤버는 도메인 필드에 **LAB** 을 입력하는 경우 **IdM** 사용자 이름과 암호를 입력하여 대시보드에 로그인할 수 있습니다.



#### 참고

사용자에게 **Error: Unable to retrieve container list** 가 표시되면 컨테이너 목록을 검색할 수 있으며 컨테이너를 관리할 수 있을 것으로 예상되는 경우 **SwiftOperator** 역할에 추가해야 합니다.

### 2.3.3. IdM 사용자가 프로젝트에 액세스할 수 있도록 허용

**grp-openstack IdM** 그룹의 멤버인 **IdM** 사용자에게는 대시보드에서 **프로젝트에** 로그인할 수 있는 권한을 부여할 수 있습니다.

1. **IdM** 사용자 목록을 검색합니다.

```
# openstack user list --domain LAB
+-----+-----+
| ID                               | Name      |
+-----+-----+
| 1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e | user1     |
| 12c062faddc5f8b065434d9ff6fce03eb9259537c93b411224588686e9a38bf1 | user2     |
| afaf48031eb54c3e44e4cb0353f5b612084033ff70f63c22873d181fdae2e73c | user3     |
```

```
| e47fc21dcf0d9716d2663766023e2d8dc15a6d9b01453854a898cabb2396826e | user4
|
+-----+-----+
```

2.

역할 목록을 검색합니다.

```
# openstack role list
+-----+-----+
| ID                | Name          |
+-----+-----+
| 544d48aaffde48f1b3c31a52c35f01f9 | SwiftOperator |
| 6d005d783bf0436e882c55c62457d33d | ResellerAdmin |
| 785c70b150ee4c778fe4de088070b4cf | admin          |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_      |
+-----+-----+
```

3.

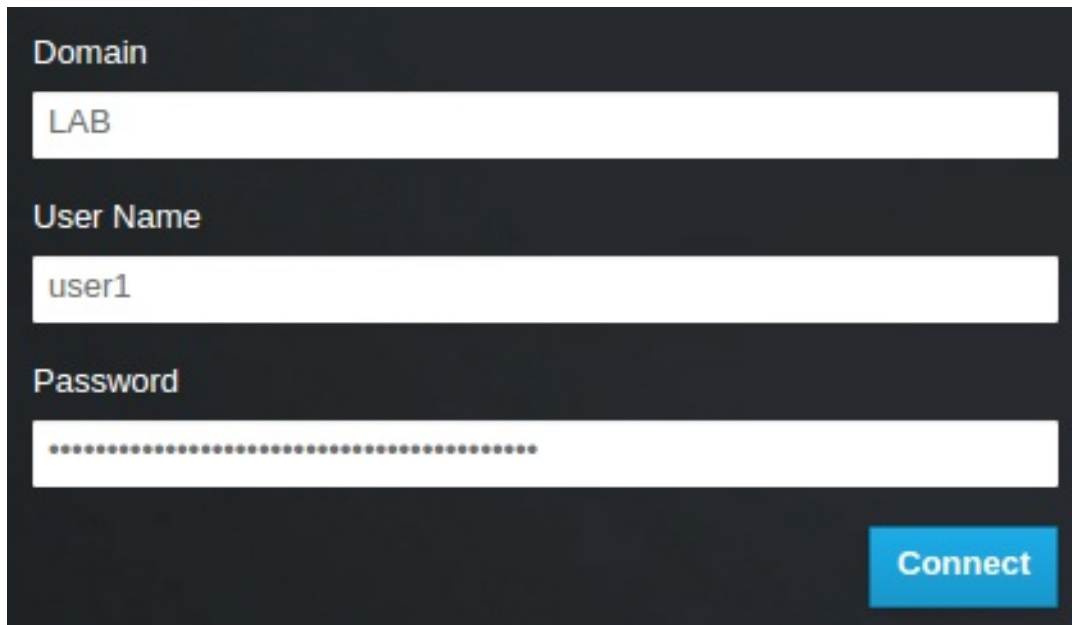
이러한 역할 중 하나 이상에 추가하여 사용자에게 프로젝트에 대한 액세스 권한을 부여합니다. 예를 들어 **user1** 이 **demo** 프로젝트의 일반 사용자가 되도록 하려면 **member** 역할에 추가합니다.

```
# openstack role add --project demo --user
1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e _member_
```

