LVM 볼륨 확장

지난번에 이어 LVM시스템에 디스크(볼륨)를 추가 확장 하는 방법을 해보겠습니다.
볼륨(20G) 추가하여 현재 /home 파티션(70G)을 확장 하는 방법을 알아 보겠습니다.

1. 기존 디스크 확장

물리장치(sdb) 또는 볼륨 추가 후 표시 용량이 설정한(20G) 용량으로 표시되어야 합니다. 용량에 변화가 변화가 없으면 재부팅 합니다.

```
# lsblk
NAME                    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
fd0                       2:0    1    4K  0 disk
sda                       8:0    0  100G  0 disk
sda1                    8:1    0    1G  0 part /boot
sda2                    8:2    0   30G  0 part /
sda3                    8:3    0   69G  0 part
vg--home-lv--home 253:0    0   69G  0 lvm  /home
sdb                       8:16   0   20G  0 disk
```

2. 파티션 생성

/dev/sdb에 sdb1파트션을 만들고 Linux LVM으로 바꿔 줍니다.

```
# fdisk /dev/sdb

Welcome to fdisk (util-linux 2.23.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
Device does not contain a recognized partition table
Building a new DOS disklabel with disk identifier 0x49a98a24.
The device presents a logical sector size that is smaller than
the physical sector size. Aligning to a physical sector (or optimal

I/O) size boundary is recommended, or performance may be impacted.

Command (m for help):
Command (m for help): n
Partition type:
```

```
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): p
Partition number (1-4, default 1):
First sector (2048-41943039, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-41943039, default
41943039):
Using default value 41943039
Partition 1 of type Linux and of size 20 GiB is set
Command (m for help): p
Disk /dev/sdb: 21.5 GB, 21474836480 bytes, 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disk label type: dos
Disk identifier: 0x49a98a24

Device Boot      Start         End      Blocks   Id  System

/dev/sdb1            2048    41943039    20970496   83  Linux

Command (m for help):
Command (m for help): t
Selected partition 1
Hex code (type L to list all codes): 8e
Changed type of partition 'Linux' to 'Linux LVM'
Command (m for help): p
Disk /dev/sdb: 21.5 GB, 21474836480 bytes, 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes

Disk label type: dos
Disk identifier: 0x49a98a24

Device Boot      Start         End      Blocks   Id  System

/dev/sdb1            2048    41943039    20970496   8e  Linux
LVM
```

```
Command (m for help): w
The partition table has been altered!
Calling ioctl() to re-read partition table.
Syncing disks.
```

3. PV 생성

pvcreate 로 물리 볼륨 생성 및 pvdisplay 으로 확인 합니다.

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

[root@localhost ~]# pvdisplay

  --- Physical volume ---
  PV Name               /dev/sda3
  VG Name               vg-home
  PV Size               <69.00 GiB / not usable 3.00 MiB
  Allocatable           yes (but full)
  PE Size               4.00 MiB
  Total PE              17663
  Free PE               0
  Allocated PE          17663
  PV UUID               dFUFjg-ETp6-aCkV-UHpw-HpP7-e40T-7L00p3
  "/dev/sdb1" is a new physical volume of "<20.00 GiB"

  --- NEW Physical volume ---

  PV Name               /dev/sdb1
  VG Name
  PV Size               <20.00 GiB
  Allocatable           NO
  PE Size               0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               VjYTMM-ECCZ-GWrd-WpNS-OxMI-qYns-pnEr9x
```

4. VG 확장

기존 볼륨 그룸 확인 하고 vgextend로 볼륨 그룹을 확장합니다.
볼륨 정보 확인

```
# vgdisplay

  --- Volume group ---

  VG Name               vg-home
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  2
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                1
  Open LV               1
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               <69.00 GiB
  PE Size               4.00 MiB
  Total PE              17663
  Alloc PE / Size       17663 / <69.00 GiB
  Free  PE / Size       0 / 0
  VG UUID               MESw3F-j0gs-yzOo-bwR9-34nn-u0dP-qHOHEC
```

추가 볼륨을 확장

```
[root@localhost ~]# vgextend vg-home /dev/sdb1

  Volume group "vg-home" successfully extended
```

볼륨확장 후 정보 확인

```
[root@localhost ~]# vgdisplay

  --- Volume group ---

  VG Name               vg-home
  System ID
  Format                lvm2
  Metadata Areas        2
  Metadata Sequence No  3
  VG Access             read/write
```

```
  VG Status              resizable
  MAX LV                 0
  Cur LV                 1
  Open LV                1
  Max PV                 0
  Cur PV                 2
  Act PV                 2
  VG Size                88.99 GiB
  PE Size                4.00 MiB
  Total PE               22782
  Alloc PE / Size        17663 / <69.00 GiB
  Free  PE / Size        5119 / <20.00 GiB
  VG UUID                MESw3F-j0gs-yzOo-bwR9-34nn-u0dP-qHOHEC
```

5. LV 확장 (lvextend)

lvdisplay로 물리 볼륨을 확인한다.

```
# lvdisplay

  --- Logical volume ---

  LV Path                /dev/vg-home/lv-home
  LV Name                lv-home
  VG Name                vg-home
  LV UUID                AY0b7g-L0pN-DrTm-Lhgb-vqU5-
T2W1-0cFe4e
  LV Write Access        read/write
  LV Creation host, time localhost.localdomain, 2022-05-11
16:20:59 +0900
  LV Status              available
  # open                 1
  LV Size                <69.00 GiB
  Current LE             17663
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     8192
  Block device           253:0
```

6. 리사이즈 (resize2fs)

작업전 용량 확인

```
[root@localhost ~]# df -h
Filesystem                      Size  Used Avail Use% Mounted
on

/dev/sda2                        30G  1.3G   29G   5% /
devtmpfs                        484M     0  484M   0% /dev
tmpfs                           493M     0  493M   0% /dev/shm
tmpfs                           493M   19M  474M   4% /run
tmpfs                                493M      0   493M    0%
/sys/fs/cgroup
/dev/sda1                      1014M  140M  875M  14% /boot
/dev/mapper/vg--home-lv--home    68G   55M   65G   1% /home
```

현재 작업하는 파티션 파일시스템 종류 확인 해보고 맞는 것으로 진행 하면 됩니다.

