Red Hat Enterprise Linux 7

고가용성 추가 기능 관리

고가용성 추가 기능 설정 및 관리

고가용성 추가 기능 설정 및 관리

법적 공지

초록

고가용성 추가 기능 관리에서는 Red Hat Enterprise Linux 7의 고가용성 추가 기능의 설정 및 관리에 대해 설명합니다.

차례

1장. 페이스메이커(Pacemaker)를 사용한 Red Hat 고가용성 클러스터 생성

다음 부분에서는 **pcs**를 사용해서 Red Hat 고가용성 2 노드 클러스를 생성하는 방법에 대해 설명합니다. 클러스터를 생성한 후 사용자가 필요로 하는 리소스와 리소스 그룹을 설정할 수 있습니다.

여기에 설명되어 있는 방법에 따라 클러스터를 설정하려면 다음과 같은 구성 요소가 시스템에 설치되어야 합니다:

- 클러스터 생성에 사용될 두 개의 노드. 예를 들어, **z1.example.com** 그리고 **z2.example.com**인 두 개의 노드가 사용됩니다.
- 클러스터 노드 그리고 네트워크 전원 스위치와 파이버 채널 스위치와 같은 다른 클러스터 하드웨어와 통신하는 데 필요한 비공개 네트워크용 네트워크 스위치
- 클러스터의 각 노드별 전원 차단(power fencing) 장치. 예를 들어, **zapc.example.com** 호스트명을 갖고 있는 APC 전원 스위치의 포트를 두개 사용합니다.

이는 다음과 같은 부분으로 구성되어 있습니다.

- 1.1절. “클러스터 소프트웨어 설치”는 클러스터 소프트웨어 설치 절차를 설명합니다.
- 1.2절. “클러스터 생성”은 2 노드 클러스 설정 절차를 설명합니다.
- 1.3절. “차단 장치 설정”은 클러스터의 각 노드에 대한 차단 장치 설정 절차를 설명합니다.

1.1. 클러스터 소프트웨어 설치

다음의 절차를 통해 클러스터를 설치하고 설정합니다.

1. 클러스터의 각 노드에서 Red Hat 고가용성 추가 기능 소프트웨어 패키지와 함께 고가용성 채널에서 사용 가능한 모든 차단 에이전트를 설치합니다.

```
# yum install pcs fence-agents-all
```

2. 사용자가 **firewalld** 데몬을 실행중이라면 다음의 명령을 실행해서 Red Hat 고가용성 추가 기능에 의해 요구되는 포트를 활성화합니다.

참고

사용자는 **rpm -q firewalld** 명령을 실행해서 **firewalld** 데몬이 시스템에 설치되어 있는지 확인할 수 있습니다. **firewalld** 데몬이 설치되어 있다면 그것이 실행중인지 확인하기 위해 **firewall-cmd --state** 명령을 실행합니다.

```
# firewall-cmd --permanent --add-service=high-availability
# firewall-cmd --add-service=high-availability
```

3. **pcs**를 사용해서 클러스터를 설정하고 노드 간에 통신하려면 **pcs** 관리 계정인 사용자 ID **hacluster**의 암호를 각 노드에 설정해야 합니다. **hacluster** 사용자의 암호를 각 노드에서 동일하게 하는 것을 권장합니다.

```
# passwd hacluster
```

```
Changing password for user hacluster.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

4. 클러스터가 설정되기 전에 **pcsd** 데몬이 반드시 시작되어야 하며 각 노드에서 시작 시 부팅되도록 활성화되어야 합니다. 이 데몬은 **pcs** 명령을 사용해서 해당 클러스터에 있는 모든 노드의 설정을 관리할 수 있습니다.

   클러스터의 각 노드에서 다음의 명령을 실행해서 시스템 시작 시 **pcsd** 서비스를 시작하고 **pcsd**를 활성화하도록 합니다.

```
# systemctl start pcsd.service
# systemctl enable pcsd.service
```

5. 사용자가 **pcs**를 실행하게 될 노드에서 **pcs** 사용자 **hacluster**를 클러스터의 각 노드에 대해 인증합니다.

   다음의 명령을 사용해서 예시와 같은 2 노드 클러스터의 두 노드 **z1.example.com**과 **z2.example.com** 모두를 위해 **z1.example.com**에서 **hacluster** 사용자를 인증합니다.

```
[root@z1 ~]# pcs cluster auth z1.example.com z2.example.com
Username: hacluster
Password:
z1.example.com: Authorized
z2.example.com: Authorized
```

1.2. 클러스터 생성

다음의 절차에 따라 **z1.example.com**과 **z2.example.com** 노드로 구성된 Red Hat 고가용성 추가 기능 클러스터를 생성합니다.

1. **z1.example.com**에서 다음의 명령을 실행해서 **z1.example.com**과 **z2.example.com** 노드로 구성된 2 노드 클러스터인 **my_cluster**를 생성합니다. 이를 통해 클러스터의 두 노드 모두에 클러스터 설정 파일을 전달합니다. 이 명령은 클러스터의 두 노드 모두에서 클러스터 서비스를 시작하게 해주는 **--start** 옵션을 포함합니다.

```
[root@z1 ~]# pcs cluster setup --start --name my_cluster \
z1.example.com z2.example.com
z1.example.com: Succeeded
z1.example.com: Starting Cluster...
z2.example.com: Succeeded
z2.example.com: Starting Cluster...
```

2. 노드가 부팅되었을 때 클러스터의 각 노드에서 클러스터 서비스가 실행되도록 활성화합니다.

참고

사용자의 특정 환경에서는 다음의 단계를 생략해서 클러스터 서비스를 비활성화된 상태로 둘 수도 있습니다. 이렇게 하면 노드가 작동하지 않게 되었을 때 사용자의 클러스터나 리소스에 문제가 있을 경우 해당 노드가 클러스터에 다시 참여하기 전에 이 문제가 해결되도록 할 수 있습니다. 클러스터 서비스를 비활성화된 상태로 두면 사용자가 노드를 재부팅할 때 해당 노드에서 **pcs cluster start** 명령을 실행해서 서비스를 수동으로 시작해야 합니다.

```
# pcs cluster enable --all
```

pcs cluster status 명령을 사용해서 클러스터의 현재 상태를 표시할 수 있습니다. 사용자가 **pcs cluster setup** 명령의 **--start** 옵션을 사용해서 클러스터 서비스를 시작했을 때 클러스터가 시작 및 실행되는 데 약간 시간이 걸릴 수 있으므로 클러스터 서비스를 시작한 후 클러스터와 클러스터 설정에 다른 동작을 실행하기 전에 먼저 클러스터가 시작 및 실행중인지 확인해야 합니다.

```
[root@z1 ~]# pcs cluster status
Cluster Status:
 Last updated: Thu Jul 25 13:01:26 2013
 Last change: Thu Jul 25 13:04:45 2013 via crmd on z2.example.com
 Stack: corosync
 Current DC: z2.example.com (2) - partition with quorum
 Version: 1.1.10-5.el7-9abe687
 2 Nodes configured
 0 Resources configured
```

1.3. 차단 장치 설정

사용자는 클러스터에 있는 각 노드에 대해 차단 장치를 설정해야 합니다. 차단 장치 설정에 관한 일반적인 정보는 *Red Hat Enterprise Linux 7 High Availability Add-On Reference*를 참조하십시오.

참고

차단 장치를 설정할 때 사용자의 차단 장치가 제어하는 노드와 그 장치가 전원을 공유하고 있지 않은지 확인합니다.

다음의 예시에서는 노드를 차단하기 위해 **zapc.example.com** 호스트명을 갖고 있는 APC 전원 스위치와 **fence_apc_snmp** 차단 에이전트를 사용합니다. 두 노드가 모두 동일한 차단 에이전트에 의해 차단될 것이므로 사용자는 **pcmk_host_map**과 **pcmk_host_list** 옵션을 사용해서 두 차단 장치를 단일 리소스로 설정할 수 있습니다.

pcs stonith create 명령을 사용해서 차단 장치를 **stonith** 리소스로 설정해서 생성합니다. 다음의 명령을 사용해서 **z1.example.com**과 **z2.example.com** 노드에 대해 **fence_apc_snmp** 차단 에이전트를 사용하는 **myapc**라는 이름의 **stonith** 리소스를 설정합니다. **pcmk_host_map** 옵션을 사용해서 **z1.example.com**을 포트 1에 매핑하고 **z2.example.com**을 포트 2에 매핑합니다. APC 장치의 로그인 값과 암호는 모두 **apc**입니다. 기본값으로 이 장치는 각 노드에 대해 60초의 모니터 간격을 둡니다.

노드를 위한 호스트명을 지정할 때 IP 주소를 사용할 수 있습니다.