또는 **user1** 이 **demo** 프로젝트의 관리자인 경우 **admin** 역할에 추가합니다.

```
# openstack role add --project demo --user
1f24ec1f11aeb90520079c29f70afa060d22e2ce92b2eba7784c841ac418091e admin
```

결과적으로 **user1** 은 **Domain** 필드에 **LAB** 을 입력하는 경우 **IdM** 사용자 이름과 암호를 입력하여 대시보드에 로그인할 수 있습니다.



Domain

LAB

User Name

user1

Password

.....

Connect



#### 참고

사용자에게 **Error: Unable to retrieve container list** 가 표시되면 컨테이너 목록을 검색할 수 있으며 컨테이너를 관리할 수 있을 것으로 예상되는 경우 **SwiftOperator** 역할에 추가해야 합니다.

## 2.4. DOMAIN(도메인) 탭에 대한 액세스 권한 부여

**admin** 사용자가 **Domain** 탭을 볼 수 있도록 하려면 기본 도메인에서 **admin** 역할을 할당해야 합니다.

1. 관리자 의 **UUID**를 찾습니다.

```
$ openstack user list | grep admin
| a6a8adb6356f4a879f079485dad1321b | admin |
```

2. **default** 도메인의 **admin** 역할을 **admin** 사용자에게 추가합니다.

```
$ openstack role add --domain default --user a6a8adb6356f4a879f079485dad1321b admin
```

결과적으로 **admin** 사용자는 **Domain** 탭을 볼 수 있습니다.

## 2.5. 새 프로젝트 생성

이러한 통합 단계를 완료한 후에는 새 프로젝트를 생성할 때 **Default** 도메인에 생성할지 또는 방금 만든

**keystone** 도메인에 생성할지 결정해야 합니다. 이러한 결정은 워크 플로우와 사용자 계정을 관리하는 방법을 고려해 볼 수 있습니다. **Default** 도메인은 서비스 계정과 **admin** 프로젝트에 사용되는 내부 도메인으로 간주될 수 있으므로 **AD** 지원 사용자가 다른 **keystone** 도메인 내에 배치되는 것이 좋습니다. **IdM** 사용자가 있는 것과 동일한 **keystone** 도메인일 필요는 없으며, 여러 **keystone** 도메인이 있을 수 있습니다.

### 2.5.1. 대시보드 로그인 프로세스 변경 **Changes to the dashboard log in process**

**ID** 서비스에서 여러 도메인을 구성하면 대시보드 로그인 페이지에서 새 도메인 필드가 활성화됩니다. 사용자는 로그인 자격 증명과 일치하는 도메인을 입력해야 합니다. 이 필드는 **keystone**에 있는 도메인 중 하나로 수동으로 입력해야 합니다. **openstack** 명령을 사용하여 사용 가능한 항목을 나열합니다.

이 예제에서 **IdM** 계정은 **LAB** 도메인을 지정해야 합니다. **admin** 등의 기본 제공 **keystone** 계정은 **Default** 를 도메인으로 지정해야 합니다.

```
$ openstack domain list
+-----+-----+-----+-----+
| ID                | Name   | Enabled | Description |
+-----+-----+-----+-----+
| 6800b0496429431ab1c4efbb3fe810d4 | LAB    | True    |              |
| default            | Default | True    | Owns users and tenants (i.e. projects) available on Identity API v2. |
+-----+-----+-----+-----+
```

### 2.5.2. 명령줄 변경 **Changes to the command line**

특정 명령의 경우 해당 도메인을 지정해야 할 수 있습니다. 예를 들어 이 명령에서 **--domain LAB** 을 추가하면 **LAB** 도메인( **grp-openstack** 그룹의 멤버)에서 사용자를 반환합니다.

```
$ openstack user list --domain LAB
```

**--domain Default** 는 기본 제공 **keystone** 계정을 반환합니다.

```
$ openstack user list --domain Default
```

### 2.5.3. **IdM** 통합 테스트

다음 절차에서는 대시보드 기능에 대한 사용자 액세스를 테스트하여 **IdM** 통합을 검증합니다.



