Oracle® Solaris에서 OpenStack(Kilo) 설치 및 구성

부품 번호: E74916
2016년 6월

ORACLE

부품 번호: E74916

Copyright © 2014, 2016, Oracle and/or its affiliates. All rights reserved.

본 소프트웨어와 관련 문서는 사용 제한 및 기밀 유지 규정을 포함하는 라이센스 합의서에 의거해 제공되며, 지적 재산법에 의해 보호됩니다. 라이센스 합의서 상에 명시적으로 허용되어 있는 경우나 법규에 의해 허용된 경우를 제외하고, 어떠한 부분도 복사, 재생, 번역, 방송, 수정, 라이센스, 전송, 배포, 진열, 실행, 발행, 또는 전시될 수 없습니다. 본 소프트웨어를 리버스 엔지니어링, 디스어셈블리 또는 디컴파일하는 것은 상호 운용에 대한 법규에 의해 명시된 경우를 제외하고는 금지되어 있습니다.

이 안의 내용은 사전 공지 없이 변경될 수 있으며 오류가 존재하지 않음을 보증하지 않습니다. 만일 오류를 발견하면 서면으로 통지해 주기 바랍니다.

만일 본 소프트웨어나 관련 문서를 미국 정부나 또는 미국 정부를 대신하여 라이센스한 개인이나 법인에게 배송하는 경우, 다음 공지사항이 적용됩니다.

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

본 소프트웨어 혹은 하드웨어는 다양한 정보 관리 애플리케이션의 일반적인 사용을 목적으로 개발되었습니다. 본 소프트웨어 혹은 하드웨어는 개인적인 상해를 초래할 수 있는 애플리케이션을 포함한 본질적으로 위험한 애플리케이션에서 사용할 목적으로 개발되거나 그 용도로 사용될 수 없습니다. 만일 본 소프트웨어 혹은 하드웨어를 위험한 애플리케이션에서 사용할 경우, 라이센스 사용자는 해당 애플리케이션의 안전한 사용을 위해 모든 적절한 비상-안전, 백업, 대비 및 기타 조치를 반드시 취해야 합니다. Oracle Corporation과 그 자회사는 본 소프트웨어 혹은 하드웨어를 위험한 애플리케이션에서의 사용으로 인해 발생하는 어떠한 손해에 대해서도 책임지지 않습니다.

Oracle과 Java는 Oracle Corporation 및/또는 그 자회사의 등록 상표입니다. 기타의 명칭들은 각 해당 명칭을 소유한 회사의 상표일 수 있습니다.

Intel 및 Intel Xeon은 Intel Corporation의 상표 내지는 등록 상표입니다. SPARC 상표 일체는 라이센스에 의거하여 사용되며 SPARC International, Inc.의 상표 내지는 등록 상표입니다. AMD, Opteron, AMD 로고, 및 AMD Opteron 로고는 Advanced Micro Devices의 상표 내지는 등록 상표입니다. UNIX는 The Open Group의 등록상표입니다.

본 소프트웨어 혹은 하드웨어와 관련문서(설명서)는 제3자로부터 제공되는 컨텐츠, 제품 및 서비스에 접속할 수 있거나 정보를 제공합니다. 사용자와 오라클 간의 합의서에 별도로 규정되어 있지 않는 한 Oracle Corporation과 그 자회사는 제3자의 컨텐츠, 제품 및 서비스와 관련하여 어떠한 책임도 지지 않으며 명시적으로 모든 보증에 대해서도 책임을 지지 않습니다. Oracle Corporation과 그 자회사는 제3자의 컨텐츠, 제품 및 서비스에 접속하거나 사용으로 인해 초래되는 어떠한 손실, 비용 또는 손해에 대해 어떠한 책임도 지지 않습니다. 단, 사용자와 오라클 간의 합의서에 규정되어 있는 경우는 예외입니다.

설명서 접근성

오라클의 접근성 개선 노력에 대한 자세한 내용은 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc에서 Oracle Accessibility Program 웹 사이트를 방문하십시오.

오라클 고객지원센터 액세스

지원 서비스를 구매한 오라클 고객은 My Oracle Support를 통해 온라인 지원에 액세스할 수 있습니다. 자세한 내용은 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info를 참조하거나, 청각 장애가 있는 경우 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs를 방문하십시오.

목차

이 설명서 사용

- **개요** - 현재 OpenStack 버전을 설치하고 OpenStack 가상 시스템을 지원되는 Oracle Solaris 시스템에 배치하는 방법에 대해 설명합니다.
- **대상** - 대규모 설치 시스템 관리자
- **필요한 지식** - Oracle Solaris 네트워크 및 대규모 시스템 관리. OpenStack에 익숙한 것이 좋습니다.

제품 설명서 라이브러리

이 제품과 관련 제품들에 대한 설명서 및 리소스는 http://www.oracle.com/pls/topic/lookup?ctx=E69402에서 사용할 수 있습니다.

피드백

http://www.oracle.com/goto/docfeedback에서 이 설명서에 대한 피드백을 보낼 수 있습니다.

♦ ♦ ♦ 1 장

1 Oracle Solaris의 OpenStack(Kilo) 새로운 기능

Oracle Solaris 11.3 SRU 9에는 전체 패키지의 일부로 OpenStack 2015.1.2(Kilo) 버전이 포함되어 있습니다. 이 장에는 이 릴리스용 OpenStack Kilo 버전의 새 기능이 나와 있습니다.

핵심 구성요소 기반 기능

이 절에서는 OpenStack의 Kilo 버전에 대한 핵심 구성요소를 도입한 기능에 대해 설명합니다.

Cinder 관련 기능

다음과 같은 Cinder 기능이 추가되었습니다.

- 원격 SAN 저장소 사용

  SAN(Storage Area Networks) 지원을 통해 Cinder 서비스를 원격으로 배치할 수 있습니다. 자세한 내용은 "저장소용 원격 시스템 배치" [87]를 참조하십시오.
- 볼륨 백업 및 복원 작업에 대한 지원

  Cinder 백업 SMF 서비스를 이제 Oracle Solaris에서 사용할 수 있습니다. 따라서 연결되지 않은 볼륨을 구성된 백엔드 간에 백업하고 복원할 수 있습니다. 현재는 Swift가 유일하게 지원되는 백엔드입니다.
- Cinder 볼륨 마이그레이션에 대한 지원

  Cinder는 ZFS 작업을 사용하여 Cinder 볼륨을 마이그레이션합니다. ZFS 전송 및 수신 프로세스를 통해 구성된 여러 Cinder 백엔드 간에 볼륨을 마이그레이션할 수 있습니다. 마이그레이션 대상이 소스와 동일한 zpool에 있을 경우 ZFS 이름 바꾸기 작업이 사용됩니다. 현재는 Cinder 볼륨 마이그레이션 지원은 단일 시스템으로 제한됩니다.
- Cinder의 볼륨 관리가 *manage* 옵션을 사용하여 향상되었기 때문에 Cinder 기능 외부에서 만든 볼륨을 가져올 수 있습니다. 볼륨을 가져온 후에는 정규 Cinder 볼륨인 것처럼 클라우드에서 관리할 수 있습니다.

마찬가지로 *unmanage* 옵션을 사용하여 표시되지 않는 Cinder 볼륨을 렌더링하여 액세스를 사용 안함으로 설정할 수 있습니다. 이 옵션은 볼륨을 삭제하지 않습니다. 따라서 볼륨을 다시 가져와서 액세스를 다시 사용으로 설정할 수 있습니다.

관리/관리 해제 기능은 Horizon 대시보드뿐 아니라 명령줄에서도 사용할 수 있습니다.

- 새 등록 정보로 업데이트된 ZFSSA Cinder 드라이버는 현재 Kilo 구현에서 사용할 수 있습니다. OpenStack에서 ZFSSA 구성 설정을 조정하려면 사후 업그레이드 작업 [19]에서 지침을 참조하십시오.
- Solaris에 대한 OpenStack Cinder NFS 볼륨 드라이버 지원을 사용할 수 있습니다. `nfs` 유형의 볼륨을 만들 수 있습니다. NFS 파일 액세스가 Cinder에서 사용자 및 그룹으로 정의됩니다. 하지만 현재는 이 드라이버의 지원이 커널 영역으로만 제한됩니다.

  자세한 내용은 "Cinder NFS 드라이버 사용" [93]을 참조하십시오.
- 여러 백엔드가 있는 구성에서 만드는 모든 컴퓨트 노드에 대한 부트 볼륨을 지정할 수 있습니다. 이 기능으로 Cinder를 구성하려면 "컴퓨트 노드에 대한 부트 볼륨 지정" [92]을 참조하십시오.

Nova 관련 기능

다음과 같은 Nova 기능이 추가되었습니다.

- 보안 라이브 마이그레이션

  Oracle Solaris 영역의 기능인 라이브 마이그레이션에 대한 지원은 Nova 노드의 VM 인스턴스로 확장되었습니다. 노드 라이브 마이그레이션에 대한 자세한 내용은 "VM 인스턴스 마이그레이션 및 비우기" [82]를 참조하십시오.
- 인스턴스 비우기에 대한 지원

  호스트 실패가 발생하거나 서비스가 호스트에서 사용 안함으로 설정되는 경우에는 `nova evacuate` 명령을 사용하여 복구를 위해 다른 노드로 인스턴스를 이동할 수 있습니다. 비우기 지원은 루트 장치가 공유 저장소에 있는 경우에만 사용할 수 있습니다. 더욱이 비우기는 커널 영역에 대해서만 지원되고, 비전역 영역에 대해서는 지원되지 않습니다.
- VM 인스턴스 크기 조정 기능

  flavor를 변경하여 VM 인스턴스의 크기를 조정할 수 있습니다. 새 flavor는 CPU 용량, 메모리 및 기타 리소스와 같은 다른 등록 정보로 VM 인스턴스를 제공합니다. 자세한 내용은 "VM 인스턴스 크기 조정" [84]을 참조하십시오.

Neutron 관련 기능

다음과 같은 Neutron 기능이 추가되었습니다.

- 커널 내 Neutron 영역 기능

  영역의 동적 MAC 주소 및 VID에 대한 지원을 통해 커널 영역에서 Neutron을 설치할 수 있습니다. 자세한 내용은 "커널 영역에 Neutron 배치" [101]를 참조하십시오.

- VPN as a Service

  VPNaaS(VPN as a Service)는 Neutron을 통해 지원됩니다. 또한 "네트워크 IPsec 관리" 프로파일이 Neutron에 이미 지정된 프로파일에 추가됩니다. 이 프로파일을 사용하면 관리자가 IPsec 및 IKE SMF(System Management Facility) 서비스를 관리할 수 있습니다.

Oracle Solaris 기능 추가

이러한 기능 추가는 Oracle Solaris에 대한 OpenStack의 드라이브 측 개선 사항입니다. 이러한 개선 사항은 핵심 업스트림 프로젝트에 이미 구현되어 있습니다.

cloudbase-init에 대한 지원

cloudbase-init 서비스를 사용하면 클라우드의 게스트 운영체제에 대한 초기화 및 구성을 원활히 수행할 수 있습니다. 이 작업에는 사용자 만들기, 암호 생성, 정적 네트워킹 구성, 호스트 이름, SSH 공개 키 및 사용자 데이터 스크립트가 포함됩니다. 서비스의 구성 파일은 `/etc/cloudbase-init.conf`입니다.

cloudbase-init의 Oracle Solaris 버전은 SMF 서비스인 `application/cloudbase-init`로 실행되며 기본적으로 사용으로 설정됩니다. 사용자 데이터를 통해 내보낸 스크립트는 일반적으로 권한이 있는 액세스가 필요한 시스템 및 응용 프로그램 구성 작업을 수행합니다. 따라서 cloudbase-init 서비스는 사용자 루트로 실행되며 모든 사용자 데이터 스크립트도 루트로 실행됩니다.

cloudbase-init 패키지는 표준 그룹 패키지에 포함되지 않습니다. 사용자는 명시적으로 클라우드 환경에 배치할 이미지에만 패키지를 설치해야 합니다.

주 - 현재 이 OpenStack 릴리스에서 `/etc/cloudbase-init.conf` 파일은 UserData 플러그인만 사용으로 설정합니다.

Cloudbase-Init에 대한 자세한 내용을 보려면 http://cloudbase-init.readthedocs.io/en/latest/tutorial.html로 이동하십시오.

OpenStackClient 구현

OpenStackClient(OSC)는 동일한 명령 구조를 사용하여 모든 구성요소 명령 세트를 단일 셸에 결합하는 OpenStack 커뮤니티의 클라이언트입니다. 따라서 이전 버전에 `keystone user-list`, `glance image-show`, `neutron net-list` 등과 같은 구성요소를 기반으로 하는 명령이 있고 대부분의 명령은 `openstack user list`와 같은 기본 명령으로 `openstack`과 함께 실행됩니다.

현재 Kilo 버전에서 모든 `keystone` 명령은 더 이상 사용되지 않습니다. `keystone` 명령을 사용하면 해당 경보가 생성됩니다.

OSC에 대한 자세한 내용은 http://docs.openstack.org/developer/python-openstackclient/index.html을 참조하십시오.

이전 명령 및 OSC에서 이에 상응하는 명령 목록은 부록 B. OpenStackClient 명령을 참조하십시오.

기타 정보 소스

https://wiki.openstack.org/wiki/ReleaseNotes/Kilo에서 OpenStack 커뮤니티의 Kilo 릴리스가 제공하는 정보도 참조하십시오.

이 OpenStack 릴리스의 현재 Oracle Solaris 구현에 관련된 문제 목록은 My Oracle Support (https://support.oracle.com)의 OpenStack 버전에 해당하는 Readme 파일을 참조하십시오.

♦ ♦ ♦ 2 장

2 OpenStack 2015.1.2(Kilo)로 업그레이드

이 장에서는 이전 버전에서 OpenStack Kilo 버전으로 업그레이드하기 위한 단계에 대해 논의합니다.

업그레이드 노트

업그레이드할 때 다음 모범 사례를 고려해 보십시오.

- 특히 전체 운영체제가 아닌 OpenStack 구성만 업그레이드하는 경우 백업을 만듭니다.
- OpenStack 구성만 업그레이드하는 경우 업그레이드된 OpenStack 버전으로 새로운 BE(부트 환경)가 만들어집니다. 업그레이드가 완료되면 새 BE로 부트합니다.
- 다중 노드 구성에서 컨트롤러 노드를 먼저 업데이트한 다음 저장소 노드를 업데이트하고 나머지 노드를 업데이트합니다.
- Kilo에서 RabbitMQ에 대한 기본 게스트 사용자 및 암호 설정이 사용 안함으로 설정됩니다. 이전 OpenStack 버전에서 기본 암호 설정을 사용한 경우 안전한 암호로 변경합니다.

업그레이드를 위한 절차

Kilo 업그레이드를 위한 단계는 시작하는 OpenStack 버전에 따라 다릅니다. Havana에서 Kilo로의 직접 업그레이드는 지원되지 않습니다. 현재 버전이 Havana인 경우 먼저 Juno로 업그레이드한 다음 Juno에서 Kilo로 다른 업그레이드를 수행해야 합니다.

주 - 업그레이드를 수행하는 경우 `pkg update` 명령을 사용하여 작업을 시작해야 합니다. `pkg install`을 사용하지 마십시오.

업그레이드 로드맵은 다음과 같습니다.

- 기존 Havana 구성에서는 Havana에서 Kilo로 업그레이드하는 방법에서 시작합니다. 그런 다음 사후 업그레이드 작업을 진행합니다.

- 기존 Juno 구성에서는 Juno에서 Kilo로 업그레이드하는 방법에서 시작합니다. 그런 다음 사후 업그레이드 작업을 진행합니다.

▼ Havana에서 Kilo로 업그레이드하는 방법

현재 OpenStack 버전이 Oracle Solaris 11.2 SRU 11 또는 이전 릴리스에서 실행 중인 Havana인 경우 이 절차를 사용합니다.

OpenStack 구성이 다중 노드에서 작성되는 경우 컨트롤러부터 시작하여 각 노드에서 이 절차를 수행해야 합니다.

Oracle Solaris 업그레이드는 운영체제에 대한 새 BE(부트 환경)를 만듭니다. 업그레이드가 완료되면 새 BE가 활성화됩니다. 재부트 시 시스템은 새 BE로 부트됩니다.

1. **시스템이 Oracle Solaris 11.2 SRU 11 또는 이전 릴리스에서 실행 중인 경우 다음 단계를 수행합니다.**

   a. **운영체제를 Oracle Solaris 11.3 이상으로 업그레이드합니다.**

   주 - 여기서 아직 Oracle Solaris 11.3 SRU 9로 업그레이드하지 마십시오.

   Oracle Solaris 11.3 패키지에는 OpenStack Juno 패키지가 포함되어 있습니다.

   Oracle Solaris 11.3으로 업그레이드하기 위한 단계는 *Updating to Oracle Solaris 11.3* 의 "How to Update a System Running 11.1 or 11.2 to Oracle Solaris 11.3"을 참조하십시오.

   b. **Neutron 데이터베이스 마이그레이션에 대한 정보가 설정에 적용되는지 여부를 확인하려면 선택합니다.**

   Migrating Neutron Database from sqlite to MySQL for Oracle OpenStack for Oracle Solaris을 참조하십시오. 시나리오가 구성에 적용되는 경우 블로그 항목의 지침을 따릅니다.

2. **Oracle Solaris 11.3 SRU 9로 운영체제를 업그레이드합니다.**

   Oracle Solaris 11.3 SRU 9 릴리스로 업데이트하는 방법에 대한 지침을 보려면 `https://support.oracle.com`에서 MOS 계정으로 로그인합니다. Oracle Solaris 11.3 SRU(Support Repository Updates) 인덱스(Doc ID 2045311.1)에서 SRU9 Readme 파일에 액세스합니다.

3. **새 BE로 시스템을 부트합니다.**

다음 순서 사후 업그레이드 작업 [19]의 모든 단계를 완료하여 업그레이드 프로세스를 완료합니다.

▼ Juno에서 Kilo로 업그레이드하는 방법

현재 OpenStack 버전이 Juno인 경우 이 절차를 사용합니다.

OpenStack 구성이 다중 노드에서 작성되는 경우 컨트롤러부터 시작하여 각 노드에서 이 절차를 수행해야 합니다.

Oracle Solaris 업그레이드는 운영체제에 대한 새 BE(부트 환경)를 만듭니다. 업그레이드가 완료되면 새 BE가 활성화됩니다. 재부트 시 시스템은 새 BE로 부트됩니다.

1. **Oracle Solaris 11.3 SRU 9로 운영체제를 업그레이드합니다.**
   Oracle Solaris 11.3 SRU 9 릴리스로 업데이트하는 방법에 대한 지침은 릴리스의 Readme 파일을 참조하십시오.

2. **업그레이드 프로세스를 완료하려면 사후 업그레이드 작업을 진행합니다.**

▼ 사후 업그레이드 작업

현재 Oracle Solaris 릴리스로 업그레이드한 후 나머지 단계를 수행하여 OpenStack Kilo로의 업그레이드를 완료합니다.

1. **Horizon 사용자정의를 Kilo 버전으로 마이그레이션합니다.**

   a. **`/etc/openstack_dashboard/local_settings.py.old`에서 `/etc/openstack_dashboard/local_settings.py`로 사용자정의 설정을 전송합니다.**

   b. **다음 중 하나가 구성에 적용되는 경우 `/etc/openstack_dashboard/local_settings.py` 파일에 추가 행을 주석 처리합니다.**

      - 평가를 위한 단일 노드 OpenStack 구성이 있음
      - Horizon 구성에서 SSL을 사용하지 않음

      다음 예를 참조하십시오.

      ```
      # SECURE_PROXY_SSL_HEADER = ('HTTP_X_FORWARDED_PROTOCOL', 'https')
      # CSRF_COOKIE_SECURE = True
      # SESSION_COOKIE_SECURE = True
      ```

   c. **샘플 Horizon Apache 구성 단편을 Apache `conf.d` 디렉토리에 복사합니다.**
      사용 중인 프로토콜에 해당하는 샘플 단편만 복사해야 합니다. 다음 명령 중 하나를 실행합니다.

      - HTTP를 사용 중인 경우 다음과 같이 합니다.

```
# cp /etc/apache2/2.4/samples-conf.d/openstack-dashboard-http.conf /etc/
apache2/2.4/conf.d
```

- TLS를 사용 중인 경우 다음과 같이 합니다.

```
# cp /etc/apache2/2.4/samples-conf.d/openstack-dashboard-tls.conf /etc/
apache2/2.4/conf.d
```

2. **다중 노드 구성을 사용 중인 경우 굵게 표시된 행으로 /etc/rabbitmq/rabbitmq.config를 업데이트합니다.**

```
% FHC read buffer has been disabled by default in later versions of
%RabbitMQ.
[
   {rabbit, [
     {fhc_read_buffering, false},
     {loopback_users, []}
   ]}
].
```

3. **Cinder v2 서비스를 업데이트합니다.**

   Keystone이 실행 중인 노드에서 다음 단계를 수행합니다. 샘플 출력은 실행하는 각 명령에 대해 포함되어 있습니다.

   a. **v2 Cinder 서비스를 만듭니다.**

```
controller# openstack --os-url http://$CONTROLLER_ADMIN_NODE:35357/v2.0 \
   --os-token ADMIN \
    service create --name cinderv2 \
   --description "Cinder Volume Service v2" volumev2
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Cinder Volume Service v2         |
| enabled     | True                             |
| id          | 2ee6fefbdcdc4f06bcb0e36e0e4dd9c3 |
| name        | cinderv2                         |
| type        | volumev2                         |
+-------------+----------------------------------+
```

   b. **끝점을 만듭니다.**

```
controller# openstack --os-url http://$CONTROLLER_ADMIN_NODE:35357/v2.0 \
   --os-token ADMIN
   endpoint create \
   --region RegionOne \
   --publicurl "http://$CONTROLLER_ADMIN_NODE:8776/v2/\$(tenant_id)s" \
   --adminurl "http://$CONTROLLER_ADMIN_NODE:8776/v2/\$(tenant_id)s" \
   --internalurl "http://$CONTROLLER_ADMIN_NODE:8776/v2/\$(tenant_id)s" cinderv2

+-------------+-----------------------------------------+
```

```
| Field        | Value                                    |
+--------------+------------------------------------------+
| adminurl     | http://controller-node:8776/v2/$(tenant_id)s |
| id           | 1b8cd962b12342429cdedb0c7e5d0440       |
| internalurl  | http://controller-node:8776/v2/$(tenant_id)s |
| publicurl    | http://controller-node:8776/v2/$(tenant_id)s |
| region       | RegionOne                                |
| service_id   | 2ee6fefbdcdc4f06bcb0e36e0e4dd9c3         |
| service_name | cinderv2                                 |
| service_type | volumev2                                 |
+--------------+------------------------------------------+
```

c. **cinderv2가 끝점 목록에 존재하는지 확인합니다.**

```
controller# openstack --os-url http://$CONTROLLER_ADMIN_NODE:35357/v2.0 --os-token
 ADMIN endpoint list
+----------------------------------+-----------+--------------+----------------+
| ID                               | Region    | Service Name | Service Type   |
+----------------------------------+-----------+--------------+----------------+
| 6891354066f84268968c8498f5f6d51b | RegionOne | neutron      | network        |
| 03121908d41e4efa98748fde8ca6d057 | RegionOne | heat         | orchestration  |
| b69e4f0373ff4a8f9560dc2644d891ba | RegionOne | glance       | image          |
| 1e6c7f52dcd34a27b7ccac98918f19f1 | RegionOne | ec2          | ec2            |
| e3236915a3dd4098b9e8e0853b5a5af2 | RegionOne | keystone     | identity       |
| fe8870c3e6ac4b529aa7ce7563fc24a4 | RegionOne | heat-cfn     | cloudformation |
| aa931a795f2c4c0ca637e0e4c351cf07 | RegionOne | swift        | object-store   |
| 1b8cd962b12342429cdedb0c7e5d0440 | RegionOne | cinderv2     | volumev2       |
| 618a8edba487417c91d0de7f3bcc786d | RegionOne | cinder       | volume         |
| 4c79d020189a44d383bdc15033a942c4 | RegionOne | nova         | compute        |
+----------------------------------+-----------+--------------+----------------+
```

4. **Apache 서비스를 다시 시작합니다.**

```
# svcadm restart apache24
```

5. **실행 중이 아닌 경우 IP 필터 서비스를 시작합니다.**

```
controller# svcadm enable -rs ipfilter
```

6. **저장에 ZFSSA를 사용 중인 경우 `/etc/cinder.cinder.conf`를 조정하여 새 드라이버 등록 정보를 사용합니다.**

   `zfssa_initiator_config` 등록 정보가 여러 개시자 또는 여러 개시자 그룹을 나열하고 OpenStack Kilo 버전에서 더 이상 사용되지 않는 `zfssa_initiator_group`을 대체합니다.

   a. **다음 형식을 사용하여 새 등록 정보에 대한 여러 개시자를 나열합니다.**

```
zfssa_initiator_config = {
    'init-grp1': [
                {'iqn':'iqn1' , 'user':'user' , 'password':'password'},
                {'iqn':'iqn2' , 'user':'user' , 'password':'password'}
                ],
    'init-grp2': [
```

```
                {'iqn':'iqn3' , 'user':'user' , 'password':'password'}
                ] }
```

예를 들어 그룹 A 및 그룹 B와 같은 개시자 그룹 두 개가 ZFS 저장소 어플라이언스에서 생성되는 경우 다음과 같이 정의합니다.

```
zfssa_initiator_config = {
        'GroupA':[
            {'iqn':'iqn.1986-03.com.sun:01:0a43b9fdcfd5.570d7fd1', 'user':'test1',
 'password':'password1234'},
            {'iqn':'iqn.1986-03.com.sun:01:0a43b9fdcfd5.570d7fd2', 'user':'',
 'password':''}
        ],
        'GroupB':[
            {'iqn':'iqn.1986-03.com.sun:01:0a43b9fdcfd5.570d7fd3', 'user':'',
 'password':''}
        ] }
```

b. 파일에서 더 이상 사용되지 않는 다음 매개변수를 주석 처리합니다.

- `zfssa_initiator_group`
- `zfssa_initiator`

c. Cinder 서비스를 다시 시작합니다.

```
controller# svcadm restart cinder-volume:default
```

♦ ♦ ♦ 3 장

3 다중 노드 OpenStack 구성을 위해 여러 시스템에 걸쳐 설치

이 장에서는 다중 노드 OpenStack 구성을 설치하는 방법에 대해 설명합니다. 여기에서는 다음 내용을 다룹니다.

3노드 아키텍처 개요

단일 노드 구성은 OpenStack을 제품으로 테스트하고 해당 기능을 익히는 데 유용합니다. 하지만 단일 노드 구성은 운용 환경에는 적합하지 않습니다. 이 환경에서는 여러 시스템 또는 노드에 걸쳐 OpenStack을 설치하고 구성합니다.

각 클라우드에는 대시보드 인스턴스, 이미지 저장소 및 ID 서비스가 하나씩만 필요하며, 저장소와 Compute 인스턴스가 여러 개 포함될 수 있습니다. 특정 클라우드 배치에 대한 요구사항과 관련하여 각 구성요소를 평가한 후 구성요소를 별도의 노드에 설치해야 할지 여부와 필요한 해당 유형의 노드 수를 결정하십시오.

이 장에서 설명되는 아키텍처는 다음과 같은 세 시스템에 배치됩니다.

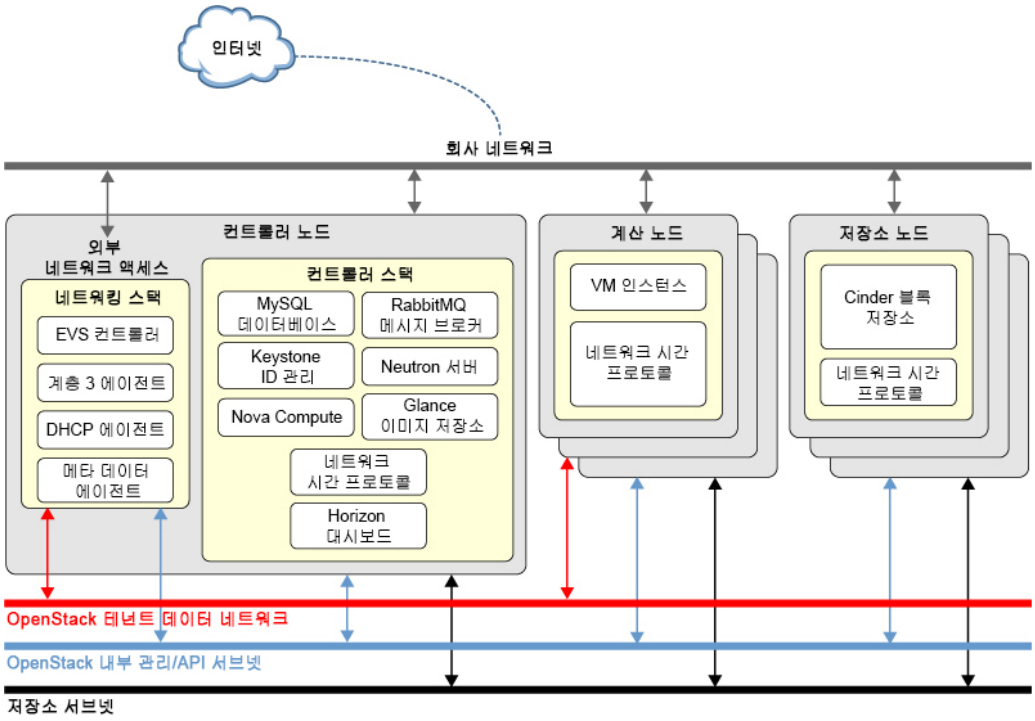
- 컨트롤러 노드 - 대부분의 공유 OpenStack 서비스 및 기타 도구가 실행되는 노드입니다. 컨트롤러 노드는 API, 예약 및 기타 클라우드용 공유 서비스를 제공합니다. 컨트롤러 노드에는 대시보드, 이미지 저장소 및 ID 서비스가 있습니다. 추가적으로 Nova Compute 관리 서비스와 Neutron 서버도 이 노드에 구성됩니다.
- 컴퓨트 노드 - Nova 컴퓨트 인스턴스라고도 하는 VM 인스턴스가 설치되는 노드입니다. 이 노드에서는 이러한 VM 인스턴스를 관리하는 컴퓨트 데몬이 실행됩니다.
- 저장소 노드 - 데이터를 호스트하는 노드입니다.

이 3노드 아키텍처는 여러 시스템에 OpenStack을 배치하는 유일한 방법입니다. 유연성 덕분에 이 아키텍처가 아닌 방법으로 OpenStack 구성요소를 배포할 수 있습니다. 따라서 설치를 시작하기 전에 클라우드 구성을 계획해야 합니다. 계획에 대한 자세한 내용은 *OpenStack 구성 계획*을 참조하십시오.

주 - 단일 Oracle SPARC 서버를 분할하고 OVM Server for SPARC(LDoms)를 실행하는 서버에서 다중 노드 OpenStack을 구성하려면 Multi-node Solaris 11.2 OpenStack on SPARC Servers를 참조하십시오. 이 문서는 Havana 버전의 OpenStack에 해당하는 내용입니다. 하지만 일반적인 단계는 현재 버전에도 적용됩니다.

다음 그림은 이 장에서 설명되는 높은 레벨의 아키텍처 뷰를 보여줍니다.

그림 1 3노드 구성 참조 아키텍처



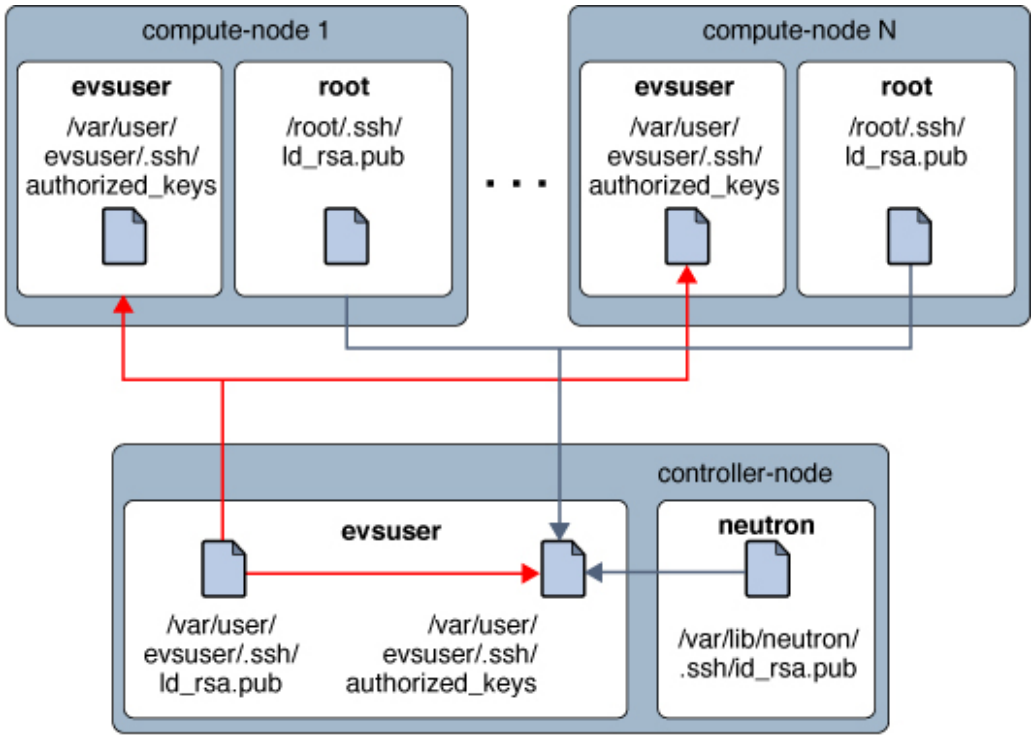
이 그림은 Cinder를 사용하여 저장소 노드를 설명합니다. 그러나 Swift 객체 저장소 서비스도 비슷하게 구성할 수 있습니다.

Oracle Solaris에서 EVS(탄력적 가상 스위치)는 OpenStack 네트워킹의 백엔드를 구성합니다. EVS를 사용하면 VLAN 또는 VXLAN에 있는 VM 인스턴스 간 통신이 용이해집니다.

VM 인스턴스는 하나의 컴퓨트 노드에 있거나 여러 컴퓨트 노드에 걸쳐 있을 수 있습니다. EVS에 대한 자세한 내용은 탄력적 가상 스위치에 대한 네트워크 가상화 및 네트워크 리소스 관리를 참조하십시오. 이 설명서는 해당 Oracle Solaris 버전의 라이브러리(Operating Systems Documentation)에 있습니다.

