고가용성 클러스터링 환경 구축(corosync, pacemaker, DRBD)

구축 목적

: 서버 및 서비스를 클러스터로 묶어서 관리하고 , 그 환경을 고가용성(HA) 환경으로 구축하여 다운타임을 최소화 하기 위하여

서비스 의미

```
★
corosync : 클러스터 내의 "노드 간" 통신 및 동기화 작업 담당
pacemaker : corosync의 기능을 이용하여 "클러스터" 리소스 제어 및 관리 담당 ( 상태에 따른 순차적 노드 서비스 시작 및 정지 )
DRBD : Failover에 따른 파일 시스템 동기화 ( 디스크 미러링 )
pcsd : 클러스터 설정을 간편하게 하기 위해 쓰는 툴
★
```

서버 구축

★웹 이중화 버전★

corosync 및 pacemaker 환경 구축

```
# cat /etc/hosts
172.16.1.6      dw-test
172.16.1.7      dw-test2
172.16.1.8      dw-vip
( 양쪽 노드에, 각 노드의 ip 주소를 등록해준다 )

# yum -y install pacemaker corosync pcs psmisc policycoreutils-python

# systemctl start pcsd
# systemctl enable pcsd

# passwd hacluster
( 양쪽 노드에서, 자동으로 생성된 계정을 양쪽 동일하게 비밀번호를 설정한다. )
dw-test : root
dw-test2 : root

# pcs cluster auth dw-test dw-test2
( 한쪽 노드에서, hacluster 사용자 인증 진행 )

# pcs cluster setup --name dw_cluster dw-test dw-test2
( 한쪽 노드에서, 클러스터를 만들고 두 대의 노드를 동기화시킨다 - corosync 구성 )

# pcs cluster start --all
```

```
( 클러스터 실행 )

# corosync-cfgtool -s
( 클러스터 통신 확인 )

# corosync-cmapctl | egrep -i members
# pcs status corosync
( 멤버쉽과 쿼럼 확인 )

# pcs status
( 클러스터 전체 정보 확인 )

# pcs property set stonith-enabled=false
# crm_verify -L -V
( 클러스터 설정 변경 전 유효성 확인 - stonith 설정 비활성화로 오류 제거 )

# pcs resource create VirtualIP ocf:heartbeat:IPaddr2 ip=172.16.1.8
cidr_netmask=24 op monitor interval=30s
( 가상 아이피 리소스 추가 - 노드가 다운되면 다른 노드로 이동함 )

# pcs status
VirtualIP   (ocf::heartbeat:IPaddr2):   Started dw-test -> 리소스 추가 확인

# ip a
( VIP 및 설정된 IP 주소 들어갔는지 확인 )

# pcs cluster stop dw-test
# pcs status
( 1번 테스트 서버 정지 후 2번으로 정상적으로 넘어가는지 확인 )
 VirtualIP  (ocf::heartbeat:IPaddr2):   Started dw-test2

# yum -y install httpd wget
( 양쪽 노드에, 설치 )

# vi /etc/httpd/conf.d/status.conf
--------------------------------------
 <Location /server-status>
    SetHandler server-status
    Require local
 </Location>
--------------------------------------
( 양쪽 노드에, 생성 및 해당 내용 추가 )

# pcs resource create WebService ocf:heartbeat:apache
configfile=/etc/httpd/conf/httpd.conf statusurl="http://localhost/server-status"
op monitor interval=1min
# pcs resource op defaults timeout=60s
# pcs resource op defaults
( 웹서비스 리소스 등록 및 설정 )

# pcs status
( 웹서비스 리소스 정상적으로 올라왔는지 확인 )

# pcs constraint colocation add WebService with VirtualIP INFINITY
```

```
( 웹서비스와 VIP의 리소스를 같은 노드에서 실행할 수 있도록 묶어줌 )

# pcs constraint order VirtualIP then WebService
( VIP가 먼저 실행된 후에 웹서비스를 실행하도록 설정 추가 )

# pcs constraint location WebService prefers dw-test2=INFINITY
( 리소스를 2번 노드로 강제이동이 필요할 경우에 설정 )

# pcs constraint
( 추가된 constraint 내용 확인 )

# pcs constraint --full
Disabled on: wolf2 (score:-INFINITY) (id:location-WebService-wolf2--INFINITY)
# pcs constraint remove location-WebService-wolf2--INFINITY
( 리소스 삭제가 필요할땐 --full로 id를 확인 후에 remove [id] 로 삭제 )
```