1. **IdM**에서 테스트 사용자를 생성하고 사용자를 **grp-openstack IdM** 그룹에 추가합니다.
2. 사용자를 **demo** 테넌트의 **\_member\_** 역할에 추가합니다.
3. **IdM test** 사용자의 자격 증명을 사용하여 대시보드에 로그인합니다.
4. 각 탭을 클릭하여 오류 메시지 없이 성공적으로 표시되는지 확인합니다.
5. 대시보드를 사용하여 테스트 인스턴스를 빌드합니다.



#### 참고

이러한 단계에 문제가 발생하면 기본 제공 관리자 계정으로 3-5단계를 수행하십시오. 성공적으로 수행되면 **OpenStack**이 여전히 예상대로 작동하고 **IdM databind Identity** 통합 설정 내의 문제가 있음을 보여줍니다. **2.7절. “문제 해결”**을 참조하십시오.

## 2.6. 관리자가 아닌 사용자에 대한 RC 파일 생성

관리자가 아닌 사용자의 **RC** 파일을 만들어야 할 수 있습니다. 예를 들면 다음과 같습니다.

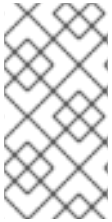
```
$ cat overcloudrc-v3-user1
# Clear any old environment that may conflict.
for key in $( set | awk '{FS="="} /^OS_/ {print $1}' ); do unset $key ; done
export OS_USERNAME=user1
export NOVA_VERSION=1.1
export OS_PROJECT_NAME=demo
export OS_PASSWORD=RedactedComplexPassword
export OS_NO_CACHE=True
export COMPUTE_API_VERSION=1.1
export no_proxy=,10.0.0.5,192.168.2.11
export OS_CLOUDNAME=overcloud
export OS_AUTH_URL=https://10.0.0.5:5000/v3
export OS_AUTH_TYPE=password
export PYTHONWARNINGS="ignore:Certificate has no, ignore:A true
SSLContext object is not available"
export OS_IDENTITY_API_VERSION=3
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=LAB
```

## 2.7. 문제 해결

### 2.7.1. LDAP 연결 테스트

**ldapsearch** 를 사용하여 **IdM** 서버에 대한 테스트 쿼리를 원격으로 수행합니다. 여기에서 성공하면 네트워크 연결이 작동 중이며 **IdM** 서비스가 작동 중임을 나타냅니다. 이 예에서 테스트 쿼리는 포트 **636**에 서버 **idm.lab.local** 에 대해 수행됩니다.

```
# ldapsearch -D "cn=directory manager" -H ldaps://idm.lab.local:636 -b "dc=lab,dc=local" -s sub "(objectclass=*)" -w RedactedComplexPassword
```



참고

**ldapsearch** 는 **openldap-clients** 패키지의 일부입니다. **# yum install openldap-clients** 를 사용하여 이 패키지를 설치할 수 있습니다.

### 2.7.2. 포트 액세스 테스트

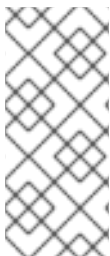
**nc** 를 사용하여 **LDAPS** 포트(**636**)에 원격으로 액세스할 수 있는지 확인합니다. 이 예에서는 서버 **idm.lab.local** 에 대해 프로브가 수행됩니다. **ctrl-c**를 눌러 프롬프트를 종료합니다.

```
# nc -v idm.lab.local 636
Ncat: Version 6.40 ( http://nmap.org/ncat )
Ncat: Connected to 192.168.200.10:636.
^C
```

연결을 설정하지 않으면 방화벽 구성 문제가 표시될 수 있습니다.

### 3장. NOVAJOIN을 사용하여 IDM과 통합

**novajoin**을 사용하면 배포 프로세스의 일부로 **Red Hat IdM(Identity Manager)**에 노드를 등록할 수 있습니다. 결과적으로 **ID, kerberos** 자격 증명, 액세스 제어를 포함하여 **IdM** 기능을 **OpenStack** 배포와 통합할 수 있습니다.