```
xfs인 경우 -> xfs_growfs
ext4 인 경우 -> resize2fs
```

파일시스템 종류 확인

```
[root@localhost ~]# blkid /dev/vg-home/lv-home

/dev/vg-home/lv-home:                  UUID="c29e7a45-
dc2f-46b0-9f9a-62aa7e17ad07" TYPE="ext4"

[root@localhost ~]# resize2fs /dev/vg-home/lv-home

resize2fs 1.42.9 (28-Dec-2013)
Filesystem at /dev/vg-home/lv-home is mounted on /home; on-
line resizing required
old_desc_blocks = 9, new_desc_blocks = 12
The filesystem on /dev/vg-home/lv-home is now 23328768 blocks
long.

[root@localhost ~]# resize2fs /dev/vg-home/lv-home

resize2fs 1.42.9 (28-Dec-2013)
Filesystem at /dev/vg-home/lv-home is mounted on /home; on-
```

```
line resizing required
old_desc_blocks = 9, new_desc_blocks = 12
The filesystem on /dev/vg-home/lv-home is now 23328768 blocks
long.
```

7. 용량 확인

/home 파티션에 추가된 용량 확인 됩니다.

```
[root@localhost ~]# df -h
Filesystem                      Size  Used Avail Use% Mounted
on
/dev/sda2                        30G  1.3G   29G   5% /
devtmpfs                        484M     0  484M   0% /dev
tmpfs                           493M     0  493M   0% /dev/shm
tmpfs                           493M   19M  474M   4% /run
tmpfs                                493M       0   493M    0%
/sys/fs/cgroup
/dev/sda1                      1014M  140M  875M  14% /boot
/dev/mapper/vg--home-lv--home    88G   59M   84G   1% /home
```

[Linux] LVM 만들어보기 – 1부

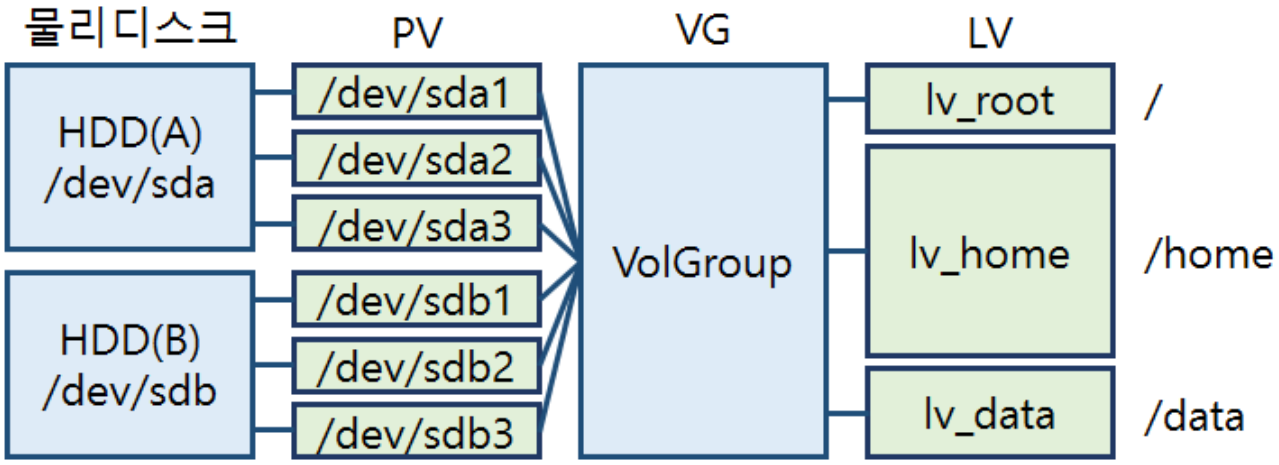
1. LVM이란?

커널에 구현된 기능으로 여러 개의 HDD 용량을 합쳐서 하나의 파일 시스템으로 구성하고 스토리지 확장 및 변경에 있어 유연한 대처를 할 수 있다

1. 구성

LVM 은 이름처럼 파티션대신 볼륨이라는 단위로 저장 장치를 다룰 수 있으며, 물리 디스크를 볼륨 그룹으로 묶고 이것을 논리 볼륨으로 분할하여 관리합니다. 스토리지의 확장이나 변경시 서비스의 변경을 할 수 있으며 특정 영역의 사용량이 많아져서 저장 공간이 부족할 경우에 유연하게 대응할 수 있습니다.

그림 예)



2. LVM 순서 및 용어 정리

1) LVM구성을 위하여 파티션 작업

하드디스크를LVM 으로 만들 수 있게 파티션 생성 또는 변경

2) PV(Physical Volume) : 물리적인 볼륨 생성 작업

/dev/sda1, /dev/sdb1, /dev/sdc1 같은 실제 파티션들을 말합니다

블록 장치 전체 또는 그 블록 장치를 이루고 있는 파티션들을 LVM에서

사용할 수 있게 변환하는 것으로 LVM으로 쓰기 위해 PV로 초기화 합니다.

(PV는 일정한 크기의 PE(Physical Extent)들로 구성)

- PE(Physical Extent) : 물리적 확장

PV를 구성하는 일정한 크기의 블록으로 기본크기는 4MB입니다

3) VG (Volume Group) : 볼륨 그룹 작업

Physical Volume을 합친 그룹을 말하며(PE가 모여서 생성되는 큰 덩어리?)

PV들의 집합으로 PV들로 초기화된 장치들은 VG로 통합되게 되며 이 공간을 쪼개서 LV로 만들 수 있습니다

4) LV (Logical Volume) : 논리적 볼륨

합쳐진 Volume Group을 쪼개어 실제 사용하는 볼륨을 의미 하며 사용자가 최종적

으로 사용하게 되는

논리적인 스토리지 입니다.

3. LVM 구성 예제

현재 시스템은 일반 파일시스템으로 구성되어 있으며 /home 파티션을 LVM 으로 만들고 파일시스템 생성 합니다.

참고로 /home 에 데이터가 있는 경우 모두 삭제 되므로 중요 데이터 있는 경우 꼭 사전에 백업을 해주어야 합니다

파티션 상태 확인

```
# df -Th

Filesystem     Type      Size  Used Avail Use% Mounted on
/dev/sda2      xfs        30G  1.3G   29G   5% /
/dev/sda1      xfs      1014M  140M  875M  14% /boot
/dev/sda3      ext4       68G   55M   65G   1% /home
```

1) LVM 패키지 확인하고 설치 안되어 있으면 설치 합니다.

- 설치 여부 확인
```
# rpm -qa |grep lvm
```

- 패키지 설치
```
# yum install lvm2
```

- 재부팅시 자동 실행 되도록 설정
```
# systemctl enable lvm2-monitor
```

2) 파티션 수정 작업

LVM구성을 위하여 먼저 디스크를 확인 합니다.(fdisk -l /dev/sda)

상태 확인시 LVM 구성이 안 되어 있어 LVM으로 변경 작업을 진행 합니다.

여기서는 대상 파티션인 /dev/sda3 (/home) 파티션만 LVM으로 변경 합니다

- 파티션 상태 확인