다른 노드가 서로 통신하도록 하려면 컨트롤러 노드의 `evsuser`, `neutron` 및 `root`의 SSH 공개 키가 구성된 모든 컴퓨트 노드에 있는 `evsuser`의 각 `authorized_keys` 파일에 있어야 합니다. 다음 그림에서 SSH 공개 키 배포를 참조하십시오. 그림에서는 여러 컴퓨트 노드가 구성된 것으로 가정합니다.

그림 2 EVS 컨트롤러 SSH 키 배포



Oracle Solaris 시스템에 OpenStack을 배치하는 데 유용한 OpenStack 구성 매개변수 목록은 부록 A. 공통 OpenStack 구성 파일 및 서비스를 참조하십시오.

예비 단계

이 절에서는 다중 노드 OpenStack 구성을 구현하기 전에 몇 가지 사전 고려 사항을 설명합니다.

호스트 이름, 변수 및 암호 준비

다중 노드 구성에서는 여러 네트워크 인터페이스를 사용하여 클라우드에 대해 만든 다양한 서브넷을 제공합니다. 이 인터페이스의 호스트 이름을 준비했는지 확인합니다. 이 이름과 해당 IP 주소를 시스템의 `/etc/hosts` 파일이나 DNS 구성에 포함하십시오.

예를 들어, 여러 유형의 네트워크 트래픽을 처리하기 위해 다음과 같은 호스트 이름을 만들 수 있습니다.

- `host-on`: 관리 및 API 트래픽을 호스트하는 OpenStack 네트워크용
- `host-tn`: 컴퓨트 노드와 L3 라우터 간 트래픽을 호스트하는 프로젝트 네트워크용
- `host-en`: 외부 네트워크 트래픽용

OpenStack 서비스를 여러 노드에 구성할 때 다음과 같이 작업을 효율화하는 변수를 만듭니다.

- `$CONTROLLER_ADMIN_NODE` - OpenStack 관리 서비스가 연결될 컨트롤러 노드에서 인터페이스 또는 IP 주소의 호스트 이름입니다.
- `$CONTROLLER_ADMIN_NODE_IP` - OpenStack 관리 서비스 및 트래픽을 처리하는 컨트롤러 포트의 IP 주소
- `$COMPUTE_ADMIN_NODE_IP` - OpenStack 관리 서비스 및 트래픽을 처리하는 컴퓨트 포트의 IP 주소
- `$VOLUME_IP` - 컨트롤러 노드의 호스트 이름

구성 프로세스에서 암호도 필요합니다. 다음은 준비해야 하는 샘플 암호 목록입니다.

- MySQL 데이터베이스에 대한 루트 암호
- `admin` 사용자의 암호
- OpenStack 서비스의 데이터베이스 암호:
  - ID 서비스
  - 이미지 서비스
  - Compute 서비스
  - 대시보드 데이터베이스
  - 블록 저장소 데이터베이스
  - 네트워킹 데이터베이스
  - 조정 데이터베이스

- OpenStack 서비스 사용자의 암호:
  - `glance`
  - `nova`
  - `cinder`
  - `neutron`
  - `heat`

주 - 사용자 또는 서비스 그룹에 대해 공통 암호를 지정할 수도 있습니다. 암호 지정을 위해 어떠한 시스템을 채용하더라도 모범 사례에 따라 환경을 보호해야 합니다. Operating Systems Documentation의 사용 중인 Oracle Solaris 버전 라이브러리에서 시스템 및 연결된 장치 보안을 참조하십시오.

샘플 Keystone 스크립트

샘플 스크립트 `/usr/demo/openstack/keystone/sample_data.sh`를 사용하면 Keystone 데이터베이스를 빠르게 채울 수 있습니다. 스크립트는 다음과 같은 기본 작업을 수행하여 시작할 수 있도록 도와줍니다.

- 다음 초기 프로젝트를 만듭니다.
  - 기본 또는 핵심 서비스가 만들어지는 `service`
  - 기본 암호로 `secrete`를 가진 사용자 `admin`이 만들어지는 `demo`
- Keystone 데이터베이스를 채웁니다.
- 다음 핵심 서비스를 만듭니다.
  - cinder
  - cinder2
  - ec2
  - glance
  - keystone
  - neutron
  - nova
  - swift
  - heat

  Keystone 서비스를 제외한 모든 서비스에는 해당 사용자 이름 및 암호가 있으며 사용자가 생성되지 않습니다. 기본적으로 사용자 이름, 암호 및 서비스 이름은 동일합니다. 예를 들어 `cinder`는 Cinder 서비스의 사용자 이름 및 암호이고 `glance`는 Glance 서비스의 사용자 이름 및 암호입니다. 선택적으로 스크립트에서 이와 같은 기본값을 대체할 사용자정의 암호를 만들 수 있습니다. 또는 스크립트의 모든 서비스에 대해 단일 암호를 설정할 수도 있습니다. 부트스트랩 Keystone으로 실행하기 전에 스크립트에 모든 필요한 변경사항을 적용하십시오.

주 - 스크립트에서 환경에 대해 설정할 수 있는 매개변수에 대한 자세한 내용을 검토하십시오. 환경설정에 따라 스크립트의 기본 설정을 바꾸십시오.

이 문서의 모든 절차에서는 암호를 제외하고 샘플 데이터 스크립트가 개정 없이 사용되고 모든 스크립트의 기본 설정이 클라우드 구성에 적용된다고 간주합니다.

구성 파일 편집 정보

OpenStack 구성의 중요한 부분은 구성요소의 구성 파일 편집입니다. 이 문서에서는 각 `*.conf` 또는 `*.ini` 파일에서 선택된 매개변수만 구성에 대해 식별됩니다. 이러한 선택된 매개변수는 클라우드 구성의 작동에 필요한 최소한의 매개변수입니다. 하지만 각 구성 파일의 전체 내용을 검토하여 해당하는 특정 클라우드 설정과 관련된 모든 매개변수가 제대로 구성되었는지 확인하십시오.

메모리 사용 최적화

Oracle Solaris 11에서 ZFS와 응용 프로그램 간 메모리 사용을 보다 잘 관리하려면 다음 예에 표시된 것과 같이, 노드에 대해 `usr_reserve_hint_pct` 매개변수를 설정합니다.

```
# echo "set user_reserve_hint_pct=80" >>/etc/system.d/site:kernel-zones-reserve
# reboot
```

여기서 *site*는 회사를 나타낼 수 있습니다.

이 매개변수를 다른 OpenStack 노드에도 설정합니다.

이 매개변수에 대한 자세한 내용은 https://support.oracle.com에서 MOS 계정에 로그인한 다음 Document 1663862.1, *Memory Management Between ZFS and Applications in Oracle Solaris 11.2*를 검토하십시오.

NTP 서버 구성

NTP(Network Time Protocol) 설치는 선택사항이지만 강력히 권장됩니다. NTP는 클라우드의 모든 서비스 노드에서 시간을 일관성 있게 유지할 수 있도록 도와줍니다. 네트워크에서 NTP를 사용으로 설정하는 경우 네트워크를 통해 시간을 가져오도록 서비스 노드를 구성하십시오.

- 서비스 노드가 상주하는 IP 서브넷에서 IP 멀티캐스트가 사용으로 설정된 경우 IP 멀티캐스트를 활용하여 NTP를 구성할 수 있습니다.
- 서비스 노드가 상주하는 IP 서브넷에서 IP 멀티캐스트가 사용으로 설정되지 않은 경우 NTP를 수동으로 구성하십시오.

NTP 사용에는 NTP 서버와 NTP 클라이언트 구성이 수반됩니다. 일반적으로 NTP 서버는 OpenStack을 구성한 시스템과 다른 별도의 시스템입니다. NTP 클라이언트는 OpenStack 구성요소를 호스트하는 시스템이나 노드에 설치 및 구성됩니다.

NTP에 대한 자세한 내용은 http://www.ntp.org/documentation.html에서 설명서를 참조하십시오.

▼ NTP 서버를 설정하는 방법

NTP 서버는 OpenStack 노드와 다른 별도의 시스템에 있습니다.

1. **NTP 패키지를 설치합니다.**

```
ntp-server# pkg install ntp
```

2. **구성 파일을 설치합니다.**

```
ntp-server# cp /etc/inet/ntp.server /etc/inet/ntp.conf
```

3. **`/etc/inet/ntp.conf` 파일에서 `server` 및 `driftfile` 키워드를 구성하여 편집합니다.**
   예를 들어 다음을 실행합니다.

```
server 127.127.1.0 prefer
...
driftfile /var/ntp/ntp.drift
```

주 - `127.127.1.0`은 IP 주소가 **아닙니다**. 서버에 정확한 시간을 제공하는 클럭을 참조할 때 사용되는 형식입니다. `ntp.conf` 파일에서 `server` 키워드를 설명하는 주석을 읽어야 합니다.

4. **이전 단계에서 정의한 대로 `/var/ntp/ntp.drift` 파일을 만듭니다.**

```
ntp-server# touch /var/ntp/ntp.drift
```

5. **`ntp` 서비스를 시작합니다.**

```
ntp-server# svcadm enable ntp
```

컨트롤러 노드 구성

컨트롤러 노드에는 대시보드 서비스, 이미지 저장소 및 ID 서비스가 하나씩 필요하며, MySQL, RabbitMQ와 Compute, 블록 저장소 및 네트워킹 서비스도 포함됩니다.

컨트롤러 노드를 구성하려면 다음 명령을 사용하여 시스템에 OpenStack 구성요소 및 서비스를 설치합니다.

```
controller# pkg install openstack
```

패키지 설치를 완료한 후 노드에서 실행할 서비스를 구성합니다. 다음 목록은 컨트롤러 노드를 구성하는 작업을 보여줍니다.

NTP 클라이언트 구성

클라우드 배치 내 각 서비스 노드에 NTP 클라이언트 서비스를 설치합니다.

▼ NTP 클라이언트를 구성하는 방법

이 절에서는 NTP 서버를 설정하는 방법 [29]에 설명된 대로 NTP 서버를 이미 설정했다고 가정합니다..

1. **클라이언트 구성 파일을 만듭니다.**

   ```
   controller# cp /etc/inet/ntp.client /etc/inet/ntp.conf
   ```

2. **클라이언트 구성 파일에서 하나 이상의 서버 옵션에 대한 주석 처리를 해제하고 NTP 서버의 특정 이름 또는 IP 주소를 제공합니다.**

   예를 들어, 구성된 NTP 서버의 호스트 이름이 `system1`이면 구성 파일은 다음 예와 비슷합니다.

   ```
   # multicastclient 224.0.1.1
   ...
   server system1.example.com iburst
   # server server_name2 iburst
   # server server_name3 iburst
   ```

3. **`ntp` 서비스를 사용으로 설정합니다.**

   ```
   controller# svcadm enable ntp
   ```

MySQL 설치

여러 OpenStack 서비스는 데이터베이스 유지 관리를 통해 중요한 리소스, 사용량 및 기타 정보를 추적합니다. 특히 다중 노드 구성의 경우 이 정보를 저장하는 데 MySQL 데이터베이스와 같은 데이터베이스를 사용합니다.

▼ MySQL 데이터베이스를 설치하는 방법

1. **RabbitMQ 서비스를 사용으로 설정합니다.**

```
controller# svcadm enable rabbitmq
controller# svcadm restart rad:local
```

2. **(옵션) 관리 및 API 트래픽에 전용 IP 주소를 사용하는 경우 `/etc/mysql/5.5/my.cnf`에 해당 주소를 추가합니다.**

```
bind-address=$CONTROLLER_ADMIN_NODE_IP
```

3. **MySQL 서비스를 사용으로 설정합니다.**

```
controller# svcadm enable mysql
```

4. **MySQL 서버 `root` 암호를 설정합니다.**

```
controller# mysqladmin -u root password MySQL-root-password
```

5. **MySQL을 구성합니다.**

OpenStack에서 사용될 테이블을 만듭니다. 해당 데이터베이스에 대해 배타적 액세스를 제공하기 위해 컨트롤러 노드의 서비스에 권한을 부여합니다.

```
controller# mysql -u root -p
Enter password: MySQL-root-password
mysql> drop database if exists nova;
mysql> drop database if exists cinder;
mysql> drop database if exists glance;
mysql> drop database if exists keystone;
mysql> drop database if exists neutron;
mysql> drop database if exists heat;
mysql> create database cinder default character set utf8 default collate utf8_general_ci;
mysql> grant all privileges on cinder.* to 'cinder'@'$CONTROLLER_ADMIN_NODE' identified
 by 'service-password';
mysql> create database glance default character set utf8 default collate utf8_general_ci;
mysql> grant all privileges on glance.* to 'glance'@'$CONTROLLER_ADMIN_NODE' identified
 by 'service-password';
mysql> create database keystone default character set utf8 default collate
 utf8_general_ci;
mysql> grant all privileges on keystone.* to 'keystone'@'$CONTROLLER_ADMIN_NODE'
 identified by 'service-password';
```

```
mysql> create database nova default character set utf8 default collate utf8_general_ci;
mysql> grant all privileges on nova.* to 'nova'@'$CONTROLLER_ADMIN_NODE' identified by
 'service-password';
mysql> create database neutron default character set utf8 default collate
 utf8_general_ci;
mysql> grant all privileges on neutron.* to 'neutron'@'$CONTROLLER_ADMIN_NODE' identified
 by 'service-password';
mysql> create database heat default character set utf8 default collate utf8_general_ci;
mysql> grant all privileges on heat.* to 'heat'@'$CONTROLLER_ADMIN_NODE' identified by
 'service-password';
mysql> flush privileges;
mysql> quit
```

Keystone 설치

Keystone 서비스는 컨트롤러 노드에서 설치 및 구성해야 합니다. 이 절차에서는 "샘플 Keystone 스크립트" [27]에 설명된 샘플 스크립트를 사용합니다. 스크립트를 사용하기 전에 이 절을 읽으십시오.

▼ Keystone을 설치 및 구성하는 방법

1. **Keystone 및 기타 OpenStack 서비스에 대한 공유 토큰을 만듭니다.**
   이 토큰은 무작위 문자열로 구성됩니다. `openssl` 명령은 국가, 시/도 등의 키를 구성하는 구성요소를 묻습니다.

   ```
   controller# openssl rand -hex 10
   token-string
   ```

2. **토큰을 셸 변수로 설정합니다.**

   ```
   controller# export MY_SERVICE_TOKEN=token-string
   ```

   여기서 *token-string*은 이전 단계 명령의 출력입니다.

3. **`/etc/keystone/keystone.conf` 파일에서 매개변수를 수정합니다.**
   구성은 다음 예제와 유사해야 합니다.

   ```
   [DEFAULT]
   admin_token = token-string
   ...
   [database]
   connection = mysql://keystone:service-password@$CONTROLLER_ADMIN_NODE/keystone

   [oslo_messaging_rabbit]
   rabbit_host=$CONTROLLER_ADMIN_NODE
   ```

4. **Keystone SMF 서비스를 사용으로 설정합니다.**

```
controller# svcadm enable keystone
```

5. **Keystone 샘플 스크립트를 사용하여 Keystone 데이터베이스를 채웁니다.**

   스크립트를 실행하기 전에 스크립트를 검토하고 환경설정에 따라 수정했는지 확인합니다. 절차에서는 샘플 스크립트가 사용자정의되지 않았다고 간주합니다.

```
controller# CONTROLLER_PUBLIC_ADDRESS=$CONTROLLER_ADMIN_NODE \
CONTROLLER_ADMIN_ADDRESS=$CONTROLLER_ADMIN_NODE \
CONTROLLER_INTERNAL_ADDRESS=$CONTROLLER_ADMIN_NODE \
SERVICE_TOKEN=$MY_SERVICE_TOKEN \
ADMIN_PASSWORD=admin-password
SERVICE_PASSWORD=service-password
/usr/demo/openstack/keystone/sample_data.sh
```

Glance 설치

Glance를 설정하려면 MySQL 및 RabbitMQ 서비스의 위치를 지정하고 인증에 대한 일부 정보를 구성해야 합니다.

▼ Glance를 설치 및 구성하는 방법

1. **다음 구성 파일에서 매개변수를 설정하거나 주석을 해제하여 Glance를 구성합니다.**

   - /etc/glance/glance-api.conf

```
[DEFAULT]
registry_host = $CONTROLLER_ADMIN_NODE

auth_strategy = keystone
default_publisher_id =image.$CONTROLLER_ADMIN_NODE

[database]
connection = mysql://glance:service-password@$CONTROLLER_ADMIN_NODE/glance

[keystone_authtoken]
auth_uri= http://$CONTROLLER_ADMIN_NODE:5000/v2.0
identity_uri = http://$CONTROLLER_ADMIN_NODE:35357
admin_user = glance
admin_password = service-password
admin_tenant_name = service

[oslo_messaging_rabbit]
```

```
rabbit_host=$CONTROLLER_ADMIN_NODE
```

- /etc/glance/glance-cache.conf

```
[DEFAULT]
auth_url = http://$CONTROLLER_ADMIN_NODE:5000/v2.0/
admin_user = glance
admin_password = service-password
admin_tenant_name = service
```

- /etc/glance/glance-registry.conf

```
[DEFAULT]
default_publisher_id = image.$CONTROLLER_ADMIN_NODE

[database]
connection = mysql://glance:service-password@$CONTROLLER_ADMIN_NODE/glance

[keystone_authtoken]
auth_uri = http://$CONTROLLER_ADMIN_NODE:5000/v2.0
identity_uri = http://$CONTROLLER_ADMIN_NODE:35357
admin_user = glance
admin_password = service-password
admin_tenant_name = service

[oslo_messaging_rabbit]
rabbit_host=$CONTROLLER_ADMIN_NODE
```

- /etc/glance/glance-scrubber.conf

```
[DEFAULT]
auth_url = http://$CONTROLLER_ADMIN_NODE:5000/v2.0/
identity_uri = http://$CONTROLLER_ADMIN_NODE:35357
admin_user = glance
admin_password = service-password
admin_tenant_name = service

[database]
connection=mysql://glance:service-password@$CONTROLLER_ADMIN_NODE/glance
```

2. **Glance SMF 서비스를 사용으로 설정합니다.**

```
controller# svcadm enable -rs glance-api glance-db glance-registry glance-scrubber
```

Nova 설치

이 절은 컴퓨트 노드 자체가 아니라 Nova 끝점 서비스 구성에 적용됩니다.

▼ Nova를 설치 및 구성하는 방법

1. **/etc/nova/nova.conf 파일에 매개변수를 설정하거나 주석 처리 해제하여 Nova를 구성합니다.**

```
[DEFAULT]
my_ip=$CONTROLLER_ADMIN_NODE_IP
host=$CONTROLLER_ADMIN_NODE
firewall_driver=nova.virt.firewall.NoopFirewallDriver

[database]
connection = mysql://nova:service-password@$CONTROLLER_ADMIN_NODE/nova

[glance]
host=$CONTROLLER_ADMIN_NODE

[keystone_authtoken]
auth_uri=http://$CONTROLLER_ADMIN_NODE:5000/v2.0/
identity_uri=http://$CONTROLLER_ADMIN_NODE:35357/
admin_user=nova
admin_password=service-password
admin_tenant_name=service

[neutron]
url=http://$CONTROLLER_ADMIN_NODE:9696
admin_username=neutron
admin_password=service-password
admin_tenant_name=service
admin_auth_url=http://$CONTROLLER_ADMIN_NODE:5000/v2.0

[oslo_messaging_rabbit]
rabbit_host=$CONTROLLER_ADMIN_NODE
```

2. **/etc/nova/api-paste.ini 파일에서 매개변수를 설정합니다.**

```
[filter:authtoken]
admin_user = nova
admin_password = service-password
admin_tenant_name = service
auth_uri = http://$CONTROLLER_ADMIN_NODE:5000/v2.0/
identity_uri = http://$CONTROLLER_ADMIN_NODE:35357
```

3. **Nova SMF 서비스를 사용으로 설정합니다.**

```
controller# svcadm enable -rs nova-conductor
controller# svcadm enable -rs nova-api-osapi-compute \
nova-cert nova-scheduler
```

Horizon 설치

Horizon는 OpenStack에 대한 웹 포털 역할을 수행합니다.

▼ Horizon을 구성하는 방법

1. **SSL/TLS에 대한 Horizon 구성을 설정합니다.**

   a. **Horizon용 인증서를 생성합니다.**

   다음 명령은 Horizon용 자체 서명된 인증서를 생성하고 OpenStack 대시보드 구성 파일을 Apache 구성 파일 디렉토리에 복사합니다. 자체 서명된 인증서를 만드는 방법은 Apache SSL/TLS Strong Encryption: FAQ를 참조하십시오.

   ```
   controller# export DASHBOARD=/etc/openstack_dashboard
   controller# openssl req -new -x509 -nodes \
   -out horizon.crt -keyout horizon.key
   ```

   여기서 메시지가 표시되면 국가, 시/도, 구/군/시, 회사, 조직, 이름 및 전자메일 주소와 같은 정보를 제공합니다. 그런 다음 키 이동을 진행합니다.

   ```
   controller# mv horizon.crt horizon.key ${DASHBOARD}
   controller# chmod 0644 ${DASHBOARD}/*
   controller# chown webservd:webservd ${DASHBOARD}/*

   controller# sed \
   -e "/SSLCertificateFile/s:/path.*:${DASHBOARD}/horizon.crt:" \
   -e "/SSLCACertificateFile/d" \
   -e "/SSLCertificateKeyFile/s:/path.*:${DASHBOARD}/horizon.key:" \
   < /etc/apache2/2.4/samples-conf.d/openstack-dashboard-tls.conf \
   > /etc/apache2/2.4/conf.d/openstack-dashboard-tls.conf
   ```

   b. **`/etc/apache2/2.4/conf.d/openstack-dashboard-tls.conf` 파일에서 다음 매개변수에 대해 Horizon 패키지의 사이트 주소 및 서버 이름을 지정합니다.**

   ```
   RedirectPermanent /horizon https://controller-fqdn/horizon
   ServerName controller-fqdn
   ```

2. **Apache 서비스를 시작합니다.**

   ```
   controller# svcadm enable apache24
   ```

Cinder 설치

Cinder 구성에서는 최소한 다음 정보를 지정해야 합니다.

- Keystone에 대해 인증할 권한 부여 정보
- 만들 볼륨의 클래스

▼ Cinder를 설치 및 구성하는 방법

이 절차의 단계는 Cinder 또는 볼륨 노드가 아니라 Cinder 끝점 서비스의 구성을 가리킵니다.

1. **`/etc/cinder/cinder.conf` 파일에서 매개변수를 설정하거나 주석 처리를 해제하여 Cinder를 구성합니다.**

   `volume_driver` 매개변수의 경우 선택할 수 있는 여러 드라이버가 있습니다. 다음 예에서는 `volume_driver`에 대해 선택한 드라이버만 표시됩니다. 주석 처리된 다른 사용 가능한 드라이버는 제외됩니다.

   ```
   [DEFAULT]
   volume_driver=cinder.volume.drivers.solaris.zfs.ZFSISCSIDriver
   my_ip=$CONTROLLER_ADMIN_NODE

   [database]
   connection = mysql://cinder:service-password@$CONTROLLER_ADMIN_NODE/cinder

   [keystone_authtoken]
   auth_uri = http://$CONTROLLER_ADMIN_NODE:5000/v2.0
   identity_uri = http://$CONTROLLER_ADMIN_NODE:35357
   admin_user = cinder
   admin_password = service-password
   admin_tenant_name = service

   [oslo_messaging_rabbit]
   rabbit_host=$CONTROLLER_ADMIN_NODE
   ```

2. **`/etc/cinder/api-paste.ini` 파일에서 매개변수를 구성합니다.**

   ```
   [filter:authtoken]
   admin_tenant_name = service
   admin_user = cinder
   admin_password = service-password
   ```

3. **iSCSI 대상이 구성된 경우 해당 SMF 서비스를 사용으로 설정합니다.**

   ```
   controller# svcadm enable iscsi/target stmf
   ```

4. **Cinder SMF 서비스를 사용으로 설정합니다.**

   ```
   controller# svcadm enable -rs cinder-db
   controller# svcadm enable -rs cinder-api cinder-scheduler
   ```

참조 How to Build OpenStack Block Storage on ZFS을 참조하십시오.

Neutron 설치

이 장에서 설명되는 아키텍처에서는 Neutron API 서비스가 컨트롤러 노드에서 실행됩니다.

▼ Neutron을 설치 및 구성하는 방법

1. 다음 구성 파일에서 매개변수를 설정하거나 주석을 해제하여 Neutron을 구성합니다.

- /etc/neutron/neutron.conf

```
[DEFAULT]
host=$CONTROLLER_ADMIN_NODE

[keystone_authtoken]
auth_uri = http://$CONTROLLER_ADMIN_NODE:5000/v2.0
identity_uri = http://$CONTROLLER_ADMIN_NODE:35357
admin_user = neutron
admin_password = service-password
admin_tenant_name = service

[database]
connection = mysql://neutron:service-password@$CONTROLLER_ADMIN_NODE/neutron

[oslo_messaging_rabbit]
rabbit_host=$CONTROLLER_ADMIN_NODE
```

- /etc/neutron/plugins/evs/evs_plugin.ini

```
[EVS]
evs_controller = ssh://evsuser@$CONTROLLER_ADMIN_NODE
```

- /etc/neutron/dhcp_agent.ini

```
[DEFAULT]
evs_controller = ssh://evsuser@$CONTROLLER_ADMIN_NODE
```

- /etc/neutron/l3_agent.ini

```
evs_controller = ssh://evsuser@$CONTROLLER_ADMIN_NODE
```

주 - 기본적으로 클라우드에서 프로젝트의 내부 네트워크는 서로 격리되어 있습니다. 프로젝트 네트워크는 다른 프로젝트의 네트워크와 통신하지 않고 서로와만 통신할 수 있습니다. 클라우드에서 모든 네트워크를 사용으로 설정하여 네트워크가 속하는 프로젝트에 상관없이 서로 연결하려면 다음 매개변수 설정을 사용하여 `/etc/neutron/l3_agent.ini` 파일을 편집합니다.

```
allow_forwarding_between_networks = true
```

2. **SSH 키 쌍이 사용되도록 설정합니다.**

   a. **`evsuser`, `neutron` 및 `root` 사용자에 대한 SSH 키 쌍을 만듭니다.**

   ```
   controller# su - evsuser -c "ssh-keygen -N '' \
   -f /var/user/evsuser/.ssh/id_rsa -t rsa"
   controller# su - neutron -c "ssh-keygen -N '' -f /var/lib/neutron/.ssh/id_rsa -t rsa"
   controller# ssh-keygen -N '' -f /root/.ssh/id_rsa -t rsa
   ```

   b. **`evsuser`의 `authorized_keys` 파일에서 `evsuser`, `neutron` 및 `root` 사용자의 SSH 키를 조합합니다.**

   ```
   controller# cat /var/user/evsuser/.ssh/id_rsa.pub \
   /var/lib/neutron/.ssh/id_rsa.pub /root/.ssh/id_rsa.pub >> \
   /var/user/evsuser/.ssh/authorized_keys
   ```

   c. **SSH 연결을 테스트하여 `known_host` 파일에 저장될 지문을 수락합니다.**
   확인을 요청하는 메시지가 표시될 때마다 `Yes`를 지정합니다.

   ```
   controller# su - evsuser -c "ssh evsuser@$CONTROLLER_ADMIN_NODE true"
   controller# su - neutron -c "ssh evsuser@$CONTROLLER_ADMIN_NODE true"
   controller# ssh evsuser@$CONTROLLER_ADMIN_NODE true
   ```

3. **EVS(탄력적 가상 스위치)를 구성합니다.**

주 - 다음 하위 단계에서는 구체적으로 VLAN 기반 네트워크에 대한 EVS를 구성합니다.

VXLAN 기반 네트워크를 구성하려면 해당 Oracle Solaris 버전의 라이브러리(Operating Systems Documentation)에서 *Managing Network Virtualization and Network Resources in Oracle Solaris*로 이동하십시오. 이 설명서에서 구체적으로 *Use Case: Configuring an Elastic Virtual Switch for a Tenant* 절을 참조하십시오.

플랫 네트워크 구성 방법의 예는 `https://blogs.oracle.com/openstack/entry/configuring_the_neutron_l3_agent`를 참조하십시오.

   a. **EVS 컨트롤러의 위치를 지정하도록 EVS 등록 정보를 설정합니다.**

```
controller# evsadm set-prop -p controller=ssh://evsuser@$CONTROLLER_ADMIN_NODE
```

b. **EVS 컨트롤러의 `l2-type`, `vlan-range` 및 `uplink-port` 등록 정보를 구성합니다.**

```
controller# evsadm set-controlprop -p property=value
```

그림 1에 나온 대로 대개 서로 다른 서브넷을 서비스하는 여러 네트워크 인터페이스를 가집니다. `uplink-port` 등록 정보를 설정할 때 서브넷을 제공하는 여러 네트워크 포트에 걸쳐 VLAN을 분할할 수 있습니다.

다음 예에서는 VLAN 분할을 포함하여 EVS 등록 정보를 설정하는 방법을 보여줍니다. 선택적으로, 최종 명령을 사용하여 EVS 등록 정보를 모두 표시합니다.

주 - 네트워크 포트에 걸쳐 VLAN을 분산하기 전에 먼저 VLAN 범위를 정의합니다. 그렇지 않으면 `uplink-port` 등록 정보를 구성할 수 없습니다.

```
controller# evsadm set-controlprop -p l2-type=vlan
controller# evsadm set-controlprop -p vlan-range=1,200-300
controller# evsadm set-controlprop -p uplink-port=net0,vlan-range=1
controller# evsadm set-controlprop -p uplink-port=net1,vlan-range=200-250
controller# evsadm set-controlprop -p uplink-port=net2,vlan-range=251-300

controller# evsadm show-controlprop -o all
```

4. **IP 전달을 사용으로 설정합니다.**

```
controller# ipadm set-prop -p forwarding=on ipv4
controller# ipadm set-prop -p forwarding=on ipv6
```

5. **IP 필터 서비스를 시작합니다.**

```
controller# svcadm enable -rs ipfilter
```

6. **Neutron 서버 서비스를 사용으로 설정합니다.**

```
controller# svcadm enable -rs neutron-server neutron-dhcp-agent
```

컴퓨트 노드 구성

`nova-compute` 데몬과 컴퓨트 노드에 VM 인스턴스를 설치합니다. VM 인스턴스는 웹 응용 프로그램 및 분석 등 다양한 서비스를 제공합니다. 클라우드에 대해 필요한 만큼의 컴퓨트 노드를 구성할 수 있습니다.

컴퓨트 노드를 구성하려면 다음 명령을 사용하여 시스템에 OpenStack 구성요소 및 서비스를 설치합니다.

```
compute# pkg install openstack
```

패키지 설치를 완료한 후 노드에서 실행할 서비스를 구성합니다.

주 - Oracle Solaris 11에서 ZFS와 응용 프로그램 간 메모리 사용을 보다 잘 관리하려면 다음 예에 표시된 것과 같이, 노드에 대해 `usr_reserve_hint_pct` 매개변수를 설정합니다.

```
# echo "set user_reserve_hint_pct=80" >>/etc/system.d/site:kernel-zones-reserve
# reboot
```

여기서 *site*는 회사 이름과 같은 무작위 식별자입니다.

이 매개변수를 다른 OpenStack 노드에도 설정합니다.

이 매개변수에 대한 자세한 내용은 https://support.oracle.com에서 MOS 계정에 로그인한 다음 Document 1663862.1, *Memory Management Between ZFS and Applications in Oracle Solaris 11.2*를 검토하십시오.

▼ 컴퓨트 노드를 구성하는 방법

1. **NTP 클라이언트를 구성합니다.**
   "NTP 클라이언트 구성" [30]을 참조하십시오.

2. **RAD(Remote Access Daemon)를 다시 시작합니다.**
   Nova는 RAD를 사용하여 Oracle Solaris 영역 프레임워크와 통신합니다.

