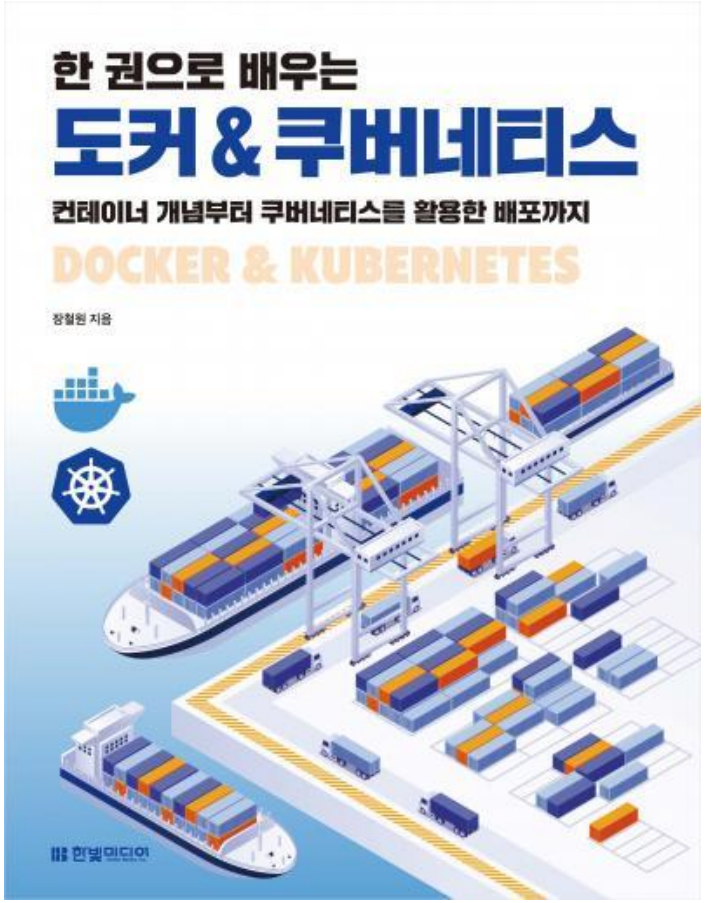
kubernetes

쿠버네티스 실습환경 구축

Autumn 2024

AI융합학과
Seongbok Baik
sbbaik@dju.ac.kr

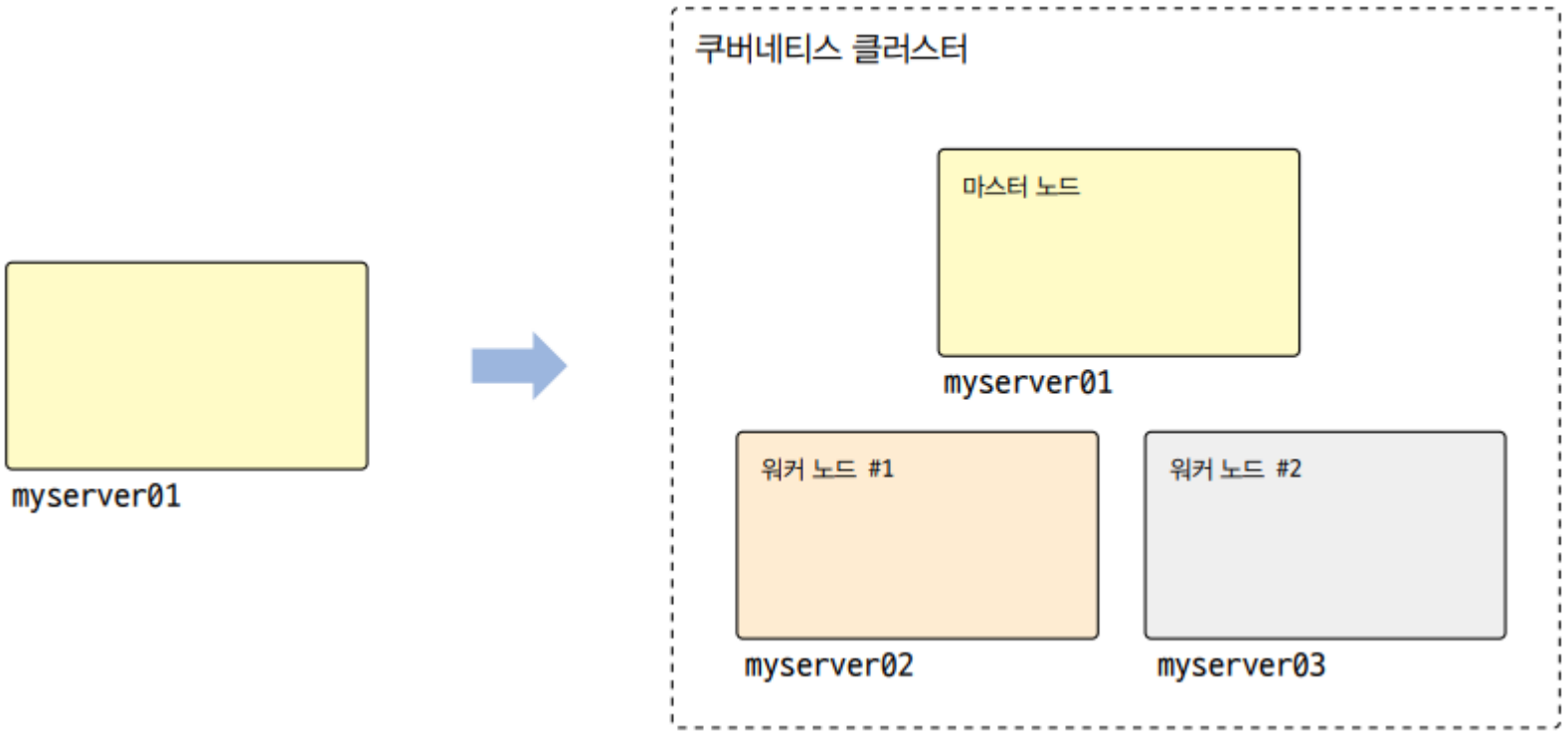
00 Textbook



장철원 소프트웨어공학자

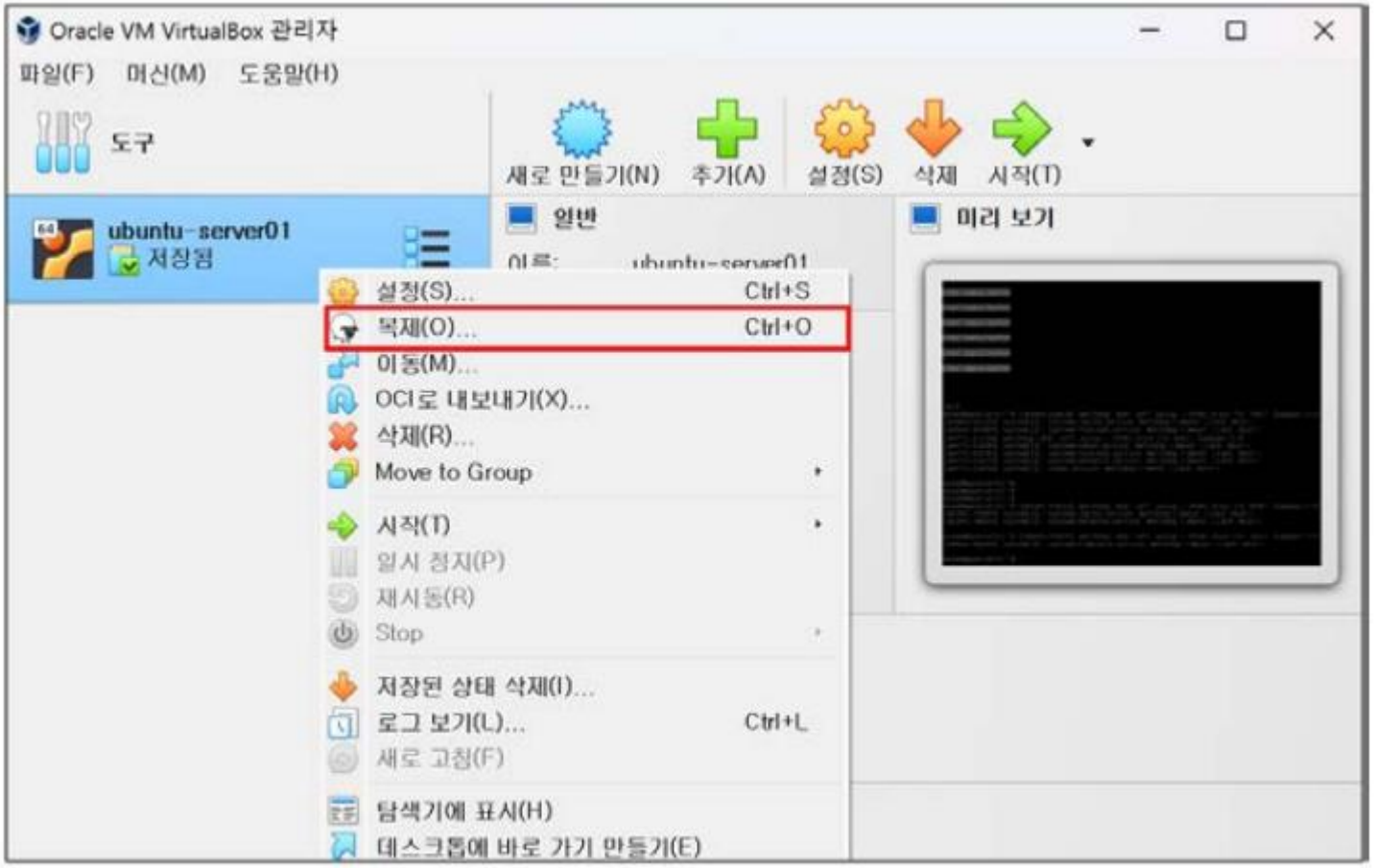
01 가상머신 복제

쿠버네티스 클러스터를 구축하기 위해 가상머신 추가



02 가상머신 복제

기존에 생성했던 가상머신 복제



03 가상머신 복제

새 머신 이름과 경로 설정

- MAC Address Policy 옵션을 "모든 네트워크 어댑터의 새 MAC 주소 생성"으로 지정
- 복제 방식은 "완전한 복제"로 지정