```
# fdisk -l /dev/sda

  Device Boot   Start     End   Blocks  Id System
```

```
/dev/sda1  *    2048   2099199   1048576  83 Linux
/dev/sda2     2099200  65013759  31457280  83 Linux
/dev/sda3    65013760  209715199  72350720  8e Linux
```

- 파티션 변경 작업

```
# fdisk /dev/sda
Command (m for help): **p**
Disk /dev/sda: 107.4 GB, 107374182400 bytes, 209715200 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disk label type: dos
Disk identifier: 0x000c8e42



  Device Boot   Start     End   Blocks  Id System
/dev/sda1  *    2048   2099199   1048576  83 Linux
/dev/sda2     2099200  65013759  31457280  83 Linux
/dev/sda3    65013760  209715199  72350720  83 Linux

Command (m for help): **t**
Partition number (1-3, default 3): **3**
Hex code (type L to list all codes): **8e**
Changed type of partition 'Linux' to 'Linux LVM'

Command (m for help): **p**



Disk /dev/sda: 107.4 GB, 107374182400 bytes, 209715200 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disk label type: dos
Disk identifier: 0x000c8e42

  Device Boot   Start     End   Blocks  Id System

/dev/sda1  *    2048   2099199   1048576  83 Linux
```

```
/dev/sda2     2099200  65013759  31457280  83 Linux
/dev/sda3    65013760  209715199  72350720  8e Linux LVM

Command (m for help): **w**
```

3) 물리적볼륨 생성(PV)

블록 장치 전체 또는 그 블록 장치를 이루고 있는 파티션들을 LVM에서 사용할 수 있게 변환합니다.
PE(Physical Extent) : PV를 구성하는 일정한 크기의 블록으로 LVM2에서 기본크기는 4MB입니다

3-1) PV생성 (pvcreate)

```
# pvcreate /dev/sda3
WARNING: ext4 signature detected on /dev/sda3 at offset 1080.
Wipe it? [y/n]: y Wiping ext4 signature on /dev/sda3. Physical
volume "/dev/sda3" successfully created.
```

```
# pvcreate /dev/sda3
WARNING: ext4
signature detected on /dev/sda3 at offset 1080.
Wipe it? [y/n]: y
Wiping ext4 signature on /dev/sda3.
Physical volume "/dev/sda3" successfully created.
```

```
pvcreate /dev/sda3
WARNING: ext4 signature detected on /dev/sda3 at offset 1080.
Wipe it? [y/n]: y
Wiping ext4 signature on /dev/sda3.
Physical volume "/dev/sda3" successfully created.
```

3-2) PV 생성 확인

```
# pvs
PV VG Fmt Attr PSize  PFree
/dev/sda3   lvm2 --- <69.00g <69.00g
```

```
# pvscan PV /dev/sda3
lvm2 [<69.00 GiB]Total: 1 [<69.00 GiB] / in use: 0 [0  ] / in
no VG: 1 [<69.00 GiB]
```

```
# pvdisplay
VG NamePV Size        <69.00 GiB
```

```
Allocatable      NO
PE Size        0
Total PE       0
Free PE        0
Allocated
PE     0
PV UUID        dFUFjg-ETp6-aCkV-UHpw-HpP7-e40T-7L00p3
```

3-3) 볼륨그룹(VG) 생성

볼륨 그룹(VG)은 많은 물리 볼륨을 묶고 장치 파일 시스템에 /dev/VG_NAME와 같은 식으로 보여준다.
- VG 생성 하기

```
# vgcreate [볼륨이름] [포함되는 장치명]

vgcreate vg-home /dev/sda3Volume group "vg-home" successfully
created
```

- 현재 VG 상태 확인

```
# vgs
VG   #PV #LV #SN Attr  VSize  VFreevg-home  1  0  0 wz--n-
<69.00g <69.00g

# vgscan
Reading volume groups from cache.Found volume group "vg-home"
using metadata type lvm2

# vgdisplay
--- Volume group ---
VG Name        vg-home
System
IDFormat        lvm2
Metadata Areas    1
Metadata Sequence No 1
VG Access       read/write
VG Status       resizable
MAX LV         0
Cur LV        0O
pen LV        0
Max PV        0
```

```
Cur PV        1
Act PV        1
VG Size        <69.00 GiB
PE Size        4.00 MiB
Total PE       17663
Alloc PE / Size    0 / 0
Free PE / Size    17663 / <69.00 GiB
VG UUID        MESw3F-j0gs-yzOo-bwR9-34nn-u0dP-qHOHEC
```

4) 논리 볼륨 생성하기

볼륨 그룹 생성이 완료 되었으면 논리 볼륨(Logical Volumes)을 생성 합니다.
- LV 생성 하기

```
# lvcreate -n datalv L 70GB testvg  --> testvg의 70G를 datalv에 할당
# lvcreate -n lv-home -l +100%FREE vg-home  --> 남은 용량을 lv-home 에 할당함
# lvcreate

lvcreate -n lv-home -l +100%FREE vg-homeLogical volume "lv-home" created.
```

- 생성된 LV 상태 확인 (lvs, lvdisplay, lvscan)

```
# lvs
LV   VG   Attr    LSize  Pool Origin Data% Meta% Move Log Cpy%Sync Convert
lv-home vg-home -wi-ao---- <69.00g

# lvscan
ACTIVE      '/dev/vg-home/lv-home' [<69.00 GiB] inherit

# lvdisplay
--- Logical volume ---
LV Path        /dev/vg-home/lv-home
LV Name        lv-home
VG Name        vg-home
LV UUID        AY0b7g-L0pN-DrTm-Lhgb-vqU5-T2W1-0cFe4W
LV Write Access    read/write
LV Creation host, time localhost.localdomain, 2022-05-11 16:20:59 +0900
LV Status       available
```

```
# open         0
LV Size        <69.00 GiB
Current LE       17663
Segments        1
Allocation       inheritRead ahead sectors   auto
- currently set to   8192
Block device      253:0
```

5) 파일시스템 생성 및 마운트

5-1) 생성된 LV를 파일시스템 생성 및 마운트

lvdisplay 에서 나오는 LV Path를 ext4 타입으로 파일시스템 포맷 한 후 /home 에 마운트 합니다.
**