   ```
   compute1# svcadm restart rad:local
   ```

3. **`/etc/nova/nova.conf` 파일에서 다음 매개변수를 설정하거나 주석 처리를 해제하여 Nova를 구성합니다.**

   ```
   [DEFAULT]
   my_ip=$COMPUTE_ADMIN_NODE_IP
   host=$COMPUTE_ADMIN_NODE_X
   firewall_driver=nova.virt.firewall.NoopFirewallDriver
   keystone_ec2_url=http:/$CONTROLLER_ADMIN_NODE:5000/v2.0/ec2tokens

   [database]
   connection = mysql://nova:service-password@$CONTROLLER_ADMIN_NODE/nova

   [glance]
   host=$CONTROLLER_ADMIN_NODE

   [keystone_authtoken]
   auth_uri=http://$CONTROLLER_ADMIN_NODE:5000/v2.0/
   identity_uri=http://$CONTROLLER_ADMIN_NODE:35357/
   admin_user=nova
   ```

```
admin_password=service-password
admin_tenant_name=service

[neutron]
url=http://$CONTROLLER_ADMIN_NODE:9696
admin_username=neutron
admin_password=service-password
admin_tenant_name=service
admin_auth_url=http://$CONTROLLER_ADMIN_NODE:5000/v2.0

[oslo_messaging_rabbit]
rabbit_host=$CONTROLLER_ADMIN_NODE
```

4. **/etc/nova/api-paste.ini 파일에서 매개변수를 설정합니다.**

```
[filter:authtoken]
admin_user = nova
admin_password = service-password
admin_tenant_name = service
auth_uri = http://$CONTROLLER_ADMIN_NODE:5000/v2.0/
identity_uri = http://$CONTROLLER_ADMIN_NODE:35357
```

5. **컴퓨트 노드에서 EVS를 설정합니다.**

   a. **EVS 패키지가 설치되었는지 확인합니다.**

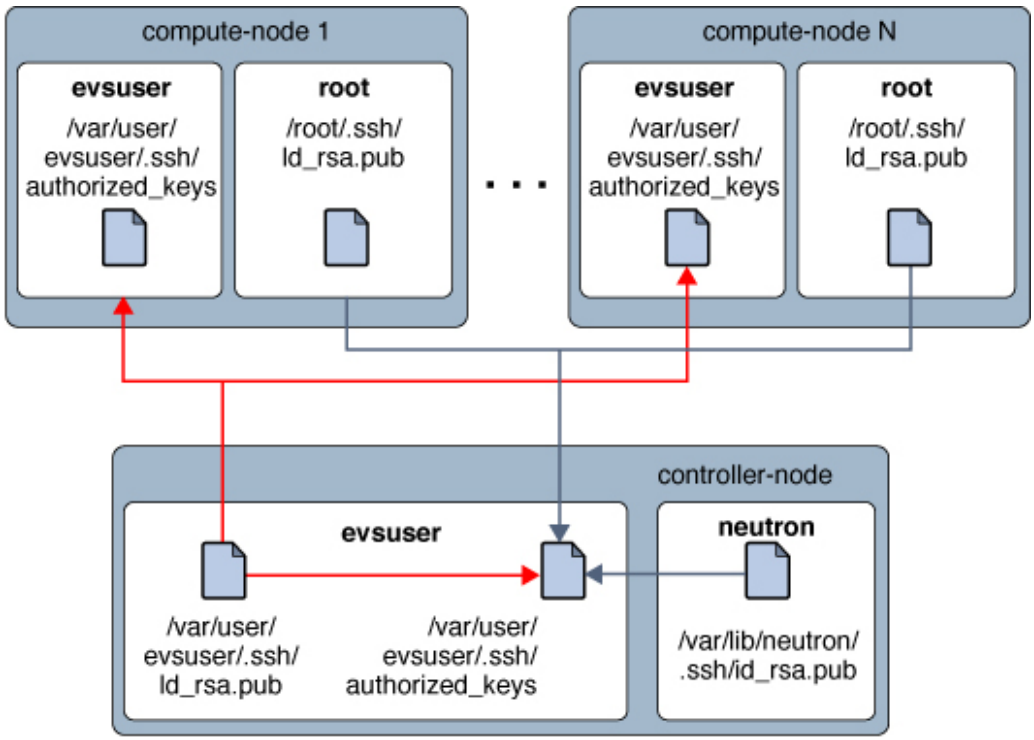
   ```
   compute1# pkg info evs
   ```

   b. **EVS 컨트롤러의 위치를 지정합니다.**

   ```
   compute1# evsadm set-prop -p controller=ssh://evsuser@$CONTROLLER_ADMIN_NODE
   ```

6. **컨트롤러 노드와 컴퓨트 노드 간 통신을 구성합니다.**

노드간 통신을 설정하는 SSH 키의 배포는 다음 그림과 유사합니다.



a. **`root` 사용자에 대해 컴퓨트 노드에서 SSH 공개 키를 만듭니다.**

```
compute1# ssh-keygen -N '' -f /root/.ssh/id_rsa -t rsa
```

b. **(옵션) SSH 키의 컨텐츠를 확인합니다.**

```
compute1# cat /root/.ssh/id_rsa.pub
```

c. **SSH 키 `/root/.ssh/id_rsa.pub`를 컨트롤러 노드의 위치로 복사합니다.**

d. **컨트롤러 노드에서 `evsuser`에 대해 SSH 키를 `authorized_keys` 파일에 추가합니다.**

```
controller# cat location/id_rsa.pub >> /var/user/evsuser/.ssh/authorized_keys
```

e. **(옵션) 컴퓨트 노드의 SSH 키가 `authorized_keys` 파일에 추가되었는지 확인합니다.**

```
controller# cat /var/user/evsuser/.ssh/authorized_keys
```

출력에는 컴퓨트 노드에서 생성한 /root/.ssh/id_rsa.pub의 내용이 포함되어야 합니다.

f. **컴퓨트 노드의 컨트롤러에 대한 SSH 연결을 테스트하고 지문을 known_host 파일에 저장하도록 허용합니다.**

확인을 요청하는 메시지가 표시되면 Yes를 지정합니다.

```
compute1# ssh evsuser@$CONTROLLER_ADMIN_NODE true
```

7. **Nova Compute 서비스를 사용으로 설정합니다.**

```
compute1# svcadm enable nova-compute
```

▼ 콘솔 액세스를 사용으로 설정하는 방법

이 절차에 따라 사용자 요청을 기반으로 브라우저에서 VM 인스턴스의 콘솔을 사용할 수 있습니다.

1. **각 컴퓨트 노드에서 적용되는 시나리오에 따라 다음 단계를 수행합니다.**

- **공용 네트워크에서 컴퓨트 노드의 IP 주소에 액세스할 수 있는 경우 /etc/nova/nova.conf 파일의 [DEFAULT] 섹션에서 다음 매개변수를 설정합니다.**

```
[DEFAULT]
...
vnc_enabled = true
vncserver_listen = 0.0.0.0
novncproxy_port = 6080
novncproxy_base_url =http://FQDN:6080/vnc_auto.html
novncproxy_host = 0.0.0.0
...
```

여기서 *FQDN*은 컴퓨트 노드의 IP 주소 또는 정규화된 도메인 이름입니다.

- **컴퓨트 노드가 개인 네트워크에 있는 경우 /etc/nova/nova.conf 파일의 [DEFAULT] 섹션에서 다음 매개변수를 설정합니다.**

```
[DEFAULT]
...
vnc_enabled = true
vncserver_listen = internal-IP
novncproxy_port=6080
novncproxy_base_url = http://public-IP:6080/vnc_auto.html
vncserver_proxyclient_address = internal-IP
```

  - *internal-IP* - 내부 네트워크에 있는 컴퓨트 노드의 IP 주소입니다.

- *public-IP* - 컨트롤러 호스트의 공용 IP 주소입니다.

2. **공용 네트워크에서 컴퓨트 노드의 IP 주소에 액세스할 수 있는 경우 다음 하위 단계를 수행합니다. 그렇지 않으면 다음 단계로 건너뜁니다.**

   a. **nova-novncproxy 서비스를 사용으로 설정합니다.**

   ```
   compute# svcadm enable nova-novncproxy
   ```

   b. **nova-compute 서비스를 다시 시작합니다.**

   ```
   compute# svcadm restart nova-compute
   ```

3. **컨트롤러 노드에서 적용되는 시나리오에 따라 다음 단계를 수행합니다.**

   - **공용 네트워크에서 컴퓨트 노드의 IP 주소에 액세스할 수 있는 경우 nova-consoleauth 서비스를 사용으로 설정합니다.**

     ```
     controller# svcadm enable nova-consoleauth
     ```

   - **컴퓨트 노드가 개인 네트워크에 있는 경우 다음 단계를 수행합니다.**

     a. **/etc/nova/nova.conf 파일의 [DEFAULT] 섹션에서 다음 매개변수를 설정합니다.**

     ```
     novncproxy_base_url=http://public-IP:6080/vnc_auto.html
     ```

     여기서 *public-IP*는 컨트롤러 호스트의 공용 IP 주소입니다.

     b. **다음과 같이 Nova 서비스를 사용으로 설정합니다.**

     ```
     controller# svcadm enable nova-consoleauth
     controller# svcadm enable nova-novncproxy
     ```

저장소 노드 구성

컨트롤러 노드는 OpenStack 설정 내에서 트랜잭션이 발생하는 모든 데이터의 저장소입니다.

컴퓨트 노드를 구성하려면 다음 명령을 사용하여 시스템에 OpenStack 구성요소 및 서비스를 설치합니다.

```
storage# pkg install openstack
```

패키지 설치를 완료한 후 노드에서 실행할 서비스를 구성합니다.

주 - Oracle Solaris 11에서 ZFS와 응용 프로그램 간 메모리 사용을 보다 잘 관리하려면 다음 예에 표시된 것과 같이, 노드에 대해 `usr_reserve_hint_pct` 매개변수를 설정합니다.

```
# echo "set user_reserve_hint_pct=80" >>/etc/system.d/site:kernel-zones-reserve
# reboot
```

여기서 *site*는 회사 이름과 같은 무작위 식별자입니다.

이 매개변수를 다른 OpenStack 노드에도 설정합니다.

이 매개변수에 대한 자세한 내용은 https://support.oracle.com에서 MOS 계정에 로그인한 다음 Document 1663862.1, *Memory Management Between ZFS and Applications in Oracle Solaris 11.2*를 검토하십시오.

▼ 블록 저장소 노드를 구성하는 방법

이 절차에서는 블록 저장소의 일반적인 구성을 설명합니다. 저장소 구성요소 구성을 위한 기타 옵션은 6장. Cinder 구성 및 배치 옵션을 참조하십시오.

1. **NTP 클라이언트를 구성합니다.**
   "NTP 클라이언트 구성" [30]을 참조하십시오.

2. **`/etc/cinder/cinder.conf` 파일에서 매개변수를 설정하거나 주석 처리를 해제하여 Cinder를 구성합니다.**

```
[DEFAULT]
san_is_local=true
volume_driver=cinder.volume.drivers.solaris.zfs.ZFSISCSIDriver
my_ip=$VOLUME_IP
glance_host=$CONTROLLER_ADMIN_NODE
zfs_volume_base=cinder/cinder

[database]
connection = mysql://cinder:service-password@$CONTROLLER_ADMIN_NODE/cinder

[keystone_authtoken]
auth_uri = http://$CONTROLLER_ADMIN_NODE:5000/v2.0
identity_uri = http://$CONTROLLER_ADMIN_NODE:35357
admin_user = cinder
admin_password = service-password
admin_tenant_name = service

[oslo_messaging_rabbit]
rabbit_host=$CONTROLLER_ADMIN_NODE
```

3. **`/etc/cinder/api-paste.ini` 파일에서 매개변수를 설정합니다.**

```
[filter:authtoken]
admin_tenant_name = service
admin_user = cinder
admin_password = service-password
```

4. **Cinder 서비스를 시작합니다..**

```
storage# svcadm enable -rs cinder-db cinder-volume:default cinder-volume:setup
storage# svcadm enable -rs iscsi/target
```

Swift 객체 저장소 구성

Swift는 OpenStack 객체 저장소 프로젝트입니다. 단순 API로 대용량 데이터를 저장하고 검색할 수 있는 클라우드 저장소 소프트웨어를 제공합니다. 이 서비스는 무제한으로 증가할 수 있는 비구조적 데이터를 저장하기에 이상적입니다.

Swift에 대한 자세한 내용은 OpenStack 커뮤니티에서 OpenStack Cloud Administrator Guide의 Object Storage 장을 참조하십시오.

OpenStack 커뮤니티 설명서에서는 운용 모드의 Swift 배치를 위해 최소한 6 노드를 권장합니다. 이 노드는 Swift 프록시 컨트롤러 1개와 Swift 컨트롤러 노드 5개로 구성됩니다. 그러나 본 설명서는 이전 구성 절차에서 사용된 3노드 아키텍처와 일관성을 맞추기 위해 3노드 배치에 대해 설명합니다. 나중에 필요에 따라 컨트롤러 노드를 추가할 수 있습니다.

▼ Swift 프록시 컨트롤러 서비스 노드를 구성하는 방법

이 작업에서는 Swift를 위해 지정된 노드에 OpenStack 패키지를 이미 설치했다고 가정합니다. “저장소 노드 구성” [45]에서 사용할 설치 명령을 참조하십시오.

1. **Swift 패키지를 설치합니다.**

```
proxy-node # pkg install swift swiftclient
```

2. **ZFS 데이터 세트를 만듭니다.**

```
proxy-node # /usr/sbin/zfs create -o mountpoint=none rpool/export/swift
proxy-node # /usr/sbin/zfs create -o mountpoint=/srv rpool/export/swift/srv
proxy-node # /usr/sbin/zfs create -p rpool/export/swift/srv/node/disk0
proxy-node # /usr/bin/chown -R swift:swift /srv
```

3. **다음 8진수 덤프를 수행합니다.**

덤프 값을 유지합니다. 이 두 값은 후속 단계에서 `$OD_1` 및 `$OPD_2`라고 합니다.

```
proxy-node # od -t x8 -N 8 -A n < /dev/random
```

```
proxy-node # od -t x8 -N 8 -A n < /dev/random
```

4. **다음 매개변수를 사용하여 `/etc/swift/swift.conf` 파일을 편집합니다.**

```
[swift-hash]
swift_hash_path_suffix = $OD_1
swift_hash_path_prefix = $OD_2
```

5. **다음 매개변수로 `/etc/swift/proxy-server.conf` 파일을 편집합니다.**

```
[DEFAULT]
bind_port = 8080

[filter:tempauth]
use = egg:swift#tempauth

operator_roles = admin, swiftoperator

[filter:authtoken]
auth_uri = http://$CONTROLLER_IP:5000/
identity_uri = http://$CONTROLLER_IP:35357
admin_user = swift
admin_password = swiftpass
admin_tenant_name = service

[filter:cache]
memcache_servers = $CONTROLLER_IP:11211

[oslo_messaging_rabbit]
rabbit_host=$CONTROLLER_ADMIN_NODE
```

6. **`memcached` 데몬을 사용으로 설정합니다.**

```
proxy-node # svcadm enable -rs memcached
```

7. **링을 구축합니다.**

```
proxy-node # cd /etc/swift
proxy-node # swift-ring-builder account.builder create 18 3 1
proxy-node # swift-ring-builder container.builder create 18 3 1
proxy-node # swift-ring-builder object.builder create 18 3 1
proxy-node # swift-ring-builder account.builder add r1z1-$STORAGE_IP_1:6002/disk0 100
proxy-node # swift-ring-builder container.builder add r1z1-$STORAGE_IP_1:6001/disk0
 100
proxy-node # swift-ring-builder object.builder add r1z1-$STORAGE_IP_1:6000/disk0 100
proxy-node # swift-ring-builder account.builder add r1z1-$STORAGE_IP_2:6002/disk0 100
proxy-node # swift-ring-builder container.builder add r1z1-$STORAGE_IP_2:6001/disk0
 100
proxy-node # swift-ring-builder object.builder add r1z1-$STORAGE_IP_2:6000/disk0 100
proxy-node # swift-ring-builder account.builder rebalance
proxy-node # swift-ring-builder container.builder rebalance
proxy-node # swift-ring-builder object.builder rebalance
proxy-node # >chown -R swift:swift /etc/swift
```

8. **Swift 서비스를 사용으로 설정합니다.**

```
proxy-node # svcadm enable swift-proxy-server
```

▼ 객체 저장소 노드를 구성하는 방법

설정할 각 객체 컨트롤러 노드에서 이 절차를 반복합니다.

1. **Swift 패키지를 설치합니다.**

```
storage-node # pkg install swift swiftclient
```

2. **ZFS 데이터 세트를 만듭니다.**

```
storage-node # /usr/sbin/zfs create -o mountpoint=none rpool/export/swift
storage-node # /usr/sbin/zfs create -o mountpoint=/srv rpool/export/swift/srv
storage-node # /usr/sbin/zfs create -p rpool/export/swift/srv/node/disk0
storage-node # /usr/bin/chown -R swift:swift /srv
```

3. **다음과 같이 프록시 서버 노드에서 파일을 복사합니다.**

   a. **`/etc/swift/swift.conf` 파일을 프록시 서버 노드에서 현재 노드의 `/etc/swift` 디렉토리로 복사합니다.**

   b. **다음 파일을 프록시 서버 노드에서 현재 노드의 `/etc/swift` 디렉토리로 복사합니다.**

   - account.ring.gz
   - container.ring.gz
   - object.ring.gz

4. **Swift 복제기 서비스를 사용으로 설정합니다.**

```
storage-node # svcadm enable swift-replicator-rsync
```

5. **현재 노드의 `/etc/swift` 디렉토리에 대한 소유권을 설정합니다.**

```
storage-node # chown -R swift:swift /etc/swift
```

6. **모든 Swift 서비스를 사용으로 설정합니다.**

```
storage-node # for x in `svcs -a -o SVC | fgrep swift |  \
   egrep "account|container|object" | sort` \
      do \
           echo Starting $x \
           svcadm enable $x \
      done
```

7. **컨트롤러 노드에서 사용자가 Swift 서비스에 액세스하고 운영할 수 있도록 합니다.**

a. **Swift에 대한 전역 셸 변수를 설정합니다.**

```
controller# export OS_USERNAME=swift
controller# export OS_PASSWORD=service-password
controller# export OS_PROJECT_NAME=service
controller# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0
```

b. **Keystone `swiftoperator` 역할을 추가합니다.**

```
controller# openstack role create --name swiftoperator
```

c. **Swift 서비스의 인증된 사용자에게 `swiftoperator` 역할을 지정합니다.**

```
controller# openstack user role add --user user-name \
    --role swiftoperator --project tenant-ID
```

다음 순서 사용자가 클라우드를 사용할 수 있도록 하려면 4장. 설치 후 구성 작업에 설명된 준비 작업을 완료하십시오.

♦ ♦ ♦ 4 장

4 설치 후 구성 작업

이 장에서는 초기 OpenStack 설치 및 구성을 완료하기 위한 절차에 대해 설명합니다. 이 장에서는 다음 항목을 다룹니다.

OpenStack 프로젝트에 대한 외부 네트워크 준비

외부 네트워크는 클라우드의 사설 네트워크와 공용 네트워크 간에 연결을 제공합니다.

공급자 라우터 정보

라우터는 더 넓은 네트워크에서 프로젝트 VM 인스턴스에 대한 연결성을 제공합니다. 라우터는 모든 프로젝트 네트워크에서 공유됩니다. 라우터가 한 개뿐이므로 프로젝트 네트워크는 겹치는 IP 주소를 사용할 수 없습니다.

해당 라우터는 인터페이스에서 라우터를 외부 네트워크에 연결하는 양방향 NAT(네트워크 주소 변환)를 수행합니다. 프로젝트는 필요한 만큼 또는 유동 IP 할당량에서 허용하는 만큼 많은 유동 IP(공용 IP)를 가질 수 있습니다. 이러한 유동 IP는 외부 연결성이 필요한 VM 인스턴스와 연관됩니다.

라우터를 만들려면 Neutron L3 에이전트를 구성해야 합니다. 이 에이전트는 Nova 인스턴스에 지정된 주소와 유동 IP 주소 간에 자동으로 일대일 NAT 매핑을 만듭니다. 또한 L3 에이전트는 개인 네트워크 간 통신을 사용으로 설정합니다.

기본적으로 같은 프로젝트의 사설 네트워크 간 경로 지정은 사용 안함으로 설정됩니다. 이 동작을 변경하려면 `/etc/neutron/l3_agent.ini` 구성 파일에서 `allow_forwarding_between_networks`를 `True`로 설정합니다. 매개변수를 설정한 후 `neutron-l3-agent` SMF 서비스를 다시 시작합니다.

▼ 외부 네트워크에 대한 라우터를 구성하는 방법

이 절차에서는 외부 네트워크에 대한 라우터를 만드는 방법을 보여줍니다. 단계 중 일부에서는 구성 파일 편집이 필요합니다. 따라서 이 절차의 경우 터미널 창을 사용하는 것이 Horizon 대시보드를 사용하는 것보다 편리합니다.

Neutron 서비스가 설치된 노드에서 다음 단계를 수행하십시오. 이 문서는 이전 장에 설명된 샘플 아키텍처를 기준으로 컨트롤러 노드에서 서비스를 찾습니다.

시작하기 전에 Neutron을 설치 및 구성하는 방법 [38]에 설명된 대로 Neutron의 구성을 완료했는지 확인하십시오.

1. **사용 안함으로 설정된 경우 IP 필터 서비스를 시작합니다.**

   ```
   controller# svcadm enable -rs ipfilter
   ```

2. **호스트에서 IP 전달이 사용 안함으로 설정된 경우 사용으로 설정합니다.**

   ```
   controller# ipadm set-prop -p forwarding=on ipv4
   ```

3. **Neutron에 대한 전역 셸 변수를 설정합니다.**

   ```
   controller# export OS_USERNAME=neutron
   controller# export OS_PASSWORD=service-password
   controller# export OS_PROJECT_NAME=service
   controller# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0
   ```

4. **공급자 라우터를 만듭니다.**

   ```
   controller# neutron router-create router-name
   ```

   이 명령은 라우터 이름을 해당 ID와 함께 표시합니다. 이 ID를 사용하여 다음 단계에서 구성 파일을 업데이트합니다.

5. **`/etc/neutron/l3_agent.ini` 파일에 다음 설정 매개변수가 포함되어 있는지 확인합니다.**

   `router_id=`*routerID* 이전 단계에서 가져온 *ID*

6. **`neutron-l3-agent` SMF 서비스를 사용으로 설정합니다.**

   ```
   controller# svcadm enable neutron-l3-agent
   ```

7. **(옵션) 라우터에 대한 정보를 표시합니다.**
   라우터에 외부 네트워크를 추가한 후 라우터에 대해 더 많은 정보가 추가됩니다.

   ```
   controller# neutron router-show router-name
   ```

예 1 라우터 만들기

이 예에서는 외부 네트워크에 대한 라우터를 만드는 방법을 보여줍니다.

```
controller# svcadm enable -rs ipfilter

controller# ipadm set-prop -p forwarding=on ipv4
controller# ipadm set-prop -p forwarding=on ipv6

controller# export OS_USERNAME=neutron
controller# export OS_PASSWORD=service-password
controller# export OS_PROJECT_NAME=service
controller# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0

controller# neutron router-create ext-router
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | f89b24ed-42dd-48b0-8f4b-fd41887a3370 |
| name                  | ext-router                           |
| status                | ACTIVE                               |
| project_id            | 7d1caf0854b24becb28df5c5cabf72cc     |
+-----------------------+--------------------------------------+
```

이 시점에서 `/etc/neutron/l3_agent.ini` 파일의 `router_id`를 업데이트합니다.

```
router_id = f89b24ed-42dd-48b0-8f4b-fd41887a3370
```

그런 다음 L3 에이전트 서비스를 사용으로 설정합니다.

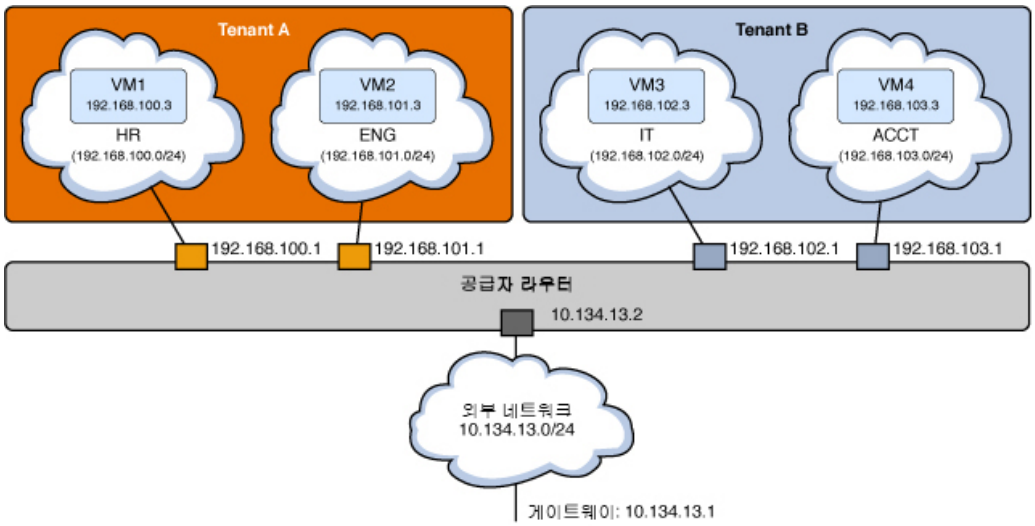
```
controller# svcadm enable neutron-l3-agent
```

외부 네트워크 만들기

라우터를 만든 후 외부 네트워크를 구성합니다. 외부 네트워크의 게이트웨이를 통해 클라우드의 내부 네트워크 사용자가 인터넷에 연결할 수 있습니다.

다음 그림은 단일 라우터가 클라우드 프로젝트의 네트워크 통신을 어떻게 지원하는지 보여줍니다.

그림 3 개인 네트워크 모델을 사용하는 공급자 라우터



그림은 다음을 보여줍니다.

- 프로젝트 2개(`Tenant A` 및 `Tenant B`)
- VM 4개(`VM1`, `VM2`, `VM3` 및 `VM4`)
- 서브넷 4개(`HR`, `ENG`, `IT` 및 `ACCT`)
- 라우터
- 외부 네트워크

외부 네트워크에 대한 내부 네트워크 액세스를 제공할 경우 VM의 IP 주소는 외부 네트워크에 지정하는 유동 주소 중 하나에 매핑됩니다.

▼ 외부 네트워크를 만드는 방법

이 절차에서는 외부 네트워크를 나타내는 가상 네트워크를 만드는 방법을 보여줍니다. 이 가상 네트워크에서는 DHCP를 사용하지 않습니다. 대신 유동 IP 주소가 만들어지고 프로젝트에 지정되며 이러한 프로젝트 아래에서 Nova VM 인스턴스에 의해 사용됩니다. 다음 단계에서는 VLAN 네트워크 유형을 만들지만, 절차는 다른 네트워크 유형(예: 플랫 네트워크) 만들기에도 적용됩니다.

내부 네트워크 만들기와 독립적으로 외부 네트워크를 만들 수 있습니다.

시작하기 전에 탄력적 가상 스위치의 구성을 완료하십시오. 자세한 내용은 Neutron을 설치 및 구성하는 방법 [38], 특히 EVS 구성 단계를 참조하십시오.

1. **Neutron에 대한 전역 셸 변수를 설정합니다.**

```
controller# export OS_USERNAME=neutron
controller# export OS_PASSWORD=service-password
controller# export OS_PROJECT_NAME=service
controller# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0
```

2. **(옵션) VLAN 범위를 가져옵니다.**

```
controller# evsadm show-controlprop -p vlan-range
```

3. **외부 네트워크를 만듭니다.**

```
controller# neutron net-create --provider:network_type=vlan \
--provider:segmentation_id=VLAN-nbr \
--router:external network-name
```

이 단계에서는 Neutron을 설치 및 구성하는 방법 [38]에 따라 VLAN 네트워크를 만든 것으로 간주합니다. `segmentation_id`는 EVS를 구성할 때 정의한 범위의 VLAN 네트워크에 대한 VLAN ID입니다.

주 - 플랫 네트워크를 만드는 경우 세그먼트화 ID를 지정할 필요가 없습니다.

---

4. **외부 네트워크의 서브넷을 만듭니다.**

할당 풀은 서브넷에 지정된 일정 범위의 유동 IP 주소로 구성됩니다.

```
controller# neutron subnet-create --name subnet-name --disable-dhcp \
  --allocation-pool start=start-IP,end=end-IP \
  network-name subnet-IP
```

5. **라우터에 외부 네트워크를 추가합니다.**

```
controller# neutron router-gateway-set router-name ext-network-ID
```

주 - 기본적으로 SNAT는 이 명령을 실행할 때 사용으로 설정됩니다. SNAT가 사용으로 설정된 경우 개인 네트워크의 VM은 외부 네트워크에 액세스할 수 있습니다. 그러나 인스턴스 자체는 클라우드 밖에서 액세스할 수 없습니다. SNAT를 사용 안함으로 설정하려면 `neutron router-gateway-set` 하위 명령에 `--disable-snat` 옵션을 지정합니다.

---

6. **(옵션) 라우터에 대한 정보를 표시합니다.**

```
controller# neutron router-show router-name
```

예 2 외부 네트워크 만들기

이 예에서는 외부 네트워크를 만들어 클라우드의 내부 네트워크에서 사용할 수 있도록 준비하는 방법을 보여줍니다.

플랫 네트워크를 만들려면 https://blogs.oracle.com/openstack/tags/juno의 2절에 제공된 예를 참조하십시오.

```
controller# export OS_USERNAME=neutron
controller# export OS_PASSWORD=service-password
controller# export OS_PROJECT_NAME=service
controller# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0

controller# evsadm show-controlprop -p vlan-range
PROPERTY          PERM VALUE       DEFAULT   HOST
vlan-range        rw   1,200-300   --        --

controller# neutron net-create --router:external \
--provider:network_type=vlan --provider:segmentation_id=1 ext_network
Created a new network:
+--------------------------+--------------------------------------+
| Field                    | Value                                |
+--------------------------+--------------------------------------+
| admin_state_up           | True                                 |
| id                       | 08cf49c8-f28f-49c1-933d-bdb1017e0294 |
| name                     | ext_network                          |
| provider:network_type    | vlan                                 |
| provider:segmentation_id | 1                                    |
| router:external          | True                                 |
| shared                   | False                                |
| status                   | ACTIVE                               |
| subnets                  |                                      |
| project_id               | 7d1caf0854b24becb28df5c5cabf72cc     |
+--------------------------+--------------------------------------+

controller# neutron subnet-create --name ext_subnet --disable-dhcp \
   --allocation-pool start=10.134.10.8,end=10.134.10.254 \
   ext_network 10.134.10.0/24
Created a new subnet:
+-------------------+------------------------------------------------------+
| Field             | Value                                                |
+-------------------+------------------------------------------------------+
| allocation_pools  | {"start": "10.134.10.8", "end": "10.134.10.254"}     |
| cidr              | 10.134.10.0/24                                       |
| dns_nameservers   |                                                      |
| enable_dhcp       | False                                                |
| gateway_ip        | 10.134.13.1                                          |
| host_routes       |                                                      |
| id                | fce503ff-f483-4024-b122-f2524e3edae1                 |
| ip_version        | 4                                                    |
| ipv6_address_mode |                                                      |
| ipv6_ra_mode      |                                                      |
| name              | ext_subnet                                           |
| network_id        | 08cf49c8-f28f-49c1-933d-bdb1017e0294                 |
| project_id        | 7d1caf0854b24becb28df5c5cabf72cc                     |
+-------------------+------------------------------------------------------+

controller# neutron router-gateway-set ext-router 08cf49c8-f28f-49c1-933d-bdb1017e0294
Set gateway for router ext-router
```

```
controller# neutron router-show ext-router
+-----------------------+-------------------------------------------------------+
| Field                 | Value                                                 |
+-----------------------+-------------------------------------------------------+
| admin_state_up        | True                                                  |
| external_gateway_info | {"network_id": "08cf49c8-f28f-49c1-933d-bdb1017e0294",|
|                       | "enable_snat": true,                                  |
|                       | "external_fixed_ips":                                 |
|                       |[{"subnet_id": "fce503ff-f483-4024-b122-f2524e3edae1", |
|                       |"ip_address": "10.134.10.8"}]} |                       |
| id                    | f89b24ed-42dd-48b0-8f4b-fd41887a3370                  |
| name                  | ext-router                                            |
| status                | ACTIVE                                                |
| project_id            | 7d1caf0854b24becb28df5c5cabf72cc                      |
+-----------------------+-------------------------------------------------------+
```

참조
- L3 에이전트 구성을 관찰하는 방법 [58].
- “알려진 제한 사항” [126].

▼ 내부 네트워크에 외부 연결을 제공하는 방법

이 절차를 사용하여 내부 네트워크에서 더 넓은 공용 네트워크에 액세스하도록 할 수 있습니다. 이 절차에서는 내부 네트워크가 특정 프로젝트에 대해 이미 존재한다고 간주합니다. 대시보드를 사용하여 프로젝트 내부 네트워크를 만들려면 “프로젝트에 대한 내부 네트워크 만들기” [70]를 참조하십시오.

시작하기 전에 계속하기 전에 공용 액세스가 필요한 서브넷 이름을 확인하십시오.

1. **Neutron에 대한 전역 셸 변수를 설정합니다.**

```
controller# export OS_USERNAME=neutron
controller# export OS_PASSWORD=service-password
controller# export OS_PROJECT_NAME=service
controller# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0
```

2. **외부 액세스가 필요한 서브넷의 ID를 식별합니다.**

```
controller# neutron subnet-list | grep subnet-name
```

3. **(옵션) 라우터의 이름을 확인합니다.**

```
controller# neutron router-list
```

4. **서브넷의 ID를 라우터에 인터페이스로 추가합니다.**

```
controller# neutron router-interface-add router-name subnetID
```

예 3 외부 네트워크에 내부 네트워크 연결

이 예에서는 프로젝트에 대한 네트워크를 구성하는 방법 [70]에서 만들어진 HR 내부 네트워크를 사용합니다. 서브넷이 HR_Subnet인 HR 네트워크에 공용 네트워크에 대한 액세스가 필요합니다.

```
controller# neutron subnet-list | grep HR_Subnet
| b6feff42-36aa-4235- | HR_Subnet  | 10.132.30.0/24 | {"start": "10.132.30.2", |
|   9fe0-ac5de6b43af3 |            |                |  "end": "10.132.30.254"} |

controller# neutron router-list
+--------------------  +------------+--------------------------------+
| id                   | name       | external_gateway_info          |
+--------------------  +------------+--------------------------------+
| f89b24ed-42dd-48b0-  | ext-router | {"network_id": "6c4c1823-a203- |
|    8f4b-fd41887a3370 |            |    43b1-9674-ddb5ff4185fc",    |
|                      |            | "enable_snat": true,           |
|                      |            | "external_fixed_ips":          |
|                      |            | [{"subnet_id": "83d9b40f-cc61- |
|                      |            |    4696-b22e-b4cbc2aa3872",    |
|                      |            | "ip_address": "10.132.10.8"}]} |
+----------  -----------+------------+--------------------------------+

controller# neutron router-interface-add ext-router b6feff42-36aa-4235-9fe0-ac5de6b43af3
Added interface b6feff42-36aa-4235-9fe0-ac5de6b43af3 to router ext-router.
```

▼ L3 에이전트 구성을 관찰하는 방법

IP 필터 명령(예: ipf, ippool 및 ipnat) 및 네트워킹 명령(예: dladm 및 ipadm)을 사용하여 neturon-l3-agent로 완료된 구성을 관찰하고 문제를 해결할 수 있습니다.

1. **neutron-l3-agent로 만들어진 VNIC를 표시합니다.**

```
network# dladm show-vnic
LINK                OVER         SPEED  MACADDRESS        MACADDRTYPE VIDS
l3i7843841e_0_0     net1         1000   2:8:20:42:ed:22   fixed       200
l3i89289b8e_0_0     net1         1000   2:8:20:7d:87:12   fixed       201
l3ed527f842_0_0     net0         100    2:8:20:9:98:3e    fixed       0
```

2. **neutron-l3-agent로 만들어진 IP 주소를 표시합니다.**

```
network# ipadm
NAME                 CLASS/TYPE STATE   UNDER      ADDR
l3ed527f842_0_0      ip         ok      --         --
  l3ed527f842_0_0/v4 static     ok      --         10.134.10.8/24
  l3ed527f842_0_0/v4a static    ok      --         10.134.10.9/32
l3i7843841e_0_0      ip         ok      --         --
  l3i7843841e_0_0/v4 static     ok      --         192.168.100.1/24
l3i89289b8e_0_0      ip         ok      --         --
  l3i89289b8e_0_0/v4 static     ok      --         192.168.101.1/24
```

3. **IP 필터 규칙을 표시합니다.**

```
network# ipfstat -io
empty list for ipfilter(out)
block in quick on l3i7843841e_0_0 from 192.168.100.0/24 to pool/4386082
block in quick on l3i89289b8e_0_0 from 192.168.101.0/24 to pool/8226578
network# ippool -l
table role = ipf type = tree number = 8226578
{ 192.168.100.0/24; };
table role = ipf type = tree number = 4386082
{ 192.168.101.0/24; };
```

4. **IP NAT 규칙을 표시합니다.**

```
network# ipnat -l
List of active MAP/Redirect filters:
bimap l3ed527f842_0_0 192.168.101.3/32 -> 10.134.13.9/32
List of active sessions:
BIMAP 192.168.101.3  22  <- -> 10.134.13.9  22 [10.132.146.13 36405]
```

Glance 저장소에 대한 이미지 준비

이미지는 클라우드에서 VM 인스턴스의 기초입니다. 이미지는 부트 가능 운영체제가 설치된 가상 디스크를 포함하는 단일 파일입니다. 이미지는 하나 이상의 VM을 만들기 위한 템플리트를 제공합니다. 따라서 클라우드에서 VM을 프로비전하려면 먼저 이미지를 만들어야 합니다.