```
[root@z1 ~]# pcs stonith create myapc fence_apc_snmp params \
```

```
ipaddr="zapc.example.com" pcmk_host_map="z1.example.com:1;z2.example.com:2"
\
pcmk_host_check="static-list" pcmk_host_list="z1.example.com,z2.example.com"
\
login="apc" passwd="apc"
```

참고

사용자가 **fence_apc_snmp stonith** 장치를 생성할 때 다음과 같은 경고 메시지가 표시될 수도 있지만 무시해도 됩니다:

```
Warning: missing required option(s): 'port, action' for resource type:
stonith:fence_apc_snmp
```

다음의 명령을 사용해서 기존 STONITH 장치의 매개 변수를 표시합니다.

```
[root@rh7-1 ~]# pcs stonith show myapc
 Resource: myapc (class=stonith type=fence_apc_snmp)
  Attributes: ipaddr=zapc.example.com
pcmk_host_map=z1.example.com:1;z2.example.com:2 pcmk_host_check=static-list
pcmk_host_list=z1.example.com,z2.example.com login=apc passwd=apc
  Operations: monitor interval=60s (myapc-monitor-interval-60s)
```

2장. Red Hat 고가용성 클러스터에 있는 활성/수동 Apache 웹 서버

다음 부분에서는 클러스터 리소스를 설정하는 데 **pcs**를 사용해서 2 노드 Red Hat Enterprise Linux 고가용성 추가 기능 클러스터에 활성/수동 Apache 웹 서버를 설정하는 방법에 대해 설명합니다. 이렇게 사용되는 경우 클라이언트가 부동 IP 주소를 통해 Apache 웹 서버에 액세스합니다. 웹 서버는 클러스터에 있는 두 노드 중 하나에서 실행됩니다. 웹 서버가 실행되고 있는 노드가 실행 불가능하게 될 경우, 웹 서버가 클러스터에 있는 두 번째 노드에 다시 시작되면서 서비스 장애를 최소화합니다.

그림 2.1. “Red Hat 고가용성 2 노드 클러스터에 있는 Apache 웹 서버”는 클러스터의 전반적인 개요를 보여줍니다. 이 클러스터는 네트워크 전원 스위치 및 공유 스토리지가 설정되어 있는 2 노드 Red Hat 고가용성 클러스터입니다. 이 클러스터 노드는 클라이언트가 가상 IP를 통해 Apache 웹 서버에 액세스할 수 있도록 공개 네트워크에 연결되어 있습니다. Apache 서버는 Node 1 또는 Node 2에서 실행되며, 각 노드는 Apache 데이터가 저장되는 스토리지에 액세스할 수 있습니다.

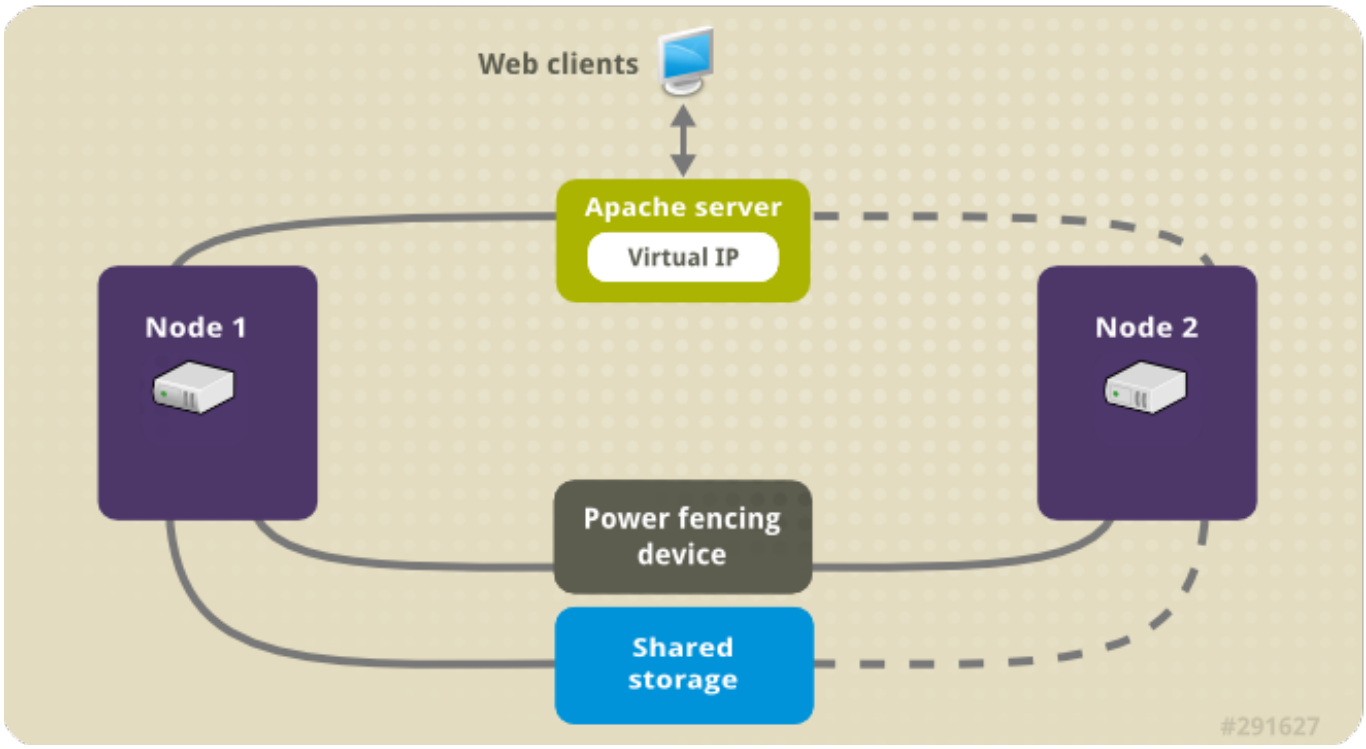


그림 2.1. Red Hat 고가용성 2 노드 클러스터에 있는 Apache 웹 서버

이렇게 사용하려면 다음과 같은 구성 요소가 시스템에 설치되어야 합니다:

- 각 노드에 전원 차단(power fencing)이 설정되어 있는 2 노드 Red Hat 고가용성 클러스터. 이 절차는 1장. *페이스메이커(Pacemaker)를 사용한 Red Hat 고가용성 클러스터 생성*에 제시된 클러스터 예시를 사용합니다.
- Apache 웹 서버에 요구되는 공개된 가상 IP 주소
- iSCSI 또는 파이버(Fibre) 채널을 사용하는 클러스터의 노드를 위한 공유 스토리지

이 클러스터는 웹 서버가 요구하는 클러스터 구성 요소인 LVM 리소스, 파일 시스템 리소스, IP 주소 리소스, 웹 서버 리소스 등이 들어 있는 Apache 리소스 그룹이 설정되어 있습니다. 이 리소스 그룹은 클러스터의 한 노드에서 다른 노드로 페일오버하므로 두 노드 중 어느 노드에서도 웹 서버를 실행할 수 있습니다. 이 클러스터를 위한 리소스 그룹을 생성하기 전에 사용자는 다음의 절차를 실행합니다:

1. 2.1절. “ext4 파일 시스템이 있는 LVM 볼륨 설정”에 있는 설명에 따라서 **my_lv** 논리 볼륨에 마운트되어 있는 **ext4** 파일 시스템을 설정합니다.
2. 2.2절. “웹 서버 설정”에 있는 설명에 따라서 웹 서버를 설정합니다.

3. 2.3절. “클러스터에서 볼륨 그룹의 단독 활성화”에 있는 설명에 따라서 오직 그 클러스터만이 **my_lv**를 포함하는 볼륨 그룹을 활성화할 수 있으며, 그 볼륨 그룹은 시작 시 클러스터 밖에서 활성화되지 않도록 확인합니다.

이 절차를 실행한 후에 2.4절. “pcs 명령을 사용한 리소스 및 리소스 그룹 생성”에 있는 설명에 따라서 리소스 그룹과 거기에 포함된 리소스를 생성합니다.

2.1. ext4 파일 시스템이 있는 LVM 볼륨 설정

이렇게 사용하는 경우 클러스터의 노드 사이에 공유된 스토리지에 있는 LVM 논리 볼륨을 생성하도록 요구됩니다.

다음의 절차에 따라 LVM 논리 볼륨을 생성한 후 그 볼륨에 **ext4** 파일 시스템을 생성합니다. 이 예시에서는 공유 파티션 **/dev/sdb1**이 LVM 물리 볼륨을 저장하는 데 사용되며, 여기에서 LVM 논리 볼륨이 생성됩니다.

참고

클러스터 노드가 사용하는 LVM 볼륨과 이와 연관된 파티션과 장치는 반드시 클러스터 노드에만 연결되어 있어야 합니다.

/dev/sdb1 파티션은 공유된 스토리지이므로, 이 절차는 단 하나의 노드에만 수행합니다.

1. **/dev/sdb1** 파티션에 LVM 물리 볼륨을 생성합니다.