```
# mkfs.ext4 /dev/vg-home/lv-home
**mke2fs 1.42.9 (28-Dec-2013)Discarding
device blocks: done
Filesystem label=OS
type: LinuxBlock
size=4096 (log=2)Fragment size=4096 (log=2)Stride=0 blocks, Stripe width=0 blocks4521984 inodes, 18086912 blocks904345 blocks (5.00%) reserved for the super userFirst data block=0Maximum filesystem blocks=2166358016552 block groups32768 blocks per group, 32768 fragments per group8192 inodes per groupSuperblock backups stored on blocks: 32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,      4096000, 7962624, 11239424 Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): doneWriting superblocks and filesystem accounting information: done

**# mount -t ext4 /dev/vg-home/lv-home /home**



**# df -Th
**Filesystem
Type   Size Used Avail Use% Mounted on/dev/sda2
xfs    30G 1.3G  29G  5% /devtmpfs           devtmpfs 484M   0 484M  0%
```

```
/devtmpfs              tmpfs   493M   0 493M  0%
/dev/shmtmpfs              tmpfs   493M  13M 480M  3%
/runtmpfs              tmpfs   493M   0 493M  0%
/sys/fs/cgroup/dev/sda1           xfs   1014M 140M 875M 14%
/boottmpfs              tmpfs   99M   0  99M  0% /run/user/0*
/dev/mapper/vg--home-lv--home ext4    68G  53M  65G  1% /home*
```

5-2) 자동 마운트를 위하여 /etc/fstab 에 등록 해줍니다.

```
# /etc/fstab 확인
/dev/vg-home/lv-home    /home           ext4  defaults    1 2
```

[Network] K8S Overlay Network 변경 실습 (IPIP -> VXLAN)

K8S Overlay Network 의 변경

IPIP -> VXLAN 변경 하기

주의) 다른 방식의 네트워크에서 생성된 서비스 POD가 있을 경우 프로덕션 레벨에서는 변경이 사실상 어렵다.
(pod 통신의 문제 발생에 대한 트러블슈팅이나 보장을 할 수가 없다)

Calico IP-IP Network 제거 후 VXLAN 으로 재 설치

Node : Controller / Worker01 / Worker02

```
## Controller
# Mode 확인 시 IPIPMODE 인 것을 확인할 수 있다
calicoctl get ippool -o wide
NAME                  CIDR              NAT    IPIPMODE
VXLANMODE   DISABLED   DISABLEBGPEXPORT   SELECTOR
default-ipv4-ippool   192.168.0.0/16   true   Always     Never
```

```
false      false
 all()

# 설치 Manifest YAML 을 이용하여 삭제한다.
kubectl delete -f calico.yml

## Contoller / Worker 공통
# 가비지 삭제 후 리부팅하면 tunl0 인터페이스가 사라진 것을 확인할 수 있다.
sudo rm -rf /var/run/calico/
sudo rm -rf /var/lib/calico/
sudo rm -rf /etc/cni/net.d/
sudo rm -rf /var/lib/cni/
sudo reboot


## Controller
# Manifest. calico.yaml 파일을 VXLAN 설치 타입으로 수정한다.

 livenessProbe:
            exec:
              command:
              - /bin/calico-node
              - -felix-live
               # - -bird-live        // VXLAN 은 bird(BGP) 를 사용
하지 않으므로 주석 처리 필수
            periodSeconds: 10
            initialDelaySeconds: 10
            failureThreshold: 6
            timeoutSeconds: 10
          readinessProbe:
            exec:
              command:
              - /bin/calico-node
              - -felix-ready
               # - -bird-ready    // 주석 필수


            # Enable IPIP
            - name: CALICO_IPV4POOL_IPIP
              value: "Never"       // Always --> Never 로 변경한
다
            # Enable or Disable VXLAN on the default IP pool.
```

```
            - name: CALICO_IPV4POOL_VXLAN
              value: "Always"         // Never --> Always 로 변경한다

kind: ConfigMap
apiVersion: v1
metadata:
  name: calico-config
  namespace: kube-system
data:
  # Typha is disabled.
  typha_service_name: "none"
  # Configure the backend to use.
  calico_backend: "vxlan"        // "bird" --> "vxlan" 으로 변경한다.


# 설치
kubectl apply -f calico.yaml

# Calico Node 체크. 모두 Ready 여야 한다. 조금 기다려보자.
kubectl get nodes -o wide -A

# Calico Pod 체크. kube-system PoD 에 에러가 있으면 안된다.
kubectl get pod -o wide -A

# Calico Type 체크. BIRD 는 꺼져야 정상
sudo calicoctl node status
Calico process is running.
The BGP backend process (BIRD) is not running.

# Network 체크 VXLANMODE 가 활성화되었다.
calicoctl get ippool -o wide
NAME                  CIDR              NAT    IPIPMODE
VXLANMODE   DISABLED   DISABLEBGPEXPORT   SELECTOR
default-ipv4-ippool    192.168.0.0/16    true    Never
Always        false        false
 all()

# 각 노드들의 tunl0 인터페이스가 사라지고 vxlan 인터페이스가 생성되었다.
# 노드의 vxlan 인터페이스 간 라우팅이 연결된 부분을 확인 가능하다.
```

```
hostway@controller:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric
Ref    Use Iface
0.0.0.0         10.10.10.1      0.0.0.0         UG    0      0
0 ens18
10.10.10.0      0.0.0.0         255.255.255.0   U     0      0
0 ens18  // External (SNAT)
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0
0 docker0 // Container Runtime Bridge
192.168.5.0     192.168.5.0     255.255.255.192 UG    0      0
0 vxlan.calico  // Worker01의 인터페이스 정보
192.168.30.64   192.168.30.64   255.255.255.192 UG    0      0
0 vxlan.calico  // Worker02의 인터페이스 정보
192.168.49.0    0.0.0.0         255.255.255.192 U     0      0
0 *              // Controller 의 vxlan 대역
192.168.49.1    0.0.0.0         255.255.255.255 UH    0      0
0 cali09ae4a7064b   // Node(Worker01)가 도달할 GW
192.168.49.2    0.0.0.0         255.255.255.255 UH    0      0
0 cali1fdac863dc5  // Node(Worker02)가 도달할 GW

# Worker 의 정보를 알고 있다.
hostway@controller:~$ ip nei | grep vxlan
192.168.5.0  dev  vxlan.calico  lladdr  66:8c:33:86:44:ce
PERMANENT
192.168.30.64  dev  vxlan.calico  lladdr  66:fb:72:20:22:a1
PERMANENT