OpenStack 이미지 서비스인 Glance는 디스크 및 서버 이미지에 대한 저장소, 검색, 등록 및 전달 서비스를 제공합니다. 레지스트리 서버는 클라이언트에 이미지 메타데이터 정보를 제공하는 이미지 서비스입니다. 이미지 캐시는 이미지가 요청될 때마다 이미지 서비스가 이미지 서버에서 이미지를 다시 다운로드하지 않고 로컬 호스트에서 이미지를 가져오는 데 사용합니다.

Glance 저장소로 여러 개의 이미지를 업로드할 수 있습니다. 클라우드에 배치할 여러 시스템 유형의 이미지를 업로드하는 것이 좋습니다. 예를 들어, 비전역 영역, 커널 영역 및 전역 영역의 아카이브된 이미지를 만듭니다. 그러면 적절한 템플리트를 선택하여 이러한 유형 중 하나의 VM을 빠르게 배치할 수 있습니다.

이미지 만들기

Oracle Solaris에서 OpenStack 이미지를 만들려면 Unified Archive 기능을 사용합니다. `archiveadm` 명령을 사용하여 전역, 비전역 및 커널 영역에서 새 UA(Unified Archive)를 만들 수 있습니다.

UA는 복제 아카이브 또는 복구 아카이브일 수 있습니다. 복제 아카이브는 현재 활성 부트 환경에 기반합니다. 이 아카이브는 비활성 BE 등 OS 인스턴스의 어떤 시스템 구성 정보도 포함하지 않습니다. 대신 설치 프로그램에서 재구성을 강제로 수행할 수도 있고, SC(시스템

구성) 프로파일에 사용자가 제공한 구성 정보를 사용할 수도 있습니다. 복구 아카이브는 모든 부트 환경과 시스템 구성 정보를 포함합니다. 따라서 모든 시스템 정보를 UA에 포함하려면 복구 아카이브를 만드십시오. UA에 대한 자세한 내용은 시스템 복구 및 복제용 *Unified Archive* 사용을 참조하십시오. 이 설명서는 Systems Operation Documentation의 사용 중인 Oracle Solaris 버전 라이브러리에 있습니다.

나중에 구성이 완전히 작동 중일 때 기존 VM 인스턴스의 스냅샷을 만들어 이미지를 만들 수도 있습니다. 이 경우 VM 인스턴스는 이미 클라우드에 있습니다. 따라서 사용할 명령은 `archiveadm` 대신 `nova image-create`입니다. `nova` 명령은 실행 중인 VM 인스턴스의 스냅샷을 만들어 이미지를 만듭니다.

데이터 백업을 위해 또는 VM 인스턴스 자동 복구를 위해 사용자정의 이미지를 사용할 수도 있습니다. 자동 복구 이미지는 VM 인스턴스가 `rescue` 모드로 설정될 때 부팅되는 특수 유형의 이미지입니다. 자동 복구 이미지를 통해 관리자는 VM 인스턴스에 대한 파일 시스템을 마운트하여 문제를 해결할 수 있습니다.

Oracle Solaris에서 OpenStack 이미지는 3단계로 만듭니다.

1. 영역을 만듭니다.
2. 영역의 UA를 만듭니다.
3. UA를 Glance로 업로드합니다.

이러한 단계는 다음 절차에서 결합됩니다.

▼ OpenStack에 대한 이미지를 만드는 방법

영역을 만들기 위한 명령을 제외하고, 나머지 모든 단계는 비전역 영역 및 커널 영역 모두의 이미지 만들기 및 업로드에 사용할 수 있습니다.

이 절차에서 영역 만들기에 대한 단계는 기본 명령만 제공합니다. 영역 만들기에 대한 전체 지침은 비전역 영역 설치, 종료, 정지, 제거 및 복제에 대한 *Oracle Solaris* 영역 만들기 및 사용을 참조하십시오. 이 설명서는 해당 Oracle Solaris 버전의 라이브러리(Operating Systems Documentation)에 있습니다.

1. **아무 시스템에서나 영역을 만든 다음 로그인합니다.**
   로그인 후 메시지가 표시되면 정보를 제공합니다.

```
global# zonecfg -z zone-name create
global# zoneadm -z zone-name install
global# zoneadm -z zone-name boot
global# zlogin -C zone-name
```

주 - 이 단계를 완료하는 데 약간의 시간이 걸릴 수 있습니다.

2. **OpenStack 루트 로그인 액세스에 대해 루트 SSH를 사용으로 설정합니다.**

```
global# zlogin zone-name
```

```
root@zone-name# sed /^PermitRootLogin/s/no$/without-password/ < /etc/ssh/sshd_config
 > /system/volatile/sed.$$
root@zone-name# cp /etc/ssh/sshd_config /etc/ssh/sshd_config.orig
root@zone-name# cp /system/volatile/sed.$$ /etc/ssh/sshd_config
root@zone-name# exit
```

3. **영역에 대한 UA를 만듭니다.**

```
global# archiveadm create -z zone-name /var/tmp/archive-name.uar
```

4. **UA를 Glance가 설치된 시스템으로 전송합니다.**
   이 문서는 Glance가 컨트롤러 노드에 있는 것으로 간주합니다.

5. **Glance에 대한 전역 셸 변수를 설정합니다.**

```
controller# export OS_USERNAME=glance
controller# export OS_PASSWORD=service-password
controller# export OS_PROJECT_NAME=service
controller# export OS_AUTH_URL=$CONTROLLER_ADMIN_NODE:5000/v2.0
```

6. **UA를 Glance 저장소로 업로드합니다.**

```
controller# glance --os-image-api-version 2 image-create \
   --container-format bare --disk-format raw \
   --visibility visibility-mode --name "image-name" \
   --property architecture=system-arch \
   --property hypervisor_type=solariszones \
   --property vm_mode=solariszones  --file path-to-archive-file
```

system-arch x86_64 또는 sparc64일 수 있는 시스템 아키텍처

visibility-mode public 또는 private일 수 있는 이미지 접근성 범위

이미지에 대한 정보 표시

이미지 정보를 표시하려면 nova 명령 또는 glance 명령을 사용할 수 있습니다.

```
$ nova image-list
+--------------------------------------+-------------------------+--------+--------+
| ID                                   | Name                    | Status | Server |
+--------------------------------------+-------------------------+--------+--------+
| 4dfbfd4f-2de5-4251-832c-e35a4a4145ee | Solaris Non-global Zone | ACTIVE |        |
+--------------------------------------+-------------------------+--------+--------+
```

glance image-list 명령은 디스크 포맷, 컨테이너 형식, 여러 이미지의 크기 등 추가 정보를 표시합니다.

nova image-show 및 glance image-show 명령은 특정 이미지에 대한 정보를 표시합니다. 각 명령은 이미지에 대한 서로 다른 출력을 생성합니다.

```
$ nova image-show 'Solaris Non-global Zone'
+-------------------------+--------------------------------------+
| Property                | Value                                |
| OS-EXT-IMG-SIZE:size    | 845025280                            |
| created                 | 2015-11-19T14:46:38Z                 |
| id                      | 4dfbfd4f-2de5-4251-832c-e35a4a4145ee |
| metadata architecture   | x86_64                               |
| metadata hypervisor_type | solariszones                         |
| metadata vm_mode        | solariszones                         |
| minDisk                 | 0                                    |
| minRam                  | 0                                    |
| name                    | Solaris Non-global Zone              |
| progress                | 100                                  |
| status                  | ACTIVE                               |
| updated                 | 2015-11-19T14:46:42Z                 |
+-------------------------+--------------------------------------+

$ glance image-show 'Solaris Non-global Zone'
+---------------------------+--------------------------------------+
| Property                  | Value                                |
+---------------------------+--------------------------------------+
| Property 'architecture'   | x86_64                               |
| Property 'hypervisor_type' | solariszones                         |
| Property 'vm_mode'        | solariszones                         |
| checksum                  | ba9b9eeddb467833d725c8750a46e004     |
| container_format          | bare                                 |
| created_at                | 2015-11-19T14:46:38                  |
| deleted                   | False                                |
| disk_format               | raw                                  |
| id                        | 4dfbfd4f-2de5-4251-832c-e35a4a4145ee |
| is_public                 | True                                 |
| min_disk                  | 0                                    |
| min_ram                   | 0                                    |
| name                      | Solaris Non-global Zone              |
| owner                     | 7d1caf0854b24becb28df5c5cabf72cc     |
| protected                 | False                                |
| size                      | 845025280                            |
| status                    | active                               |
| updated_at                | 2015-11-19T14:46:42                  |
+---------------------------+--------------------------------------+
```

주 - Horizon 대시보드에서 동일한 이미지 정보를 얻을 수 있습니다.

Glance 이미지 만들기 스크립트 사용

`glance image-create` 명령은 이미지를 업로드하고 모든 등록 정보 값을 한 번에 설정할 수 있습니다. 다음 스크립트는 `architecture` 등록 정보가 현재 호스트의 아키텍처로 설정된 이미지를 업로드하는 방법을 보여줍니다.

```
#!/bin/ksh
```

```
# Upload Unified Archive image to glance with proper Solaris decorations

arch=$(archiveadm info -p $1|grep ^archive|cut -d '|' -f 4)

if [[ "$arch" == "i386" ]]; then
        imgarch=x86_64
else
        imgarch=sparc64
fi

name=$(basename $1 .uar)

export OS_USERNAME=glance
export OS_PASSWORD=glance
export OS_TENANT_NAME=service
export OS_AUTH_URL=http://controller-name:5000/v2.0

glance image-create --name $name --container-format bare --disk-format raw --owner service
--file $1 --is-public True --property architecture=$imgarch --property
 hypervisor_type=solariszones
--property vm_mode=solariszones --progress
```

♦ ♦ ♦ 5 장

5 클라우드 사용

이 장에서는 클라우드에서 여러 관리 작업을 수행하는 방법에 대해 설명합니다. 이러한 작업에 대해 대시보드 또는 명령줄을 사용할 수 있습니다. 대시보드에서 Project(프로젝트) 탭 아래의 작업에는 멤버 역할만 필요하지만, Admin(관리자) 탭 아래의 작업에는 관리 권한이 필요합니다. 대시보드에서 단일 로그인 세션으로 프로젝트에 대한 모든 작업을 수행하려면 해당 프로젝트에 대해 멤버 및 관리자 역할을 모두 보유해야 합니다.

이 장에서는 다음 항목을 다룹니다.

OpenStack 대시보드 액세스

OpenStack에 대한 설치 및 설치 후 구성 작업을 완료한 후 OpenStack 대시보드에 로그인하여 클라우드에서 사용 가능한 리소스를 확인하십시오.

▼ OpenStack 대시보드에 액세스하는 방법

1. **OpenStack 시스템에 연결할 수 있는 시스템에 로그인합니다.**
2. **사용 중인 브라우저를 구성합니다.**
   a. **JavaScript를 사용으로 설정합니다.**
   b. **쿠키를 보관합니다.**
3. **브라우저의 위치 또는 주소 필드에 다음 위치를 입력합니다.**

`http://`*system*`/horizon/`

*system*은 OpenStack 통합 아카이브가 설치되고 Apache 웹 서버에서 Horizon OpenStack 서비스를 실행 중인 OpenStack 시스템의 이름 또는 IP 주소입니다.

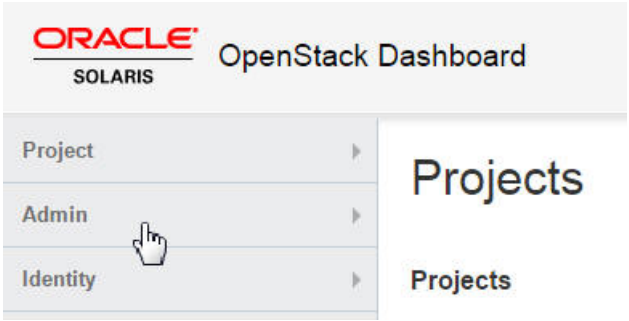
커널 영역에 통합 아카이브를 설치한 경우 OpenStack 시스템은 커널 영역이며 *system*은 커널 영역의 이름 또는 IP 주소입니다.

4. **로그인 화면에서 다음 정보를 제공합니다.**

   - 사용자 이름: `admin`
   - 암호: `secrete`

대시보드 살펴보기

처음에 `admin` 사용자로 Horizon 대시보드에 로그인하면 프로젝트 `demo`의 기본 페이지가 열립니다. `admin`은 `demo`에 대한 관리 권한을 가지므로 페이지의 왼쪽 패널에 Project(프로젝트), Admin(관리자) 및 Identity(ID)의 3개 탭이 표시됩니다. 관리 권한이 없으면 사용자는 Project(프로젝트) 및 Identity(ID) 탭만 보게 됩니다.



Admin(관리자) 패널의 Usage Summary Overview(사용량 요약 개요) 페이지는 기본 클라우드 관리자 뷰입니다.

그림 4 OpenStack 대시보드 관리자 개요 창

Admin(관리자) 패널의 선택 항목은 다음 기능을 제공합니다.

- 클라우드에서 사용 중인 Nova 인스턴스 및 Cinder 볼륨의 전체 뷰
- 다음과 같은 VM 인스턴스 특성을 정의하는 flavor 정의를 확인 및 편집할 수 있는 기능
  - 가상 CPU 수
  - 메모리 양
  - 지정된 디스크 공간
  - 기본 Oracle Solaris 영역의 브랜드: `solaris`(비전역 영역의 경우) 및 `solaris-kz`(커널 영역의 경우)
- 클라우드 관리자용 가상 네트워크 및 라우터를 만들 수 있는 기능
- 가상 컴퓨팅 리소스의 소유권을 그룹화하고 격리시켜서 프로젝트를 확인 및 편집할 수 있는 기능
- 클라우드 리소스를 사용하는 개인 또는 서비스인 사용자를 확인 및 편집할 수 있는 기능

단일 노드 구성에 대한 OpenStack UA는 다음 사전 구성된 리소스를 제공합니다.

- 이미지 2개: Solaris 비전역 영역 및 Solaris 커널 영역
- 두 가지 프로젝트 또는 테넌트: `demo` 및 `service`

  `demo` 프로젝트는 `admin`을 단일 멤버로 사용하는 기본 프로젝트입니다.

  `service` 프로젝트는 클라우드 관리자가 여러 프로젝트에 걸쳐 공유되는 리소스를 만드는 데 사용됩니다. 예를 들어, 이 문서에서는 Neutron 라우터를 `service` 프로젝트에 만들어서 이 라우터를 모든 프로젝트에서 공유하게 됩니다. OpenStack 설정에서 다른 목적으로 `service` 프로젝트를 사용하지 마십시오. OpenStack 서비스는 서비스별 사용자를 매개로 서로 통신하는데, 이들은 모두 `service` 프로젝트에서 관리 권한을 가집니다.
- flavor 10개

OpenStack Unified Archive에 포함된 저장소의 사전 구성된 이미지를 보려면 다음 탭 중 하나를 누릅니다.

- Admin(관리자) -> System(시스템) -> Images(이미지) 탭

- Project(프로젝트) -> Compute(컴퓨트) -> Images(이미지) 탭

자동으로 사용 가능한 Oracle Solaris flavor를 나열하려면 Admin(관리자) > System(시스템) > Flavors 탭을 누릅니다.

다음 비디오 프리젠테이션은 대시보드의 전체 개요를 제공합니다.

- The OpenStack Dashboard - Part 1
- The OpenStack Dashboard - Part 2

대시보드에서 수행할 수 있는 작업에 대한 자세한 내용은 5장. 클라우드 사용을 참조하십시오.

프로젝트 및 사용자 만들기

처음에 `admin` 사용자로 OpenStack에 로그인하면 `demo` 프로젝트의 기본 페이지가 표시됩니다. 이 프로젝트에서 다른 프로젝트를 만들 수 있습니다.

▼ 프로젝트를 만들고 사용자를 지정하는 방법

이 절차를 사용하여 새 프로젝트 또는 테넌트를 만들고 새 사용자로 채울 수 있습니다.

1. **브라우저에서 다음 링크와 유사한 URL을 사용하여 클라우드 관리자로 로그인합니다.**

   `http://`*system*`/horizon/`

   `demo` 프로젝트 기본 페이지가 표시됩니다.

   대시보드에 대한 간략한 개요는 "대시보드 살펴보기" [66]를 참조하십시오..

2. **왼쪽 패널에서 Identity(ID) -> Projects(프로젝트) 탭을 선택합니다.**
   기본 프로젝트인 `demo` 및 `service`가 표시됩니다.

3. **Create Project(프로젝트 생성)를 누릅니다.**

4. **Project Information(프로젝트 정보) 탭에서 프로젝트의 이름 및 설명을 지정합니다.**
   만들어지면 새 프로젝트가 프로젝트 목록에 추가됩니다.

5. **Identity(ID) -> Users(사용자) 탭을 선택합니다.**
   `demo` 및 `service`에 대한 기본 사용자가 표시됩니다.

6. **Create User(사용자 생성)를 누릅니다.**

7. **해당하는 필드에 필요한 정보를 제공합니다.**

   a. **새 사용자 이름 및 지정된 암호를 지정합니다.**

   b. **Primary Project(기본 프로젝트) 드롭다운 메뉴에서 새 사용자가 속할 프로젝트를 선택합니다.**

   c. **(옵션) Role(역할) 드롭다운 메뉴에서 프로젝트 사용자에 대한 역할을 선택합니다.**
   기본적으로 프로젝트의 새 사용자는 멤버 역할을 가집니다.

▼ 프로젝트를 기존 사용자로 채우는 방법

이 절차를 사용하여 새로 만들어진 프로젝트에 기존 사용자를 추가할 수 있습니다.

1. **기본 화면의 왼쪽 패널에서 Identity(ID) -> Projects(프로젝트) 탭을 선택합니다.**

2. **기존 사용자를 추가하려는 프로젝트에 대해 Modify Users(사용자 수정)를 누릅니다.**
   Edit Project(프로젝트 편집) 대화 상자가 표시됩니다.

3. **All Users(모든 사용자) 목록에서 프로젝트에 추가하려는 사용자 이름의 오른쪽에 있는 플러스(+) 기호를 누릅니다.**
   기본적으로 추가된 사용자는 해당 프로젝트에서 멤버 역할을 가집니다.

주 - `service` 프로젝트의 다음 사용자를 다른 프로젝트에 추가하지 **마십시오**.

- glance
- cinder
- swift
- neutron
- nova
- ec2

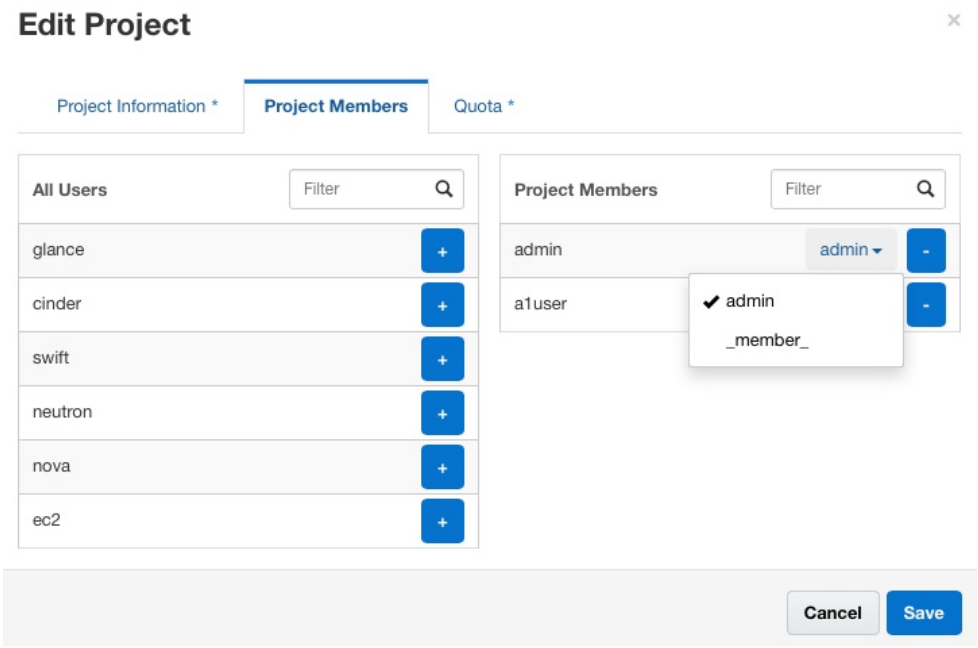
---

4. **(옵션) 프로젝트의 사용자에 대한 역할을 수정하려면 다음 단계를 수행하십시오.**

   a. **프로젝트 멤버 목록에서 역할을 변경하고자 하는 사용자에 대한 드롭다운 메뉴를 엽니다.**

   b. **사용자에게 지정할 새 역할을 선택합니다.**

이 샘플 그림에서 현재 프로젝트의 프로젝트 멤버는 `a1user` 및 `admin`입니다. `admin` 사용자에게는 프로젝트에 대한 관리 권한이 부여되었습니다. 사용자에게 `member` 및 `admin` 역할을 모두 지정할 수 있습니다.



프로젝트에 대한 내부 네트워크 만들기

프로젝트에는 각각 해당하는 가상 시스템 인스턴스를 서비스하는 여러 내부 네트워크가 있을 수 있습니다. 기본적으로 사용자 통신은 네트워크 내부만으로 제한됩니다. 클라우드 네트워크를 구성할 수 있으려면 먼저 사이트에 작동하는 네트워크가 있어야 합니다.

▼ 프로젝트에 대한 네트워크를 구성하는 방법

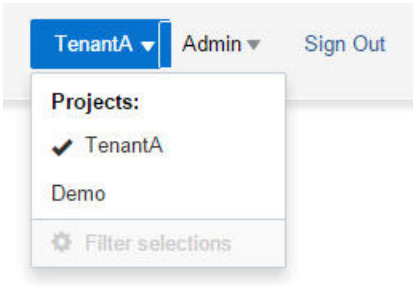
프로젝트에 대한 내부 네트워크를 만들려면 적어도 프로젝트의 멤버여야 합니다. 이 절차를 수행하기 위해 관리 권한이 필요하지는 않습니다.

1. **브라우저에서 다음 주소와 유사한 URL을 사용하여 Horizon 대시보드에 로그인합니다.**

   `http://`*system*`/horizon/`

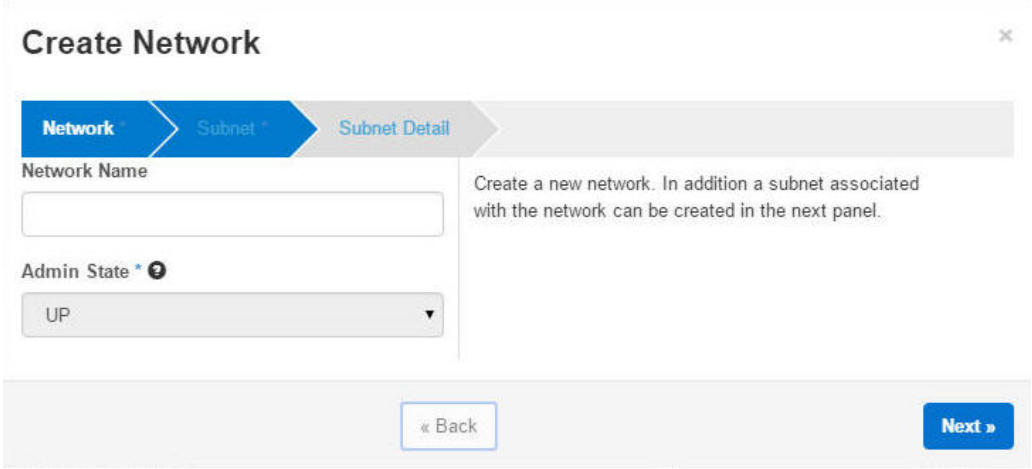
2. **기본 화면의 상단 오른쪽에 있는 프로젝트 이름을 확인하여 올바른 프로젝트에 로그인되었는지 확인합니다.**

   다음 예는 2개의 사용 가능한 프로젝트 중에서 `TenantA`가 선택되었음을 보여줍니다. `admin` 사용자가 현재 로그인되어 있습니다.



3. **왼쪽 패널에서 Project(프로젝트) -> Network(네트워크) -> Networks(네트워크) 탭을 선택한 다음 Create Network(네트워크 생성)를 누릅니다.**

   Create Network(네트워크 생성) 대화 상자가 표시됩니다.



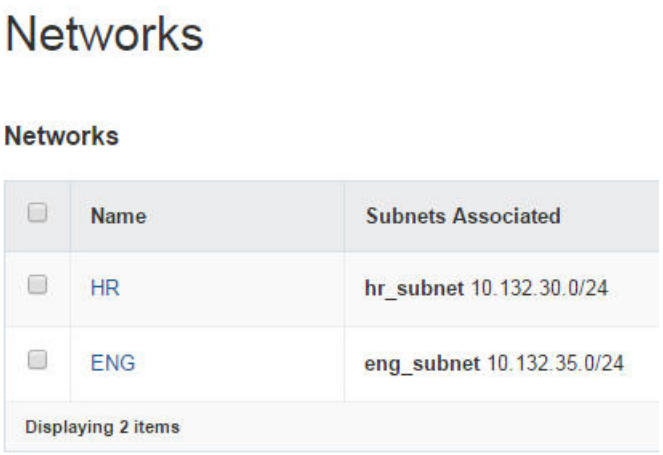
4. **각 탭을 눌러 필요에 따라 정보를 제공합니다.**

주 - Next(다음)를 누르면 각 탭에 대한 화면이 표시됩니다.

각 탭에서는 다음 정보를 요구합니다.

- Network(네트워크) 탭
  - Network Name(네트워크 이름)
  - Admin State(관리자 상태) - 기본값을 사용합니다.
- Subnet(서브넷) 탭
  - Subnet Name(서브넷 이름)
  - Network Address(네트워크 주소)
  - IP Version(IP 버전)
  - Gateway IP(게이트웨이 IP) - 기본값을 사용하려면 비워 둡니다.
- Subnet Detail(서브넷 상세정보) 탭
  - Use DHCP(DHCP 사용) - DHCP를 사용하지 않는 경우 선택 해제합니다.
  - Allocation Pools(Pools 할당)
  - DNS Server Names(DNS 서버 이름)
  - Host Routes(호스트 경로)

정보를 제공한 후 Create(생성) 버튼을 누르면 Networks(네트워크) 화면에 다음 예와 유사한 네트워크 및 연관된 서브넷이 표시됩니다.



다음 순서 내부 네트워크를 공용 네트워크에 연결하려는 경우 클라우드의 외부 네트워크에 해당 서브넷을 추가합니다. 내부 네트워크에 외부 연결을 제공하는 방법 [57]을 참조하십시오.

▼ 유동 IP 주소를 프로젝트와 연결하는 방법

외부 네트워크 구성에는 유동 IP 주소 만들기가 포함됩니다. 외부 네트워크를 만드는 방법 [54]을 참조하십시오. 이 절차를 사용하여 이러한 IP 주소 중 일부를 프로젝트에 할당할 수 있습니다.

1. **왼쪽 패널에서 Project(프로젝트) -> Compute(컴퓨트) -> Access & Security(접근 & 보안) 탭을 선택합니다.**
2. **Floating IPs(유동 IP) 탭을 누릅니다.**
3. **Allocate IP To Project(프로젝트에 IP 할당) 버튼을 누릅니다.**
   Allocate Floating IP(유동 IP 할당) 대화 상자가 열립니다.
4. **드롭다운 메뉴에서 유동 IP를 할당할 풀을 선택합니다.**
5. **대화 상자에서 Allocate IP(IP 할당) 버튼을 누릅니다.**
   IP 주소가 유동 IP 목록에 추가됩니다. IP는 원하는 만큼 또는 할당량으로 허용된 만큼 할당할 수 있습니다.

VM 인스턴스 만들기 및 부팅

이 절의 절차를 수행하려면 최소한 프로젝트의 유효한 멤버여야 합니다. 관리 권한이 필요하지는 않습니다.

▼ SSH 키 쌍을 만드는 방법

1. **브라우저에서 다음 주소와 유사한 URL을 사용하여 Horizon 대시보드에 로그인합니다.**

   `http://system/horizon/`

2. **기본 화면의 상단 오른쪽에 있는 프로젝트 이름을 확인하여 올바른 프로젝트에 로그인되었는지 확인합니다.**
3. **왼쪽 패널에서 Project(프로젝트) -> Compute(컴퓨트) -> Access & Security(접근 & 보안) 탭을 누릅니다.**
4. **Key Pairs 탭에서 키 쌍을 만들지 또는 키 쌍을 가져올지 결정합니다.**

- **키 쌍을 만듭니다.**

  a. **Create Key Pair(Keypair 생성) 버튼을 누릅니다.**

  b. **Key Pair Name(Keypair 이름) 필드에 이름을 지정합니다.**

  c. **Create Key Pair(Keypair 생성) 버튼을 누릅니다.**
  새 키 쌍이 자동으로 다운로드됩니다. 그렇지 않을 경우 제공된 링크를 눌러 키 쌍을 다운로드합니다.
  Access & Security(접근 & 보안) 패널의 Key Pairs 탭에 새 키 쌍이 나열됩니다.

- **키 쌍을 가져옵니다.**

  a. **Import Key Pair(Keypair 가져오기) 버튼을 누릅니다.**

  b. **Key Pair Name(Keypair 이름) 필드에 이름을 지정합니다.**

  c. **터미널 창에서 `root` 사용자의 `.ssh/id_rsa.pub` 파일 내용을 복사하고 Public Key (공개 키) 필드에 붙여 넣습니다.**

  d. **Import Key Pair(Keypair 가져오기) 버튼을 누릅니다.**
  Access & Security(접근 & 보안) 패널의 Key Pairs 탭에 새 키 쌍이 나열됩니다.

▼ VM 인스턴스를 만드는 방법

시작하기 전에 SSH 키 쌍이 있는지 확인하십시오. SSH 키 쌍을 만드는 방법 [73]을 참조하십시오.

내부 네트워크가 정의되었는지 확인합니다. “프로젝트에 대한 내부 네트워크 만들기” [70]를 참조하십시오.

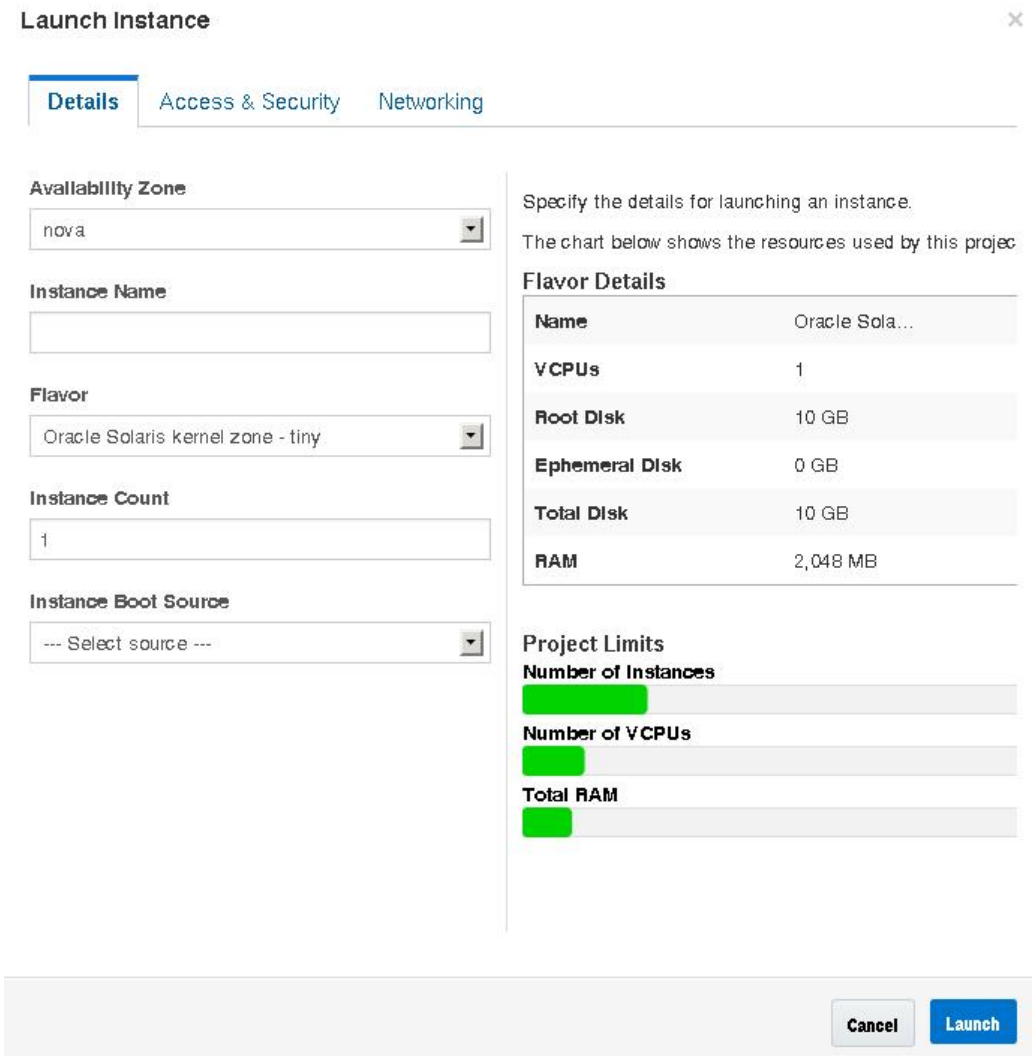
1. **브라우저에서 다음 주소와 유사한 URL을 사용하여 Horizon 대시보드에 로그인합니다.**

   `http://`*system*`/horizon/`

2. **기본 화면의 상단 오른쪽에 있는 프로젝트 이름을 확인하여 올바른 프로젝트에 로그인되었는지 확인합니다.**

3. **왼쪽 패널에서 Project(프로젝트) -> Compute(컴퓨트) -> Instances(인스턴스) 탭을 누른 다음 Launch Instance(인스턴스 구동)를 누릅니다.**

다음 Launch Instance(인스턴스 구동) 대화 상자가 표시됩니다.



4. **각 탭에서 메시지가 표시되면 정보를 제공합니다.**
   다음 필드에 대한 정보를 지정합니다.

   - Details(상세정보) 탭
     - Instance Name(인스턴스 이름)
     - Flavor - 드롭다운 메뉴에서 알맞은 flavor를 선택합니다. 이 OpenStack 시스템이 커널 영역이며 베어 메탈 시스템이 아닌 경우 비전역 영역 flavor를 선택해야 합니다.