DRBD 환경 구축

```
# rpm --import https://www.elrepo.org/RPM-GPG-KEY-elrepo.org
# rpm -ivh http://www.elrepo.org/elrepo-release-7.0-2.el7.elrepo.noarch.rpm
# yum install drbd90-utils
# yum install kmod-drbd90
( 양쪽 모두 설치 , 레포지토리 및 drbd 설치 )

# modprobe drbd
# lsmod | grep drbd
( 양쪽 노드 모두, 커널에 drbd 모듈 로딩 )

** 작업 전 먼저 두개의 서버에 각각 디스크 하나씩 추가 진행!!

# parted -l
( 새로운 디스크 생성 확인 )

# vi /etc/drbd.d/cluster_disk.res
----------------------------------------
resource disk_dw {
        on dw-test {
                device /dev/drbd0;
                disk /dev/xvdb;
                address 172.16.1.6:7789;
                meta-disk internal;
        }
        on dw-test2 {
                device /dev/drbd0;
                disk /dev/xvdb;
                address 172.16.1.7:7789;
                meta-disk internal;
        }
}
----------------------------------------
( 양쪽 노드에, 해당 설정파일 생성 및 내용 확인 후 추가 )
```

```
# vi /etc/drbd.d/global_common.conf
-----------------------------------------
net {
        # protocol timeout max-epoch-size max-buffers
        # connect-int ping-int sndbuf-size rcvbuf-size ko-count
        # allow-two-primaries cram-hmac-alg shared-secret after-sb-0pri
        # after-sb-1pri after-sb-2pri always-asbp rr-conflict
        # ping-timeout data-integrity-alg tcp-cork on-congestion
        # congestion-fill congestion-extents csums-alg verify-alg
        # use-rle
        protocol C;
    }
-----------------------------------------
( net 설정 맨 아래에 protocol C; 추가 )

# drbdadm create-md disk_dw
( 메타디스크 생성 )

# systemctl start drbd
# systemctl enable drbd

# drbdadm status
( 각 서버의 drbd 상태 확인 - Primary 와 Secondary로 나뉨 )

disk_dw role:Primary
  disk:UpToDate
  dw-test2 role:Secondary
    peer-disk:UpToDate
( 1번 서버 )

# drbdadm status
disk_dw role:Secondary
  disk:UpToDate
  dw-test role:Primary
    peer-disk:UpToDate
( 2번 서버 )

# drbdadm primary disk_dw
( Primary를 1번서버로 바꾸어주는 명령어 )

# mkfs -t xfs /dev/drbd0
( 파일시스템 생성 )

# mount /dev/drbd0 /mnt
( 원하는 위치에 마운트 )
# umount /mnt


# pcs cluster cib drbd_cfg
( drbd 리소스를 클러스터에 통합 - cib로부터 raw xml 파일을 생성 )

# pcs -f drbd_cfg resource create DrbdData ocf:linbit:drbd drbd_resource=disk_dw
op monitor interval=60s
```

```
( DRBD data 리소스 생성 )

# pcs -f drbd_cfg resource master DrbdDataClone DrbdData master-max=1 master-node-
max=1 clone-max=2 clone-node-max=1 notify=true
( DRBD clone 리소스 생성 )

# pcs cluster cib-push drbd_cfg
( 작성한 cib를 라이브 cib로 push )


# pcs cluster cib fs_cfg
( DRBD 파일시스템 리소스 생성 )

# pcs  -f fs_cfg resource create DrbdFS Filesystem device="/dev/drbd0"
directory="/var/www/html" fstype="xfs"
( DRBD의 마운트포인트를 /var/www/html로 설정 )

# pcs  -f fs_cfg constraint colocation add DrbdFS with DrbdDataClone INFINITY
with-rsc-role=Master
( DrbdFS와 DrbdClone 리소스를 같은 노드에서 실행할 수 있도록 묶어줌 )

# pcs  -f fs_cfg constraint order promote DrbdDataClone then start DrbdFS
( DrbdFS 보다 Clone이 먼저 실행되도록 설정 )

# pcs -f fs_cfg constraint colocation add WebService with DrbdFS INFINITY
( DrbdFS와 웹서비스 리소스를 같은 노드에서 실행할 수 있도록 묶어줌 )

# pcs cluster cib-push fs_cfg
( 작성한 cib를 push )

# pcs constraint order DrbdFS then WebService
( 웹서비스 보다 DrbdFS 가 먼저 실행되도록 설정 )
```