```
# pvcreate /dev/sdb1
  Physical volume "/dev/sdb1" successfully created
```

2. 물리 볼륨 **/dev/sdb1**으로 구성된 볼륨 그룹 **my_vg**를 생성합니다.

```
# vgcreate my_vg /dev/sdb1
  Volume group "my_vg" successfully created
```

3. 볼륨 그룹 **my_vg**를 사용해서 논리 볼륨을 생성합니다.

```
# lvcreate -L450 -n my_lv my_vg
  Rounding up size to full physical extent 452.00 MiB
  Logical volume "my_lv" created
```

lvs 명령을 사용해서 논리 볼륨을 표시할 수 있습니다.

```
# lvs
  LV      VG      Attr      LSize   Pool Origin Data%  Move Log Copy%
Convert
  my_lv   my_vg   -wi-a---- 452.00m
  ...
```

4. 논리 볼륨 **my_lv**에 **ext4** 파일 시스템을 생성합니다.

```
# mkfs.ext4 /dev/my_vg/my_lv
mke2fs 1.42.7 (21-Jan-2013)
Filesystem label=
```

```
OS type: Linux
...
```

2.2. 웹 서버 설정

다음의 절차에 따라 Apache 웹 서버를 설정합니다.

1. Apache HTTPD 서버가 클러스터의 각 노드에 설치되어 있는 것을 확인합니다. 또한 Apache 웹 서버의 상태를 확인할 수 있기 위해서는 **wget** 도구가 클러스터에 설치되어 있어야 합니다.

   각 노드에서 다음의 명령을 실행합니다.

   ```
   # yum install -y httpd wget
   ```

2. Apache 리소스 에이전트가 Apache 웹 서버의 상태를 받기 위해서는 다음의 텍스트가 클러스터의 각 노드에 있는 **/etc/httpd/conf/httpd.conf** 파일에 존재하며 주석으로 처리되지 않도록 해야 합니다. 해당 텍스트가 존재하지 않는다면 파일 마지막에 이 텍스트를 추가합니다.

   ```
   <Location /server-status>
     SetHandler server-status
     Order deny,allow
     Deny from all
     Allow from 127.0.0.1
   </Location>
   ```

3. Apache가 제공할 웹 페이지를 생성합니다. 클러스터에 있는 하나의 노드에서 사용자가 2.1절. “ext4 파일 시스템이 있는 LVM 볼륨 설정”에서 생성한 파일 시스템을 마운트하고 그 파일 시스템에 **index.html** 파일을 생성한 후 파일 시스템 마운트를 해제합니다.

   ```
   # mount /dev/my_vg/my_lv /var/www/
   # mkdir /var/www/html
   # mkdir /var/www/cgi-bin
   # mkdir /var/www/error
   # restorecon -R /var/www
   # cat <<-END >/var/www/html/index.html
   <html>
   <body>Hello</body>
   </html>
   END
   # umount /var/www
   ```

2.3. 클러스터에서 볼륨 그룹의 단독 활성화

다음의 절차에 따라 볼륨 그룹을 설정하면 오직 클러스터만이 볼륨 그룹을 활성화시킬 수 있고 시작 시 볼륨 그룹이 클러스터 밖에서 활성화되지 않도록 설정됩니다. 만약 볼륨 그룹이 클러스터 밖에 있는 시스템에 의해 활성화되면 볼륨 그룹의 메타데이터가 손상될 수 있는 위험이 있습니다.

다음의 절차에 따라 **/etc/lvm/lvm.conf** 설정 파일에 있는 **volume_list** 항목을 수정합니다. **volume_list** 목록에 있는 볼륨 그룹은 클러스터 관리자의 제어 범위를 벗어난 로컬 노드에서 자동으로 활성화되는 게 허용됩니다. 그 노드의 로컬 루트와 홈 디렉토리에 관련된 볼륨 그룹이 이 목록에 포함되어야 합니다. 클러스터 관리자가 관리하는 모든 볼륨 그룹은 **volume_list** 항목에서 제외되어야 합니다. 이 절차는 **clvmd**의 사용을 필요로 하지 않습니다.

클러스터의 각 노드에 다음과 같은 절차를 수행합니다.

1. **locking_type**이 1로 설정되어 있고 **/etc/lvm/lvm.conf** 파일에서 **use_lvmetad**가 0으로 설정되어 있도록 확실히 하기 위해 다음의 명령을 실행합니다. 이 명령은 모든 **lvmetad** 프로세스를 즉시 비활성화 및 중지합니다.

```
# lvmconf --enable-halvm --services --startstopservices
```

2. 다음의 명령을 사용해서 어느 볼륨 그룹이 현재 사용자의 로컬 스토리지에 설정되어 있는지 확인합니다. 이 명령을 통해 현재 설정되어 있는 볼륨 그룹 목록이 표시됩니다. 이 노드에 루트와 홈 디렉토리를 위해 각각의 볼륨 그룹에 공간을 할당해 놓았다면 다음의 예시에서와 같이 표시된 목록에서 그 볼륨을 볼 수 있습니다.

```
# vgs --noheadings -o vg_name
  my_vg
  rhel_home
  rhel_root
```

3. **my_vg** (사용자가 방금 클러스터를 위해 정의한 볼륨 그룹) 이외의 볼륨 그룹을 **/etc/lvm/lvm.conf** 설정 파일에 항목으로 추가합니다. 예를 들어, 각각의 볼륨 그룹에 루트와 홈 디렉토리를 위한 공간을 할당해 놓았다면 **lvm.conf** 파일의 **volume_list** 행을 주석 해제 처리하고 그 볼륨 그룹을 다음과 같이 **volume_list**에 항목으로 추가합니다.