- Instance Boot Source(인스턴스 부팅 소스) - 드롭다운 메뉴에서 Boot from image (이미지로 부팅)를 선택합니다. 그런 다음 사용할 이미지 이름을 선택합니다. 이 OpenStack 시스템이 커널 영역이며 베어 메탈 시스템이 아닌 경우 비전역 영역 이미지를 선택해야 합니다.

  flavor 및 이미지 유형이 일치해야 합니다. 예를 들어, 이미지가 비전역 영역 유형이면 flavor도 비전역 영역 유형이어야 합니다.
- Access and Security(액세스 및 보안) 탭 - 사용할 키 쌍을 선택합니다.
- Networking(네트워킹) 탭 - 사용 가능한 네트워크에서 새 VM을 연결할 네트워크를 선택합니다.

5. **대화 상자 하단에 있는 Launch(구동) 버튼을 누릅니다.**

   새 VM 인스턴스가 만들어지고 설치 및 부팅됩니다.

   이 단계를 완료하는 데 약간의 시간이 걸릴 수 있습니다.

6. **유동 IP 주소를 새 VM 인스턴스와 연결합니다.**

   새 VM 인스턴스를 설치하는 동안 이러한 단계를 수행할 수 있습니다. VM 인스턴스에서는 연결된 유동 IP 주소가 있어야만 사용자가 로그인할 수 있습니다.

   a. **Actions(작업) 열의 드롭다운 메뉴에서 Associate Floating IP(유동 IP 연결)를 선택합니다.**

      Manage Floating IP Associations(유동 IP 연관 관리) 대화 상자가 열립니다.

   b. **IP Address(IP 주소) 드롭다운 메뉴에서 주소를 선택합니다.**

      사용 가능한 IP 주소가 없는 경우 + 버튼을 누릅니다. 유동 IP 주소를 프로젝트와 연결하는 방법 [73]을 참조하십시오.

   c. **만든 VM에 해당하는 포트를 선택합니다.**

   d. **대화 상자 하단에 있는 Associate(연결) 버튼을 누릅니다.**

다음 순서

- Instances(인스턴스)를 누르고 인스턴스 이름을 누르면 인스턴스에 대한 세부정보가 표시되고 인스턴스의 콘솔 로그가 표시됩니다. 페이지를 다시 로드하여 로그 업데이트를 확인합니다.
- Volumes(볼륨)를 눌러 만들어진 Cinder 볼륨을 확인합니다.
- Network Topology(네트워크 토폴로지)를 눌러 모든 서브넷 세그먼트, 가상 라우터, 활성 인스턴스 등 클라우드 네트워크의 표현을 확인합니다.
- Images & Snapshots(이미지 & 스냅샷)를 눌러 Glance 이미지 저장소로 업로드된 통합 아카이브를 확인합니다.
- 새 VM 인스턴스 설치가 완료되고 인스턴스가 Active(활성) 상태에 도달하면 인스턴스에 로그인합니다. 다음 명령은 키 쌍과 유동 IP 주소를 사용하여 루트로 영역에 로그인합니다.

```
# ssh root@floating-IP-address
```

▼ VM 인스턴스에 사용자를 추가하는 방법

Oracle Solaris에서 VM 인스턴스는 Oracle Solaris 영역 기술을 사용하여 클라우드에 가상 시스템을 프로비전할 수 있도록 합니다. VM 인스턴스에 사용자를 추가하려면 영역 관리자로 명령을 실행해야 합니다. 이 단계는 대시보드에서 지원되지 않습니다. 그러므로 터미널 창에 액세스해야 합니다.

시작하기 전에 대시보드의 Admin(관리자) -> System(시스템) -> Instances(인스턴스) 탭을 눌러 VM 인스턴스가 연관된 외부 네트워크 유동 IP 주소를 확인합니다.

1. **터미널 창에서 VM 인스턴스를 나열합니다.**

   ```
   # zoneadm list -cv
   ```

   클라우드의 VM 이름에는 접두어 `instance`가 붙습니다.

2. **특정 영역에 로그인합니다.**

   ```
   # zlogin zonename
   ```

3. **사용자에 대한 홈 디렉토리를 만듭니다.**

   ```
   root@zone# mkdir -p /export/home/username
   ```

4. **사용자를 만듭니다.**

   ```
   root@zone# useradd -d home-dir options
   ```

   여기서 *home-dir*는 사용자에 대해 만든 디렉토리입니다. `useradd` 명령과 함께 사용할 수 있는 기타 옵션은 useradd(8) 매뉴얼 페이지를 참조하십시오.

5. **사용자 암호를 만들려면 이 명령을 실행한 다음 프롬프트를 따릅니다.**

   ```
   root@zone# passwd username
   ```

6. **(옵션) 암호가 만들어졌는지 확인합니다.**

   ```
   root@zone# grep username /etc/passwd
   ```

7. **`root` 암호를 만들려면 이 명령을 실행한 다음 프롬프트를 따릅니다.**

   ```
   zone# passwd root
   ```

8. **영역을 편집한 다음 로그아웃합니다.**

9. **보안 셸에서 가상 시스템에 로그인합니다.**

```
# ssh username@floating-IP
```

여기서 *floating-IP*는 VM의 연관된 유동 IP 주소입니다.

예 4 VM 인스턴스에 사용자 추가

이 예에서는 사용자 이름 `jsmith`가 VM1의 사용자로 추가됩니다.

```
# zoneadm list -cv
 ID NAME             STATUS     PATH                           BRAND    IP
  0 global           running    /                              solaris  shared
  6 instance-00000006 running    /system/zones/instance-00000006 solaris  excl
  - myzone           installed  /system/zones/myzone           solaris  excl

# zlogin instance-00000006
[Connected to zone 'instance-00000006' pts/3]
Last login: Wed Jan  6 14:31:18 2016 on pts/2
Oracle Corporation      SunOS 5.11      11.3    September 2015

root@VM1# mkdir -p /export/home/jsmith
root@VM1# useradd -d /export/home/jsmith -m -s /usr/bin/bash jsmith
```

`jsmith` 사용자는 기본 셸로 `bash`를 사용하여 만들어집니다.

```
root@VM1# passwd jsmith
New Password: password
Re-enter new Password: password
passwd: password successfully changed for jsmith

root@VM1# passwd root
New Password: password
Re-enter new Password: password
passwd: password successfully changed for root

root@VM1# exit
logout

[Connection to zone 'instance-00000006' pts/3 closed]

# ssh jsmith@10.132.10.9
```

Flavor 관리

*flavor*는 VM 인스턴스 유형 또는 가상 하드웨어 템플리트입니다. flavor는 일련의 가상 시스템 리소스(예: 가상 CPU 수, 메모리 양 및 VM 인스턴스에 지정된 디스크 공간)를 지정합니다. Oracle Solaris에서 flavor에는 기본 영역의 브랜드인 `solaris`(비전역 영역의 경우) 및

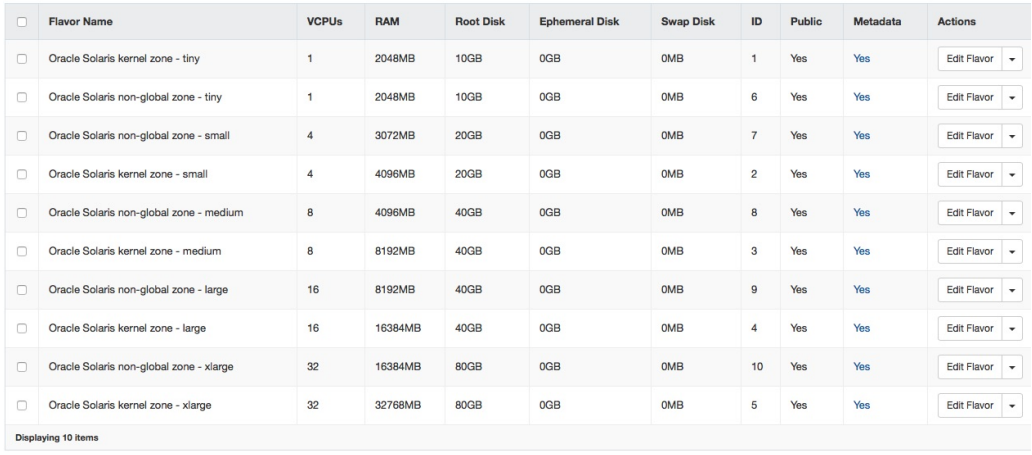
`solaris-kz`(커널 영역의 경우)도 포함됩니다. 가상 CPU가 16개이며 RAM이 16384MB인 커널 영역을 인스턴스 flavor의 예로 들 수 있습니다.

flavor에 대한 일반적인 정보는 *OpenStack Cloud Administrator Guide*의 "Flavors" 섹션을 참조하십시오.

Flavor에 대한 정보 표시

대시보드에 클라우드 관리자로 로그인되어 있으면 Admin(관리자) -> System(시스템) -> Flavor 탭에서 사용 가능한 flavor를 볼 수 있습니다.

그림 5 Oracle OpenStack용 Oracle Solaris의 Flavor



Flavor Name	VCPUs	RAM	Root Disk	Ephemeral Disk	Swap Disk	ID	Public	Metadata	Actions
Oracle Solaris kernel zone - tiny	1	2048MB	10GB	0GB	0MB	1	Yes	Yes	Edit Flavor
Oracle Solaris non-global zone - tiny	1	2048MB	10GB	0GB	0MB	6	Yes	Yes	Edit Flavor
Oracle Solaris non-global zone - small	4	3072MB	20GB	0GB	0MB	7	Yes	Yes	Edit Flavor
Oracle Solaris kernel zone - small	4	4096MB	20GB	0GB	0MB	2	Yes	Yes	Edit Flavor
Oracle Solaris non-global zone - medium	8	4096MB	40GB	0GB	0MB	8	Yes	Yes	Edit Flavor
Oracle Solaris kernel zone - medium	8	8192MB	40GB	0GB	0MB	3	Yes	Yes	Edit Flavor
Oracle Solaris non-global zone - large	16	8192MB	40GB	0GB	0MB	9	Yes	Yes	Edit Flavor
Oracle Solaris kernel zone - large	16	16384MB	40GB	0GB	0MB	4	Yes	Yes	Edit Flavor
Oracle Solaris non-global zone - xlarge	32	16384MB	80GB	0GB	0MB	10	Yes	Yes	Edit Flavor
Oracle Solaris kernel zone - xlarge	32	32768MB	80GB	0GB	0MB	5	Yes	Yes	Edit Flavor

Displaying 10 items

이러한 열에 대한 자세한 내용은 OpenStack Command-Line Interface Reference를 참조하십시오.

Flavor 사양 수정

각 flavor에 대한 Actions(실행) 열에서 처음 3개의 옵션 중 하나를 누르면 flavor의 등록 정보가 표시되며 등록 정보를 수정할 수 있습니다. 다음 3가지 작업이 가능합니다.

- `Edit Flavor`(Flavor 편집) flavor에 대한 정보 및 수정할 수 있는 등록 정보를 표시합니다. Flavor Access(Flavor 접근 권한) 탭에서는 프로젝트에서 액세스할 수 있는 flavor를

제한할 수 있습니다. 기본 설정은 없음이며, flavor가 공용이고 모든 프로젝트에서 액세스할 수 있음을 의미합니다.

- `Modify Access`(접근 권한 수정) flavor의 Flavor Access(Flavor 접근 권한) 탭을 바로 열어 액세스 설정을 수정할 수 있습니다.
- `Update Metadata`(메타데이터 업데이트) flavor의 메타데이터를 수정할 수 있습니다.

flavor의 사양을 수정하면 수정된 flavor를 사용하는 이후에 만드는 모든 게스트에 수정 사항이 적용됩니다.

대시보드에서 모든 flavor 수정 작업을 수행할 수 있는 것은 아닙니다. 예를 들어, `extra_specs` 등록 정보의 키는 명령줄에서만 수정할 수 있습니다. 등록 정보의 키는 일반적으로 `zonecfg` 명령으로 구성되고 OpenStack에서 지원되는 영역 등록 정보 세트를 가리킵니다.

다음 키는 커널 영역과 비전역 영역 flavor에서 지원됩니다.

- `zonecfg:bootargs`
- `zonecfg:brand`
- `zonecfg:hostid`
- `zonecfg:cpu-arch`

다음 키는 비전역 영역 flavor에서만 지원됩니다.

- `zonecfg:file-mac-profile`
- `zonecfg:fs-allowed`
- `zonecfg:limitpriv`

이러한 영역 구성 등록 정보에 대한 설명은 zonecfg(8) 매뉴얼 페이지를 참조하십시오.

주 - 모든 영역 구성 등록 정보가 OpenStack에서 지원되는 것은 아닙니다.

또한 `sc_profile` 키는 명령줄에서만 수정할 수 있습니다. 이 키를 사용하여 flavor에 대한 시스템 구성 프로파일을 지정합니다.

명령줄에서 flavor를 수정하려면 다음 구문을 사용합니다.

`nova flavor-key` *flavor action key=value* `[`*key=value* `...]`

flavor	flavor의 이름 또는 ID입니다.
action	`set` 또는 `unset`입니다.
key=value	*key*는 사양 이름입니다. *value*는 해당 사양에 대한 새 값입니다. *action*이 `unset`인 경우 *key*만 지정합니다.

예를 들어, flavor 목록의 8번째 flavor(`Oracle Solaris kernel zone - large`)에 대한 특정 시스템 구성 파일을 설정하려면 다음 명령을 실행합니다.

```
$ nova flavor-key 4 set sc_profile=/system/volatile/profile/sc_profile.xml
```

flavor 삭제 및 생성에 대한 자세한 내용은 OpenStack Admin User Guide를 참조하십시오.

▼ Flavor의 extra_specs 등록 정보를 수정하는 방법

1. **Neutron에 대한 전역 셸 변수를 설정합니다.**

```
controller# export OS_USERNAME=nova
controller# export OS_PASSWORD=service-password
controller# export OS_PROJECT_NAME=service
controller# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0
```

2. **사용 가능한 flavor를 표시합니다.**

```
controller# nova flavor-list
```

3. **수정할 flavor의 ID를 확인합니다.**

4. **해당 flavor의 extra_specs 키를 수정합니다.**

```
controller# nova flavor-key flavor action es-key=value
```

여기서 *es-key*는 extra_specs 등록 정보의 특정 키를 가리킵니다.

5. **(옵션) flavor의 등록 정보를 표시합니다.**

```
controller# nova flavor-show flavor
```

예 5 zonecfg:bootargs 키 변경

이 예는 ID가 8인 Oracle Solaris non-global zone - medium flavor의 zonecfg:bootargs 키를 수정하는 방법을 보여줍니다.

공간 절약을 위해 RTX_Factor 및 Is_Public 열은 다음 nova flavor-list 샘플 출력에서 생략되었습니다.

```
controller# nova flavor-list
+----+-----------------------------------------+-----------+------+-----------+------+-------+
| ID | Name                                    | Memory_MB | Disk | Ephemeral | Swap | VCPUs |
+----+-----------------------------------------+-----------+------+-----------+------+-------+
| 1  | Oracle Solaris kernel zone - tiny       | 2048      | 10   | 0         |      | 1     |
| 10 | Oracle Solaris non-global zone - xlarge | 16384     | 80   | 0         |      | 32    |
| 2  | Oracle Solaris kernel zone - small      | 4096      | 20   | 0         |      | 4     |
| 3  | Oracle Solaris kernel zone - medium     | 8192      | 40   | 0         |      | 8     |
| 4  | Oracle Solaris kernel zone - large      | 16384     | 40   | 0         |      | 16    |
| 5  | Oracle Solaris kernel zone - xlarge     | 32768     | 80   | 0         |      | 32    |
| 6  | Oracle Solaris non-global zone - tiny   | 2048      | 10   | 0         |      | 1     |
| 7  | Oracle Solaris non-global zone - small  | 3072      | 20   | 0         |      | 4     |
```

```
| 8  | Oracle Solaris non-global zone - medium | 4096      | 40   | 0         |      | 8     |
| 9  | Oracle Solaris non-global zone - large  | 8192      | 40   | 0         |      | 16    |
+----+-----------------------------------------+-----------+------+-----------+------+-------+

controller# nova flavor-key 8 set zonecfg:bootargs=-v
controller# nova flavor-show 8

+---------------------------+----------------------------------------+
| Property                  | Value                                  |
+---------------------------+----------------------------------------+
| OS-FLV-DISABLED:disabled  | False                                  |
| OS-FLV-EXT-DATA:ephemeral | 0                                      |
| disk                      | 40                                     |
| extra_specs               | {"zonecfg:brand": "solaris"}           |
|                           |    u'zonecfg:bootargs': u'-v'}         |
| id                        | 8                                      |
| name                      | Oracle Solaris non-global zone - medium |
| os-flavor-access:is_public | True                                   |
| ram                       | 4096                                   |
| rxtx_factor               | 1.0                                    |
| swap                      |                                        |
| vcpus                     | 8                                      |
+---------------------------+----------------------------------------+
```

수정된 *bootargs*

VM 인스턴스 관리

이 절에서는 VM 마이그레이션 또는 크기 조정 및 VM 부트 옵션 설정과 같이 클라우드에서 만든 VM 인스턴스를 수정하는 방법에 대해 설명합니다.

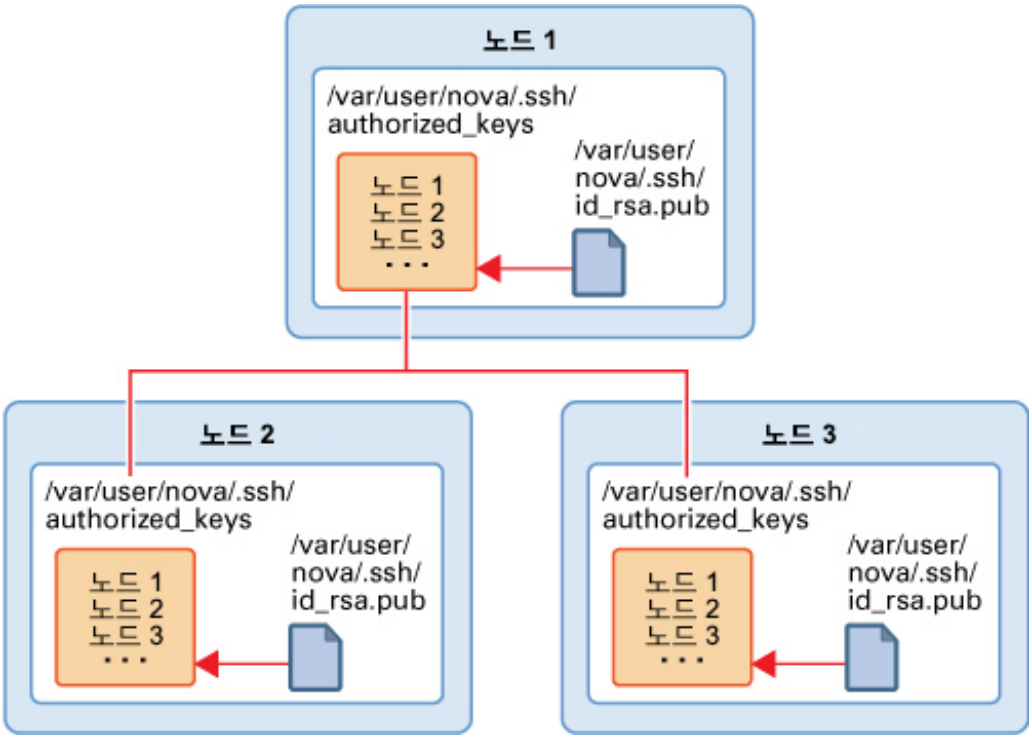
VM 인스턴스 마이그레이션 및 비우기

Oracle Solaris용 Oracle OpenStack에서 VM 인스턴스는 Oracle Solaris의 확장성 높은 고밀도 가상 환경인 커널 또는 비전역 영역입니다. 영역 라이브 마이그레이션에 대한 지원은 Nova VM 인스턴스까지 확장됩니다.

Horizon 서비스의 대시보드 또는 `nova` 명령을 사용하여 라이브 마이그레이션을 시작할 수 있습니다. 스케줄러는 참여 노드 중에서 마이그레이션의 대상 호스트를 선택합니다. 보안을 보장하기 위해 마이그레이션 방식이 마이그레이션 수행에 적합한 암호화 알고리즘을 자동으로 선택합니다. 하지만 `/etc/nova/nova.conf` 파일의 매개변수를 사용하면 사용할 선호 암호화를 선택할 수 있습니다.

영역 라이브 마이그레이션에 대한 자세한 내용은 Operating Systems Documentation에서 사용 중인 Oracle Solaris 버전 라이브러리의 커널 영역 마이그레이션에 대한 *Oracle Solaris* 커널 영역 만들기 및 사용을 참조하십시오. `zoneadm`(8) 및 `solaris-kz`(7) 매뉴얼 페이지도 참조하십시오.

노드 마이그레이션을 성공하려면 각 컴퓨트 노드의 SSH 키가 각 노드의 권한이 부여된 키 파일에 추가되었는지 확인해야 합니다. 그러면 노드는 다음 그림에 나온 대로 권한이 부여된 키의 동일한 파일을 공유합니다.



다음 단계를 완료하여 VM 인스턴스 마이그레이션을 준비하십시오.

1. 각 노드에서 SSH 키를 만듭니다.

   ```
   # su - nova -c "ssh-keygen -N '' -f /var/user/nova/.ssh/id_rsa -t rsa"
   ```

2. 여러 노드의 모든 키 파일을 노드 중 하나의 공통 위치로 가져옵니다.
3. 모든 키를 `authorized_keys` 파일로 결합합니다.

   예를 들면 다음과 같습니다.

   ```
   # cat nova(1)/id_rsa.pub nova(n)/id_rsa.pub >> /var/user/nova/.ssh/authorized_keys
   ```

   여기서 *nova(1)* - *nova(n)*은 참여 노드의 SSH 키를 나타냅니다.
4. `authorized_keys` 파일을 다른 모든 참여 노드의 `/var/user/nova/.ssh` 디렉토리에 배포합니다.
5. 선택적으로 각 컴퓨트 노드의 `/etc/nova/nova.conf` 파일에 있는 `live_migration_cipher` 매개변수에서 마이그레이션 중 사용할 암호화를 지정합니다.

하지만 프로세스가 적당한 암호화를 자동으로 선택하도록 하려면 매개변수를 설정하지 않은 상태로 둡니다.

실행 중인 서버를 다른 시스템으로 라이브 마이그레이션을 수행하려면 전역 셸 변수를 설정한 후 다음 구문을 사용합니다.

```
# nova live-migration server [host]
```

여기서 *server*는 서버의 이름 또는 ID가 될 수 있고, 선택적 *host*는 대상 서버의 이름입니다.

현재 인스턴스의 노드가 실패하거나 Nova 서비스 자체가 일정 기간 동안 사용 안함으로 설정되면 인스턴스를 이동하거나 비우고 다른 노드에서 재구성할 수 있습니다. 따라서 노드를 복구할 수 있습니다.

주 - 커널 영역만 비울 수 있습니다. 비우기는 루트 장치가 공유 저장소에 있는 구성에서 지원됩니다.

한 호스트에서 다른 호스트로 모든 VM 인스턴스의 라이브 마이그레이션을 수행하려면 전역 셸 변수를 설정한 후 다음 구문을 사용합니다.

```
# nova host-evacuate-live [--target-host target] server
```

VM 인스턴스 크기 조정

VM의 크기는 VM이 실행되는 기반인 flavor로 지정됩니다. VM 인스턴스 만들기 단계는 VM 인스턴스를 만드는 방법 [74]을 참조하십시오. 다음 그림은 Horizon 대시보드에 표시된 대로 샘플 VM `hr_vm1`에 대한 세부정보를 보여줍니다.

그림 6 VM 인스턴스 크기

그림은 hr_vm1의 크기가 Oracle Solaris non-global zone - tiny flavor에 대해 정의된 크기임을 보여줍니다. 인스턴스 이름을 누르면 특정 flavor에 대한 세부정보를 볼 수 있습니다.

- RAM = 2GB
- VCPU = 1 VCPU
- 디스크 = 10GB

인스턴스 크기 조정은 인스턴스에 대해 다른 flavor 사용을 의미합니다.

기본적으로 크기 조정 프로세스는 새 노드에서 새로운 크기의 인스턴스를 만듭니다. 하지만 비전역 영역 flavor로 크기를 조정하는 경우 크기 조정 프로세스가 동일한 노드에서 완료되도록 설정해야 합니다. 비전역 영역은 원래 전역 영역과 동일한 버전이어야 합니다. 비전역 영역 flavor로 크기를 조정하고 다른 노드에서 만들어진 인스턴스가 있을 경우 다른 전역 영역 버전을 가지는 노드의 인스턴스가 위험해집니다. 위험을 방지하려면 동일한 노드에 새로 크기 조정된 인스턴스를 만드는 /etc/nova/nova.conf 파일에서 다음 매개변수를 편집합니다.

```
allow_resize_to_same_host=true
```

주 - 이 위험은 커널 영역에 적용되지 않습니다. 따라서 다른 노드에 대한 커널 영역의 크기를 안전하게 조정할 수 있습니다.

▼ VM 인스턴스 크기를 조정하는 방법

시작하기 전에 수정할 VM 인스턴스의 현재 크기를 확인합니다. 이 정보는 대시보드에서 얻을 수 있습니다. 예는 그림 6를 참조하십시오.

1. **전역 셸 변수를 설정합니다.**

```
# export OS_USERNAME=nova
# export OS_PASSWORD=service-password
# export OS_PROJECT_NAME=service
# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0"
```

2. **비전역 영역 flavor로 크기를 조정하는 경우 다음과 같이 /etc/nova/nova.conf 파일에서 다음 매개변수를 편집합니다.**

```
allow_resize_to_same_host=true
```

3. **flavor 목록을 표시합니다.**

```
# openstack flavor list
```

4. **VM 인스턴스에서 사용할 새 flavor에 대한 ID 번호를 확인합니다.**

5. **flavor의 ID 번호를 참조하여 VM 인스턴스에서 새 flavor를 사용하도록 변경합니다.**

```
# openstack server resize --flavor flavor-ID instance-name
```

예를 들어, 현재 `hr_vm1` 인스턴스의 크기가 ID 번호가 6인 `Oracle Solaris non-global zone - tiny` flavor로 설정되어 있다고 가정해 보겠습니다. 인스턴스의 크기를 ID번호가 8인 `Oracle Solaris non-global zone - medium` flavor로 저장하기를 원합니다. 다음을 입력합니다.

```
# openstack server resize --flavor 8 hr_vm1
```

인스턴스 이름을 지정하기 전에 추가 옵션(크기 조정을 완료할 때까지 대기하려는 경우 `--wait` 또는 서버 크기 조정이 완료되면 확인하려는 경우 `--confirm`)을 삽입할 수 있습니다.

6. **(옵션) 대시보드에서 인스턴스의 크기가 새 flavor로 변경되었는지 확인합니다.**

   또는 다음 명령을 사용하여 동일한 확인 작업을 수행할 수 있습니다.

   ```
   # openstack server show instance-name
   ```

   `flavor` 필드에 대한 값에 이전 단계의 새 flavor가 지정되었는지 확인합니다.

   다음 예는 추출된 출력입니다. `hr_vm1` 인스턴스는 `TenantA`라는 프로젝트에서 만들어졌고 `Oracle Solaris non-global zone - medium`으로 크기가 조정되었습니다.

   ```
   # export OS_USERNAME=admin
   # export OS_PASSWORD=admin-password
   # export OS_PROJECT_NAME=TenantA
   # export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0"

   # openstack server show hr_vm1
   +---------------+-----------------------------------------------+
   | Property      | Value                                         |
   +---------------+-----------------------------------------------+
   | ...                                                           |
   | created       | 2016-01-26T12:38:47Z                          |
   | flavor        | Oracle Solaris non-global zone - medium (8)   |
   | ...                                                           |
   +---------------+-----------------------------------------------+
   ```

♦ ♦ ♦ 6 장

6 Cinder 구성 및 배치 옵션

이 장에서는 OpenStack 설정의 저장소 구성요소 및 Cinder를 구성할 수 있는 다른 방법에 대해 설명합니다.

주 - 블록 저장소를 구성하기 위해 “저장소 노드 구성” [45]에서 설명하는 절차는 기본 방법을 사용하여 저장소 노드를 설정합니다. 이 장의 옵션은 저장소 구성요소를 배치하는 다른 방법을 제공합니다. 그러나 기본 저장소 구성 및 대체 구성을 동시에 사용하여 클라우드 프레임워크에 Cinder를 설정하는 것은 현재 권장되지 않습니다.

이 장에서는 다음 내용을 다룹니다.

저장소용 원격 시스템 배치

SAN을 지원하지 않는 이전 OpenStack 릴리스에서는 ZFS iSCSI 드라이버를 사용할 경우 Cinder 볼륨 서비스를 대상 호스트에서 실행되도록 구성해야 했습니다.

Cinder의 SAN(Storage Area Networks)에 대한 지원을 통해 SSH를 사용하여 다중 저장소 호스트 백엔드를 구성할 수 있을 뿐 아니라 볼륨 유형을 정의할 수도 있습니다. 이 배치 유형에서 Cinder 패키지 및 모든 Cinder 서비스는 일반적으로 컴퓨트 노드처럼 작동하는 호스트인 개시자 호스트에만 설치됩니다.

OpenStack 패키지를 원격 대상 호스트에 설치할 필요가 없습니다. 이러한 호스트는 COMSTAR 기반의 개시자 호스트에 단순히 LUN 디스크를 제공합니다.

저장소용 원격 시스템의 적합한 배치를 위한 요구사항은 다음과 같습니다.

- `/etc/cinder/cinder.conf` 파일 구성
- 지정된 사용자에게 적합한 권한 프로파일 부여
- 추가 패키지 수동 설치

다음 절에서는 이러한 요구사항에 대해 자세히 설명합니다.

cinder.conf 파일 구성

Cinder 패키지를 개시자 호스트에 설치한 후 /etc/cinder/cinder.conf 파일의 [DEFAULT] 섹션을 편집합니다.

구성 파일의 정의를 이해하려면 다음 예를 참조하십시오.

```
[DEFAULT]
my_ip = localhost
osapi_volume_workers = 1
auth_strategy = keystone
#num_volume_device_scan_tries = 6
os-volume-api-version = 2
scheduler_driver=cinder.scheduler.filter_scheduler.FilterScheduler

enabled_backends=zfsdriver-1, zfsdriver-2, zfsdriver-3, zfsdriver-4, zfsdriver-5

[zfsdriver-1]
volume_group=zvolumes-1
volume_driver=cinder.volume.drivers.solaris.zfs.ZFSVolumeDriver
volume_backend_name=ZFS_LOCAL
zfs_volume_base = rpool/cinder
san_is_local = True
debug=true
verbose=true
[zfsdriver-2]
volume_group=zvolumes-2
volume_driver=cinder.volume.drivers.solaris.zfs.ZFSISCSIDriver
volume_backend_name=ZFS_REMOTE
zfs_volume_base = rpool/cinder
san_ip = 10.134.13.38
san_login = user-name
san_password = password
debug=true
verbose=true
[zfsdriver-3]
volume_group=zvolumes-3
volume_driver=cinder.volume.drivers.solaris.zfs.ZFSFCDriver
volume_backend_name=ZFS_REMOTE_FC
zfs_volume_base = rpool/fc
san_ip = 10.134.13.38
san_login = user-name
san_password = password
debug=true
verbose=true
[zfsdriver-4]
volume_group=zvolumes-4
volume_driver=cinder.volume.drivers.solaris.zfs.ZFSISCSIDriver
volume_backend_name=ZFS_REMOTE
```

```
zfs_volume_base = rpool/cinder/zq
san_ip = 10.134.13.38
san_login = user-name
san_password = password
debug=true
verbose=true
[zfsdriver-5]
volume_group=zvolumes-5
volume_driver=cinder.volume.drivers.solaris.zfs.ZFSISCSIDriver
volume_backend_name=ZFS_REMOTE
zfs_volume_base = rpool/zq
san_ip = 10.134.63.182
san_login = user-name
san_password = password
debug=true
verbose=true
```

`enabled_backends` 사용으로 설정된 백엔드 호스트를 나열합니다. 이 예에서는 5개의 백엔드 호스트가 정의되어 있습니다.

백엔드 호스트는 3개의 호스트에서 3개의 ZFS 드라이버(`ZFSVolumeDriver`, `ZFSISCSIDriver`, `ZFSFCDriver`)를 사용합니다. 이러한 호스트 중 하나는 로컬(`localhost`)이고, 나머지는 원격(`10.134.13.38` 및 `10.134.63.182`)입니다.

`volume_backend_name` 볼륨 유형을 지정된 이름으로 정의합니다. 이 매개변수는 볼륨 유형을 식별합니다. 하지만 다음 명령을 사용하여 볼륨 유형을 수동으로 만들어야 합니다.

```
# cinder type-create vol-type
# cinder type-key vol-type set volume_backend_name=backend-name
# cinder create --display-name display --volume-type vol-type
```

이러한 명령은 각각 다음 작업을 실행합니다.

- 새 볼륨 유형을 만듭니다.
- 백엔드 이름을 새 볼륨 유형에 지정합니다.
- 새 볼륨 유형에 따라 새 볼륨을 만듭니다.

이전 샘플 Cinder 구성 파일을 바탕으로 하여 다음 명령을 입력합니다.

```
# cinder type-create type-remote
# cinder type-key type-remote set volume_backend_name=ZFS_REMOTE
# cinder create --display-name t1 --volume-type type-remote
```

마지막 명령은 새 볼륨 `t1`을 만듭니다. t1은 필터링 규칙을 기준으로 이름이 ZFS_REMOTE인 백엔드 중 하나에서 생성됩니다.

`ZFS_LOCAL` 및 `ZFS_REMOTE_FC`에 대한 볼륨 유형을 만들기 위한 동일한 명령 세트를 실행합니다.

`zfs_volume_base` 새 ZFS 볼륨의 기본 데이터 세트를 각 볼륨 백엔드에 지정합니다.

`san_is_local`
`san_ip san_login`
`san_password`

모든 ZFS 드라이버가 기반으로 하는 SAN 드라이버의 매개변수입니다. 이러한 매개변수를 설정하여 백엔드 호스트에서 로컬로 또는 SSH를 사용하여 원격으로 실행할 명령을 사용으로 설정해야 합니다.

각 백엔드의 경우 다음 두 가지 방법 중 하나로 SAN 매개변수를 설정합니다.

- `san_is_local = True`만 설정
- `san_ip`, `san_login` 및 `san_password`를 함께 설정

SAN 매개변수 4개를 모두 설정하지 마십시오.

`debug=true`
`verbose=true`

디버깅을 위한 선택적 매개변수입니다. 이러한 매개변수 구성은 생략할 수 있습니다.