★DB 이중화 버전★

- 구축 전 디스크 추가 먼저 진행

```
# vi /etc/hosts
------------------------
172.16.1.8      dw1
172.16.1.9      dw2
------------------------
( 양쪽 노드에서 진행, 맨 밑줄에 추가 )

# yum install -y pacemaker pcs fence-agents-all psmisc policycoreutils-python
# rpm --import https://www.elrepo.org/RPM-GPG-KEY-elrepo.org
# yum -y install https://www.elrepo.org/elrepo-release-7.0-3.el7.elrepo.noarch.rpm
# yum -y install mariadb-server
# yum -y install drbd84-utils kmod-drbd84
( 양쪽 노드에서 진행, 설치 진행, drbd90 버전도 상관없음 )

# modprobe drbd
```

```
# lsmod | grep drbd
( 양쪽 노드에서 진행, drbd 모듈 추가 및 확인 )

# drbdadm create-md disk_dw
# systemctl start drbd
# drbdadm status
( 양쪽 노드에서 진행, 정상적으로 작동됐는지 테스트 )

# drbdadm primary disk_dw
( 마스터 노드에서 진행, 마스터노드를 primary로 변경 )

# mkfs.xfs /dev/drbd0
( drbd에 파일시스템 생성 )

# mkdir /data
( 양쪽 노드에서 진행 )

# mount /dev/drbd0 /data
( 마스터 노드에서 진행, 마운트 할 디렉토리 생성 후 drbd 마운트 )


# vi /etc/my.cnf
--------------------------------------------------------------------------------
[mysqld]
datadir=/data/mysql
socket=/data/mysql/mysql.sock
# Disabling symbolic-links is recommended to prevent assorted security risks
symbolic-links=0
# Settings user and group are ignored when systemd is used.
# If you need to run mysqld under a different user or group,
# customize your systemd unit file for mariadb according to the
# instructions in http://fedoraproject.org/wiki/Systemd

[mysqld_safe]
log-error=/var/log/mariadb/mariadb.log
pid-file=/var/run/mariadb/mariadb.pid

#
# include all files from the config directory
#
!includedir /etc/my.cnf.d
--------------------------------------------------------------------------------
( 마스터 노드에서 진행 )
( datadir 부분과 socket 부분을 /var/lib -> /data 로 변경 )


# vi /etc/drbd.d/clusterdata.res
----------------------------------------
resource disk_dw {
        on dw1 {
                device /dev/drbd0;
                disk /dev/xvdb;
                address 172.16.1.8:7789;
                meta-disk internal;
```

```
        }
        on dw2 {
                device /dev/drbd0;
                disk /dev/xvdb;
                address 172.16.1.9:7789;
                meta-disk internal;
        }
}
----------------------------------------
( 양쪽 노드에서 진행 )
( resource [ 생성한 Drbd ] , on [ hostname ] , device [ drbd 장치 경로 ] , disk [
새로 생성한 디스크 ] , address [ 해당 서버 주소 ] )


# vi /etc/drbd.d/global_common.conf
----------------------------------------
net {
        # protocol timeout max-epoch-size max-buffers
        # connect-int ping-int sndbuf-size rcvbuf-size ko-count
        # allow-two-primaries cram-hmac-alg shared-secret after-sb-0pri
        # after-sb-1pri after-sb-2pri always-asbp rr-conflict
        # ping-timeout data-integrity-alg tcp-cork on-congestion