#### 참고

**novajoin**을 통한 **IdM** 등록은 현재 언더클라우드 및 오버클라우드 노드에만 사용할 수 있습니다. 오버클라우드 인스턴스의 **novajoin** 통합은 이후 릴리스에서 지원될 예정입니다.

### 3.1. 언더클라우드에 NOVAJOIN 설치 및 설정

#### 3.1.1. CA에 언더클라우드 추가

오버클라우드를 배포하기 전에 **CA(인증 기관)**에 언더클라우드를 추가해야 합니다.

1. 언더클라우드 노드에서 **python-novajoin** 패키지를 설치합니다.

```
$ sudo yum install python-novajoin
```

2. 언더클라우드 노드에서 **novajoin-ipa-setup** 스크립트를 실행하여 배포에 맞게 값을 조정합니다.

```
$ sudo /usr/libexec/novajoin-ipa-setup \
  --principal admin \
  --password <IdM admin password> \
  --server <IdM server hostname> \
  --realm <overcloud cloud domain (in upper case)> \
  --domain <overcloud cloud domain> \
  --hostname <undercloud hostname> \
  --precreate
```

다음 섹션에서는 생성된 **TTL(단시간 암호)**을 사용하여 언더클라우드를 등록합니다.

#### 3.1.2. IdM에 언더클라우드 추가

이 절차에서는 **IdM**에 언더클라우드를 등록하고 **novajoin**을 구성합니다. **undercloud.conf** (**[DEFAULT]** 섹션에 있는)에서 다음 설정을 설정합니다.

1. **novajoin** 서비스는 기본적으로 비활성화되어 있습니다. 활성화하려면 다음을 수행합니다.

```
[DEFAULT]
enable_novajoin = true
```

2. 언더클라우드 노드를 **IdM**에 등록하려면 **One-Time Password(OTP)**를 설정해야 합니다.

```
ipa_otp = <otp>
```

3. **neutron**의 **DHCP** 서버에서 제공하는 오버클라우드의 도메인 이름이 **IdM** 도메인(대소문자의 **kerberos** 영역)과 일치하는지 확인합니다.

```
overcloud_domain_name = <domain>
```

4. 언더클라우드에 적절한 호스트 이름을 설정합니다.

```
undercloud_hostname = <undercloud FQDN>
```

5. **IdM**을 언더클라우드의 네임서버로 설정합니다.

```
undercloud_nameservers = <IdM IP>
```

6. 대규모 환경의 경우 **novajoin** 연결 시간 초과 값을 검토해야 합니다. **undercloud.conf** 에서 이름이 **undercloud-timeout.yaml** 인 새 파일에 대한 참조를 추가합니다.

```
hieradata_override = /home/stack/undercloud-timeout.yaml
```

**undercloud-timeout.yaml** 에 다음 옵션을 추가합니다. 시간 제한 값을 초 단위로 지정할 수 있습니다(예: 5):

```
nova::api::vendordata_dynamic_connect_timeout: <timeout value>
nova::api::vendordata_dynamic_read_timeout: <timeout value>
```

7. **undercloud.conf** 파일을 저장합니다.

8. 언더클라우드 배포 명령을 실행하여 기존 언더클라우드에 변경 사항을 적용합니다.

```
$ openstack undercloud install
```

## 검증

1. 언더클라우드의 키 항목의 키탭 파일을 확인합니다.

```
[root@undercloud-0 ~]# klist -kt
Keytab name: FILE:/etc/krb5.keytab
KVNO Timestamp      Principal
-----
1 04/28/2020 12:22:06 host/undercloud-0.redhat.local@REDHAT.LOCAL
1 04/28/2020 12:22:06 host/undercloud-0.redhat.local@REDHAT.LOCAL
```

```
[root@undercloud-0 ~]# klist -kt /etc/novajoin/krb5.keytab
Keytab name: FILE:/etc/novajoin/krb5.keytab
KVNO Timestamp      Principal
-----
1 04/28/2020 12:22:26 nova/undercloud-0.redhat.local@REDHAT.LOCAL
1 04/28/2020 12:22:26 nova/undercloud-0.redhat.local@REDHAT.LOCAL
```