# VXLAN Traffic Port 는 UDP 이다.
udp        0      0 0.0.0.0:4789            0.0.0.0:*

# PoD 생성 시 특이사항 없음
hostway@controller:~$ kubectl create deployment sampleos --
image=gcr.io/google-samples/kubernetes-bootcamp:v1     --
replicas=3
deployment.apps/sampleos created
hostway@controller:~$ kubectl get pod -o wide
NAME                         READY   STATUS    RESTARTS   AGE
IP              NODE       NOMINATED NODE   READINESS GATES
sampleos-646dc9654b-8xjw9   1/1     Running   0          45s
192.168.5.11    worker01   <none>           <none>
```

```
sampleos-646dc9654b-gxn75   1/1     Running   0          45s
192.168.5.10    worker01   <none>           <none>
sampleos-646dc9654b-snkxg   1/1     Running   0          45s
192.168.30.75   worker02   <none>           <none>
```

VXLAN 패킷 확인하기

// Controller

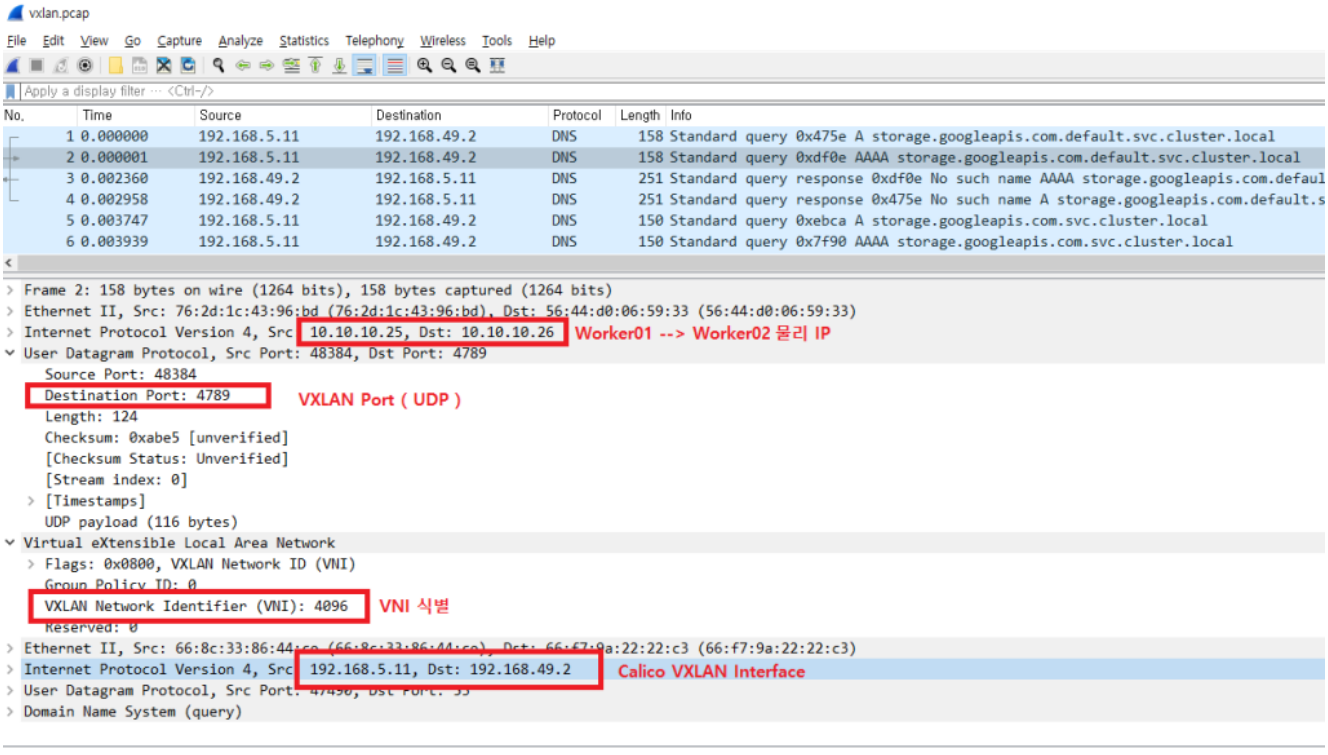
1) worker01 과 worker02 에 위치하는 POD 간 Ping 을 날려본다.

```
hostway@controller:~$ kubectl exec -it
sampleos-646dc9654b-8xjw9 -- ping 192.168.30.75
PING 192.168.30.75: 56 data bytes
64 bytes from 192.168.30.75: icmp_seq=0 ttl=115 time=92.124 ms
64 bytes from 192.168.30.75: icmp_seq=1 ttl=115 time=79.735 ms
64 bytes from 192.168.30.75: icmp_seq=2 ttl=115 time=79.233 ms
```

2) 실제 통신은 물리적 인터페이스에서 진행되므로 실제 인터페이스에서 tcpdump 를 추출한다.

```
sudo tcpdump -i ens18 -w vxlan.pcap
```

3) Wireshark 로 확인한다. UDP 패킷을 확인할 수 있다.



[기타] CentOS 7 에 Kubernetes 기본 Install

CentOS 7 에 Kubernetes 기본 설치하기

```
OS : CentOS 7.6.1810 Minimal
Account : root
- 퍼블릭 SNAT IP 세팅
Controller : 10.10.10.237 SSH:4223
Worker-01 : 10.10.10.204 SSH:4224
Worker-02 : 10.10.10.190 SSH:4225
```

전 노드 공통

```
# root 로 초기 로그인. sudo 를 사용하기 위해 일반유저 생성 및 패스워드 지정
useradd -d /home/username username
echo "password" | passwd username --stdin

# 일반유저는 su 사용하지 못하도록 권한 조정
chmod 700 /usr/bin/su

# sudoer 의 기본 wheel 그룹에 계정 추가
sed -ie '/wheel/s/$/\:username/' /etc/group

# Timezone 설정
sudo timedatectl set-timezone Asia/Seoul

# SWAP OFF
sudo swapoff -a
sudo sed -i -e '/swap/d' /etc/fstab

# firewalld off
sudo systemctl stop firewalld && sudo systemctl disable firewalld
```

```
# Selinux
setenforce 0
sudo  sed  -i  's/SELINUX=enforcing/SELINUX=disabled/g'
/etc/selinux/config

# Hostname
sudo hostnamectl set-hostname controller
sudo hostnamectl set-hostname worker-01
sudo hostnamectl set-hostname worker-02