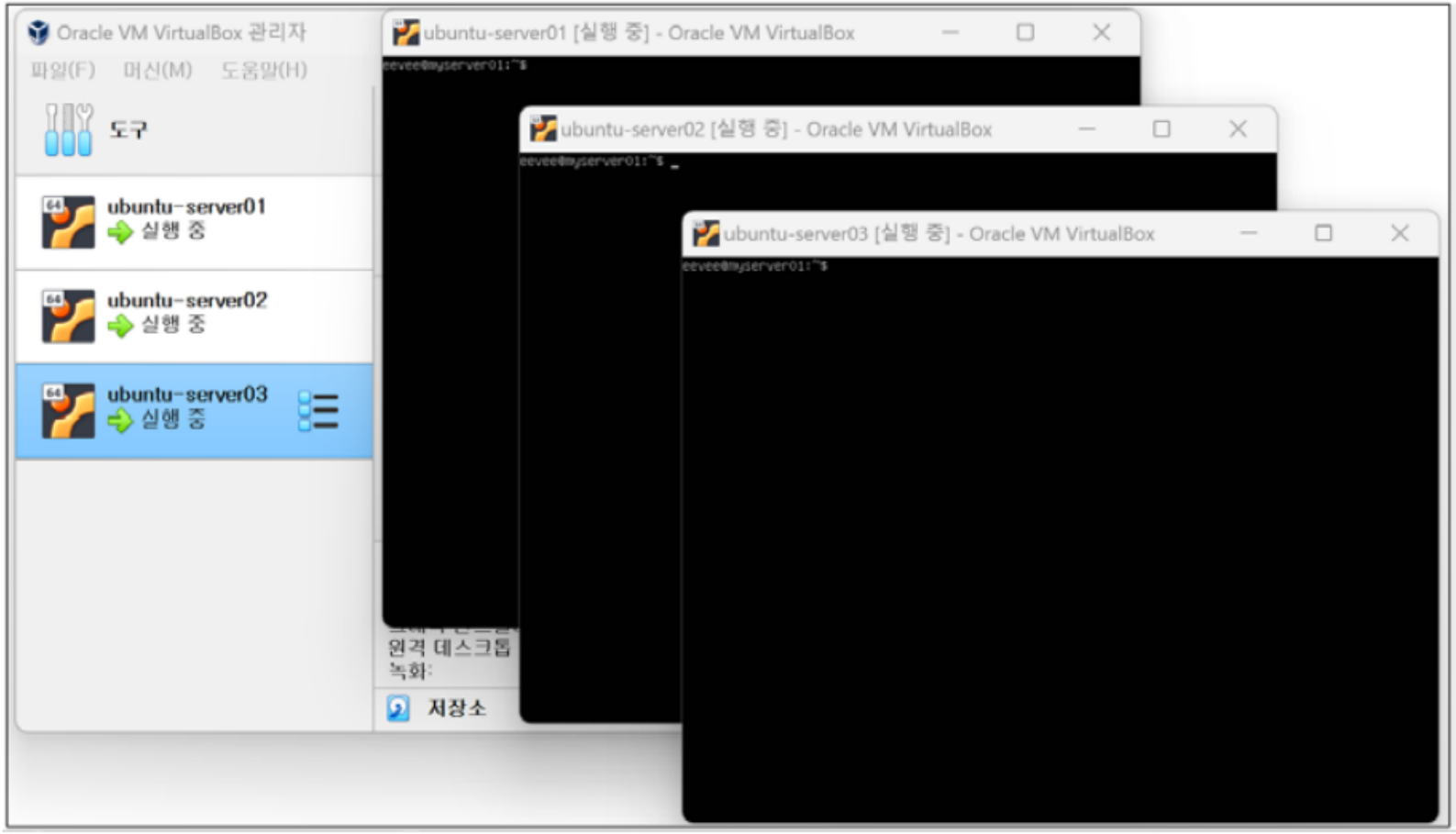
04 가상머신 복제

세번째 가상머신 복제

- 두번째 머신 복제 방식 그대로 반복

05 가상머신 복제

가상머신 3 개 동시 실행



06 호스트 이름 변경

두 번째 가상머신의 호스트 이름 변경

```
eevee@myserver01:/$ sudo hostnamectl set-hostname myserver02    ❶
eevee@myserver01:/$ cat /etc/hostname                           ❷
myserver02
eevee@myserver01:/$ sudo reboot now                             ❸
```

- 두번째 가상머신의 콘솔에서 직접 입력
- 수정 후 재부팅

07 호스트 이름 변경

세 번째 가상머신의 호스트 이름 변경

```