2. **/etc/krb.keytab** 파일을 호스트 원칙으로 테스트합니다.

```
[root@undercloud-0 ~]# kinit -k
[root@undercloud-0 ~]# klist
Ticket cache: KEYRING:persistent:0:0
Default principal: host/undercloud-0.redhat.local@REDHAT.LOCAL

Valid starting    Expires          Service principal
05/04/2020 10:34:30 05/05/2020 10:34:30  krbtgt/REDHAT.LOCAL@REDHAT.LOCAL
```

```
[root@undercloud-0 ~]# kdestroy
Other credential caches present, use -A to destroy all
```

3. **novajoin /etc/novajoin/krb.keytab** 파일을 **nova** 원칙으로 테스트합니다.

```
[root@undercloud-0 ~]# kinit -kt /etc/novajoin/krb5.keytab 'nova/undercloud-0.redhat.local@REDHAT.LOCAL'
[root@undercloud-0 ~]# klist
Ticket cache: KEYRING:persistent:0:0
Default principal: nova/undercloud-0.redhat.local@REDHAT.LOCAL

Valid starting    Expires          Service principal
05/04/2020 10:39:14 05/05/2020 10:39:14  krbtgt/REDHAT.LOCAL@REDHAT.LOCAL
```

### 3.2. 오버클라우드에 NOVAJOIN 설치 및 설정

이 섹션에서는 IdM에 오버클라우드 노드를 등록하는 방법을 설명합니다.

#### 3.2.1. 오버클라우드 DNS 설정

IdM 환경을 자동으로 감지하고 쉽게 등록할 수 있도록 하려면 IdM을 DNS 서버로 사용하는 것이 좋습니다.

1. 언더클라우드에 연결합니다.

```
$ source ~/stackrc
```

2. DNS 이름 서버로 IdM을 사용하도록 컨트롤 플레인 서브넷을 구성합니다.

```
$ openstack subnet set ctlplane-subnet --dns-nameserver <idm_server_address>
```

3. IdM 서버를 사용하도록 환경 파일에서 DnsServers 매개변수를 설정합니다.

```
parameter_defaults:
  DnsServers: ["<idm_server_address>"]
```

이 매개변수는 일반적으로 사용자 지정 `network-environment.yaml` 파일에 정의되어 있습니다.

#### 3.2.2. novajoin을 사용하도록 오버클라우드 설정

1. IdM 통합을 활성화하려면 `/usr/share/openstack-tripleo-heat-templates/environments/predictable-placement/custom-domain.yaml` 환경 파일의 사본을 생성합니다.

```
$ cp /usr/share/openstack-tripleo-heat-templates/environments/predictable-
placement/custom-domain.yaml \
/home/stack/templates/custom-domain.yaml
```

2. `/home/stack/templates/custom-domain.yaml` 환경 파일을 편집하고 배포에 맞게 `CloudDomain` 및 `CloudName*` 값을 설정합니다. 예를 들면 다음과 같습니다.

-

```
parameter_defaults:
  CloudDomain: lab.local
  CloudName: overcloud.lab.local
  CloudNameInternal: overcloud.internalapi.lab.local
  CloudNameStorage: overcloud.storage.lab.local
  CloudNameStorageManagement: overcloud.storagemgmt.lab.local
  CloudNameCtlplane: overcloud.ctlplane.lab.local
```

3.

오버클라우드 배포 프로세스에 다음 환경 파일을 포함합니다.

- `/usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-internal-tls.yaml`
- `/usr/share/openstack-tripleo-heat-templates/environments/ssl/tls-everywhere-endpoints-dns.yaml`
- `/home/stack/templates/custom-domain.yaml`

예를 들면 다음과 같습니다.

```
openstack overcloud deploy \
  --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-internal-tls.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/tls-everywhere-endpoints-dns.yaml \
  -e /home/stack/templates/custom-domain.yaml \
```

결과적으로 배포된 **Overcloud** 노드가 **IdM**에 자동으로 등록됩니다.