## Controller / Worker
#curl -s https://get.docker.com | sudo sh
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh

## Check
sudo docker -v
sudo docker ps -a

## Controller / Worker
sudo mkdir /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
EOF

## Docker enable && restart
sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker

## Packages Repo
sudo cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
```

```
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes
-el7-x86_64
enabled=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
EOF

## Install
sudo yum install -y kubelet kubeadm kubectl --
disableexcludes=kubernetes
```

Controller Init

```
# Controller. 일반적으로 컨트롤러 자신의 공인IP 를 통해 API 서버임을 알
린다 (Advertise)
sudo kubeadm init --ignore-preflight-errors=all --pod-network-
cidr=192.168.0.0/16 --apiserver-advertise-address=10.10.10.237

# Regular User Privileges
  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

# Network Plugin Setting ( Calico )
curl
https://projectcalico.docs.tigera.io/manifests/calico.yaml -O
kubectl apply -f calico.yaml

# System Namespace ( kube-system ) check. CoreDNS 가 올라오면 정상
kubectl get pods -o wide -A

NAMESPACE     NAME                                       READY
STATUS     RESTARTS    AGE       IP                NODE
NOMINATED NODE   READINESS GATES
kube-system   calico-kube-controllers-7c845d499-p85pm   1/1
Running   0           3m6s    192.168.49.3   controller
<none>           <none>
kube-system   calico-node-fnm2q                         1/1
Running   0           3m6s    10.10.10.237   controller
<none>           <none>
```

```
kube-system   coredns-64897985d-cgvml                  1/1
Running   0          5m41s   192.168.49.2   controller
<none>           <none>
kube-system   coredns-64897985d-vdckf                  1/1
Running   0          5m42s   192.168.49.1   controller
<none>           <none>
kube-system   etcd-controller                          1/1
Running   0          5m54s   10.10.10.237   controller
<none>           <none>
kube-system   kube-apiserver-controller                1/1
Running   0          5m54s   10.10.10.237   controller
<none>           <none>
kube-system   kube-controller-manager-controller       1/1
Running   0          6m      10.10.10.237   controller
<none>           <none>
kube-system   kube-proxy-nn5zn                         1/1
Running   0          5m42s   10.10.10.237   controller
<none>           <none>
kube-system   kube-scheduler-controller                1/1
Running   0          5m54s   10.10.10.237   controller
<none>           <none>
```

(공통) Multi NIC 를 가지는 환경에서 INTERNAL-IP 설정
다수의 네트워크가 존재하는 서버의 경우 K8S 에서 첫 번째 NIC 의 IP 가 INTERNAL-IP 로 자동 설정된다.
해당 INTERNAL-IP 를 Init 시 설정한
kubeadm --apiserver-advertise-address 와 동일한 IP 대역을 사용하도록 수동 설정한다.

```
cat << EOF | sudo tee /etc/default/kubelet
KUBELET_EXTRA_ARGS='--node-ip $(hostname -I | cut -d ' ' -f2)'
EOF
sudo systemctl daemon-reload
sudo systemctl restart kubelet
kubectl cluster-info
```

Worker Join

Worker-01과 Woker-02 Node 에도 User Privileges 설정 복사 및 적용한다

```
sudo scp /etc/kubernetes//admin.conf username@10.10.10.204:/home/username/admin.conf
sudo scp /etc/kubernetes//admin.conf username@10.10.10.190:/home/username/admin.conf

# Worker
mkdir -p $HOME/.kube
sudo cp -i ./admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

# 각 Worker 에서 kubeadm 를 사용해서 Join 한다.
sudo kubeadm join 10.10.10.237:6443 --token jgocer.fu65ql39kdod5qi0 \
                        --discovery-token-ca-cert-hash sha256:3cb85267e89913d7865d219922daaa8fc6e788dd2be0e2f80fae27176e2dfe3b

# 토큰값을 나중에 재 확인해야 할 경우
kubeadm token create --print-join-command

# Check
kubectl get nodes -o wide
NAME         STATUS     ROLES                  AGE   VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE                KERNEL-VERSION          CONTAINER-RUNTIME
controller   Ready      control-plane,master   16m   v1.23.5   10.10.10.237   <none>        CentOS Linux 7 (Core)   3.10.0-1062.el7.x86_64   docker://20.10.14
worker-01    Ready      <none>                 55s   v1.23.5   10.10.10.204   <none>        CentOS Linux 7 (Core)   3.10.0-1062.el7.x86_64   docker://20.10.14
worker-02    NotReady   <none>                 38s   v1.23.5   10.10.10.190   <none>        CentOS Linux 7 (Core)   3.10.0-1062.el7.x86_64   docker://20.10.14

# Check Pod Create
kubectl run hello --image=nginx --dry-run=client -o yaml | kubectl apply -f-
pod/hello created
[myungin.baek@controller ~]$ kubectl get pods -o wide
NAME      READY    STATUS      RESTARTS    AGE    IP       NODE          NOMINATED NODE    READINESS GATES
```

```
hello   1/1     Running   0          42s   192.168.171.1
worker-01   <none>           <none>
```

[OS] CentOS 7 에서 iptables 로 초기 접근 제어하기

iptables 로 초기 접속 제어 하기

CentOS 7 기본 설치 후, 공인 네트워크 환경에 연결해야 할 때 우선 최소 SSH 접근만 허용하기

(Pre) CentOS 7 에서 firewalld 대신 iptables 사용하기

firewalld 사용이 권장되지만, iptables 에 익숙할 경우 iptables.target 을 service 에 연결하는 패키지를 운영할 수 있다.

```
# firewalld disable
systemctl stop firewalld && systemctl disable firewalld

# firewalld 대신 service 를 통한 운영.
# 설치 후 서비스 재시작하면 /etc/sysconfig/iptables 에 실시간 룰이 저장된다.
yum install iptables-services
service iptables reload
service iptables status

# 룰 저장
service iptables save

# 룰 리로드
service iptables reload

# 룰 형식으로 텍스트 저장. 반드시 -c ( ALL Rule ) 옵션을 지정해서 저장하여
```

야 방화벽 정책과 ROUTE(NAT) 룰이 같이 저장된다.
```
iptables-save -c > rules.txt

# 룰 복원
iptables-restore < rules.txt
```

iptables 초기 설정 (모두 차단 후 특정 IP만 허용하기)

```
# 모든 룰 초기화
iptables -F

# lo 인터페이스 모든 ACCEPT
iptables -A INPUT -i lo -j ACCEPT

# 모든 접근 허용할 소스 IP 지정. 포트(SSH) 까지 지정할 경우 -p tcp (-m tcp 생략 가능) --dport 22 를 추가한다
iptables -A INPUT -s 1.2.3.4/32 -m comment --comment "회사" -j ACCEPT