eevee@myserver01:/$ sudo hostnamectl set-hostname myserver02
eevee@myserver01:/$ cat /etc/hostname
myserver02
eevee@myserver01:/$ sudo reboot now
```

myserver02

```
eevee@myserver01:/$ sudo hostnamectl set-hostname myserver03
eevee@myserver01:/$ cat /etc/hostname
myserver03
eevee@myserver01:/$ sudo reboot now
```

myserver03

- 두번째 가상머신의 호스트 이름 변경 과정 반복

08 IP 주소 변경

복제된 가상머신의 IP 주소 변경

```
eevee@myserver02:/$ ifconfig          ❶
...(중략)
enp0s3: inet 10.0.2.4
...(생략)
```

- 두 번째, 세 번째 가상머신은 복제된 머신이므로 위와 같이 IP 주소 동일

09 IP 주소 변경

복제된 가상머신의 IP 주소 변경

- 네트워크 config 파일(yaml 파일) 수정하여 IP 주소 변경

```
eevee@myserver02:/$ sudo vim /etc/netplan/00-installer-config.yaml
```

- myserver02 노드의 IP 주소를 10.0.2.5로 변경

```
network:
ethernets:
  enp0s3:
addresses: [10.0.2.5/24]
routes:
- to: default
  via: 10.0.2.1
nameservers:
  addresses: [8.8.8.8]
version: 2
```