        # congestion-fill congestion-extents csums-alg verify-alg
        # use-rle
        protocol C;
    }
----------------------------------------
( 양쪽 노드에서 진행 )
( net 설정 맨 아래에 protocol C; 추가 )



# cp -rfp /var/lib/mysql /data/mysql
# mv /var/lib/mysql /var/lib/mysql.back
# ln -sf /data/mysql /var/lib/mysql
( 마스터 노드에서 진행, mysql 디렉토리 복사, 백업 및 링크 설정 )

# mysql_install_db --datadir=/data/mysql
( 마스터 노드에서 진행, DB 설정 진행 )

# systemctl start mariadb
# mysqladmin -u root password root
( 마스터 노드에서 진행, DB 패스워드 설정 )

# mysql -u root -p
( 마스터 노드에서 진행, 패스워드 설정 및 정상 접속 확인 )

# systemctl stop mariadb
# systemctl disable mariadb
( 양쪽 노드에서 진행, 마리아 DB 종료 )

# systemctl enable drbd
# systemctl start drbd
# systemctl start pcsd
```

```
# systemctl enable pcsd
( 양쪽 노드에서 진행 )

# passwd hacluster
( 양쪽 노드에서 진행, 암호 양쪽 동일하게 설정 )

# pcs cluster auth dw1 dw2
( 마스터 노드에서 진행, Username : hacluster 로 인증 진행 )

# pcs cluster setup --name dw_cluster dw1 dw2
( 마스터 노드에서 진행,  클러스터 생성 )

# pcs cluster start --all
# pcs cluster enable --all
# pcs property set stonith-enabled=false
( 마스터 노드에서 진행, 클러스터 시작 및 stonith 비활성화 )

# pcs cluster cib drbd_cfg
# pcs -f drbd_cfg resource create Data ocf:linbit:drbd drbd_resource=disk_dw
# pcs -f drbd_cfg resource master DataSync Data master-max=1 master-node-max=1
clone-max=2 clone-node-max=1 notify=true
# pcs -f drbd_cfg resource create storage Filesystem device="/dev/drbd0"
directory="/data" fstype="xfs"
# pcs -f drbd_cfg resource create DBmaria ocf:heartbeat:mysql
binary="/usr/bin/mysqld_safe" socket="/var/lib/mysql/mysql.sock"
config="/etc/my.cnf" datadir="/data/mysql" op start timeout=60s op stop
timeout=60s op monitor interval=20s timeout=30s
# pcs -f drbd_cfg resource group add p-group storage DBmaria
# pcs -f drbd_cfg constraint colocation add p-group DataSync INFINITY with-rsc-
role=Master
# pcs -f drbd_cfg constraint order promote DataSync then start p-group
# pcs cluster cib-push drbd_cfg
( cib로 묶어서 리소스 설정 진행 )
```

TEST

```
1.해당 리소스 설정 내용들이 정상적으로 들어갔는지 확인
# pcs constraint

2.
설정내용 모두 잘 들어갔는지 확인
# pcs status

3.
Failover 정상적으로 이루어지는지 확인
# pcs cluster stop dw-test
# pcs cluster start dw-test

# pcs cluster stop dw-test2
# pcs cluster start dw-test2
- 양쪽 노드에서 pcs status를 확인해가며 정상적으로 작동하는지 확인 및 마운트 내용,
```

```
/data 파일 내용 등 확인

4.  해당하는 서비스 종료 시 자동으로 올라오는지 확인
- kill -9 [ 서비스 PID ]
- netstat -plunt
```

- fail count 리셋 : pcs resource failcount reset
- drbd StandAlone 발생 시 [Slave] : drbdadm -- --discard-my-data connect all [Master] : drbdadm connect all

---

★ 정상적으로 작동된다면 고가용성 클러스터링 환경 구축(corosync, pacemaker, DRBD) 완료 ! ★

참조

- https://blog.boxcorea.com/wp/archives/1784
- https://www.nextree.co.kr/p12211/
- https://it-sunny-333.tistory.com/135
- https://open-infra.tistory.com/7