Cinder를 올바르게 구성한 후 서비스를 나열하면 각각의 서비스 상태가 표시됩니다.

```
# cinder service-list
+------------+--------------+------+---------+-------+--------------------+----------+
|  Binary    |     Host     | Zone |  Status | State |      Updated_at    | Disabled |
|            |              |      |         |       |                     |  Reason  |
+------------+--------------+------+---------+-------+--------------------+----------+
|  cinder-   |              |      |         |       |     2015-10-13     |          |
|  backup    |    host-2    | nova | enabled |   up  |  T19:22:45.000000  |  None    |
|            |              |      |         |       |                    |          |
|  cinder-   |              |      |         |       |     2015-10-13     |          |
| scheduler  |    host-2    | nova | enabled |   up  |  T19:22:43.000000  |  None    |
|            |              |      |         |       |                    |          |
|  cinder-   |              |      |         |       |     2015-10-13     |          |
|  volume    |    host-2    | nova | enabled |  down |  T18:31:41.000000  |  None    |
|            |              |      |         |       |                    |          |
|  cinder-   |    host-2    |      |         |       |     2015-10-13     |          |
|  volume    | @zfsdriver-1 | nova | enabled |   up  |  T19:22:46.000000  |  None    |
|            |              |      |         |       |                    |          |
|  cinder-   |    host-2    |      |         |       |     2015-10-13     |          |
|  volume    | @zfsdriver-2 | nova | enabled |   up  |  T19:22:47.000000  |  None    |
|            |              |      |         |       |                    |          |
|  cinder-   |    host-2    |      |         |       |     2015-10-13     |          |
|  volume    | @zfsdriver-3 | nova | enabled |   up  |  T19:22:48.000000  |  None    |
|            |              |      |         |       |                    |          |
|  cinder-   |    host-2    |      |         |       |     2015-10-13     |          |
|  volume    | @zfsdriver-4 | nova | enabled |   up  |  T19:22:47.000000  |  None    |
|            |              |      |         |       |                    |          |
|  cinder-   |    host-2    |      |         |       |     2015-10-13     |          |
|  volume    | @zfsdriver-5 | nova | enabled |   up  |  T19:22:48.000000  |  None    |
|            |              |      |         |       |                    |          |
|  cinder-   |    host-2    |      |         |       |     2015-10-13     |          |
|  volume    | @zfsdriver-6 | nova | enabled |  down |  T18:32:55.000000  |  None    |
+------------+--------------+------+---------+-------+--------------------+----------+
```

지정된 사용자에게 권한 부여

san_login에 정의된 사용자가 원격 대상을 사용할 수 있도록 하려면 해당 사용자에게 적절한 권한 프로파일을 부여해야 합니다. 다음 예에서는 사용자 권한 프로파일을 만드는 방법을 보여줍니다.

```
# useradd -s /usr/bin/pfbash -m jdoe
# passwd jdoe password
# profiles -p "Cinder Storage management"
profiles:Cinder Storage management> set desc="Cinder Storage management on target host"
profiles:Cinder Storage management> add profiles="File System Management"
profiles:Cinder Storage management> add auths="solaris.smf.modify.stmf"
profiles:Cinder Storage management> add cmd=/usr/sbin/itadm
profiles:Cinder Storage management:itadm> set euid=0
profiles:Cinder Storage management:itadm> end
profiles:Cinder Storage management> add cmd=/usr/sbin/fcadm
profiles:Cinder Storage management:itadm> set privs=file_dac_read,sys_devices
profiles:Cinder Storage management:itadm> end
profiles:Cinder Storage management> add cmd=/usr/sbin/fcinfo
profiles:Cinder Storage management:itadm> set privs=file_dac_read,sys_devices
profiles:Cinder Storage management:itadm> end
profiles:Cinder Storage management> add cmd=/usr/sbin/stmfadm
profiles:Cinder Storage management:stmfadm> set euid=0
profiles:Cinder Storage management:stmfadm> end
profiles:Cinder Storage management> add cmd=/usr/lib/rad/module/mod_zfsmgr.so.1
profiles:Cinder Storage management:mod_zfsmgr.so.1> set privs={zone}:/system/volatile/*, \
   sys_config,sys_mount
profiles:Cinder Storage management:mod_zfsmgr.so.1> end
profiles:Cinder Storage management> add cmd=/usr/sbin/zfs
profiles:Cinder Storage management:zfs> set priv=sys_config,sys_mount
profiles:Cinder Storage management:zfs> end
profiles:Cinder Storage management> exit

# usermod -P "Cinder Storage management" jdoe
```

프로파일 및 권한에 대한 자세한 내용은 Operating Systems Documentation의 사용 중인 Oracle Solaris 버전 라이브러리에서 *Securing Users and Processes in Oracle Solaris*를 참조하십시오.

원격 호스트를 대상으로 설정

대상인 원격 호스트를 사용으로 설정하려면 다음 단계를 수행합니다.

1. 원격 호스트에 group/feature/storage-server 패키지를 설치합니다.

   ```
   # pkg install storage-server
   ```

2. 개시자 및 원격 호스트 모두에 다음 서비스를 사용으로 설정합니다.

- svc:/system/stmf:default
- svc:/network/iscsi/target:default
- svc:/system/rad:remote

예를 들어 다음을 실행합니다.

```
remote-host# svcadm enable stmf
remote-host# svcadm enable -r svc:/network/iscsi/target:default
remote-host# svcadm enable rad
```

3. 정의된 zfs_volume_base에 대해 ACL(액세스 제어 목록)을 초기화하고 설정합니다.

   예를 들어 구성 파일에 zfs_volume_base=rpool/fc 정의가 있다고 가정합니다. 이 경우 다음 명령을 실행해야 합니다.

```
# chmod A+user:cinder:add_subdirectory:allow /rpool/fc
# zfs allow cinder clone,create,destroy,mount,snapshot rpool/fc
```

컴퓨트 노드에 대한 부트 볼륨 지정

다중 저장소 백엔드를 사용하는 구성 또는 다중 랙이나 다중 ZFS 저장소 ZS3 환경에서 만드는 모든 새 인스턴스에 대한 루트 볼륨을 배치할 위치를 제어해야 하는 경우가 있습니다. Oracle Solaris용 OpenStack에서 /etc/nova/nova.conf 파일의 두 가지 매개변수를 통해 이를 제어할 수 있습니다.

▼ 컴퓨트 인스턴스에 대한 루트 저장소 볼륨을 만드는 방법

이 절차는 "저장소용 원격 시스템 배치" [87]에 설명되어 있는 대로 다중 원격 백엔드 정의에 대한 일반적인 작업에 속합니다. 따라서 이 절차는 동일한 예제를 사용합니다.

1. **/etc/cinder/cinder.conf 파일에서 Cinder 백엔드를 정의합니다.**

   "cinder.conf 파일 구성" [88]에 나와 있는 예제, 특히 enabled_backends 및 각 백엔드에 대한 볼륨 이름 지정에 대한 예제를 참조하십시오.

2. **설정에서 Cinder 가용성 영역을 사용하는 경우 Cinder 구성 파일에서도 정의합니다.**

   예를 들어 다음을 실행합니다.

```
[DEFAULT]
...
storage_availability_zone=cinder_az
```

3. **구성 파일에 정의된 각 백엔드의 경우 다음 예제에 나와 있는 것처럼 해당하는 볼륨을 만듭니다.**

```
# cinder type-create type-remote
# cinder type-key type-remote set volume_backend_name=ZFS_REMOTE
```

필요한 경우 동일한 명령 세트를 실행하여 다른 볼륨 유형을 만듭니다. 그러면 `solariszones` 드라이버가 Nova 인스턴스에 대한 실제 Cinder 부트 볼륨을 만듭니다.

4. **Cinder 노드에서 Cinder 서비스를 다시 시작합니다.**

```
# svcadm restart cinder-volume:default
```

5. **각 컴퓨트 노드의 `/etc/nova/nova.conf` 파일에서 Cinder 구성 파일을 기반으로 하는 다음 매개변수 중 하나 또는 두 개를 정의합니다.**

- `boot_volume_type`
- `boot_volume_az`

예를 들어 이전 단계에 따라 다음과 같이 Nova 구성을 편집합니다.

```
boot_volume_type=type-remote
boot_volume_az=cinder_az
```

6. **컴퓨트 노드에서 Nova 서비스를 다시 시작합니다.**

```
compute-node# svcadm restart nova-compute
```

Cinder NFS 드라이버 사용

NFS용 Cinder 드라이버는 Oracle Solaris에서 지원됩니다. 이 드라이버는 Cinder 프리미티브 및 API를 기본 백엔드 저장소에 매핑합니다. 특히 이 드라이버는 NFS를 백엔드 저장소로 제공합니다.

Cinder 드라이버는 프로비저닝 및 저장 장치에 국한된 기타 관리 조작을 담당합니다. 하지만 드라이버 자체는 I/O 작업에 대한 데이터 경로에 있지 않습니다. 드라이버는 블록 레벨에 있는 저장 장치에 액세스하기 위한 인스턴스를 허용하지 않습니다. 대신, 파일이 NFS 공유에서 생성되고 인스턴스에 매핑됩니다. 각 NFS 파일은 블록 장치처럼 작동합니다.

주 - 현재 이 드라이버는 커널 영역만 지원합니다. 이 드라이버를 비전역 영역에서 사용하지 마십시오.

▼ Cinder NFS 드라이버를 사용하는 방법

시작하기 전에 드라이버를 사용하려면 클라이언트에 대한 NFS 공유를 만들기 위한 기존 NFS 서버가 있어야 합니다. NFS 서버 구성에 대한 내용은 이 문서에서 다루지 않습니다. 서버를 설정하려면 기타 NFS 설명서를 참조하십시오.

일반적으로 NFS 서버 하나면 충분합니다. 하지만 필요에 따라 NFS 서버 여러 개를 사용할 수 있습니다.

1. **사용 가능한 NFS 공유를 `/etc/cinder/nfs_shares` 파일에 추가합니다.**
   NFS 공유를 *host:share* 형식으로 나열합니다. 예를 들어 다음을 실행합니다.

   ```
   nfs-server-system1:/scratch/volume1
   nfs-server-system2:/scratch/volume2
   ```

2. **`/etc/cinder/cinder.conf` 파일을 편집합니다.**

   a. **사용할 NFS 드라이버를 지정합니다.**

      ```
      volume_driver=cinder.volume.drivers.solaris.nfs.ZfsNfsVolumeDriver
      ```

   b. **`nfs_shares_config` 매개변수가 1단계에서 사용된 파일을 지정하는지 확인합니다.**

      ```
      nfs_shares_config=/etc/cinder/nfs_shares
      ```

   c. **필요한 경우 기타 NFS 관련 매개변수를 구성합니다.**

      - `nfs_mount_attempts` - 오류를 생성하기 전에 NFS 공유 마운트를 시도하는 최대 횟수입니다.
      - `nfs_mount_point_base` - NFS 공유 마운트 지점의 기본 디렉토리입니다.
      - `nfs_oversub_ratio` - 볼륨 대상에서 할당된 볼륨 공간과 사용 가능한 공간 사이의 최대 비율 제한입니다. 비율을 초과하면 볼륨 대상이 잘못된 상태가 됩니다.
      - `nfs_sparsed_volumes` - 기본값인 True를 사용하면 볼륨이 스파스 파일로 생성됩니다. 그렇지 않은 경우 볼륨이 일반 파일로 생성됩니다.
      - `nfs_used_ratio` - 볼륨 대상에 대한 새 볼륨을 더 이상 할당할 수 없는 기본 볼륨의 실제 사용 비율입니다.
      - `nfs_round_robin` - 기본값 True를 사용하면 NFS 공유에 걸쳐 라운드 로빈을 예약합니다. 이 매개변수가 설정되지 않은 경우 가장 많은 사용 가능 공간이 있는 NFS 공유가 볼륨 배치에 선택됩니다.

3. **Cinder 서비스를 다시 시작합니다.**

   ```
   # svcadm cinder-volume restart
   ```

   서비스가 시작되면 NFS 공유에 대한 디렉토리가 NFS 공유 마운트 지점의 기본 디렉토리에 추가됩니다.

4. **볼륨을 만듭니다.**

   주 - Cinder NFS 드라이버를 사용하는 경우 스냅샷 만들기가 지원되지 않습니다.

   a. **필요한 셸 변수를 정의합니다.**

예를 들어 다음을 실행합니다.

```
# export OS_USERNAME=nova
# export OS_PASSWORD=service-password
# export OS_PROJECT_NAME=service
# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0
```

b. **볼륨을 만듭니다.**

예를 들어 다음을 실행합니다.

```
# nova volume-create --display-name nfsvol 5
```

c. **(옵션) 볼륨의 등록 정보를 표시합니다.**

예를 들어 다음을 실행합니다.

```
# nova volume-show nfsvol
```

Oracle ZFS Storage Appliance와 함께 OpenStack 사용

이 절에서는 Oracle ZFS Storage Appliance(ZFSSA)를 백엔드 저장소로 사용하는 OpenStack 구성을 설명합니다.

Oracle ZFS Storage Appliance 정보

Oracle ZFS Storage Appliance 제품군은 네트워크 클라이언트에 효율적인 파일 및 블록 데이터 서비스를 제공할 뿐 아니라 다음 기술을 비롯하여 시스템에 저장된 데이터에 적용할 수 있는 데이터 서비스 세트를 제공합니다.

- Analytics - 시스템 동작을 동적으로 실시간 관찰하고 데이터를 그래픽으로 볼 수 있습니다.
- ZFS 하이브리드 저장소 풀 - 선택적 플래시 메모리 장치, 저전원 고용량 디스크 및 단일 데이터 계층인 DRAM 메모리와 같은 다양한 장치를 관리할 수 있습니다.
- 다양한 하드웨어 지원

Oracle Solaris용 Oracle OpenStack의 `cloud/openstack/cinder` 패키지에는 Oracle ZFSSA iSCSI Cinder 드라이버가 포함됩니다. 드라이버를 통해 어플라이언스를 Cinder 구성요소에 대한 블록 저장소 리소스로 완벽하게 사용할 수 있습니다. 특히 드라이버를 통해 Cinder 서버가 Nova 서비스에 의해 인스턴스화된 가상 시스템에 할당할 수 있는 iSCSI 볼륨을 만들 수 있습니다. 저장소에 대해 Oracle ZFSSA를 사용하려면 먼저 어플라이언스가 ZFSSA 소프트웨어 릴리스 버전 2013.1.2.0 이상을 실행 중인지 확인합니다.

Oracle ZFSSA로 OpenStack 구성

이 절에서는 Oracle ZFSSA 설정 절차를 자세히 다루지 않습니다. 구성 단계를 포함하여 어플라이언스에 대한 자세한 내용은 다음 소스를 참조하십시오.

- Oracle ZFSSA 제품 설명서(Oracle 도움말 센터 저장소 설명서 페이지)
- Using the Oracle ZFS Storage Appliance as Storage Back End for OpenStack Cinder (`http://www.oracle.com/technetwork/server-storage/sun-unified-storage/documentation/openstack-cinder-zfssa-120915-2813178.pdf`). 이 백서에서는 클라우드에서 사용을 위한 저장소 어플라이언스 설정 방법에 대해 자세히 다룹니다.

Oracle ZFSSA에서 OpenStack을 설정하기 위한 절차를 시작하기 전에 데이터 저장소 트래픽을 호스트하는 서브넷을 이미 만들었는지 확인하십시오. 이 절의 시나리오에서는 그림 1을 이 시작점에 대한 기반으로 사용하지만 약간 다릅니다. 여기에서는 저장소 데이터 트래픽이 해당 트래픽에 대한 인터페이스로 `net2` 및 `net3`의 두 서브넷에 걸쳐 호스트됩니다. 두 서브넷은 데이터 저장소 서비스의 효율적인 성능과 가용성을 보장합니다.

ZFSSA 측

Oracle ZFSSA는 네트워크를 통한 SCSI 명령 교환을 위해 iSCSI를 사용합니다. iSCSI 노드간 통신을 설정하려면 다음 정보를 알고 있어야 합니다.

- SCSI 노드 DNS 이름 또는 IP 주소. SCSI 노드는 개시자 및 대상 노드로 구성됩니다.
- 대상 노드의 TCP iSCSI 인터넷 포트. 기본적으로 포트 번호는 3260입니다.
- 개시자 및 대상 노드의 IQN(iSCSI Qualified Name).
- CHAP(Challenge Handshake Authentication Protocol) 인증을 사용한 선택적 인증 정보

이 준비 단계에 대한 관련 iSCSI 정보를 얻으려면 Oracle Solaris `iscsiadm` 명령을 사용합니다.

이 문서에서는 어플라이언스 구성 단계를 설명하지 않습니다. 작동 및 사용을 위해 어플라이언스를 준비하는 절차는 Oracle의 저장소 설명서에서 해당하는 릴리스 설명서를 참조하십시오.

OpenStack 측

OpenStack ZFSSA Cinder 드라이버는 어플라이언스에서 관련 LUN을 만들고, 이러한 LUN의 알맞은 등록 정보를 설정하며, 볼륨이 사용되는 OpenStack 컴퓨트 노드 및 게스트 인스턴스에 LUN의 표시 여부를 관리함으로써 볼륨을 만들고 관리합니다.

Oracle ZFSSA를 설정한 후에 ZFSSA iSCSI Cinder 드라이버를 구성할 수 있습니다.

▼ OpenStack에 대한 Oracle ZFSSA를 구성하는 방법

이 절차에서는 다음 작업을 수행하는 `cinder.akwf` 워크플로우를 사용합니다.

- 사용자가 없을 경우 사용자를 만듭니다.
- Cinder 드라이버 작업을 수행할 수 있는 역할 권한 부여를 설정합니다.
- 현재 서비스가 사용 안함으로 설정된 경우 RESTful 서비스를 사용으로 설정합니다.

시작하기 전에 Oracle ZFS Storage Appliance에서 풀을 구성하십시오. 기존 풀을 사용하도록 선택할 수 있습니다.

1. **다음 방법 중 하나를 사용하여 `cinder.akwf` 워크플로우를 실행합니다.**

   - **CLI에서 워크플로우를 실행합니다.**

```
zfssa:maintenance workflows> download
zfssa:maintenance workflows download (uncommitted)> show
Properties:
                          url = (unset)
                         user = (unset)
                     password = (unset)

zfssa:maintenance workflows download (uncommitted)> set url= "url-to-cinder.akwf-file"
                          url = "url-to-cinder.akwf-file"
zfssa:maintenance workflows download (uncommitted)> commit
Transferred 2.64K of 2.64K (100%) ... done

zfssa:maintenance workflows> ls
Properties:
                   showhidden = false

Workflows:

WORKFLOW     NAME                                          OWNER SETID ORIGIN
 VERSION
workflow-000 Clear locks                                   root  false Oracle Corporation
 1.0.0
workflow-001 Configuration for OpenStack Cinder Driver root  false Oracle Corporation
 1.0.0

zfssa:maintenance workflows> select workflow-001

zfssa:maintenance workflow-001> execute
zfssa:maintenance workflow-001 execute (uncommitted)>

zfssa:maintenance workflow-001 execute (uncommitted)> set name=user
                         name = user
zfssa:maintenance workflow-001 execute (uncommitted)> set password=password
                     password = password
zfssa:maintenance workflow-001 execute (uncommitted)> commit
```

```
User openstack created.
```

user 및 *password*의 경우 값은 `cinder.conf` 파일의 `san_login` 및 `san_password` 매개변수에 대해 정의된 값입니다.

- **BUI에서 워크플로우를 실행합니다.**

  a. **Maintenance(유지 관리) -> Workflows(워크플로우)를 선택하고 더하기 아이콘을 사용하여 새 워크플로우를 업로드합니다.**

  b. **Browse(찾아보기) 버튼을 누르고 `cinder.akwf` 파일을 선택합니다.**

  c. **UPLOAD(업로드) 버튼을 눌러 워크플로우 업로드를 완료합니다.**

  d. **BUI Workflows(BUI 워크플로우) 페이지에 나타나는 새 행을 눌러 Cinder 드라이버 워크플로우를 실행합니다.**

  워크플로우에서 사용자 이름 및 암호를 입력하라는 메시지를 표시합니다. 이 사용자 이름과 암호는 `cinder.conf` 파일에서 `san_login` 및 `san_password`로도 사용됩니다.

2. **`/etc/cinder/cinder.conf` 파일에서 매개변수를 설정합니다.**

`cinder.conf` 파일에서 다음 필수 등록 정보를 지정합니다.

주 - 다음은 부분 목록입니다. 특정 설정 작업을 수행하는 데 필요한 구성 파일의 모든 등록 정보를 검토하고 설정했는지 확인합니다.

- `volume_driver` - `cinder.volume.drivers.zfssa.zfssaiscsi.ZFSSAISCSIDriver`의 주석 처리가 해제되었는지 확인합니다. 다른 세 가지 선택 항목이 주석 처리되었는지 확인합니다.
- `san_ip` - ZFSSA 관리 호스트의 이름 또는 IP 주소입니다.
- `san_login` - ZFSSA에서 Cinder 사용자의 사용자 이름입니다.
- `san_password` - ZFSSA에서 Cinder 사용자의 암호입니다.
- `zfssa_pool` - 볼륨 할당에 사용할 풀입니다.
- `zfssa_target_portal` - ZFSSA iSCSI 대상 포털(*data-IP:port* 형식)입니다. 기본 포트는 3260입니다.
- `zfssa_project` - ZFSSA 프로젝트의 이름입니다. 어플라이언스에 프로젝트가 없을 경우 시작 시 드라이버가 해당 이름으로 프로젝트를 만듭니다. 이 프로젝트에는 드라이버가 만든 모든 볼륨이 포함됩니다. 볼륨 특성(예: 블록 크기) 및 액세스(예: 개시자, 대상, 보안) 설정을 위해 추가 ZFSSA 등록 정보가 제공됩니다.
- `zfssa_initiator_config` - 여러 개시자 또는 여러 개시자 그룹을 나열하는 등록 정보입니다. 이 등록 정보는 OpenStack Kilo에서 더 이상 사용되지 않는 이전 `zfssa_initiator_group` 매개변수를 대체합니다.

  여러 개시자를 나열하려면 다음 형식을 사용합니다.

```
zfssa_initiator_config = {
    'init-grp1': [
                {'iqn':'iqn1' , 'user':'user' , 'password':'password'},
                {'iqn':'iqn2' , 'user':'user' , 'password':'password'}
                ],
    'init-grp2': [
                {'iqn':'iqn3' , 'user':'user' , 'password':'password'}
                ] }
```

이 등록 정보에 대한 개시자를 나열하는 방법을 보여 주는 특정 예제는 예 6. "zfssa_initiator_config 드라이버 등록 정보 사용"을 참조하십시오.

- `zfssa_target_interfaces` - ZFSSA iSCSI 대상 네트워크 인터페이스입니다. 다음 명령을 사용하여 인터페이스를 확인할 수 있습니다.

```
zfssa:configuration net interfaces> show
Interfaces:

  INTERFACE STATE CLASS LINKS    ADDRS          LABEL
  e1000g0   up    ip    e1000g0  1.10.20.30/24  Untitled Interface
```

- `connection` - 매개변수를 다음과 같이 설정합니다.

```
connection=mysql://cinder:service-password@controller-fqdn/cinder
```

3. **ZFSSA iSCSI 서비스가 온라인 상태인지 확인합니다.**

   ZFSSA iSCSI 서비스가 온라인 상태가 아니면 어플라이언스에서 BUI 또는 CLI를 통해 사용으로 설정합니다. 다음 예에서는 어플라이언스에서 CLI를 사용하는 방법을 보여줍니다.

```
zfssa:> configuration services iscsi
zfssa:configuration services iscsi> enable
zfssa:configuration services iscsi> show
Properties:
<status> = online
...
```

4. **Cinder 볼륨 SMF 서비스를 사용으로 설정합니다.**

```
controller# svcadm enable cinder-volume:default cinder-volume:setup
```

예 6 `zfssa_initiator_config` 드라이버 등록 정보 사용

이 예제는 Cinder 구성 파일에서 `zfssa_imitator_config` 등록 정보에 대한 여러 개시자를 나열하는 방법을 보여 줍니다.

예제에서 그룹 A 및 그룹 B인 개시자 그룹 두 개는 ZFS 저장소 어플라이언스에 생성됩니다. `/etc/cinder/cinder.conf` 파일에 이와 같은 개시자가 다음과 같이 나열됩니다.

```
zfssa_initiator_config = {
        'GroupA':[
```

```
            {'iqn':'iqn.1986-03.com.sun:01:0a43b9fdcfd5.570d7fd1', 'user':'test1',
'password':'password1234'},
            {'iqn':'iqn.1986-03.com.sun:01:0a43b9fdcfd5.570d7fd2', 'user':'', 'password':''}
        ],
        'GroupB':[
            {'iqn':'iqn.1986-03.com.sun:01:0a43b9fdcfd5.570d7fd3', 'user':'', 'password':''}
        ] }
```

♦ ♦ ♦ 7 장

Neutron 배치에 대한 옵션

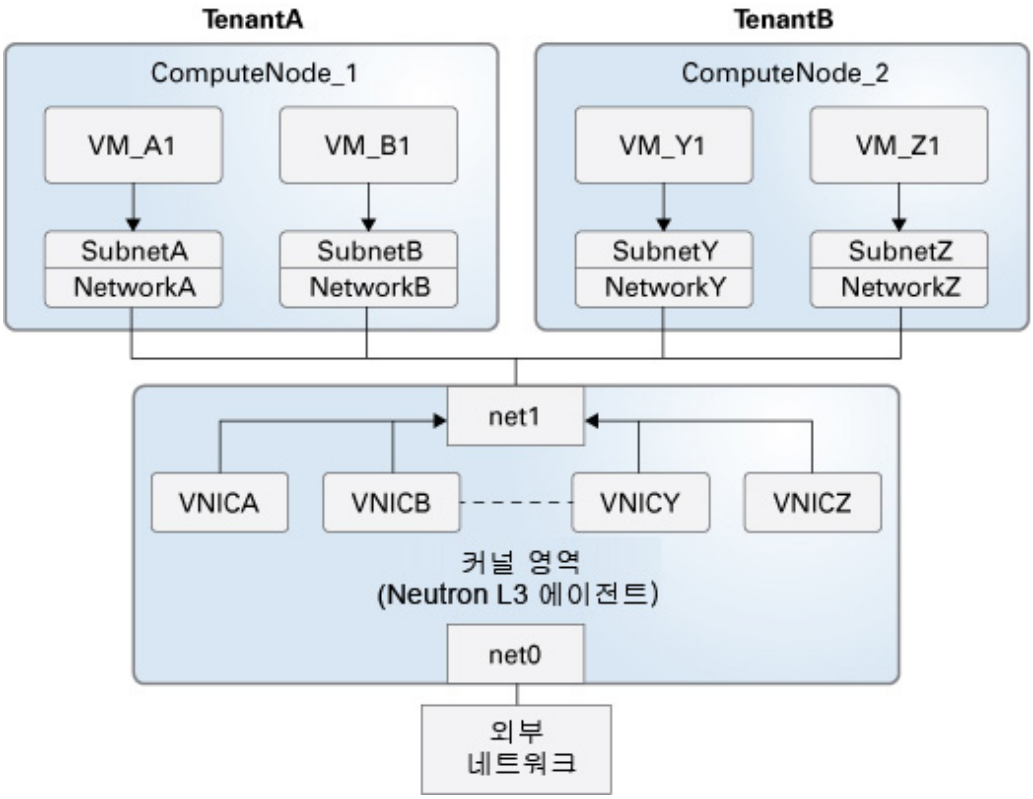
이 설명서에 사용된 3노드 OpenStack 구성 모델에서 Neutron 구성요소는 단일 시스템 그림 1의 컨트롤러 노드와 함께 설치됩니다. 이 장에서는 시스템의 다른 핵심 구성요소와 격리하여 커널 영역 내에 Neutron 구성요소를 설치할 수 있는 방법에 대해 설명합니다. 여기에서는 다음 내용을 다룹니다.

커널 영역에 Neutron 배치

이전 Oracle Solaris 버전에서는 주소를 동적으로 지정할 수 없기 때문에 Neutron이 커널 영역에 설치되지 않았습니다. 새로 도입된 영역 리소스 등록 정보는 이러한 제한을 해결했습니다.

다음 그림은 Neutron의 커널 영역 배치를 보여 줍니다.

그림 7 커널 영역에 Neutron 배치



그림에서 VM 인스턴스가 VM_A1, VM_B1 등과 같은 컴퓨트 노드에 생성되었기 때문에 커널 영역의 L3 에이전트는 각각의 네트워크에 해당 VNIC를 구성합니다. 클라우드 관리자가 VM 인스턴스의 네트워크를 실행하므로 에이전트는 동적 주소 및 VID를 사용하여 이를 자동으로 관리할 수 있습니다.

동적 MAC 주소 및 VID에 대한 지원은 영역 리소스 등록 정보 두 개를 설정하여 사용으로 설정됩니다.

- `allowed-mac-address`
- `allowed-vlan-ids`

주 - 이러한 등록 정보는 `solaris-kz` 브랜드에서만 사용할 수 있습니다.

▼ 커널 영역에 Neutron 구성요소를 설치하는 방법

기타 OpenStack 구성요소와 전역 영역을 공유하는 대신 Neutron 구성요소를 격리된 커널 영역에 배치하려는 경우 이 절차를 사용합니다.

이 절차의 단계에서는 Neutron 관련 구성에만 초점을 맞춥니다. 커널 영역 구성에 대한 자세한 지침은 해당 영역 설명서를 참조하십시오.

1. **커널 영역을 만들기 위한 단계를 완료합니다.**

   전체 지침은 커널 영역을 구성하는 방법에 대한 *Oracle Solaris* 커널 영역 만들기 및 사용을 참조하십시오. 이 설명서는 해당 Oracle Solaris 버전의 라이브러리(Operating Systems Documentation)에 있습니다.

2. **커널 영역에서 L3 에이전트가 VNIC에 동적으로 지정할 수 있는 MAC 주소 접두어 목록을 지정합니다.**

   접두어 길이는 fa:16:3f or fa:80:20:21:22와 같이 1~5 옥테트여야 합니다.

   각 접두어에 대해 새 `add` 명령을 실행합니다. 예를 들면 다음과 같습니다.

   ```
   # zonecfg -z kernel-zone
   zonecfg:kernel-zone> add anet
   zonecfg:kernel-zone:anet> add mac
   zonecfg:kernel-zone:anet:mac> add allowed-mac-address prefix
   zonecfg:kernel-zone:anet:mac> add allowed-mac-address prefix
   ...
   zonecfg:kernel-zone:anet:mac> end
   zonecfg:kernel-zone:anet> end
   zonecfg:kernel-zone>
   ```

3. **커널 영역에서 L3 에이전트가 VNIC에 동적으로 지정할 수 있는 VLAN ID의 범위를 정의합니다.**

   허용된 각 VLAN ID 범위에 대해 새 `add` 명령을 실행합니다. 예를 들면 다음과 같습니다.

   ```
   # zonecfg -z kernel-zone
   zonecfg:kernel-zone> add anet
   zonecfg:kernel-zone:anet> add vlan
   zonecfg:kernel-zone:anet:vlan> add allowed-vlan-ids id-range
   zonecfg:kernel-zone:anet:vlan> add allowed-vlan-ids id-range
   ...
   zonecfg:kernel-zone:anet:vlan> end
   zonecfg:kernel-zone:anet> end
   zonecfg:kernel-zone>
   ```

   범위를 제공하는 대신 `allowed-vlan-ids` 등록 정보에 대한 키워드 `any`를 지정할 수도 있습니다. 그러면 L3 에이전트가 유효한 VLAN ID를 에이전트가 만드는 VNIC에 지정합니다.

4. **커널 영역에서 Neutron의 설치 및 구성에 대한 단계를 완료합니다.**

   지침은 Neutron을 설치 및 구성하는 방법 [38]을 참조하십시오.

MAC 주소 및 VID 정보 표시

여러 명령은 다양한 MAC 주소 및 VID 정보를 표시합니다. 사용할 수 있는 명령은 게스트 VM에 있는지 또는 호스트에 있는지에 따라 달라질 수도 있습니다.

게스트 VM 내에서 표시

VM 인스턴스 내에서 `dladm show-phys` 명령을 사용하여 VM 사용에 제공되는 MAC 주소 및 VID의 범위를 표시할 수 있습니다. 이러한 등록 정보를 표시하려면 출력에 표시하려는 열과 함께 `-o` 옵션을 사용해야 합니다. 열 이름 `ALLOWED-ADDRESSES` 및 `ALLOWED-VIDS`는 MAC 주소 및 VID 범위를 표시합니다. 예를 들면 다음과 같습니다.

```
VM-instance# dladm show-phys -o link,media,device,allowed-addresses,allowed-vids
LINK    MEDIA       DEVICE   ALLOWED-ADDRESSES   ALLOWED-VIDS
net0    Ethernet    zvnet0   fa:16:3f,           100-199,
                             fa:80:20:21:22      400-498,500
```

호스트에서 표시

VM 인스턴스 외부에 있는 경우 `zonecfg info` 명령 또는 `zonecfg export` 명령을 사용하여 MAC 주소 및 VLAN ID 범위를 표시할 수 있습니다. 선택적으로 두 명령과 함께 `-r` 옵션을 사용할 수 있습니다.

다음 예는 명령으로 생성되는 더 완벽한 출력을 추출한 내용을 보여 줍니다.

- `zonecfg info` 또는 `zonecfg -r info`

  ```
  global-zone# zonecfg -z kernel-zone -r info
  anet:
      ...
        mac:
            ...
            allowed-mac-address: fa:16:3f
            allowed-mac-address: fa:80:20:21:22
            ...
       vlan:
            ...
            allowed-vlan-ids: 100-199
            allowed-vlan-ids: 400-498
            allowed-vlan-ids: 500
            ...
  ```

- `zonecfg export` 또는 `zone -r export`

```
global-zone# zonecfg -z kernel-zone -r export
   add anet
   ...
   add mac
   add allowed-mac-address: fa:16:3f
   add allowed-mac-address: fa:80:20:21:22
   ...
   end
   add vlan:
   add allowed-vlan-ids: 100-199
   add allowed-vlan-ids: 400-498
   add allowed-vlan-ids: 500
   end
```

영역 명령은 사용 가능한 MAC 주소 또는 VID의 범위를 표시합니다.

실제로 사용되는 주소 및 VID를 확인하려면 `dladm show-vnic -m` 명령을 실행합니다. 다음 예에서 사용되는 실제 주소 및 ID 정보는 `zonecfg` 명령의 이전 샘플 출력을 기반으로 합니다.

```
global-zone# dladm show-vnic -m
LINK            OVER     SPEED  MACADDRESSES      MACADDRTYPES IDS
kz1/net0        net0     1000   2:8:20:31:ab:46   random       VID:0,100-109
                                2:8:20:ad:29:e8   random
                                fa:80:20:21:22:00 random
                                fa:80:20:21:22:ff random
                                fa:16:3f:0:0:1    random
                                fa:16:3f:0:0:2    random
```

출력에는 에이전트가 VNIC 4개를 만들었다고 표시되어 있습니다. VNIC 두 개는 fa:80:20:21:22 범위의 주소를 사용하고 나머지 두 개는 fa:16:3f 범위의 주소를 사용합니다. 게스트 VM 4개가 현재 컴퓨트 노드에 존재한다고 출력을 추론할 수도 있습니다. VM은 모두 VID 10개를 사용합니다.