4.

이는 내부 엔드포인트에 대해서만 **TLS**를 설정합니다. 외부 끝점의 경우 `/usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-tls.yaml` 환경 파일을 사용하여 **TLS**를 추가하는 일반 수단을 사용할 수 있습니다(사용자 정의 인증서 및 키를 추가하도록 수정해야 함). 결과적으로 **openstack deploy** 명령은 다음과 유사합니다.

```
openstack overcloud deploy \
  --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-internal-tls.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/tls-everywhere-endpoints-dns.yaml \
  -e /home/stack/templates/custom-domain.yaml \
  -e /home/stack/templates/enable-tls.yaml
```

5.

또는 **IdM**을 사용하여 공용 인증서를 발행할 수도 있습니다. 이 경우 `/usr/share/openstack-tripleo-heat-templates/environments/services/haproxy-public-tls-certmonger.yaml` 환경 파일을 사용해야 합니다. 예를 들면 다음과 같습니다.

```
openstack overcloud deploy \
  --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/enable-internal-tls.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/ssl/tls-everywhere-endpoints-dns.yaml \
  -e /home/stack/templates/custom-domain.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/haproxy-public-tls-certmonger.yaml
```

### 3.3. IDM에서 노드 검증

1.

**IdM**에서 오버클라우드 노드를 찾고 호스트 항목에 **Keytab:True** 가 포함되어 있는지 확인합니다.

```
$ ipa host-show overcloud-node-01
Host name: overcloud-node-01.lab.local
Principal name: host/overcloud-node-01.lab.local@LAB.LOCAL
Principal alias: host/overcloud-node-01.lab.local@LAB.LOCAL
SSH public key fingerprint: <snip>
Password: False
Keytab: True
Managed by: overcloud-node-01.lab.local
```

2.

**SSH**를 노드에 연결하고 **sssd**가 **IdM** 사용자를 쿼리할 수 있는지 확인합니다. 예를 들어 **susan**이라는 **IdM** 사용자를 쿼리하려면 다음을 수행합니다.

```
$ getent passwd susan
uid=1108400007(susan) gid=1108400007(bob) groups=1108400007(susan)
```

### 3.4. NOVAJOIN의 DNS 항목 구성

**haproxy-public-tls-certmonger.yaml** 템플릿을 사용하여 끝점의 공용 인증서를 발급하는 경우, **Novajoin**에서 사용하는 **VIP** 엔드포인트에 대한 **DNS** 항목을 수동으로 생성해야 합니다.

1.

오버클라우드 네트워크를 확인합니다. `/home/stack/virt/network/network-environment.yaml`에서 이러한 항목을 찾을 수 있습니다.

```
parameter_defaults:
  ControlPlaneDefaultRoute: 192.168.24.1
```



```

ExternalAllocationPools:
- end: 10.0.0.149
  start: 10.0.0.101
InternalApiAllocationPools:
- end: 172.17.1.149
  start: 172.17.1.10
StorageAllocationPools:
- end: 172.17.3.149
  start: 172.17.3.10
StorageMgmtAllocationPools:
- end: 172.17.4.149
  start: 172.17.4.10

```

2.

각 오버클라우드 네트워크에 대한 **VIP(가상 IP 주소)** 목록을 생성합니다. 예:  
**/home/stack/virt/public\_vip.yaml**

```

parameter_defaults:
  ControlFixedIPs: [{'ip_address':'192.168.24.101'}]
  PublicVirtualFixedIPs: [{'ip_address':'10.0.0.101'}]
  InternalApiVirtualFixedIPs: [{'ip_address':'172.17.1.101'}]
  StorageVirtualFixedIPs: [{'ip_address':'172.17.3.101'}]
  StorageMgmtVirtualFixedIPs: [{'ip_address':'172.17.4.101'}]
  RedisVirtualFixedIPs: [{'ip_address':'172.17.1.102'}]

```

3.