# state 상태를 지정한 ACCEPT.
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

# (선택) Ping request 요청에 대해 호스트 접근 불가 응답 설정. 서버가 존재한다는 힌트를 주는 것이다.
iptables -A INPUT -j REJECT --reject-with icmp-host-prohibited

# (선택) Ping request 요청에 거절 하고 싶다면.
iptables -A INPUT -p icmp --icmp-type echo-request -j REJECT

# (선택) Ping 접근 자체를 막고 싶다면 DROP. 허용하려면 ACCEPT 를 주면 된다.
iptables -A INPUT -p icmp -j DROP

# 그 외 모든 TCP 접근 DROP
iptables -A INPUT -p tcp -j DROP

# 저장
service iptables save
```

추가 접근 정책 추가

기본 설정 이후에 -A 로 추가하면 DROP 이후의 Line 에 들어가므로 Line 순서 상 제대로 허용되지 않는다
-I INPUT [DROP Line] 을 기입하여 DROP 라인 위로 추가되도록 한다.

```
iptables -nL --line-number
------------------------------------------------------------------
------------------
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  1.2.3.4                    0.0.0.0/0
/* 회사 */
2    DROP       tcp  --  0.0.0.0/0            0.0.0.0/0
------------------------------------------------------------------
------------------

# 2번 DROP 정책 위로 추가한다.
iptables -I INPUT 2 -s 5.6.7.8 -j ACCEPT -m comment --comment "
허용 추가"
iptables -nL --line-number
------------------------------------------------------------------
------------------
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  1.2.3.4                    0.0.0.0/0
/* 회사 */
2    ACCEPT     all  --  5.6.7.8                    0.0.0.0/0
/* 허용 추가 */
3    DROP       tcp  --  0.0.0.0/0            0.0.0.0/0
------------------------------------------------------------------
------------------
```

라인 정렬이 어려울 경우 /etc/sysconfig/iptables 파일을 직접 수정 후 reload 가능하다.

라인 정책 삭제

```
iptables -D INPUT [Number]
```

CentOS 7 설치 후 할 일 – 필수 패키지 및 권장 패키지 설치 (2)

CentOS 7 패키지 설치

CentOS 7 기본 설정 후 서버 관리 및 운영을 도와주는 몇 가지 패키지들을 설치해주어야 한다.

해당 포스트에서는 필수 패키지 / 권장 패키지로 나누어 설명하도록 하겠다.

1. 패키지 업데이트

커널 및 각종 패키지들이 업데이트 되며 서버의 구성환경을 고려하여 진행한다.

```
## 패키지명을 지정하지 않을경우 설치되어있는 모든 패키지를 업데이트 한다.
yum update [패키지명]
## 설치된 패키지 중 업데이트된 패키지 목록 확인
yum list updates
```

2. 필수 패키지

일반적으로 설치해야하는 패키지들과 서버 모니터링을 위한 패키지로 구분하여 설명한다.

일반 패키지

epel-release : 각종 패키지의 최신 버전을 사용하기 위해 설치
해당 패키지 설치후 다른 패키지 설치시에 최신 버전으로 설치되므로 disable 설정 후 enablerepo 옵션 사용

vi /etc/yum.repos.d/epel.repo
enabled=1 을 0으로

[epel]
...
enabled=0
...

yum repolist
yum --enablerepo=epel install [패키지명]

net-tools : 각종 네트워크 명령어들을 관리하는 패키지로 IP 확인을 위한 ifconfig, 네트워크 연결 상태 확인을 위한 netstat 둘을 포함한 명령어를 사용하기 위해 설치한다.

<ifconfig>
ifconfig -a
ifconfig [interface] up
ifconfig [interface] down

<netstat>
netstat -nap
netstat -an | grep [Port]
netstat -nlpt

unzip : zip으로 압축된 파일을 풀기 위해 사용한다.

unzip [file_name].zip
unzip -l [file_name].zip
unzip -t [file_name].zip

wget : 웹 서버로부터 콘텐츠를 가져오는 명령어로 HTTP, HTTPS, FTP 프로토콜클 지원한다.

wget -O [파일 이름] [URL 주소]

wget --no-check-certificate [URL 주소]

curl : 커맨드 라인용 데이터 전송 도구로 HTTP, FTP등 다양한 프로토콜을 지원한다.
Web 파일 다운받을 경우 wget 권장

curl -o [파일 이름] [URL 주소]
curl -T [파일 이름] [서버 IP]
curl -L [URL 주소]

chrony : NTP를 구현한 Server/Client로 ntpd를 대체하며 서버의 시간을 동기화한다.

동기화 시킬 NTP 서버의 IP 추가
vi /etc/chrony.conf

server [NTP server IP] iburst <iburst = 동기화 시간을 단축 시켜주는 옵션>

systemctl restart chronyd
chronyc sources

gcc, gcc-c++ : 각각 C언어, C++언어 컴파일러로 서버에 개발환경을 구축하기 위해 설치한다.

openssl-devel : openssl을 설치하기 위한 의존성 패키지로 openssl을 사용할 애플리케이션 개발용 파일이다.

서버 모니터링 관련 패키지

htop : 리눅스에서 사용중인 리소스등을 확인 (윈도우의 작업 관리자와 비슷한 역할을 함)

프로세스

PID

USER
PR
NI
VIRT
RES
SHR
S
%CPU
%MEM
TIMR+
COMMAND

iftop : 네트워크 인터페이스의 트래픽 모니터링

```
iftop -i eno1
iftop -f "dst port 22"
```

dstat : 실시간으로 디스크별 I/O 사용량 체크 및 부하 유발을 주는 프로세스를 쉽게 확인 가능하다.

```
dstat -tcdml
```

sysstat : 리눅스 성능 측정 도구 패키지로 sar, iostat 등을 사용하기 위해 설치

<iostat> : CPU와 디스크의 처리량, 대기열 길이등을 실시간으로 모니터링 한다.

```
iostat -d 3
iostat -c 3
```

<sar> : 과거 모니터링 정보를 확인 /var/log/sa 폴더에 기록된 sa파일들을 조회한다.