`zonecfg` 및 `dladm` 명령에 대한 자세한 내용은 `zonecfg`(8) 및 `dladm`(8) 매뉴얼 페이지를 참조하십시오.

♦ ♦ ♦ 8 장

8 Ironic 작업

이 장에서는 Oracle Solaris에서 구현 및 지원되는 Ironic에 대해 설명합니다. 여기에서는 다음 내용을 다룹니다.

Ironic 구성요소 정보

이전 장에서 클라우드를 만드는 OpenStack 핵심 구성요소를 설명했습니다. 추가 구성요소는 클라우드 관리와 관련된 다른 서비스를 제공합니다. 이 장에서는 Kilo 릴리스에서 제공하는 Ironic 구성요소에 대해 설명합니다.

OpenStack 핵심 구성요소가 가상 시스템이나 VM 인스턴스를 프로비전하는 반면, Ironic은 베어 메탈 인스턴스나 노드를 등록, 프로비전, 폐기하는 서비스를 제공합니다. Ironic은 PXE 부트나 IPMI와 같은 공통 기술을 사용하여 프로비전할 수 있는 광범위한 하드웨어를 지원합니다. 또한 Ironic은 플러그형 드라이버 방식으로 공급업체별 하드웨어를 관리하고 지원할 수 있습니다.

Ironic에 대한 추가 정보와 제공 이점은 OpenStack 커뮤니티 웹 사이트에서 Ironic 개발자 설명서를 참조하십시오.

Ironic은 세 가지 기본 구성요소로 이루어집니다. Oracle Solaris에서 이 구성요소는 SMF 서비스로 제공됩니다. 다음 표는 이 구성요소를 나열하고 설명합니다.

구성요소	설명	SMF 서비스
OpenStack Ironic API 서비스	운영자와 다른 서비스가 관리 베어 메탈 노드와 상호 작용할 수 있는 RESTful API를 제공하는 서비스입니다.	`svc:/application/openstack/ironic/ironic-api`
OpenStack Ironic 공급자 서비스	참조 및 공급업체별 드라이버를 사용하여 베어 메탈 노드를 실제 프로비전하는 주 컨트롤러입니다. 공급자 서비스와 API 서비스는 RPC를 사용하여 통신합니다.	`svc:/application/openstack/ironic/ironic-conductor`

구성요소	설명	SMF 서비스
OpenStack Ironic 데이터베이스 서비스	Ironic 백엔드 데이터베이스를 만들고 동기화하기 위한 일시적 SMF 서비스입니다.	`svc:/application/openstack/ironic/ironic-db`

Ironic 설치 및 구성

Ironic은 다른 OpenStack 서비스 없이 독립형 구성요소로 사용할 수 있습니다. 또는 일반적으로 컴퓨트 노드에서 다른 OpenStack 구성요소와 함께 배치할 수 있습니다. 구성은 Ironic 배치 방법에 따라 달라집니다.

Oracle Solaris 베어 메탈 인스턴스를 프로비전하려면 자동 설치 프로그램(AI)이 필요합니다. AI는 Ironic 서비스와 동일한 노드에 위치할 수 있습니다. 또는 원격 서버에 AI를 사용하여 Ironic과 작동할 수도 있습니다.

AI 구성 및 사용에 대한 자세한 내용은 Operating Systems Documentation에서 사용 중인 Oracle Solaris 버전 라이브러리의 해당 설치 설명서를 참조하십시오.

▼ Ironic을 설치 및 구성하는 방법

시작하기 전에 Ironic을 나머지 OpenStack 구성요소와 함께 배치하는 경우 먼저 핵심 구성요소를 구성해야 합니다. 최소한, Ironic을 작동하기 전에 Keystone 및 Glance 구성을 완료해야 합니다.

마찬가지로, Ironic의 데이터베이스가 설정되었는지 확인합니다. 이 절차에서는 그림 1의 3 노드 참조 아키텍처를 따르며, 이 데이터베이스는 나머지 OpenStack 데이터베이스와 함께 컨트롤러 노드에 설정됩니다.

1. **Ironic 데이터베이스를 만듭니다.**

   a. **컨트롤러 관리자 노드에 대한 전역 셸 변수를 설정합니다.**

   ```
   controller# export CONTROLLER_ADMIN_NODE=controller-node
   ```

   여기서 *controller-node*는 컨트롤러의 IP 주소 또는 호스트 이름입니다. 변수 설정에 대한 자세한 내용은 "호스트 이름, 변수 및 암호 준비" [26]를 참조하십시오.

   b. **다음 명령을 사용하여 데이터베이스를 만듭니다.**

   ```
   controller# mysql -u root -p
   Enter password: MySQL-root-password
   mysql> create database ironic default character set utf8 default collate
    utf8_general_ci;
   mysql> grant all privileges on ironic.* to 'ironic'@'$CONTROLLER_ADMIN_NODE'
    identified by 'service-password';
   mysql> flush privileges;
   ```

```
mysql> quit
```

2. **OpenStack Ironic 패키지를 설치합니다.**

```
# pkg install ironic ironicclient rabbitmq
# pkg update stevedore stevedore-27
```

3. **AI 서버와 관리 도구를 설치합니다.**

   동일한 명령을 사용하여 AI 서버 패키지를 Ironic과 함께 로컬에 설치할지, 다른 호스트에 원격으로 설치할지 결정합니다.

```
# pkg install pkg:/install/installadm
```

4. **AI가 원격에 위치한 경우 해당 서버에 Ironic 사용자를 구성합니다.**

```
remote-AI# useradd -d /var/lib/ironic -m -g 88 -u 88 \
   -P "Install Service Management" ironic
remote-AI# passwd ironic
New Password: password
Re-enter new Password: password
```

주 - AI가 Ironic과 함께 로컬에 설치된 경우 Ironic 설치 프로세스에서 자동으로 시스템에 `ironic` 사용자를 만듭니다.

---

5. **Ironic 사용자의 SSH 키를 만들고 관리합니다.**

   - **AI가 Ironic과 함께 로컬에 있는 경우 해당 시스템에서 다음 명령을 실행합니다.**

```
ironic-localhost# su - ironic
ironic-localhost# mkdir /var/lib/ironic/.ssh
ironic-localhost# ssh-keygen -N '' -t rsa \
   -f /var/lib/ironic/.ssh/id_rsa
ironic-localhost# cat /var/lib/ironic/.ssh/id_rsa.pub > \
   /var/lib/ironic/.ssh/authorized_keys
```

   - **AI가 원격에 위치한 경우 다음 단계를 수행합니다.**

     a. **원격 AI에서 다음 명령을 실행합니다.**

```
remote-AI# su - ironic
remote-AI# mkdir /var/lib/ironic/.ssh
remote-AI# ssh-keygen -N '' -t rsa \
   -f /var/lib/ironic/.ssh/id_rsa
remote-AI# cat /var/lib/ironic/.ssh/id_rsa.pub > \
   /var/lib/ironic/.ssh/authorized_keys
```

     b. **Ironic 호스트에서 다음 명령을 실행합니다.**

```
ironic-localhost# mkdir /var/lib/ironic/.ssh
```

```
ironic-localhost# scp ironic@AI-server:~/.ssh/id_rsa /var/lib/ironic/.ssh
ironic-localhost# scp ironic@AI-server:~/.ssh/id_rsa.pub /var/lib/ironic/.ssh
ironic-localhost# cat /var/lib/ironic/.ssh/id_rsa.pub > \
   /var/lib/ironic/.ssh/authorized_keys
ironic-localhost# chown -R ironic:ironic /var/lib/ironic/.ssh
```

여기서 *AI-server*는 AI 서버의 IP 주소 또는 호스트 이름일 수 있습니다.

6. **`/etc/ironic/ironic.conf` 파일을 편집합니다.**

   파일에서 특정 매개변수의 구성과 관련하여 다음 사항을 고려하십시오.

   `auth_strategy` — Ironic을 독립형 구성요소로 사용하는 경우 `[DEFAULT]` 섹션에서 `noauth`를 지정합니다. Ironic을 다른 OpenStack 구성요소와 함께 배치하는 경우 `keystone`을 지정합니다.

   `server` — 자동 설치 프로그램(AI)이 Ironic과 함께 로컬에 있는 경우 `[ai]` 섹션에서 `localhost`를 지정합니다. AI가 원격에 있는 경우 해당 서버의 IP 주소 또는 호스트 이름을 지정합니다.

   `connection` — `[database]` 섹션에서 MySQL 데이터베이스에 대한 연결을 지정합니다.

   `glance_host` — `auth_strategy` 매개변수가 `keystone`으로 설정된 경우 `glance` 섹션에서 Glance 서버의 IP 주소 또는 호스트 이름을 지정합니다.

   `glance_api_servers` — 또한 `glance` 섹션에서 호스트 이름 또는 IP 주소와 함께 해당 포트 번호를 지정합니다(예: 192.168.0.150:9292). `localhost`를 지정하지 마십시오.

   다음 샘플 파일은 Ironic이 작동하는 데 필요한 모든 매개변수 구성을 안내하는 데 사용됩니다. 샘플 파일에서 *glance-serverIP* 변수는 Glance 서버의 IP 주소 또는 호스트 이름을 나타냅니다.

```
[DEFAULT]
enabled_drivers=solaris
auth_strategy=     설정은 Ironic이 독립형인지 여부에 따라 다릅니다.
pybasedir = /usr/lib/python2.6/vendor-packages/ironic
bindir = /usr/lib/ironic
host = ironic

[ai]
server=     설정은 AI가 로컬 또는 원격인지에 따라 다릅니다.
username=ironic
port=22
timeout=10
ssh_key_file=/var/lib/ironic/.ssh/id_rsa
deploy_interval=30

[api]
port=6385
```

```
[conductor]
api_url=http://localhost:6385/
heartbeat_timeout=60
heartbeat_interval=60
sync_power_state_interval=300
check_provision_state_interval=120

[database]
connection= mysql://ironic:service-password@$CONTROLLER_ADMIN_NODE/ironic

[solaris_ipmi]
imagecache_dirname = /var/lib/ironic/images
imagecache_lock_timeout = 60

[glance]
glance_host = glance-serverIP
glance_port = 9292
glance_protocol = http
glance_api_servers = glance-serverIP:port
auth_strategy =      Ironic이 Keystone을 사용하여 Glance와 상호 작용하도록 하려면 keystone을 지정합니다.

[keystone_authtoken]     DEFAULT에서 auth_strategy = keystone이면 구성합니다.
auth_host = localhost
auth_uri = http://$CONTROLLER_ADMIN_NODE:5000/v2.0
identity_uri = http://$CONTROLLER_ADMIN_NODE:35357
admin_user = ironic
admin_password = service-password
admin_tenant_name = tenant
signing_dir = $state_path/keystone-signing

[neutron]
auth_strategy =      Ironic이 Keystone을 사용하여 Neutron과 상호 작용하도록 하려면 keystone을 지정합니다.

[oslo_messaging_rabbit]
rabbit_host=$CONTROLLER_ADMIN_NODE
```

7. Ironic의 SMF 서비스가 실행 중인지 확인합니다.

```
ironic-localhost# svcs -a | grep rabbitmq
ironic-localhost# svcs -a | grep ironic
```

Ironic이 독립형 모드인 경우 수동으로 서비스를 사용으로 설정해야 할 수도 있습니다.

```
ironic-localhost# svcadm enable rabbitmq
ironic-localhost# svcadm enable ironic-db
ironic-localhost# svcadm enable ironic-api ironic-conductor
```

8. (옵션) 명령줄 유틸리티를 테스트합니다.

a. Ironic에 대한 전역 셸 변수를 설정합니다.

- **auth_strategy가 noauth로 설정된 경우 다음과 같이 셸 변수를 설정합니다.**

```
ironic-localhost# export OS_AUTH_TOKEN=fake-token
ironic-localhost# export IRONIC_URL=http://localhost:6385
```

- **auth_strategy가 keystone으로 설정된 경우 다음과 같이 셸 변수를 설정합니다.**

  Keystone 서비스가 Ironic과 다른 노드에 있는 경우 OS_AUTH_URL에 대해 Keystone이 설치된 IP 주소 또는 호스트 이름을 지정합니다.

```
ironic-localhost# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0
ironic-localhost# export OS_PROJECT_NAME=service
ironic-localhost# export OS_USERNAME=ironic
ironic-localhost# export OS_PASSWORD=service-password
ironic-localhost# export IRONIC_URL=http://localhost:6385/
```

b. **Ironic 명령줄을 실행합니다.**

```
# ironic driver-list
+---------------------+----------------+
| Supported driver(s) | Active host(s) |
+---------------------+----------------+
| solaris             | ironic         |
+---------------------+----------------+
```

현재 solaris가 Ironic에서 유일하게 사용으로 설정되고 테스트된 드라이버입니다. 그러나 /etc/ironic/ironic.conf 파일에서 [enabled_drivers] 섹션 아래에 드라이버 이름을 넣으면 목록에 추가할 수 있습니다. 드라이버를 추가한 후에 ironic-api 및 ironic-conductor SMF 서비스를 다시 시작해야 합니다.

개요: Ironic과 함께 베어 메탈 배치

Ironic solaris 드라이버와 함께 Oracle Solaris UAR(Unified Archive) 기능이나 IPS(이미지 패키징 시스템)를 사용하여 베어 메탈 노드를 프로비전할 수 있습니다. 노드를 만들 때 구성 가능한 노드 요소를 통해 solaris 드라이버로 정보를 전달합니다.

다음 표는 노드를 만들기 위한 요소를 나열합니다. driver_info/archive_uri 요소는 UAR 파일을 사용한 노드 프로비전에 적용됩니다. 표의 나머지 요소는 IPS를 사용한 노드 프로비전에 적용됩니다.

요소	설명	예
driver_info/archive_uri	베어 메탈 프로비전에 사용할 Unified Archive의 URI입니다.	http://host.example.com/sol-11_3-x86.uar

요소	설명	예
`driver_info/ai_service`	사용할 AI 서비스의 이름입니다.	default-x86
`driver_info/publishers`	이름 지정 형식 *publisher-name@origin*을 사용하여 플러스(+) 기호로 구분된 IPS 게시자 목록입니다.	solaris@http://ipkg.us.oracle.com/solaris*N*/dev+userland@http://my.example.repo
`driver_info/fmri`	플러스(+) 기호로 구분된 설치할 PKG FMRI 목록입니다.	pkg:/group/system/solaris-small-server +pkg:/cloud/openstack/nova
`driver_info/install_profiles`	설치 환경에 적용할 구성 프로파일의 URI 목록입니다. URI는 플러스(+) 기호로 구분됩니다.	http://host.example.com/profile1.xml +glance://*glance-image*
`driver_info/sc_profiles`	설치된 시스템에 적용할 시스템 구성 프로파일의 URI 목록입니다. URI는 플러스(+) 기호로 구분됩니다.	http://host.example.com/profile1.xml +glance://*glance-image*
`driver_info/manifest`	베어 메탈 프로비전에 사용할 AI 매니페스트의 URI입니다.	http://host.example.com/my-manifest.xml
`driver_info/ipmi_address`	프로비전할 노드의 ILOM에 연결하기 위한 직렬 콘솔의 IP 주소 또는 호스트 이름입니다.	192.168.2.200
`driver_info/ipmi_username`	IPMI 연결의 사용자 이름입니다.	root
`driver_info/ipmi_password`	IPMI 연결의 암호입니다.	*password*

UAR 파일을 사용하여 노드를 프로비전하는 경우 `driver_info/archive_uri` 정보만 제공해야 합니다. 아카이브 URI는 다음 목록의 스키마 중 하나를 사용할 수 있습니다. IPS를 사용하여 노드를 프로비전하려는 경우 프로파일 지정에 동일한 스키마 옵션이 적용됩니다.

- `file://`
- `http://`
- `https://`
- `glance://`

주 - Glance는 일반적으로 설치 이미지 저장에 사용되지만 저장소에 프로파일을 추가해도 됩니다.

IPS를 사용하여 노드를 프로비전하는 경우, 다음 예제와 같이 노드 아키텍처에 최소한의 기본 AI 서비스가 존재하는지 먼저 확인하십시오.

```
# installadm list
Service Name            Status Arch  Type Alias Aliases Clients Profiles Manifests
------------            ------ ----  ---- ----- ------- ------- -------- ---------
default-i386            on     i386  pkg  yes   0       0       0        1
default-sparc           on     i386  pkg  yes   0       0       0        1
ironic-x86              on     i386  pkg  no    0       0       0        1
ironic-sparc            on     i386  pkg  no    0       0       0        1
```

```
# installadm list -vcn ironic-x86
There are no clients configured for service 'ironic-x86'.

# installadm list -vmn ironic-x86
Service Name       Manifest Name Status  Criteria
------------       ------------- ------  --------
ironic-x86         orig_default  default none
ironic-sparc       orig_default  default none
```

IPS를 사용하여 노드를 프로비전할 때 `driver_info/ai_service` 지정은 선택사항입니다. AI 서비스 이름을 생략하면 해당 노드 아키텍처의 기본 AI 서비스가 사용됩니다.

`driver_info/fmri`에 사용자정의 패키지를 지정할 경우 `driver_info/publishers` 요소에 게시자도 지정해야 합니다.

Ironic을 사용하여 베어 메탈 배치

다음 작업은 노드 만들기부터 실제 배치까지 기본 단계를 제공합니다.

Oracle Solaris의 Unified Archive 기능 사용에 대한 자세한 내용은 *Using Unified Archives for System Recovery and Cloning in Oracle Solaris* 을 참조하십시오. 이 설명서는 Oracle Solaris 버전의 library에 있습니다.

▼ UAR 파일에서 베어 메탈을 배치하는 방법

다음 단계를 사용하는 구체적인 예는 예 7. “UAR 파일을 사용하여 노드 배치”를 참조하십시오.

시작하기 전에 사용할 UAR 파일이 존재하는지 확인합니다.

1. **Ironic 명령줄 유틸리티 사용을 효율화하기 위해 다음 변수를 만듭니다.**
   - `IP` - 노드의 ILOM 연결에 사용되는 IP 주소입니다.
   - `USER` - 일반적으로 사용자는 `root`입니다.
   - `PASS` - `root` 암호입니다.
   - `HOST_MAC` - 시스템의 MAC 주소입니다.
2. **Ironic 노드를 만듭니다.**

   ```
   # ironic node-create options
   ```

   새 노드의 UUID를 포함한 등록 정보가 표시됩니다.
3. **쉬운 참조를 위해 노드의 UUID에 대한 변수를 만듭니다.**

4. **설치할 UAR 위치를 지정하여 노드를 업데이트합니다.**

   ```
   # ironic node-update options
   ```

5. **이 노드에 연관된 포트를 만듭니다.**

   ```
   # ironic port-create options
   ```

6. **(옵션) 노드에 대해 지정된 필드를 검증합니다.**

   ```
   # ironic node-validate options
   ```

7. **노드를 프로비전합니다.**

   ```
   # ironic node-set-provision-state options
   ```

8. **(옵션) 배치 상태를 표시합니다.**

   ```
   # ironic node-show options
   ```

주 - 프로세스 완료 후 프로비전이 진행 중인 동안 명령을 실행할 경우 출력이 달라집니다.

예 7 UAR 파일을 사용하여 노드 배치

이 예에서는 다음 설정을 가정합니다.

- 노드를 호스트하는 시스템의 기본 사항
  - 호스트 이름: `mynewnode.example.com`
  - 아키텍처: x86
  - IP 주소: 1.1.1.1
  - MAC 주소: 01:02:03:04:05:06
- ILOM 호스트의 기본 사항
  - 호스트 이름: `mynewnode-aa.example.com`
  - IP 주소: 2.2.2.2
  - 사용자: root
  - 암호: password
- UAR 파일 이름: `myuar.server/sol11-3-x86.uar`

```
# export ILOM_IP=2.2.2.2
# export ILOM_USER=root
# export ILOM_PASS=password
# export HOST_MAC=01:02:03:04:05:06

# ironic node-create -d solaris -i ipmi_address=$ILOM_IP \
   -i ipmi_username=$ILOM_USER -i ipmi_password=$ILOM_PASS
+--------------+--------------------------------------------------------+     노드가 만들어
졌습니다.
| Property     | Value                                                  |
```

```
+--------------+-------------------------------------------------------+
| uuid         | 4eacbfde-4977-4d8c-8043-8cbe8f876187                  |
| driver_info  | {u'ipmi_address': u'2.2.2.2', u'ipmi_username': u'root', |
|              | u'ipmi_password': u'password'}                        |
| extra        | {}                                                    |
| driver       | solaris                                               |
| chassis_uuid | None                                                  |
| properties   | {}                                                    |
+--------------+-------------------------------------------------------+
```

 # export NODE=4eacbfde-4977-4d8c-8043-8cbe8f876187 *UUID*가 저장되었습니다.

```
# ironic node-update $NODE \
   add driver_info/archive_uri=http://myuar.server/sol11-3-x86.uar
+-----------------------+-------------------------------------------------------------+
| Property              | Value                                                       |
+-----------------------+-------------------------------------------------------------+
| instance_uuid         | None                                                        |
| target_power_state    | None                                                        |
| properties            | {}                                                          |
| maintenance           | False                                                       |
| driver_info           | {u'archive_uri': u'http://myuar.server/sol11-3-x86.uar',    |
|                       | u'ipmi_address': u'2.2.2.2', u'ipmi_username': u'root',     |
|                       | u'ipmi_password': u'password'}                              |
| extra                 | {}                                                          |
| last_error            | None                                                        |
| created_at            | 2014-10-03T15:38:43+00:00                                   |
| target_provision_state | None                                                       |
| driver                | solaris                                                     |
| updated_at            | 2014-10-03T15:53:04+00:00                                   |
| instance_info         |                                                             |
| chassis_uuid          | None                                                        |
| provision_state       | None                                                        |
| reservation           | None                                                        |
| power_state           | None                                                        |
| console_enabled       | False                                                       |
| uuid                  | 4eacbfde-4977-4d8c-8043-8cbe8f876187                        |
+-----------------------+-------------------------------------------------------------+

# ironic port-create -n $NODE -a $HOST_MAC
+-----------+--------------------------------------+
| Property  | Value                                |
+-----------+--------------------------------------+
| node_uuid | 4eacbfde-4977-4d8c-8043-8cbe8f876187 |
| extra     | {}                                   |
| uuid      | 4c765ab0-2529-4463-a51b-e5544dd15a32 |
| address   | 01:02:03:04:05:06                    |
+-----------+--------------------------------------+

# ironic node-validate $NODE
+------------+--------+---------------+
| Interface  | Result | Reason        |
+------------+--------+---------------+
| console    | None   | not supported |
```

```
| deploy     | True   |               |
| management | True   |               |
| power      | True   |               |
+------------+--------+---------------+
```

 # ironic node-set-provision-state $NODE active 노드가 프로비전되었습니다.

```
# ironic node-show $NODE
+-----------------------+------------------------------------------------------------+
| Property              | Value                                                      |
+-----------------------+------------------------------------------------------------+
|instance_uuid          | None                                                       |
|target_power_state     | None                                                       |
|properties             | {}                                                         |
|maintenance            | False                                                      |
|driver_info            | {u'archive_uri': u'http://myuar.server/sol11-3-x86.uar',   |
|                       | u'ipmi_address': u'2.2.2.2', u'ipmi_username': u'root',    |
|                       | u'ipmi_password': u'password'}                             |
|extra                  | {}                                                         |
|last_error             | None                                                       |
|created_at             | 2014-10-03T15:38:43+00:00                                  |
|target_provision_state | deploy_complete                                            |
|driver                 | solaris                                                    |
|updated_at             | 2014-10-03T15:53:04+00:00                                  |
|instance_info          |                                                            |
|chassis_uuid           | None                                                       |
|provision_state        | active                                                     |
|reservation            | None                                                       |
|power_state            | power on                                                   |
|console_enabled        | False                                                      |
|uuid                   | 4eacbfde-4977-4d8c-8043-8cbe8f876187                       |
+-----------------------+------------------------------------------------------------+
```

프로비전이 진행 중인 동안 `ironic node-show` 명령을 실행한 경우 `provision_state`가 `active` 대신 다른 상태로 나타납니다.

▼ 노드를 폐기하는 방법

프로비전 작업을 실패한 경우에도 이 절차를 사용하십시오.

1. **노드를 deleted 상태로 설정합니다.**

   # ironic node-set-provision-state $NODE deleted

2. **노드 정보를 표시합니다.**

   다음 예제에서 별표(*)가 붙은 등록 정보는 deleted 상태의 노드를 나타냅니다.

   ```
   # ironic node-show $NODE
   +-----------------------+------------------------------------------------------------+
   ```

```
| Property               | Value                                                    |
+------------------------+----------------------------------------------------------+
|instance_uuid           | None                                                     |
|target_power_state      | None                                                     |
|properties              | {}                                                       |
|maintenance             | True                                                     |    *
|driver_info             | {u'archive_uri': u'http://myuar.server/sol11-3-x86.uar', |
|                        | u'ipmi_address': u'2.2.2.2', u'ipmi_username': u'root',  |
|                        | u'ipmi_password': u'password'}                           |
|extra                   | {}                                                       |
|last_error              | None                                                     |
|created_at              | 2014-10-03T15:38:43+00:00                                |
|target_provision_state  | None                                                     |    *
|driver                  | solaris                                                  |
|updated_at              | 2014-10-03T15:53:04+00:00                                |
|instance_info           | {}                                                       |
|chassis_uuid            | None                                                     |    *
|provision_state         | None                                                     |    *
|reservation             | None                                                     |
|power_state             | power off                                                |    *
|console_enabled         | False                                                    |
|uuid                    | 4eacbfde-4977-4d8c-8043-8cbe8f876187                     |
+------------------------+----------------------------------------------------------+
```

3. **실패한 프로비전 프로세스의 문제를 해결하려면 노드의 필드를 검증합니다.**

   노드에 대해 지정된 `driver_info` 요소에 문제가 있으면 `Reason` 열 아래에 나타납니다.

```
# ironic node-validate $NODE
+------------+--------+---------------+
| Interface  | Result | Reason        |
+------------+--------+---------------+
| console    | None   | not supported |
| deploy     | True   |               |
| management | True   |               |
| power      | True   |               |
+------------+--------+---------------+
```

♦ ♦ ♦ 9 장

9 Heat 작업

이 장에서는 Oracle Solaris에서 구현 및 지원되는 Heat에 대해 설명합니다. 여기에서는 다음 내용을 다룹니다.

Heat 구성요소 정보

Heat는 직접 만든 Heat 조정 템플리트를 기반으로 클라우드 응용 프로그램을 배치할 수 있는 OpenStack 조정 엔진입니다. 이 템플리트는 HOT 템플리트라고도 합니다.

HOT 템플리트를 사용하면 인스턴스, 유동 IP, 볼륨, 사용자 등과 같은 다양한 OpenStack 리소스 유형을 만들 수 있습니다. 템플리트를 통해 인스턴스 고가용성, 인스턴스 자동 크기 조정 및 중첩된 스택과 같은 고급 기능을 배치할 수도 있습니다. 따라서 Heat를 사용하면 모든 OpenStack 코어 프로젝트가 더 큰 사용자 기반을 수신할 수 있습니다. 템플리트를 사용하면 Heat는 템플리트에서 엔진으로 전달된 매개변수를 대신 사용하여 기본 리소스 구현을 대체할 수 있는 다양한 방법을 제공합니다.

Heat는 RESTful 웹 서비스 API를 통해 해당 서비스를 제공합니다. 모든 OpenStack 응용 프로그램과 마찬가지로 Python WSGI 인터페이스가 사용되며 Paste를 사용하여 응용 프로그램이 함께 구성됩니다. 응용 프로그램의 HTTP 끝점은 WSGI(웹 서버 게이트웨이 인터페이스) 미들웨어의 파이프라인으로 구성됩니다. Heat는 특히 Heat API용 포트 8004 및 Heat Cloud Formation용 포트 8000과 같은 끝점 두 개를 사용합니다.

Heat의 고유 구성은 `/etc/heat/heat.conf` 구성 파일에서 제어합니다. 현재는 기본 Heat 구성 파일에 Solaris 관련 구성 매개변수가 없습니다.

Heat 구성요소에 대한 자세한 내용은 OpenStack 커뮤니티의 Heat documentation을 참조하십시오.

Heat 설치

일반적인 구성에서 Keystone과 동일한 노드에 Heat 서비스를 설치합니다. 다음 명령을 사용하여 노드에 OpenStack을 설치한 경우 Heat 패키지가 자동으로 포함됩니다.

```
# pkg install openstack
```

▼ Heat를 구성하는 방법

시작하기 전에 이 작업을 수행하기 전에 먼저 Keystone을 설치 및 구성하는 방법에 설명된 대로 Keystone을 구성해야 합니다.

1. **/etc/heat/heat.conf에서 매개변수의 주석 처리를 해제하거나 매개변수를 설정하여 Heat를 구성합니다.**

```
[database]
connection = mysql://heat:service-password@$CONTROLLER_ADMIN_NODE/heat

[keystone_authtoken]
auth_uri = http://$CONTROLLER_ADMIN_NODE:5000/v2.0
identity_uri = http://$CONTROLLER_ADMIN_NODE:35357
admin_user = heat
admin_password = service-password
admin_tenant_name = tenant

[oslo_messaging_rabbit]
rabbit_host=$CONTROLLER_ADMIN_NODE
```

2. **Heat 서비스를 사용으로 설정합니다.**

```
controller# svcadm enable -rs heat-api heat-db heat-engine \
heat-api-cfn heat-api-cloudwatch
```

HOT 템플리트 정보

Heat를 사용하여 OpenStack 구성에서 여러 합성 클라우드 응용 프로그램을 조정하려면 HOT(Heat 조정 템플리트)를 정의해야 합니다. HOT 템플리트에는 기입해야 하는 사양이 포함되어 있습니다. 프로세스를 실행하여 리소스 유형 및 기타 고급 기능을 만들 때 사용자가 제공하는 매개변수가 읽힙니다.

HOT 템플리트 사양 및 해당 설명은 http://docs.openstack.org/developer/heat/template_guide/hot_spec.html#hot-spec을 참조하십시오.

HOT 템플리트를 기록하는 방법에 대한 자세한 내용은 http://docs.openstack.org/developer/heat/template_guide/hot_guide.html#hot-guide를 참조하십시오.

HOT 템플리트를 배치하려면 다음 명령을 사용합니다.

```
# heat stack-create -f template stack-name
```

template 처리할 템플리트 파일의 전체 경로

stack-name 만들 스택

-P *key1=value1;key2=value2...* 형식을 사용하여 명령에 추가 매개변수 값을 삽입할 수도 있습니다.

다음 예는 서브넷을 사용하여 개인 네트워크 3개를 만드는 이름이 stack3인 HOT 템플리트 컨텐츠를 보여 줍니다.

주 - heat_template_version: 2013-05-23 행은 템플리트 맨 위에 표시되어야 합니다.

```
heat_template_version: 2013-05-23
description: Create a few networks.

resources:
  heat_net1:
    type:OS::Neutron::Net
    properties:
      name:heat_net1

  heat_subnet1:
    type:OS::Neutron::Subnet
    properties:
      name:heat_subnet1
      network_id: { get_resource: heat_net1 }
      cidr: 192.168.50.0/24

  heat_net2:
    type:OS::Neutron::Net
    properties:
      name: heat_net2

  heat_subnet2:
    type:OS::Neutron::Subnet
    properties:
      name:heat_subnet2
      network_id: { get_resource: heat_net2 }
      cidr: 192.168.51.0/24

  heat_net3:
    type:OS::Neutron::Net
    properties:
      name: heat_net3

  heat_subnet3:
    type:OS::Neutron::Subnet
    properties:
      name:heat_subnet3
```

```
network_id: { get_resource: heat_net3 }
cidr: 192.168.52.0/24
```

Cloudbase-Init가 제공되는 Heat 사용

Cloudbase-Init는 사용자 만들기, 암호 생성, 스크립트 실행 등과 같이 클라우드에서 새 게스트 이미지를 자동으로 초기화하는 타사 서비스입니다. Oracle Solaris OpenStack은 Kilo에서 이 서비스를 지원하지만 UserData 플러그인에 국한됩니다.

Cloudbase-Init 패키지는 표준 그룹 패키지에 포함되지 않습니다. 이미지가 명시적으로 클라우드 환경에 배치되는 시스템에만 이 패키지를 설치해야 합니다.

▼ 게스트 이미지를 자동으로 초기화하는 방법

시작하기 전에 다음을 완료한 상태여야 합니다.

- 이미지를 클라우드에 배치할 시스템의 UA(Unified Archive)를 만들었습니다. UA를 만들기 전에 해당 시스템에 Cloudbase-init 패키지가 포함되어 있는지 확인합니다.

  UA 만들기에 대한 지침은 *Oracle Solaris*의 시스템 복구 및 복제용 *Unified Archive* 사용 설명서의 *Unified Archives* 작업 장을 참조하십시오. 이 설명서는 Operating Systems Documentation의 사용 중인 Oracle Solaris 버전 라이브러리에 있습니다.
- Heat 템플리트의 사용자 데이터 섹션에서 사용자 데이터 정보를 제공합니다. 템플리트를 `*.yaml` 파일로 저장합니다.

1. **Cloudbase-Init 패키지를 포함하는 이미지를 Glance에 업로드합니다.**
2. **다음 명령을 실행합니다.**

```
# heat stack-create -f yaml-template \
  -P key_name=your key name;image=image name\
  private_net=name of tenant private network  stack-name
```

예 8 Cloudbase-Init를 사용하는 Heat 템플리트

다음 예는 Cloudbase-Init로 처리되는 Heat 템플리트를 보여줍니다. 제공해야 하는 정보는 굵게 표시되고 편의를 위해 사용자 데이터 정보가 표시됩니다. 템플리트 이름은 `test.yaml`입니다.

```
heat_template_version: 2013-05-23

description: HOT template to deploy one server into an existing neutron tenant
network and assign a floating IP address so it's routable from the public network.

parameters:
  key_name:Server1Key
```

```
    type: string
    description: Name of keypair to assign to server
  image:Solaris Non-global Zone
    type: string
    description: Name of image to use for server
  flavor:8
    type: string
    description: Flavor to use for server
    default: 1
  public_net:
    type: string
    description:
      Name or ID of public network for which floating IP address will
be allocated
    default: oracle
  private_net:HR
    type: string
    description: Name or ID of private network into which server is
deployed

resources:
  server1:
    type: OS::Nova::Server
    properties:
      name: { get_param: 'OS::stack_name' }
      image: { get_param: image }
      flavor: { get_param: flavor }
      key_name: { get_param: key_name }
      networks:
        - port: { get_resource: server1_port }
      user_data_format: RAW
    ---------- 사용자 데이터 섹션 시작 ----------
      user_data:
        str_replace:
          template: |
            #!/bin/ksh
            print "This is a test."

  server1_port:
    type: OS::Neutron::Port
    properties:
      network: { get_param: private_net }

  server1_floating_ip:
    type: OS::Neutron::FloatingIP
    properties:
      floating_network: { get_param: public_net }
      port_id: { get_resource: server1_port }

outputs:
  server1_private_ip:
    description: IP address of server1 in private network
    value: { get_attr: [ server1, first_address ] }
  server1_public_ip:
```

```
    description: Floating IP address of server1 in public network
    value: { get_attr: [ server1_floating_ip, floating_ip_address ] }
```

템플리트를 배치하려면 다음을 입력합니다.

```
# heat stack-create -f test.yaml -P key_name=Server1Key \
   -P image=Solaris Non-global Zone\
...-P flavor=8
   -P private_net=HR teststack
```

명령의 각 옵션에 대한 특정 값은 `test.yaml` 템플리트 파일의 정보에서 온 것입니다. 공용 네트워크는 `oracle`이며 기본값입니다.