```
volume_list = [ "rhel_root", "rhel_home" ]
```

참고

클러스터 관리자 밖에서 활성화되는 노드에 로컬 볼륨 그룹이 하나도 없다 하더라도 사용자는 **volume_list** 항목을 **volume_list = []**(으)로 초기화해야 합니다.

4. 부트 이미지가 클러스터에 의해 제어되는 볼륨 그룹을 활성화하려고 시도하지 않도록 확실히 하기 위해 **initramfs** 부트 이미지를 재구축합니다. 다음의 명령을 사용해서 **initramfs** 장치를 업데이트합니다. 이 명령이 완료되려면 1분 정도 소요될 것입니다.

```
# dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
```

5. 노드를 재부팅합니다.

참고

사용자가 부트 이미지를 생성한 노드를 부팅한 후 새로운 Linux 커널을 설치했다면 새로운 **initrd** 이미지는 사용자가 노드를 재부팅했을 때 실행중인 새로운 커널이 아니라 사용자가 부트 이미지를 생성했을 때 실행중이었던 커널을 위한 것입니다. 올바른 **initrd** 장치를 사용중인지 확인하려면 재부팅 전과 후에 **uname -r** 명령을 실행해서 실행중인 커널 릴리즈를 확인합니다. 릴리즈가 동일하지 않다면 새로운 커널로 재부팅하고 나서 **initrd** 파일을 업데이트한 후 노드를 재부팅합니다.

6. 노드가 재부팅된 후 그 노드에 클러스터 서비스가 다시 시작되었는지 확인하기 위해 그 노드에서 **pcs cluster status** 명령을 실행합니다. 이를 통해 **Error: cluster is not currently running on this node**라는 메시지가 표시된다면 다음의 명령을 실행합니다.

```
# pcs cluster start
```

다른 방법으로, 클러스터의 각 노드가 재부팅될 때까지 기다린 후에 다음의 명령을 실행해서 각 노드에 클러스터 서비스를 시작합니다.

```
# pcs cluster start --all
```

2.4. pcs 명령을 사용한 리소스 및 리소스 그룹 생성

이렇게 사용하는 경우 사용자가 네 개의 클러스터 리소스를 생성하도록 요구됩니다. 이 리소스가 모두 같은 노드에서 실행되도록 하기 위해 **apachegroup** 리소스 그룹의 일부로 설정됩니다. 사용자가 생성할 리소스 목록은 다음과 같이 리소스가 시작하는 순서대로 정리되어 있습니다.

1. 사용자가 2.1절. “ext4 파일 시스템이 있는 LVM 볼륨 설정”에서 생성한 LVM 볼륨 그룹을 사용하는 **my_lvm**이라는 이름의 **LVM** 리소스

2. 사용자가 2.1절. “ext4 파일 시스템이 있는 LVM 볼륨 설정”에서 생성한 파일 시스템 장치인 **/dev/my_vg/my_lv**를 사용하는 **my_fs**라는 이름의 **Filesystem** 리소스

3. **apachegroup** 리소스 그룹을 위한 부동 IP 주소인 **IPaddr2** 리소스. 해당 IP 주소는 이미 연결된 물리적 노드가 없어야 합니다. **IPaddr2** 리소스의 NIC 장치가 지정되지 않은 경우 해당 부동 IP는 반드시 클러스터 노드에서 사용되는 정적으로 할당된 IP 주소와 같은 네트워크에 존재해야 하며, 그렇지 않을 경우 부동 IP 주소를 할당할 NIC 장치가 제대로 인식되지 않습니다.

4. 사용자가 2.2절. “웹 서버 설정”에서 정의한 **index.html** 파일과 Apache 설정을 사용하는 **Website**라는 이름의 **apache** 리소스

다음의 절차에 따라 **apachegroup** 리소스 그룹과 그 그룹에 포함되는 리소스를 생성합니다. 사용자가 리소스를 그룹에 추가한 순서대로 리소스가 시작되며, 그룹에 추가된 역순으로 중지됩니다. 클러스터의 단 하나의 노드에서만 이 절차를 실행합니다.

1. 다음의 명령을 사용해서 **my_lvm**이라는 LVM 리소스를 생성합니다. 오직 클러스터만이 LVM 논리 볼륨을 활성화할 수 있도록 하기 위해 이 명령은 **exclusive=true** 매개 변수를 지정합니다. 아직 **apachegroup** 리소스 그룹이 존재하지 않으므로 이 명령은 리소스 그룹을 생성합니다.

```
[root@z1 ~]# pcs resource create my_lvm LVM volgrpname=my_vg \
exclusive=true --group apachegroup
```

사용자가 리소스를 생성했을 때 그 리소스는 자동적으로 시작됩니다. 리소스가 생성 및 시작되었다는 것을 확인하기 위해 다음의 명령을 사용할 수 있습니다.

```
# pcs resource show
 Resource Group: apachegroup
     my_lvm (ocf::heartbeat:LVM): Started
```

사용자는 **pcs resource disable** 및 **pcs resource enable** 명령을 사용해서 개별 리소스를 수동으로 중지하고 시작할 수 있습니다.

2. 다음의 명령을 사용해서 설정을 위해 남은 리소스를 생성하고 이 리소스를 기존의 **apachegroup** 리소스 그룹에 추가합니다.

```
[root@z1 ~]# pcs resource create my_fs Filesystem \
device="/dev/my_vg/my_lv" directory="/var/www" fstype="ext4" --group \
apachegroup

[root@z1 ~]# pcs resource create VirtualIP IPaddr2 ip=198.51.100.3 \
cidr_netmask=24 --group apachegroup

[root@z1 ~]# pcs resource create Website apache \
configfile="/etc/httpd/conf/httpd.conf" \
statusurl="http://127.0.0.1/server-status" --group apachegroup
```

3. 리소스와 그 리소스를 포함하는 리소스 그룹을 생성한 후 사용자는 클러스터의 상태를 확인할 수 있습니다. 4개의 리소스가 모두 동일한 노드에서 실행중인 것을 확인합니다.

```
[root@z1 ~]# pcs status
Cluster name: my_cluster
Last updated: Wed Jul 31 16:38:51 2013
Last change: Wed Jul 31 16:42:14 2013 via crm_attribute on
z1.example.com
Stack: corosync
Current DC: z2.example.com (2) - partition with quorum
Version: 1.1.10-5.el7-9abe687
2 Nodes configured
6 Resources configured


Online: [ z1.example.com z2.example.com ]

Full list of resources:
 myapc (stonith:fence_apc_snmp): Started z1.example.com
 Resource Group: apachegroup
     my_lvm (ocf::heartbeat:LVM): Started z1.example.com
     my_fs (ocf::heartbeat:Filesystem): Started z1.example.com
     VirtualIP (ocf::heartbeat:IPaddr2): Started z1.example.com
     Website (ocf::heartbeat:apache): Started z1.example.com
```

사용자의 클러스터를 위한 차단 장치를 1.3절. “차단 장치 설정”에 나와 있는 것처럼 설정하지 않았다면 기본값으로 리소스가 시작되지 않습니다.

4. 클러스터가 시작 및 실행중이면 사용자가 **IPaddr2** 리소스로 정의한 IP 주소를 브라우저에 입력해서 디스플레이 예시를 볼 수 있으며, 이 디스플레이는 "Hello"라는 간단한 단어로 구성되어 있습니다.