```
sar -u
sar -r
sar -dp
sar -n DEV
```

lsof : 시스템에서 열려있는 파일에 대한 정보를 출력

lsof -u [사용자 이름]
lsof -i
lsof -c [서비스 이름]

psmisc : proc 파일 시스템을 사용하는 유틸리티 패키지로 fuser, killall, pstree등을 사용

<fuser> : 특정 파일시스템을 umount 할 때 어떠한 프로세스 때문에 진행이 안될 경우, 그 프로세스를 kill 하기 위해 사용
fuser -v [디렉터리]
fuser -ck [디렉터리]
<killall> : 특정 프로세스를 한 번에 죽이기 위해 사용
killall -i [프로세스]
killall -v [프로세스]
killall -w [프로세스]

<pstree> : 프로세스의 상태를 Tree 구조로 출력하여 확인
pstree -anp

3. 권장 패키지

mlocate : 파일을 검색하기 위해 find 명령어 대신 데이터베이스 기반으로 파일을 검색하기 때문에 빠른 검색을 위해 사용한다.

updatedb
locate [파일 이름]
locate -n [숫자] [파일 이름]

ncat : 서버와 클라이언트가 소켓 통신을 하기위해 사용한다.

<서버>
ncat -l [Port]
ncat -lk [Port]

<클라이언트>
ncat [Server IP] [Port]

whois : 도메인 소유자나 IP 주소 위치 등 이외 다른 정보를 확인할 때 사용한다.

whois [도메인 주소]
whois [IP 주소]

cloud-utils-growpart : LVM 구성이 아닌 상태에서 root 파티션 용량을 추가하기 위해 사용한다.

growpart [디스크 명] [파티션 번호]
resize2fs [파티션 경로]

tcping : 클라이언트에서 서버의 TCP 포트를 ping 유틸리티로 체크하기 위해 사용한다.

tcping [Server IP] [Port]

CentOS 7 설치 후 할 일 – 파티셔닝 및 설정 (1)

CentOS 7 설치 후 할 일

실제로 서비스를 제공하는 서버라면 몇 가지 기본적인 설정, 패키지, 보안에 신경을 써줘야 한다.

파티셔닝 및 설정, 필수 및 권장 패키지 설치, 서버 운영을 위한 보안 설정(하드닝) 세가지로 분류해 작성한다.

파티셔닝

LVM : Default로 설정되어 있으며 스토리지 확장 시 유연하게 변경 가능, 관리 난이도가 높아 일반적으로 **표준 파티션** 사용 권장

표준 파티션 : 디스크 성능이 좋아짐에 따라 속도 면에서 **LVM**과 크게 차이가 없음, xfs 형식 사용하여 이전보다 파티션 조절 수월함

1. 네트워크 설정

(OS 설치 과정에서 네트워크 기설정되어 있는 경우 해당 내용 생략)

```
# ip addr
# vi /etc/sysconfig/network-scripts/ifcfg-eth0

. . .
BOOTPROTO=none
. . .
IPV6INIT=no
IPV6_AUTOCONF=no
IPV6_DEFROUTE=no
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
. . .
ONBOOT=yes                                  # 부팅/네트워크 재시작 시  자동으로 인터페이스 활성화를 위해  yes  설정
IPV6_PRIVACY=no
IPADDR=192.168.122.243
NETMASK=255.255.255.0
GATEWAY=192.168.122.1
DNS1=8.8.8.8
DNS2=8.8.4.4

# systemctl restart network
# ip addr
eth0 영역  IP 확인



# ping -c 4 google.com                  # 네트워크 상태 확인
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3094ms
rtt min/avg/max/mdev = 82.692/83.010/83.554/0.492 ms
```

2. 바이오스 타이머 조정

CentOS 7 부터 timedatectl 명령어로 시간, 타임존 설정을 할 수 있다.

```
# timedatectl

      RTC time : 하드웨어 시간
      NTP enabled : NTP 활성화 여부
      NTP synchronized : NTP 동기화 여부
      RTC in local TZ : RTC 와 Time zone 시간 동기화 여부

# timedatectl list-timezones | grep -i Asia*

# timedatectl set-timezone Asia/Seoul

# timedatectl
```

3. Hostname 설정

CentOS 7 설치 후 기본적으로 hostname이 localhost.localdomain 으로 설정되어있으므로 변경해주어야 함

```
# hostnamectl
Static hostname: localhost.localdomain

# hostnamectl set-hostname newhostname

# hostnamectl
Static hostname: newhostname
```

4. SELinux 비활성화

SELinux는 기본적으로 활성화 되어있으나, 해당 메뉴얼에서는 disabled(=미사용) 설정한다.

```
# vi /etc/sysconfig/selinux
```

```
. . .
SELINUX=disabled     . . .

# shutdown -r now

# getenforce
Disabled
```

5. root 계정 원격 접근 불가 설정

보안을 위해 root로의 원격 접속은 거부하고 타 계정을 사용하여 su 명령어를 통해 사용하도록 한다.

```
# ps -ef | grep sshd
# systemctl enable sshd

# vi /etc/ssh/sshd_config
. . .
PermitRootLogin=no
. . .

# systemctl restart sshd
```

6. 자동 로그아웃 설정

보안을 위해 일정 시간 응답이 없으면 자동으로 로그아웃 하도록 설정한다.

```
# vi /etc/profile.d/timeout.sh
TMOUT=600
export TMOUT

chmod +x /etc/profile.d/timeout.sh

# source /etc/profile
# echo $TMOUT
```

600

7. 히스토리 포맷 설정

명령어의 내역을 출력하는 명령어인 history의 날짜 및 시간을 확인하기 위해 설정한다.

```
# vi /etc/profile.d/history.sh

HISTTIMEFORMAT="%F %T -- "
export HISTTIMEFORMAT


# chmod 644 /etc/profile.d/history.sh
# source /etc/profile.d/history.sh

# hisotry
999  2022-04-06 14:50:10 -- vi /etc/profile.d/history.sh
1000       2022-04-06    14:50:19    --    chmod    644
/etc/profile.d/history.sh
1001  2022-04-06 14:50:28 -- source /etc/profile.d/history.sh
1002  2022-04-06 14:50:30 -- history
```

8. 문자셋 설정

설치시 언어와 문자셋을 잘못 선택했을 경우에 설정한다.

```
# localectl
  System Locale: LANG=en_US.UTF-8
      VC Keymap: us
     X11 Layout: us

# localectl list-locales | grep -i kr
ko_KR
ko_KR.euckr
ko_KR.utf8
```

```
# localectl set-locale LANG=ko_KR.UTF-8
# localectl set-keymap kr
# localectl set-x11-keymap kr

# localectl
   System Locale: LANG=ko_KR.UTF-8
       VC Keymap: kr
      X11 Layout: kr
```

다음 장에선 필수적으로 설치해야할 패키지 및 권장 패키지들에 대해서 설명한다.