10 IP 주소 변경

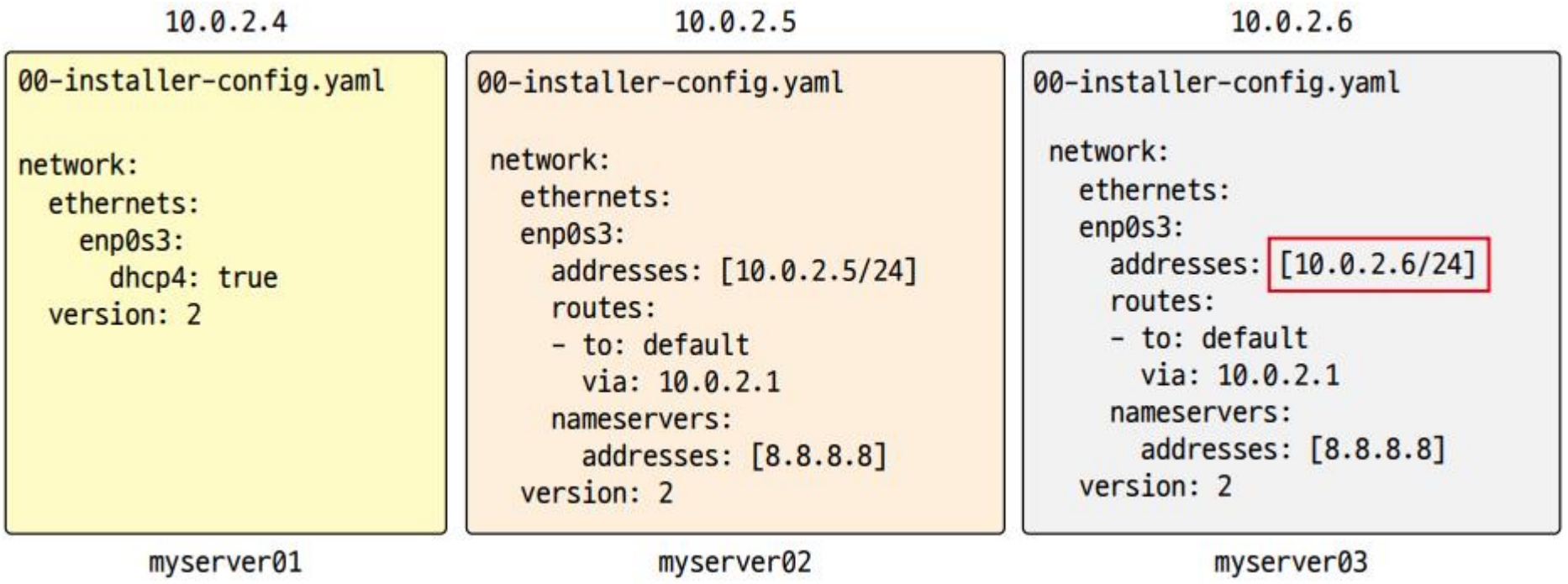
복제된 가상머신의 IP 주소 변경

```
eevee@myserver02:/$ sudo netplan apply          ❸
eevee@myserver02:/$ ifconfig                    ❹
...(중략)
enp0s3: inet 10.0.2.5
...(생략)
```

- netplan apply 명령어를 통해 IP 주소 변경 사항 적용
- ifconfig 로 IP 주소 변경 확인

11 IP 주소 변경

복제된 가상머신의 IP 주소 변경



- 동일한 방식으로 myserver03 노드의 IP 주소를 10.0.2.6으로 변경

12 DNS 설정

세대의 가상머신간 통신을 위한 DNS 설정

```
eevee@myserver02:~$ sudo vim /etc/hosts
```

- Vi 에디터로 /etc/hosts 파일 아래와 같이 수정

```
127.0.0.1 localhost
127.0.1.1 myserver02          ❶

10.0.2.4 myserver01           ❷
10.0.2.5 myserver02           ❸
10.0.2.6 myserver03           ❹

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

13 DNS 설정

Ping 명령으로 통신 확인