```
Hello
```

사용자가 설정한 리소스가 실행되고 있지 않다면, 그 리소스의 설정을 테스트하기 위해 **pcs resource debug-start *resource*** 명령을 실행할 수 있습니다. **pcs resource debug-start** 명령에 대한 자세한 내용은 *High Availability Add-On Reference* 매뉴얼을 보세요.

2.5. 리소스 설정 테스트

2.4절. “pcs 명령을 사용한 리소스 및 리소스 그룹 생성”에서 표시되는 클러스터 상태 디스플레이에서는 모든 리소스가 **z1.example.com** 노드에서 실행중입니다. 사용자는 다음의 절차를 통해 첫 노드를 **standby** 모드로 놓아서 해당 노드가 더 이상 리소스를 호스트할 수 없게 만들어서 리소스 그룹이 **z2.example.com** 노드로 페일오버하는지 테스트할 수 있습니다.

1. 다음의 명령을 사용해서 **z1.example.com** 노드를 **standby** 모드로 놓습니다.

```
root@z1 ~]# pcs cluster standby z1.example.com
```

2. **z1** 노드를 대기 모드로 놓은 후 클러스터 상태를 확인합니다. 이제 모든 리소스가 **z2**에서 실행중이어야 합니다.

```
[root@z1 ~]# pcs status
Cluster name: my_cluster
Last updated: Wed Jul 31 17:16:17 2013
Last change: Wed Jul 31 17:18:34 2013 via crm_attribute on
z1.example.com
Stack: corosync
Current DC: z2.example.com (2) - partition with quorum
Version: 1.1.10-5.el7-9abe687
2 Nodes configured
6 Resources configured


Node z1.example.com (1): standby
Online: [ z2.example.com ]

Full list of resources:

 myapc (stonith:fence_apc_snmp): Started z1.example.com
 Resource Group: apachegroup
     my_lvm (ocf::heartbeat:LVM): Started z2.example.com
     my_fs (ocf::heartbeat:Filesystem): Started z2.example.com
     VirtualIP (ocf::heartbeat:IPaddr2): Started z2.example.com
     Website (ocf::heartbeat:apache): Started z2.example.com
```

정의된 IP 주소에 있는 웹 사이트는 여전히 중단 없이 표시되어야 합니다.

3. **z1**을 **standby** 모드에서 제거하려면 다음의 명령을 실행합니다.

```
root@z1 ~]# pcs cluster unstandby z1.example.com
```

참고

standby 모드에서 어떤 노드를 제거하는 것만으로 리소스가 해당 노드로 페일오버하게 만들지는 않습니다. 리소스가 실행되는 노드를 제어하는 데 대한 자세한 정보는 *Red Hat High Availability Add-On Reference*에 있는 클러스터 리소스 설정에 대한 설명을 확인합니다.

3장. Red Hat 고가용성 클러스터에 있는 활성/수동 NFS 서버

다음 부분에서는 공유 스토리지를 사용해서 2 노드 Red Hat Enterprise Linux 고가용성 추가 기능 클러스터에 고가용성 활성/수동 NFS 서버를 설정하는 방법에 대해 설명합니다. 이 절차는 **pcs**를 사용해서 페이스메이커(Pacemaker) 클러스터 리소스를 설정합니다. 이렇게 사용되는 경우 클라이언트가 부동 IP 주소를 통해 NFS 파일 시스템에 액세스합니다. NFS 서버는 클러스터에 있는 두 노드 중 하나에서 실행됩니다. NFS 서버가 실행되고 있는 노드가 실행 불가능하게 될 경우, NFS 서버가 클러스터에 있는 두번째 노드에 다시 시작되면서 서비스 장애를 최소화합니다.

이렇게 사용하려면 다음과 같은 구성 요소가 시스템에 설치되어야 합니다:

- Apache 웹 서버를 실행하는 클러스터를 만드는 데 사용될 두 개의 노드. 예를 들어, **z1.example.com** 그리고 **z2.example.com**인 두 개의 노드가 사용됩니다.
- webfarm 클러스터의 각 노드별 전원 차단(power fencing) 장치. 예를 들어, **zapc.example.com** 호스트명을 갖고 있는 APC 전원 스위치의 포트를 두개 사용합니다.
- NFS 서버에 요구되는 공개된 가상 IP 주소
- iSCSI 또는 파이버(Fibre) 채널을 사용하는 클러스터의 노드를 위한 공유 스토리지

2 노드 Red Hat Enterprise Linux에 고가용성 활성/수동 NFS 서버를 설정하기 위해서는 다음의 단계를 수행해야 합니다.

1. 3.1절. “NFS 클러스터 생성”에 있는 설명에 따라서 NFS 서버를 실행하는 클러스터를 생성하고 클러스터에 있는 각 노드에 대한 차단을 설정합니다.
2. 3.2절. “ext4 파일 시스템이 있는 LVM 볼륨 설정”에 있는 설명에 따라서 클러스터 노드를 위한 공유 스토리지에 있는 LVM 논리 볼륨 **my_lv**에 마운트된 **ext4** 파일 시스템을 설정합니다.
3. 3.3절. “NFS 공유 설정”에 있는 설명에 따라서 LVM 논리 볼륨에 있는 공유 스토리지에 NFS 공유를 설정합니다.
4. 3.4절. “클러스터에서 볼륨 그룹의 단독 활성화”에 있는 설명에 따라서 오직 그 클러스터만이 논리 볼륨 **my_lv**을 포함하는 LVM 볼륨 그룹을 활성화할 수 있으며 그 볼륨 그룹은 시작 시 클러스터 밖에서 활성화되지 않도록 확인합니다.
5. 3.5절. “클러스터 리소스 설정”에 있는 설명에 따라서 클러스터 리소스를 생성합니다.
6. 3.6절. “리소스 설정 테스트”에 있는 설명에 따라서 사용자가 설정한 NFS 서버를 테스트합니다.

3.1. NFS 클러스터 생성

다음의 절차에 따라 NFS 클러스터를 설치하고 생성합니다.

1. 1.1절. “클러스터 소프트웨어 설치”에 설명된 절차에 따라 **z1.example.com**과 **z2.example.com** 노드에 클러스터 소프트웨어를 설치합니다.
2. 1.2절. “클러스터 생성”에 설명된 절차에 따라 **z1.example.com**과 **z2.example.com**으로 구성된 2 노드 웹팜을 생성합니다. 이렇게 사용하는 경우 절차에 있는 예시처럼 클러스터를 **my_cluster**라고 이름 짓습니다.
3. 1.3절. “차단 장치 설정”에 설명된 절차에 따라 웹팜 클러스터의 각 노드에 펜싱 장치를 설정합니다. 절차에 있는 예시에서는 **zapc.example.com**이라는 호스트명을 갖고 있는 APC 전원 스위치의 두 포트를 사용해서 차단을 설정합니다.

3.2. ext4 파일 시스템이 있는 LVM 볼륨 설정

3.2. ext4 파일 시스템이 있는 LVM 볼륨 생성

이렇게 사용하는 경우 클러스터의 노드 사이에 공유된 스토리지에 있는 LVM 논리 볼륨을 생성하도록 요구됩니다.

다음의 절차에 따라 LVM 논리 볼륨을 생성한 후 그 볼륨에 **ext4** 파일 시스템을 생성합니다. 다음의 예시에서는 공유 파티션 **/dev/sdb1**이 LVM 물리 볼륨을 저장하는 데 사용되며, 여기에서 LVM 논리 볼륨이 생성됩니다.

참고

클러스터 노드가 사용하는 LVM 볼륨과 이와 연관된 파티션과 장치는 반드시 클러스터 노드에만 연결되어 있어야 합니다.

/dev/sdb1 파티션이 공유된 스토리지이므로, 이 절차는 단 하나의 노드에만 수행합니다.

1. **/dev/sdb1** 파티션에 LVM 물리 볼륨을 생성합니다.

```
[root@z1 ~]# pvcreate /dev/sdb1
  Physical volume "/dev/sdb1" successfully created
```

2. 물리 볼륨 **/dev/sdb1**으로 구성된 볼륨 그룹 **my_vg**를 생성합니다.