각 **VIP**에 대해 **IdM**에 **DNS** 항목을 추가합니다. 또한 새 영역을 생성해야 할 수도 있습니다. 다음 예제에서는 **IdM**의 **DNS** 레코드 및 영역 생성을 보여줍니다.

```

ipa dnsrecord-add lab.local overcloud --a-rec 10.0.0.101
ipa dnszone-add ctlplane.lab.local
ipa dnsrecord-add ctlplane.lab.local overcloud --a-rec 192.168.24.101
ipa dnszone-add internalapi.lab.local
ipa dnsrecord-add internalapi.lab.local overcloud --a-rec 172.17.1.101
ipa dnszone-add storage.lab.local
ipa dnsrecord-add storage.lab.local overcloud --a-rec 172.17.3.101
ipa dnszone-add storagemgmt.lab.local
ipa dnsrecord-add storagemgmt.lab.local overcloud --a-rec 172.17.4.101

```

## 4장. DIRECTOR와 함께 도메인별 LDAP 백엔드 사용

Red Hat OpenStack Platform director는 하나 이상의 LDAP 백엔드를 사용하도록 keystone을 구성할 수 있습니다. 이 방법을 사용하면 각 keystone 도메인에 대해 별도의 LDAP 백엔드가 생성됩니다.

### 4.1. 구성 옵션 설정

Red Hat OpenStack Platform director를 사용하는 배포의 경우 heat 템플릿에서 KeystoneLDAPDomainEnable 플래그를 true 로 설정해야 합니다. 따라서 keystone에서 domain\_specific\_drivers\_enabled 옵션을 구성합니다( ID 구성 그룹 내).



#### 참고

도메인 구성 파일의 기본 디렉터리는 `/etc/keystone/domains/` 로 설정됩니다. `keystone::domain_config_directory hiera` 키를 사용하여 필요한 경로를 설정하고 환경 파일 내의 `ExtraConfig` 매개변수로 추가하여 이를 덮어쓸 수 있습니다.

LDAP 백엔드 구성의 사양도 추가해야 합니다. 이 작업은 `tripleo-heat-templates` 의 `KeystoneLDAPBackendConfigs` 매개변수를 사용하여 수행되며, 여기에서 필요한 LDAP 옵션을 지정할 수 있습니다.

### 4.2. DIRECTOR 배포 설정

1.

`keystone_domain_specific_ldap_backend.yaml` 환경 파일의 사본을 생성합니다.

```
$ cp /usr/share/openstack-tripleo-heat-templates/environments/services/keystone_domain_specific_ldap_backend.yaml /home/stack/templates/
```

2.

`/home/stack/templates/keystone_domain_specific_ldap_backend.yaml` 환경 파일을 편집하고 배포에 맞게 값을 설정합니다. 예를 들어, 이러한 항목은 `testdomain` 이라는 keystone 도메인에 대한 LDAP 구성을 생성합니다.

```
parameter_defaults:
  KeystoneLDAPDomainEnable: true
  KeystoneLDAPBackendConfigs:
    testdomain:
      url: ldaps://192.0.2.250
      user: cn=openstack,ou=Users,dc=director,dc=example,dc=com
      password: RedactedComplexPassword
      suffix: dc=director,dc=example,dc=com
```

```
user_tree_dn: ou=Users,dc=director,dc=example,dc=com
user_filter: "(memberOf=cn=OSuser,ou=Groups,dc=director,dc=example,dc=com)"
user_objectclass: person
user_id_attribute: cn
```

3.

환경 파일을 구성하여 여러 도메인을 지정할 수도 있습니다. 예를 들면 다음과 같습니다.

```
KeystoneLDAPBackendConfigs:
  domain1:
    url: ldaps://domain1.example.com
    user: cn=openstack,ou=Users,dc=director,dc=example,dc=com
    password: RedactedComplexPassword
    ...
  domain2:
    url: ldaps://domain2.example.com
    user: cn=openstack,ou=Users,dc=director,dc=example,dc=com
    password: RedactedComplexPassword
    ...
```

그러면 **domain1** 및 **domain2** 라는 두 도메인이 생성됩니다. 각각 고유한 구성이 있는 다른 LDAP 도메인이 있습니다.