♦ ♦ ♦ 10 장

10 OpenStack 문제 해결

이 장에서는 현재 릴리스와 연관된 OpenStack 문제에 대해 설명합니다. 또한 구성 중 발생할 수 있는 기본 문제를 해결하도록 문제 해결 팁과 임시해결책도 제공합니다. 다음 항목을 다룹니다.

명령줄 도움말 얻기

OpenStack에서 명령은 OpenStack 구성요소와 일치합니다. 예를 들어, `nova` 명령은 컴퓨트 작업에 적용되고, `cinder`는 저장소에 적용되며, `neutron`은 네트워킹에 적용됩니다.

올바른 구문, 지원되는 하위 명령, 가능한 옵션 등과 같이 이러한 명령 사용에 대한 도움말을 얻으려면 *command-component* `help` 명령(예: `nova help` 또는 `neutron help`)을 사용합니다. *command-component* 명령과 함께 사용할 수 있는 하위 명령 목록은 `grep` 명령을 사용하여 필터링할 수 있습니다. 예를 들어, 라우터와 관련된 `neutron` 하위 명령을 나열하려면 다음 명령을 입력합니다.

```
# neutron help | grep router
  l3-agent-list-hosting-router   List L3 agents hosting a router.
  l3-agent-router-add            Add a router to a L3 agent.
  l3-agent-router-remove         Remove a router from a L3 agent.
  net-gateway-connect            Add an internal network interface to a router.
  router-create                  Create a router for a given tenant.
  router-delete                  Delete a given router.
  router-gateway-clear           Remove an external network gateway from a router.
  router-gateway-set             Set the external network gateway for a router.
  router-interface-add           Add an internal network interface to a router.
  router-interface-delete        Remove an internal network interface from a router.
  router-list                    List routers that belong to a given tenant.
  router-list-on-l3-agent        List the routers on a L3 agent.
```

```
  router-port-list              List ports that belong to a given tenant, with specified
router.
  router-show                   Show information of a given router.
  router-update                 Update router's information.
```

그런 다음 클라우드에서 라우터를 식별하는 `router-list`와 같은 하위 명령에 대한 특정 세부 정보를 얻으려면 다음 명령을 입력합니다.

```
# neutron help router-list
```

OpenStackClient(OSC) 구현으로 인해 특정 구성요소 기반 명령이 더 이상 사용되지 않습니다. 대신, 해당 하위 명령과 함께 `openstack`을 기본 명령으로 사용합니다. OSC에 대한 간략한 설명은 "OpenStackClient 구현" [15]을 참조하십시오.

`openstack` 명령 및 하위 명령에 대한 정보를 가져오려면 다음 명령 중 하나를 사용합니다.

- `openstack help` *subcommand*
- `openstack --help`

하위 명령 없이 `openstack`을 입력하면 대화식 모드로 전환됩니다. 여기서 정보를 보려면 `help [`*subcommand*`]`를 입력합니다. 대화식 모드를 종료하려면 `quit`를 입력합니다.

OSC에 대한 자세한 내용은 `http://docs.openstack.org/developer/python-openstackclient/index.html`을 참조하십시오.

이전 명령 및 OSC에서 이에 상응하는 명령 목록은 부록 B. OpenStackClient 명령을 참조하십시오.

알려진 제한 사항

Oracle Solaris의 OpenStack(Kilo)과 관련하여 알려진 문제는 다음과 같습니다.

- Neutron은 네트워크 가상화용 플러그인을 하나만 지원하므로 Oracle Solaris를 실행하는 Nova 노드만 완전히 지원됩니다.
- 현재 Cinder 볼륨 첨부가 비전역 영역에서 지원되지 않습니다.
- VM 인스턴스는 현재 Oracle Solaris 버전을 실행 중이어야 합니다.
- Cinder 백업이 지원되지 않습니다.

  `cinder` 패키지를 설치할 때 `cinder-backup` 서비스가 설치됩니다. 그러나 "저장소 노드 구성" [45]에 설명된 것과 같은 기본 Cinder 배치에서는 이 서비스가 `backup_volume`에 대해 작동하지 않습니다.
- 대시보드의 Launch Instance(인스턴스 구동) 대화 상자에서 Instance Boot Source(인스턴스 부팅 소스)로 Boot from image(이미지로 부팅)만 지원됩니다. Project(프로젝트) -> Images & Snapshots(이미지 및 스냅샷) -> Actions(작업) 메뉴에서 `UploadToImage`가 지원되지 않습니다.
- `/etc/neutron/l3_agent.ini` 파일의 `external_network_datalink` 옵션에 대한 값으로 VXLAN 데이터 링크가 지원되지 않습니다. `external_network_datalink` 옵션에 대한 값

으로 VXLAN 데이터 링크를 설정하면 Neutron L3 에이전트가 외부 네트워크에서 VNIC를 만들고 연결하지 못합니다.

- 명령줄을 사용하여 프로젝트에 대한 네트워크 리소스의 쿼터를 수정해야 합니다.

  Horizon에서는 네트워크 리소스의 쿼터를 수정할 수 없습니다. Horizon 대시보드에서 프로젝트를 만들거나 기존 프로젝트의 비네트워크 리소스를 수정할 수 있습니다. 프로젝트의 네트워크, 서브넷, 포트, 라우터 또는 유동 IP 주소에 대한 쿼터를 수정하려면 `neutron quota-update` 명령을 사용해야 합니다.

  비네트워크 리소스를 수정하는 경우에도 다음 오류 메시지가 표시됩니다. 이 메시지는 무시해도 됩니다. 이 메시지와 달리 비네트워크 리소스에 대한 쿼터가 적용되었습니다.

  ```
  Error: Modified project information and members, but unable to modify project quotas.
  ```

- SMF와 OpenStack에서 서비스 상태를 다르게 보고할 수 있습니다.

  다음 예에서는 `nova-cert` 서비스가 OpenStack에서 사용 안함으로 설정되었음을 보여주지만 SMF는 서비스가 `online` 상태임을 보여줍니다.

  ```
  root@c190-133:~# nova service-disable c190-133 nova-cert
  +----------+-----------+----------+
  | Host     | Binary    | Status   |
  +----------+-----------+----------+
  | c190-133 | nova-cert | disabled |
  +----------+-----------+----------+
  root@c190-133:~# svcs nova-cert
  STATE          STIME    FMRI
  online         21:14:11 svc:/application/openstack/nova/nova-cert:default
  ```

로그 파일 검사

SMF 서비스 및 다양한 Oracle Solaris 프로세스가 오류 메시지를 검색하거나 화면에 표시된 메시지에 대한 세부정보를 수집할 수 있는 로그 파일을 생성합니다. SMF 서비스 로그 파일에는 유용한 디버깅 정보가 포함되어 있습니다.

OpenStack은 대개 여러 시스템에 걸쳐 설치되므로 확인해야 하는 로그 파일도 여러 위치에 있습니다. 더 체계적으로 문제를 해결하려면 노드별로 로그를 검토하십시오.

SMF 서비스와 관련된 문제를 수정하는 데 대한 일반적인 도움말은 Operating Systems Documentation의 사용 중인 Oracle Solaris 버전 라이브러리에서 서비스 문제 해결에 대한 *Managing System Services in Oracle Solaris*를 참조하십시오.

서비스 로그를 보려면 적절한 권한이 있어야 합니다. 적절한 RBAC 프로파일을 사용하여 OpenStack 서비스 로그 파일을 확인하거나 `pfedit` 명령으로 OpenStack 서비스 구성 파일을 수정하십시오. 다음 프로파일을 지정할 수 있습니다.

- OpenStack 블록 저장소 관리
- OpenStack Compute 관리

- OpenStack ID 관리
- OpenStack 이미지 관리
- OpenStack 네트워크 관리
- OpenStack 객체 저장소 관리
- OpenStack 관리

문제를 해결하려면 다음과 같은 일반적인 명령을 사용합니다.

- 특정 노드에서 실행 중인 OpenStack 서비스를 찾으려면 다음을 수행합니다.

```
# svcs -a | grep openstack
```

- 유지 관리 모드일 수 있는 서비스를 나열하려면 다음을 수행합니다.

```
# svcs -x
svc:/application/openstack/swift/swift-replicator-rsync:
      default (OpenStack Swift Replication Service)
State: maintenance since Fri May 22 04:06:11 2015
Reason: Start method exited with $SMF_EXIT_ERR_FATAL.
   See: http://support.oracle.com/msg/SMF-8000-KS
   See: rsync(1)
   See: rsyncd.conf(5)
   See: /var/svc/log/application-openstack-swift-swift-replicator-rsync:default.log
Impact: This service is not running.
```

  서비스가 유지 관리 모드인 경우 서비스 로그 파일을 확인하십시오.

- 특정 OpenStack 서비스에 대한 로그를 식별하려면 다음을 수행합니다.

```
# svcs -L openstack-service
```

  예를 들어 다음을 실행합니다.

```
# svcs -L neutron-server
/var/svc/log/application-openstack-neutron-neutron-server:default.log
```

  적절한 권한이 있는 경우 -Lv와 같이 옵션을 조합하여 서비스에 대한 로그를 나열하고 확인할 수 있습니다.

- 특정 로그에 기록된 오류 인스턴스를 즉시 식별하려면 grep과 같은 일반적인 UNIX 명령을 사용할 수 있습니다.

```
# grep keyword `svcs -L openstack-service`
```

  error, warning 및 기타 중요 키워드를 검색하여 오류 메시지를 직접 읽을 수 있습니다.

- 네트워킹 문제를 해결할 때 EVS 등록 정보를 확인하려면 evsadm show-prop와 같은 다양한 evsadm 하위 명령을 사용하십시오.

다음 로그에도 문제 해결에 유용한 정보가 포함될 수 있습니다.

- nova-compute

- `nova-scheduler`
- `cinder-scheduler`
- `neutron-server`

SMF 서비스 로그 파일 이외에도 `/var/log` 디렉토리의 로그를 확인할 수 있습니다. 다른 Oracle Solaris 프로세스와 마찬가지로 OpenStack 서비스도 `/var/log/`*openstack-service* 디렉토리에 자체 로그 파일을 생성합니다.

예를 들어, OpenStack 이미지 저장소 로그 파일은 `/var/log/glance`에 있습니다. VM 인스턴스를 만들고 부트하는 동안 발생하는 문제는 `/var/log/zones` 디렉토리에 기록될 수 있습니다. 메시징 로그는 `/var/log/rabbitmq/rabbit@`*hostname*`.log`로 저장됩니다.

대부분의 OpenStack 구성 파일은 `/etc` 디렉토리의 OpenStack 서비스 이름 아래에 있습니다. 예를 들어, OpenStack 네트워킹 구성 파일은 `/etc/neutron`에 있습니다. Horizon에 대한 구성 파일은 `/etc/openstack_dashboard`에 있습니다. Nova의 경우 `/etc/nova`에 있습니다. 서비스의 구성 파일에서 다음 매개변수를 설정하거나 주석 처리를 해제하여 특정 서비스의 문제를 해결할 수 있습니다.

- `debug=true`
- `verbose=true`

이러한 매개변수를 사용하면 해당 구성 파일의 영향을 받는 작업에 대해 더 많은 출력을 확인할 수 있습니다. `http://www.oracle.com/technetwork/articles/servers-storage-admin/getting-started-openstack-os11-2-2195380.html`의 "Common Configuration Parameters for OpenStack" 및 OpenStack 설명서 사이트의 *OpenStack Configuration Reference*에 나오는 구성 옵션 표를 참조하십시오.

주 - 개별 OpenStack 서비스 명령에도 `--debug` 옵션을 사용할 수 있습니다. 이 옵션은 구성 파일에 `debug=true`를 설정하는 것과 동일합니다.

문제 조사 및 해결

이 절에서는 OpenStack을 설치 및 구성할 때 발생할 수 있는 몇 가지 문제에 대해 설명합니다.

다음 예에서는 대시보드와 관련된 오류를 보여줍니다.

```
Error: Unauthorized: Unable to retrieve usage information.
Error: Unauthorized: Unable to retrieve quota information.
Error: Unauthorized: Unable to retrieve project list information.
Error: Unauthorized: Unable to retrieve instance list information.
```

이 메시지는 RSA 소스 키가 변경되었지만 일부 구성요소에 전파되지 않았음을 나타낼 수 있습니다. RSA 키 구성에 대한 자세한 내용은 Neutron을 설치 및 구성하는 방법 및 컴퓨트 노드를 구성하는 방법을 참조하십시오.

다음 오류 보고서는 `nova-scheduler` 로그에 포함될 수 있습니다.

```
controller# grep error `svcs -L nova-scheduler`
2014-12-03 12:49:19.271 3475 TRACE
nova.openstack.common.rpc.common error: [Errno 32] Broken pipe
```

손상된 파이프 오류는 대개 한 OpenStack 서비스를 새로 고치고 다른 OpenStack 서비스를 고치지 않은 경우 보고됩니다. 노드에서 구성 파일을 변경한 경우 해당 노드의 모든 서비스를 새로 고치십시오. 다음 명령을 실행하면 온라인 서비스 중 새로 고쳐야 하는 서비스가 다시 시작됩니다.

```
controller# svcs \*openstack\* | grep online  \
| awk -e '{print $3}' | xargs svcadm restart
```

오류는 리소스 부족으로 인해 발생할 수도 있습니다. VM 인스턴스를 만드는 중 `nova-compute` 로그에 다음과 비슷한 메시지가 표시될 수 있습니다.

```
[abc-123-def-456] Build of instance
abc-123-def-456 aborted: Image
xyz-987-uvw-654 is unacceptable: Image query failed.
Possibly invalid or corrupt. Log file location: controller:/tmp/archive_log.4249
```

또한 로그에 `out of space/storage`가 표시될 수도 있습니다. 시스템의 리소스를 보려면 `top` 명령을 사용하십시오. 시스템의 메모리가 1GB 이하인 경우 메모리를 추가하는 것이 좋습니다.

네트워크 만들기

내부 네트워크 및 해당 구성요소를 구성하는 동안 다음 메시지가 콘솔 화면에 지속적으로 표시될 수 있습니다.

```
To: root@controller.mydomain.com
From: neutron@controller.mydomain.com
Subject: *** SECURITY information for controller ***
Content-Length: 143

controller: datetime : neutron user NOT in sudoers; TTY = unknown ; PWD=var/lib/neutron;
user=ROOT ; COMMAND=command:
```

이 메시지가 생성되지 않도록 하려면 다음과 같이 `/etc/neutron/neutron.conf` 파일을 편집합니다.

1. `root_helper` 옵션에 대한 행의 주석 처리를 해제합니다.
2. 매개변수가 없음으로 설정되었는지 확인합니다.

   ```
   root_helper =
   ```

VM 인스턴스 설치 및 구성

이 절에서 설명되는 문제는 특별히 VM 인스턴스와 관련되어 있습니다.

VM 인스턴스가 오류 상태임

호스트 시스템과 다른 아키텍처인 VM 인스턴스를 설치하려고 시도한 경우 VM 인스턴스가 오류 상태일 수 있습니다. 이 경우 구체적으로 아키텍처 불일치를 나타내는 오류 메시지가 표시되지 않을 수도 있습니다. 이 문제가 발생하지 않도록 하려면 `glance` 이미지 저장소로 이미지를 업로드할 때 이미지의 `architecture` 등록 정보를 올바르게 설정해야 합니다. Horizon을 사용하여 이미지를 업로드하는 경우 업로드 후 이미지에 대한 등록 정보를 설정해야 합니다. 또는 명령줄을 사용하여 이미지를 업로드하고 하나의 `glance image-create` 명령에서 등록 정보 값을 설정할 수 있습니다. 예는 "Glance 저장소에 대한 이미지 준비" [59]를 참조하십시오.

VM 인스턴스 등록 정보 값이 영역 등록 정보 값과 일치하지 않음

OpenStack이 VM 인스턴스에 대해 보고하는 일부 정보와 Oracle Solaris가 해당하는 영역에 대해 보고하는 정보가 일치하지 않습니다. Horizon에 표시되는 정보 또는 `nova` 명령으로 표시되는 정보가 `zoneadm` 명령 또는 기타 Oracle Solaris 명령으로 표시되는 정보와 일치하지 않을 수 있습니다.

이름
: Horizon에 표시되거나 `nova list` 명령으로 표시되는 VM 인스턴스의 이름은 인스턴스를 만들 때 지정한 이름(예: `example-instance`)입니다. `zoneadm list` 명령으로 표시되는 영역의 이름은 `instance-00000001`과 유사합니다. `nova show` 명령을 사용하여 VM 인스턴스와 연관된 영역을 확인할 수 있습니다. `nova show` 출력에서 `OS-EXT-SRV-ATTR:instance_name` 등록 정보의 값은 영역 이름이며, `name` 등록 정보의 값은 VM 인스턴스 이름입니다.

UUID
: Horizon에 표시되거나 `nova show` 명령으로 표시되는 VM 인스턴스의 UUID가 `zoneadm list -p` 명령으로 표시되는 동일한 영역의 UUID와 일치하지 않습니다. `zoneadm` 명령이 표시하는 UUID는 Nova에 대해 사용된 식별자와 다른 식별자입니다.

CPU
: Horizon에 표시되는 VM 인스턴스의 VCPU 수는 상한값이 설정된 CPU 수로, 인스턴스가 사용할 수 있는 부분 CPU 수까지만 가상화됩니다. 이 수는 상한값이 설정된 CPU의 인스턴스 내에서 관찰성을 제공하지 않습니다. `psrinfo` 명령은 영역에 할당된 전용 CPU를 보고합니다.

메모리
: Horizon에 표시되는 VM 인스턴스의 메모리 양이 해당 VM 인스턴스에 로그인할 때 `prtconf` 명령으로 표시되는 메모리 양과 다를 수 있습니다. Horizon은 VM 인스턴스를 만들 때 사용되는 flavor로 지정된 메모리 양을 보여줍니다. `prtconf` 명령은 모든 시스템 메모리를 보고합니다.

저장소
: VM 인스턴스가 ZOSS(공유 저장소의 영역)를 사용하는 비전역 영역이 아닌 경우 Horizon에 표시되는 VM 인스턴스의 저장소 양이 해당 VM 인스턴스에 로그인할 때 표시되는 저장소 양과 다를 수 있습니다.

자격 증명 문제

특정 상황에서 서비스 명령을 실행하지 못하게 되는 잘못된 자격 증명과 관련된 오류 메시지가 나타날 수 있습니다. 예를 들어, glance 명령을 실행할 때 다음 오류 메시지가 생성될 수 있습니다.

```
Invalid OpenStack Identity credentials.
```

이 메시지의 근본 원인은 매번 다를 수 있습니다. 따라서 로그를 조사하여 가능한 원인을 찾아야 합니다. 예와 같은 glance 서비스의 경우 Glance SMF 서비스 로그의 내용을 확인하십시오. /var/log/glance/api.log에서는 다음을 보고할 수 있습니다.

```
WARNING keystonemiddleware.auth_token [-] Authorization failed for token
```

glance 구성 파일에서 Debug = True 및 Verbose = True를 설정한 경우 다음과 같이 더 많은 세부정보가 /var/svc/log/application-openstack-glance-glance-api:default.log에 제공됩니다.

```
DEBUG keystonemiddleware.auth_token [-] Received request from user:
 user_id None, project_id None, roles None service: user_id None,
 project_id None, roles None
__call__/usr/lib/python2.7/vendor-packages/keystonemiddleware/auth_token.py:821
```

다음 영역을 살펴보고 문제를 해결할 수 있습니다.

- 서비스 구성 파일에서 관련 매개변수가 올바르게 정의되었는지 확인합니다.
- 서비스에 대한 전역 셸 변수가 올바른지 확인합니다. 예를 들어, Glance 서비스의 경우 다음 변수가 설정되어야 합니다.
  - OS_USERNAME=glance
  - OS_PASSWORD=*service-password*
  - OS_PROJECT_NAME=service
  - OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0

동일한 오류 메시지가 보고되면서 명령을 계속 실패하는 경우 서비스 사용자를 다시 만들어 새 자격 증명을 생성해야 합니다. 다음 예를 살펴보십시오.

```
# export OS_USERNAME=admin
# export OS_PASSWORD=service-password
# export OS_PROJECT_NAME=project
# export OS_AUTH_URL=http://$CONTROLLER_ADMIN_NODE:5000/v2.0

# openstack user list
```

이 명령의 출력에서 손상된 서비스 사용자의 ID 번호를 확인합니다. 해당 사용자를 삭제한 다음 올바른 자격 증명으로 다시 만듭니다.

```
# openstack user delete user-ID

# openstack user create --name glance --password service-password
# openstack user role add --user=glance --project=service --role=admin
```

Horizon 관련 문제

VM 인스턴스를 실행한 후 Horizon 대시보드에 액세스할 수 없고 `404 Not Found` 오류 메시지가 표시됩니다. Apache 서비스 로그에 다음과 같은 항목이 포함되어 있습니다.

```
Oct 13  16:13:53  Executing start method (" /lib/svc/method/http-apache24 start"). ]
Apache version is 2.4
(125) Address already in use: AH000/2: make_sock: could not bind to address [::]:80
Oct 13  16:13:55 Method "start" exited with status 0. ]
```

로그는 포트 80이 사용 중이므로 주소를 해당 포트에 바인드할 수 없음을 나타냅니다.

Kilo 버전부터 OpenStack은 이전 OpenStack 버전의 Apache 2.2 대신 Apache 2.4를 사용합니다. 올바른 Apache 버전이 사용으로 설정되었으며 해당 포트를 수신하고 있는지 확인하십시오.

포트를 해제하려면 다음 단계를 수행하십시오.

1. 포트에서 현재 수신 중인 프로세스 ID를 확인합니다.

   ```
   # netstat -uan -f inet | grep "*.80"
   ```

   이 절차는 설정이 IPv4 주소를 사용하고 있으며 포트 80을 보유한 프로세스가 이러한 모든 주소에서 수신하고 있다고 가정합니다. 프로세스가 IPv6 트래픽을 수신하는 경우 명령이 어떠한 결과도 제공하지 않을 수 있습니다.
2. 프로세스 ID를 기준으로 실제 프로세스 또는 서비스를 식별합니다. 다음 명령 중 하나를 사용할 수 있습니다:

   ```
   # svcs -p | egrep "online|pid http" | ggrep -B1 pid
   ```

   또는

   ```
   # ps -lf -p pid
   ```
3. 잘못된 Apache 버전이 포트를 사용 중인 경우 서비스를 사용 안함으로 설정합니다.
4. Kilo 버전에 대해 올바른 Apache 버전을 사용으로 설정합니다.

   올바른 Apache 버전이 유지 관리 모드일 경우 먼저 서비스를 해제한 후 사용으로 설정합니다.

다음 예에서는 포트 80을 해제하고 올바른 Apache 버전으로 전환하는 방법을 보여줍니다.

```
# netstat -uan -f inet | grep "*.80"
*.80       *.*     root         5560 httpd     0     0  128000     0 LISTEN
*.8080     *.*     webservd     1124 java      0     0  128000     0 LISTEN
*.8009     *.*     webservd     1124 java      0     0  128000     0 LISTEN

# svcs -p | egrep "online|5560 http" | ggrep -B1 5560
online     Aug_31   svc:/network/http:apache22
           Sep_09   5560 httpd
```

```
# svcadm disable apache22
# svcadm clear apache24
# svcadm enable apache24
```

확장성 문제

기본적으로 RabbitMQ는 파일 설명자를 255자로 제한합니다. 이 제한이 있으면 소수의 컴퓨트 노드를 만든 후 클라우드가 추가로 확장되는 것을 손쉽게 방지할 수 있습니다. 이 차단을 방지하려면 `/etc/rabbitmq/rabbitmq-env.conf` 파일에서 제한 값을 늘리십시오.

```
# Increase soft limit on file descriptors for RabbitMQ
ulimit -n 8192
```

네트워크 해체

네트워크 노드에서 Neutron 구성에 문제가 발생할 때 구성을 해체하고 처음부터 다시 시작해야 하는 경우 이 절차를 따르십시오. 구성 취소를 시작해야 하는 시점에 따라 절차에 제공된 순서를 따르십시오.

▼ Neutron에서 네트워크 구성을 제거하는 방법

1. **Horizon 대시보드에서 이 단계를 수행합니다.**
   a. **모든 유동 IP 주소의 연관을 해제합니다.**
   b. **모든 유동 IP 주소를 제거합니다.**
2. **터미널 창에서 다음 명령을 입력합니다.**

   ```
   # neutron router-gateway-clear router-ID external-network-ID
   ```

   ```
   # neutron router-interface-delete router-ID subnet-ID
   ```

   a. **라우터 게이트웨이 인터페이스를 제거하려면 다음 명령을 입력합니다.**

      ```
      # neutron router-gateway-interface-delete router-ID external-network-ID
      ```

   b. **남은 라우터 인터페이스를 제거하려면 다음 명령을 입력합니다.**

      ```
      # neutron router-interface-delete router-ID subnet-ID
      ```

3. **Horizon 대시보드에서 다음을 수행합니다.**

a. **모든 VM 인스턴스를 종료합니다.**

b. **서브넷을 삭제합니다.**
서브넷을 삭제하는 중 문제가 발생하면 Vport를 제거하는 방법 [135]을 참조하십시오.

c. **네트워크를 삭제합니다.**

▼ Vport를 제거하는 방법

서브넷 삭제를 방해하는 문제가 발생할 경우 이 절차를 사용하십시오.

1. **어떤 vport가 현재 사용 중인지 확인합니다.**

```
# evsadm
```

2. **사용 중인 vport를 재설정합니다.**

```
# evsadm reset-vport vport
```

3. **vport를 제거합니다.**

```
# evsadm remove-vport vport
```

디버깅에 대한 일반적인 팁과 힌트

다음 일반적인 팁은 OpenStack에서 문제를 해결할 때 시작할 수 있도록 도와줍니다.

- 다양한 구성 파일에서 `debug = true` 및 `verbose = true`를 설정하면 문제를 진단하는 데 도움이 됩니다. 일부 구성 파일에서는 주석 처리되고 토글하여 상세 정보 로깅을 사용으로 설정할 수 있는 위치 표시자 값을 찾아볼 수 있습니다.
- 구성요소의 구성 파일을 변경할 경우 구성요소에서 변경사항을 적용하려면 서비스를 다시 시작합니다.
- SMF 로그의 정보를 제공하는 `tail -30 `svcs -L` *service-name* 명령을 사용합니다. 디버깅 및 상세 정보 로깅을 사용으로 설정한 경우 `tail` 명령에 대해 지정하는 라인을 늘려야 할 수 있습니다.
- Horizon 프로세스는 Apache를 거칩니다. 따라서 Horizon을 진단하려면 `/etc/openstack_dashboard/local_settings.py` 파일에서 `debug = True`를 사용으로 설정합니다. 그러면 Django 오류가 웹 페이지에 생성됩니다.
- Nova 구성요소는 Oracle Solaris 영역을 기반으로 만들어집니다. 따라서 `/var/log/zones`의 로그를 참조하여 Nova 문제를 해결할 수도 있습니다.

유용한 사이트

다음 사이트에서 문제 해결 팁을 참조하여 여러 가지 OpenStack 문제를 해결하십시오.

- https://blogs.oracle.com/openstack/
- https://ask.openstack.org/en/questions/
- https://raymii.org/s/tags/openstack.html

♦ ♦ ♦ 부 록 A

공통 OpenStack 구성 파일 및 서비스

이 부록에서는 핵심 OpenStack 구성요소의 일반적인 구성 파일 및 OpenStack SMF 서비스를 나열합니다.

구성 파일

Cinder 파일

- `/etc/cinder/api_paste.ini`
- `/etc/cinder/cinder.conf`

Glance 파일

- `/etc/glance/glance-api.conf`
- `/etc/glance/glance-cache.conf`
- `/etc/glance/glance-registry.conf`
- `/etc/glance/glance-scrubber.conf`
- `/etc/glance/glance-registry-paste.ini`
- `/etc/glance/glance-api-paste.ini`

Keystone 파일

- `/etc/keystone/keystone.conf`
- `/etc/keystone/keystone-paste.ini`

Neutron 파일

- `/etc/neutron/neutron.conf`

- /etc/neutron/dhcp_agent.ini
- /etc/neutron/l3_agent.ini
- /etc/neutron/api-paste.ini
- /etc/neutron/metadata_agent.ini
- /etc/neutron/plugins/evs/evs_plugin.ini

Nova 파일

- /etc/nova/nova.conf
- /etc/nova/api_paste.ini

Horizon 파일

- /etc/openstack_dashboard/local_settings.py
- /etc/apache2/2.4/httpd.conf
- /etc/apache2/2.4/conf.d/openstack-dashboard-http.conf
  또는
  /etc/apache2/2.4/conf.d/openstack-dashboard-tls.conf

Swift 파일

- /etc/swift/swift.conf
- /etc/swift/account-server.conf
- /etc/swift/container-server.conf
- /etc/swift/object-server.conf
- /etc/swift/proxy-server.conf
- /etc/swift/rsyncd.conf

OpenStack SMF 서비스

Cinder

```
svc:/application/openstack/cinder/cinder-db:default
svc:/application/openstack/cinder/cinder-backup:default
svc:/application/openstack/cinder/cinder-scheduler:default
```

```
svc:/application/openstack/cinder/cinder-api:default
svc:/application/openstack/cinder/cinder-volume:setup
svc:/application/openstack/cinder/cinder-volume:default
```

Glance

```
svc:/application/openstack/glance/glance-db:default
svc:/application/openstack/glance/glance-registry:default
svc:/application/openstack/glance/glance-scrubber:default
svc:/application/openstack/glance/glance-api:default
```

Keystone

```
svc:/application/openstack/keystone:default
```

Neutron

```
svc:/application/openstack/neutron/neutron-server:default
svc:/application/openstack/neutron/neutron-dhcp-agent:default
svc:/application/openstack/neutron/neutron-metadata-agent:default
svc:/application/openstack/neutron/neutron-l3-agent:default
```

Nova *

```
svc:/application/openstack/nova/nova-objectstore:default
svc:/application/openstack/nova/nova-consoleauth:default
svc:/application/openstack/nova/nova-novncproxy:default
svc:/application/openstack/nova/nova-api-metadata:default
svc:/application/openstack/nova/nova-api-ec2:default
svc:/application/openstack/nova/nova-api-osapi-compute:default
svc:/application/openstack/nova/nova-conductor:default
svc:/application/openstack/nova/nova-cert:default
svc:/application/openstack/nova/nova-compute:default
svc:/application/openstack/nova/nova-scheduler:default
```

* 컴퓨트 노드 설정에 따라 다른 Nova 서비스도 나열될 수 있습니다.

Swift

```
svc:/application/openstack/swift/swift-object-expirer:default
svc:/pplication/openstack/swift/swift-account-reaper:default
svc:/application/openstack/swift/swift-container-replicator:default
```

```
svc:/application/openstack/swift/swift-account-replicator:default
svc:/application/openstack/swift/swift-object-auditor:default
svc:/application/openstack/swift/swift-container-updater:default
svc:/application/openstack/swift/swift-container-sync:default
svc:/application/openstack/swift/swift-object-updater:default
svc:/application/openstack/swift/swift-account-auditor:default
svc:/application/openstack/swift/swift-replicator-rsync:default
svc:/application/openstack/swift/swift-container-auditor:default
svc:/application/openstack/swift/swift-object-replicator:default
svc:/application/openstack/swift/swift-container-reconciler:default
svc:/application/openstack/swift/swift-container-server:default
svc:/application/openstack/swift/swift-object-server:default
svc:/application/openstack/swift/swift-proxy-server:default
svc:/application/openstack/swift/swift-account-server:default
```

♦ ♦ ♦ 부 록 B

OpenStackClient 명령

OpenStackClient(OSC)

OpenStack 클라이언트는 동일한 명령 구조를 사용하여 단일 셸에 컴퓨트, ID, 이미지, 객체 저장소 및 블록 저장소 API에 대한 전체 명령 세트를 한 곳에서 제공합니다. 새 명령 세트는 이전 OpenStack 버전에 사용됐던 구성요소 이름 대신 `openstack`을 기본 명령으로 사용합니다. 클라이언트 구현은 Kilo 버전부터 시작되었습니다. 이 부록은 이전 OpenStack 하위 명령을 새 클라이언트와 매핑합니다.

주 - 해당하는 목록은 부분 목록입니다.

`openstack` 명령 및 하위 명령에 대한 정보를 가져오려면 다음 명령 중 하나를 사용합니다.

- `openstack help` *subcommand*
- `openstack --help`

하위 명령 없이 `openstack`을 입력하면 대화식 모드로 전환됩니다. 여기서 정보를 보려면 `help [`*subcommand*`]`를 입력합니다. 대화식 모드를 종료하려면 `quit`를 입력합니다.

`openstack` 하위 명령, 옵션 및 구문에 대한 자세한 목록은 http://docs.openstack.org/developer/python-openstackclient/index.html을 참조하십시오.

표 1 OpenStackClient 명령에 상응하는 명령

기본 명령	하위 명령	openstack 기본 명령에 해당하는 하위 명령	주
cinder			
	type-create	volume type create	
	type-key	volume type set/volume type unset	
	service-list	service list	
glance			
	image-create	image create	API 버전 1만
	image-show	image show	
keystone			

기본 명령	하위 명령	openstack 기본 명령에 해당하는 하위 명령	주
	user-list	user list	
	user-role-add	user role add	
	user-role-list	user role list	
neutron			
	net-create	network create	
	router-create	router create	
	router-list	router list	
	router-show	router show	
	subnet-list	subnet list	
nova			
	flavor-key	flavor set/flavor unset	OpenStack Liberty 버전의 경우
	flavor-list	flavor list	
	flavor-show	flavor show	
	image-create	image create	API 버전 1만
	image-list	image list	
	image-show	image show	
	migrate	server migrate	
	resize	server resize	
	service-disable	compute service set/compute service unset	Keystone 버전 3만
	show	server show	

색인

ㅍ

ㅎ