```
[root@z1 ~]# vgcreate my_vg /dev/sdb1
  Volume group "my_vg" successfully created
```

3. 볼륨 그룹 **my_vg**를 사용해서 논리 볼륨을 생성합니다.

```
[root@z1 ~]# lvcreate -L450 -n my_lv my_vg
  Rounding up size to full physical extent 452.00 MiB
  Logical volume "my_lv" created
```

lvs 명령을 사용해서 논리 볼륨을 표시할 수 있습니다.

```
[root@z1 ~]# lvs
  LV      VG      Attr      LSize   Pool Origin Data%  Move Log Copy%
Convert
  my_lv   my_vg   -wi-a---- 452.00m
  ...
```

4. 논리 볼륨 **my_lv**에 **ext4** 파일 시스템을 생성합니다.

```
[root@z1 ~]# mkfs.ext4 /dev/my_vg/my_lv
mke2fs 1.42.7 (21-Jan-2013)
Filesystem label=
OS type: Linux
...
```

3.3. NFS 공유 설정

다음의 절차에 따라 데몬 페일오버(daemon failover)를 위한 NFS 공유를 설정합니다. 이 절차는 클러스터에 있는 단 하나의 노드에만 수행합니다.

1. **/nfsshare** 디렉토리를 생성합니다.

```
[root@z1 ~]# mkdir /nfsshare
```

2. 3.2절. “ext4 파일 시스템이 있는 LVM 볼륨 설정”에서 사용자가 생성한 **ext4** 파일 시스템을 **/nfsshare** 디렉토리에 마운트합니다.

```
[root@z1 ~]# mount /dev/my_vg/my_lv /nfsshare
```

3. **exports** 디렉토리 트리(tree)를 **/nfsshare** 디렉토리에 생성합니다.

```
[root@z1 ~]# mkdir -p /nfsshare/exports
[root@z1 ~]# mkdir -p /nfsshare/exports/export1
[root@z1 ~]# mkdir -p /nfsshare/exports/export2
```

4. NFS 클라이언트가 액세스할 수 있도록 파일을 **exports** 디렉토리에 넣습니다. 다음의 예시에서는 **clientdatafile1**과 **clientdatafile2**라는 이름의 테스트 파일을 생성합니다.

```
[root@z1 ~]# touch /nfsshare/exports/export1/clientdatafile1
[root@z1 ~]# touch /nfsshare/exports/export2/clientdatafile2
```

5. ext4 파일 시스템의 마운트를 해제하고 LVM 볼륨 그룹을 비활성화합니다.

```
[root@z1 ~]# umount /dev/my_vg/my_lv
[root@z1 ~]# vgchange -an my_vg
```

3.4. 클러스터에서 볼륨 그룹의 단독 활성화

다음의 절차에 따라 LVM 볼륨 그룹을 설정하면 오직 클러스터만이 볼륨 그룹을 활성화시킬 수 있고 시작 시 클러스터 밖에서 볼륨 그룹이 활성화되지 않도록 설정됩니다. 만약 볼륨 그룹이 클러스터 밖에 있는 시스템에 의해 활성화되면 볼륨 그룹의 메타데이터가 손상될 수 있는 위험이 있습니다.

이 절차는 **/etc/lvm/lvm.conf** 설정 파일에 있는 **volume_list** 항목을 수정합니다. **volume_list** 목록에 있는 볼륨 그룹은 클러스터 관리자의 제어 범위를 벗어난 로컬 노드에서 자동으로 활성화되는 게 허용됩니다. 그 노드의 로컬 루트와 홈 디렉토리에 관련된 볼륨 그룹이 이 목록에 포함되어야 합니다. 클러스터 관리자가 관리하는 모든 볼륨 그룹은 **volume_list** 항목에서 제외되어야 합니다. 이 절차는 **clvmd**의 사용을 필요로 하지 않습니다.

클러스터의 각 노드에 다음과 같은 절차를 수행합니다.

1. **locking_type**이 1로 설정되어 있고 **/etc/lvm/lvm.conf** 파일에서 **use_lvmetad**가 0으로 설정되어 있도록 확실히 하기 위해 다음의 명령을 실행합니다. 이 명령은 모든 **lvmetad** 프로세스를 즉시 비활성화 및 중지합니다.

```
# lvmconf --enable-halvm --services --startstopservices
```

2. 다음의 명령을 사용해서 어느 볼륨 그룹이 현재 사용자의 로컬 스토리지에 설정되어 있는지 확인합니다. 이 명령을 통해 현재 설정되어 있는 볼륨 그룹 목록이 표시됩니다. 이 노드에 루트와 홈 디렉토리를 위해 각각의 볼륨 그룹에 공간을 할당해 놓았다면 다음의 예시에서와 같이 그 볼륨을 표시된 목록에서 볼 수 있습니다.

```
# vgs --noheadings -o vg_name
  my_vg
  rhel_home
  rhel_root
```

3. **my_vg** (사용자가 방금 클러스터를 위해 정의한 볼륨 그룹) 이외의 볼륨 그룹을 **/etc/lvm/lvm.conf** 설정 파일에 항목으로 추가합니다. 예를 들어, 각각의 볼륨 그룹에 루트와 홈 디렉토리를 위한 공간을 할당해 놓았다면 **lvm.conf** 파일의 **volume_list** 행을 주석 해제 처리하고 그 볼륨 그룹을 다음과 같이 **volume_list**에 항목으로 추가합니다.

```
volume_list = [ "rhel_root", "rhel_home" ]
```

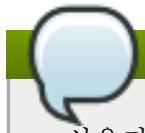
참고

클러스터 관리자 밖에서 활성화되는 노드에 로컬 볼륨 그룹이 하나도 없다 하더라도 사용자는 **volume_list** 항목을 **volume_list = []**로 초기화해야 합니다.

4. 부트 이미지가 클러스터에 의해 제어되는 볼륨 그룹을 활성화하려고 시도하지 않도록 보장하기 위해 **initramfs** 부트 이미지를 재구축합니다. 다음의 명령을 사용해서 **initramfs** 장치를 업데이트합니다. 이 명령이 완료되려면 1분 정도 소요될 것입니다.

```
# dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
```

5. 노드를 재부팅합니다.



참고

사용자가 부트 이미지를 생성한 노드를 부팅한 후 새로운 Linux 커널을 설치했다면 새로운 **initrd** 이미지는 사용자가 노드를 재부팅했을 때 실행중인 새로운 커널이 아니라 사용자가 부트 이미지를 생성했을 때 실행중이었던 커널을 위한 것입니다. 올바른 **initrd** 장치를 사용 중인지 확인하려면 재부팅 전과 후에 **uname -r** 명령을 실행해서 실행중인 커널 릴리즈를 확인합니다. 릴리즈가 동일하지 않다면 새로운 커널로 재부팅하고 나서 **initrd** 파일을 업데이트한 후 노드를 재부팅합니다.

6. 노드가 재부팅된 후 그 노드에 클러스터 서비스가 다시 시작되었는지 확인하기 위해 그 노드에서 **pcs cluster status** 명령을 실행합니다. 이를 통해 **Error: cluster is not currently running on this node**라는 메시지가 표시된다면 다음의 명령을 실행합니다.

```
# pcs cluster start
```

다른 방법으로, 클러스터의 각 노드가 재부팅될 때까지 기다린 후에 다음의 명령을 실행해서 클러스터의 모든 노드에 클러스터 서비스를 시작합니다.

```
# pcs cluster start --all
```

3.5. 클러스터 리소스 설정

다음 부분에서는 아래와 같은 용도로 클러스터 리소스를 설정하는 절차가 설명됩니다.

참고

pcs resource create를 사용해서 클러스터 리소스를 생성하면 그 리소스가 아직 실행중인지 확인하기 위해 즉시 **pcs status** 명령을 실행하는 것이 권장됩니다. 사용자의 클러스터를 위한 차단 장치를 1.3절. “차단 장치 설정”에 나와 있는 것처럼 설정하지 않았다면 기본값으로 리소스가 시작되지 않습니다.

사용자가 설정한 리소스가 실행되고 있지 않다면, 그 리소스의 설정을 테스트하기 위해 **pcs resource debug-start *resource*** 명령을 실행할 수 있습니다. 이렇게 하면 클러스터의 제어와 인식 밖에서 서비스가 시작됩니다. 설정된 리소스가 다시 실행되는 시점에 **pcs cluster cleanup *resource*** 를 실행해서 클러스터가 업데이트에 대해 알고 있도록 합니다. **pcs resource debug-start** 명령에 대한 자세한 내용은 *High Availability Add-On Reference* 설명을 참조하십시오.

다음의 절차에 따라 시스템 리소스를 설정합니다. 이 리소스가 모두 같은 노드에서 실행되도록 하기 위해 **nfsgroup** 리소스 그룹의 일부로 설정됩니다. 사용자가 리소스를 그룹에 추가한 순서대로 리소스가 시작되며, 그룹에 추가된 역순으로 중지됩니다. 클러스터의 단 하나의 노드에서만 이 절차를 실행합니다.

1. 다음의 명령을 사용해서 **my_lvm**이라는 이름의 LVM 리소스를 생성합니다. 오직 클러스터만이 LVM 논리 볼륨을 활성화할 수 있도록 하기 위해 이 명령은 **exclusive=true** 매개 변수를 지정합니다. 아직 **nfsgroup** 리소스 그룹이 존재하지 않으므로 이 명령은 리소스 그룹을 생성합니다.

```
[root@z1 ~]# pcs resource create my_lvm LVM volgrpname=my_vg \
exclusive=true --group nfsgroup
```

   리소스가 실행중인지 검증하기 위해 클러스터 상태를 확인합니다.

```
root@z1 ~]#  pcs status
Cluster name: my_cluster
Last updated: Thu Jan  8 11:13:17 2015
Last change: Thu Jan  8 11:13:08 2015
Stack: corosync
Current DC: z2.example.com (2) - partition with quorum
Version: 1.1.12-a14efad
2 Nodes configured
3 Resources configured

Online: [ z1.example.com z2.example.com ]

Full list of resources:
 myapc  (stonith:fence_apc_snmp):       Started z1.example.com
 Resource Group: nfsgroup
     my_lvm     (ocf::heartbeat:LVM):   Started z1.example.com

PCSD Status:
  z1.example.com: Online
  z2.example.com: Online
```

```
Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

2. 클러스터를 위한 **Filesystem** 리소스를 설정합니다.

참고

options=*options* 매개 변수를 가지고 있는 **Filesystem** 리소스를 위한 리소스 설정의 일부로써 마운트 옵션을 지정할 수 있습니다. 전체 설정 옵션을 표시하려면 **pcs resource describe Filesystem** 명령을 실행합니다.

다음의 명령을 사용해서 **nfsgroup** 리소스 그룹의 일부로써 **nfsshare**라는 이름의 ext4 **Filesystem** 리소스를 설정합니다. 이 파일 시스템은 사용자가 3.2절. “ext4 파일 시스템이 있는 LVM 볼륨 설정”에서 생성한 LVM 볼륨 그룹과 ext4 파일 시스템을 사용하며 사용자가 3.3절. “NFS 공유 설정”에서 생성한 **/nfsshare** 디렉토리에 마운트됩니다.

```
[root@z1 ~]# pcs resource create nfsshare Filesystem \
device=/dev/my_vg/my_lv directory=/nfsshare \
fstype=ext4 --group nfsgroup
```

my_lvm 및 **nfsshare** 리소스가 실행중인지 확인합니다.

```
[root@z1 ~]# pcs status
...
Full list of resources:
 myapc  (stonith:fence_apc_snmp):       Started z1.example.com
 Resource Group: nfsgroup
     my_lvm     (ocf::heartbeat:LVM):   Started z1.example.com
     nfsshare   (ocf::heartbeat:Filesystem):    Started z1.example.com
...
```

3. **nfsgroup** 리소스 그룹의 일부로써 **nfs-daemon**이라는 이름의 **nfsserver** 리소스를 생성합니다.

```
[root@z1 ~]# pcs resource create nfs-daemon nfsserver \
nfs_shared_infodir=/nfsshare/nfsinfo nfs_no_notify=true \
--group nfsgroup
[root@z1 ~]# pcs status
...
```

4. **/nfsshare/exports** 디렉토리를 내보내기 위해 **exportfs** 리소스를 추가합니다. 이 리소스는 **nfsgroup** 리소스 그룹의 일부입니다. 이것은 NFSv4 클라이언트를 위한 가상 디렉토리를 구축합니다. NFSv3 클라이언트도 이렇게 내보낸 디렉토리에 액세스할 수 있습니다.

```
[root@z1 ~]# pcs resource create nfs-root exportfs \
clientspec=192.168.122.0/255.255.255.0 \
options=rw,sync,no_root_squash \
directory=/nfsshare/exports \
fsid=0 --group nfsgroup

[root@z1 ~]# # pcs resource create nfs-export1 exportfs \
```

```
clientspec=192.168.122.0/255.255.255.0 \
options=rw,sync,no_root_squash directory=/nfsshare/exports/export1 \
fsid=1 --group nfsgroup

[root@z1 ~]# # pcs resource create nfs-export2 exportfs \
clientspec=192.168.122.0/255.255.255.0 \
options=rw,sync,no_root_squash directory=/nfsshare/exports/export2 \
fsid=2 --group nfsgroup
```

5. nfs 클라이언트가 nfs 공유에 액세스하기 위해 사용할 부동 IP 주소를 추가합니다. 사용자가 지정하는 부동 IP 주소는 DNS 역방향 검색이 요구되거나 또는 클러스터에 있는 모든 노드의 **/etc/hosts**에 지정되어 있어야 합니다. 이 리소스는 **nfsgroup** 리소스 그룹의 일부입니다. 다음의 배포 예시에서는 192.168.122.200을 부동 IP 주소로 사용하고 있습니다.

```
[root@z1 ~]# pcs resource create nfs_ip IPaddr2 \
ip=192.168.122.200 cidr_netmask=24 --group nfsgroup
```

6. 전체 NFS 배포가 초기화된 후 NFSv3 재부팅 알림을 보내기 위한 **nfsnotify** 리소스를 추가합니다.

참고

NFS 알림이 올바르게 처리되려면 부동 IP 주소와 연결된 호스트명이 반드시 있어야 하며, 이 호스트명은 nfs 서버와 nfs 클라이언트 모두에서 일관되어야 합니다.

```
[root@z1 ~]# pcs resource create nfs-notify nfsnotify \
source_host=192.168.122.200
```

리소스와 리소스 제한을 생성한 후 클러스터의 상태를 확인할 수 있습니다. 모든 리소스가 동일한 노드에서 실행중인 것을 확인합니다.

```
[root@z1 ~]# pcs status
...
Full list of resources:
 myapc  (stonith:fence_apc_snmp):       Started z1.example.com
 Resource Group: nfsgroup
     my_lvm     (ocf::heartbeat:LVM):   Started z1.example.com
     nfsshare   (ocf::heartbeat:Filesystem):    Started z1.example.com
     nfs-daemon (ocf::heartbeat:nfsserver):     Started z1.example.com
     nfs-root   (ocf::heartbeat:exportfs):      Started z1.example.com
     nfs-export1        (ocf::heartbeat:exportfs):      Started
z1.example.com
     nfs-export2        (ocf::heartbeat:exportfs):      Started
z1.example.com
     nfs_ip     (ocf::heartbeat:IPaddr2):       Started  z1.example.com
     nfs-notify (ocf::heartbeat:nfsnotify):     Started z1.example.com
...
```

3.6. 리소스 설정 테스트

다음의 절차를 사용해서 사용자의 시스템 설정을 검증할 수 있습니다. 사용자는 NFSv3나 NFSv4를 사용해서 내보낸 파일 시스템을 마운트할 수 있습니다.

1. 클러스터 밖에 있으며 배포된 것과 동일한 네트워크에 있는 노드에서 NFS 공유가 표시되는지 검증하기 위해 NFS 공유를 마운트합니다. 다음의 예시에서는 192.168.122.0/24 네트워크를 사용하고 있습니다.

```
# showmount -e 192.168.122.200
Export list for 192.168.122.200:
/nfsshare/exports/export1 192.168.122.0/255.255.255.0
/nfsshare/exports         192.168.122.0/255.255.255.0
/nfsshare/exports/export2 192.168.122.0/255.255.255.0
```

2. NFSv4에 NFS 공유를 마운트할 수 있는지 검증하려면 클라이언트 노드에 있는 디렉토리에 NFS 공유를 마운트합니다. 마운트 후 내보내기 디렉토리의 내용이 보이는지 확인합니다. 테스트 후에 공유 마운트를 해제합니다.

```
# mkdir nfsshare
# mount -o "vers=4" 192.168.122.200:export1 nfsshare
# ls nfsshare
clientdatafile1
# umount nfsshare
```

3. NFSv3에 NFS 공유를 마운트할 수 있는지 검증합니다. 마운트 후 **clientdatafile1** 테스트 파일이 보이는지 확인합니다. NFSv4와 달리 NFSV3는 가상 파일 시스템을 사용하지 않기 때문에 사용자가 특정 내보내기를 마운트해야 합니다. 테스트 후에 마운트를 해제합니다.

```
# mkdir nfsshare
# mount -o "vers=3" 192.168.122.200:/nfsshare/exports/export2 nfsshare
# ls nfsshare
    clientdatafile2
# umount nfsshare
```

4. 페일오버를 테스트하기 위해 다음의 단계를 실행합니다.

    a. 클러스터 밖의 노드에 nfs 공유를 마운트하고 사용자가 3.3절. “NFS 공유 설정”에서 생성한 **clientdatafile1**에 액세스할 수 있는지 확인합니다.

```
# mkdir nfsshare
# mount -o "vers=4" 192.168.122.200:export1 nfsshare
# ls nfsshare
clientdatafile1
```

    b. 클러스터의 어느 노드에서 **nfsgroup**이 실행중인지 클러스터 내의 노드에서 확인합니다. 다음의 예시에서 **nfsgroup**은 **z1.example.com**에서 실행중입니다.

```
[root@z1 ~]# pcs status
...
Full list of resources:
 myapc  (stonith:fence_apc_snmp):       Started z1.example.com
 Resource Group: nfsgroup
     my_lvm     (ocf::heartbeat:LVM):   Started z1.example.com
     nfsshare   (ocf::heartbeat:Filesystem):    Started
z1.example.com
     nfs-daemon (ocf::heartbeat:nfsserver):     Started
z1.example.com
     nfs-root   (ocf::heartbeat:exportfs):      Started
```

```
z1.example.com
     nfs-export1        (ocf::heartbeat:exportfs):      Started
z1.example.com
     nfs-export2        (ocf::heartbeat:exportfs):      Started
z1.example.com
     nfs_ip     (ocf::heartbeat:IPaddr2):       Started
z1.example.com
     nfs-notify (ocf::heartbeat:nfsnotify):     Started
z1.example.com
...
```

c. 클러스터 내의 노드에서 **nfsgroup**을 실행중인 노드를 대기 모드로 놓습니다.

```
[root@z1 ~]#pcs cluster standby z1.example.com
```

d. 다른 클러스터 노드에서 **nfsgroup**이 성공적으로 시작되는지 확인합니다.

```
[root@z1 ~]# pcs status
...
Full list of resources:
 Resource Group: nfsgroup
     my_lvm     (ocf::heartbeat:LVM):   Started z2.example.com
     nfsshare   (ocf::heartbeat:Filesystem):    Started
z2.example.com
     nfs-daemon (ocf::heartbeat:nfsserver):     Started
z2.example.com
     nfs-root   (ocf::heartbeat:exportfs):      Started
z2.example.com
     nfs-export1        (ocf::heartbeat:exportfs):      Started
z2.example.com
     nfs-export2        (ocf::heartbeat:exportfs):      Started
z2.example.com
     nfs_ip     (ocf::heartbeat:IPaddr2):       Started
z2.example.com
     nfs-notify (ocf::heartbeat:nfsnotify):     Started
z2.example.com
...
```

e. 사용자가 nfs 공유를 마운트한 클러스터 밖의 노드에서 이 외부 노드가 NFS 마운트 내의 테스트 파일에 아직 액세스할 수 있는지 확인합니다.

```
# ls nfsshare
clientdatafile1
```

페일오버가 잠시 진행되는 동안 클라이언트에 대한 서비스가 잠시 끊기지만 클라이언트는 사용자 작업 없이 복구됩니다. 기본적으로 NFSv4를 사용하는 클라이언트는 마운트 복구에 90초 정도 소요됩니다. 이 90초는 시작 시 서버가 준수하는 NFSv4 파일 임대 유예 시간을 나타냅니다. NFSv3 클라이언트는 단 몇초 만에 마운트에 대한 액세스를 회복할 수 있습니다.

f. 클러스터 내의 노드에서 초기에 **nfsgroup**을 대기 모드에서 실행 중이던 노드를 제거합니다. 이 작업 자체가 클러스터 리소스를 이 노드로 다시 옮기지는 않습니다.

```
[root@z1 ~]# pcs cluster unstandby z1.example.com
```

부록 A. 고친 과정

고침 1.2-3.1 **Fri Jul 22 2016** **Red Hat Localization Services**
XML 소스 1.2-3 버전과 번역 파일을 동기화

고침 1.2-3 **Mon Nov 9 2015** **Steven Levine**
7.2 GA 공개 용 문서 준비

고침 1.2-2 **Tue Aug 18 2015** **Steven Levine**
7.2 Beta 공개 용 문서 준비

고침 1.1-19 **Mon Feb 16 2015** **Steven Levine**
7.1 GA 릴리즈 버전

고침 1.1-10 **Thu Dec 11 2014** **Steven Levine**
7.1 Beta 릴리즈 버전

고침 1.1-9 **Tue Dec 9 2014** **Steven Levine**
nfs 클러스터 설정 절차 추가

고침 1.1-6 **Mon Dec 8 2014** **Steven Levine**
로드 밸런서 클러스터 절차 업데이트

고침 1.1-5 **Thu Dec 05 2014** **Steven Levine**
7.1 Beta 릴리즈 버전

고침 0.1-34 **Fri Dec 4 2014** **Steven Levine**
Red Hat Enterprise Linux 스플래시 페이지에 새로운 정렬 순서 도입을 위한 업데이트

고침 0.1-33 **Mon Jun 2 2014** **Steven Levine**
7.0 GA 릴리즈 버전

고침 0.1-31 **Wed May 21 2014** **Steven Levine**
문제 해결: #886235
volume_list 사용 기록

고침 0.1-29 **Tue May 20 2014** **Steven Levine**
스타일 변경과 업데이트된 초안을 위해 다시 빌드

고침 0.1-20 **Wed Apr 9 2014** **Steven Levine**
업데이트된 Beta 초안

고침 0.1-8 **Fri Dec 6 2013** **Steven Levine**
Beta 초안

고침 0.0-1 **Wed Jan 16 2013** **Steven Levine**
첫번째 Red Hat Enterprise Linux 7 버전