```
eevee@myserver02:~$ ping myserver01          ❶
PING myserver01 (10.0.2.4) 56(84) bytes of data.
64 bytes from myserver01 (10.0.2.4): icmp_seq=1 ttl=64 time=0.706 ms
64 bytes from myserver01 (10.0.2.4): icmp_seq=2 ttl=64 time=0.824 ms
64 bytes from myserver01 (10.0.2.4): icmp_seq=3 ttl=64 time=1.71 ms
64 bytes from myserver01 (10.0.2.4): icmp_seq=4 ttl=64 time=1.26 ms
^C                                           ❷
--- myserver01 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3023ms
rtt min/avg/max/mdev = 0.706/1.127/1.714/0.397 ms
```

- Myserver02에서 myserver01로 ping 작동 확인
- Ctrl + c 로 ping 명령 종료

14 DNS 설정

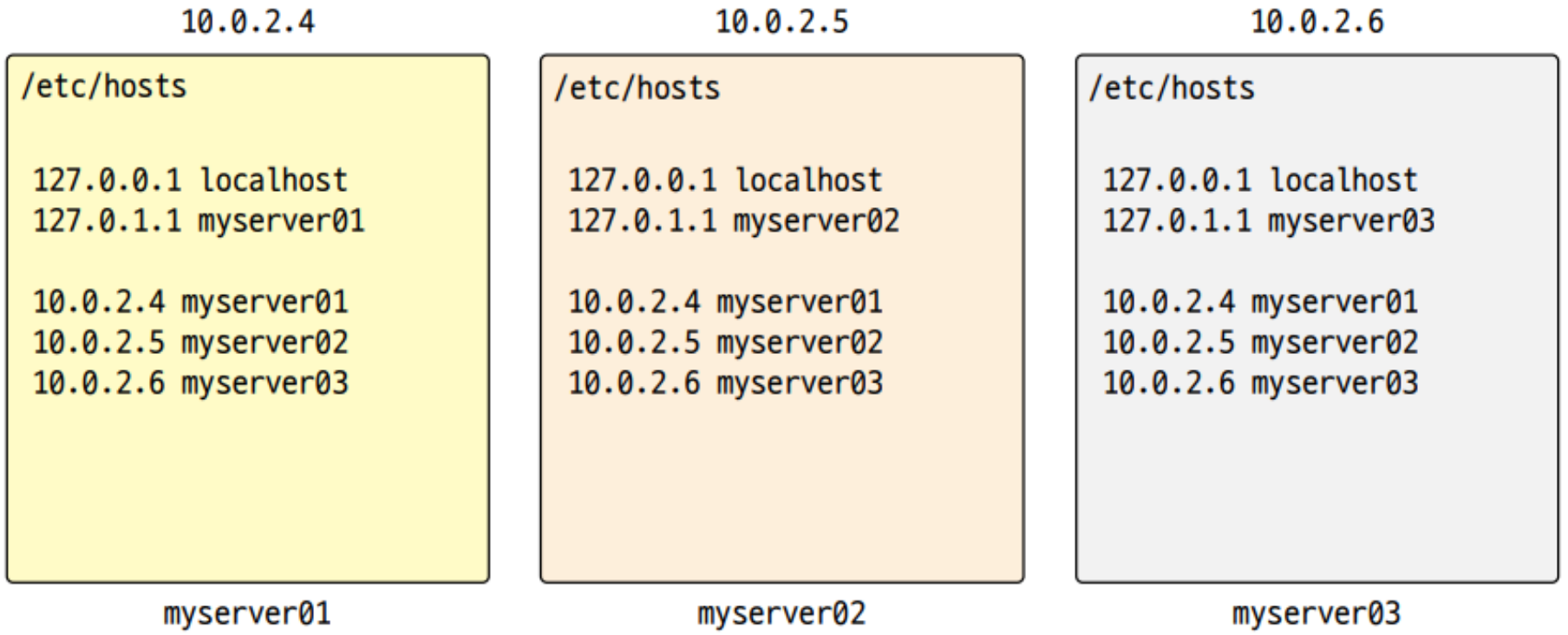
ssh 명령으로 서버 접속 확인

```
eevee@myserver02:~$ ssh myserver01          ❶
eevee@myserver01:~$                         ❷
eevee@myserver01:~$ exit                    ❸
eevee@myserver02:~$                         ❹
```

- 접속 후 프롬프트의 호스트 이름 변경 확인

15 DNS 설정

세개의 가상머신의 DNS를 동일한 방식으로 설정



- 가상머신 끼리 ssh를 통해 접속 확인

16 UFW 방화벽 설정

노드간 자유로운 통신을 위해 ufw(Uncomplicated FireWall) 정지 시킴

```
eevee@myserver01:~$ sudo ufw status          ❶
Status: inactive
```

- Ufw의 status 상태 확인 -> inactive 상태이면 방화벽 정지 작업 불필요
- Active 상태이면 정지 작업 실행

17 UFW 방화벽 설정

노드간 자유로운 통신을 위해 ufw(Uncomplicated FireWall) 정지 시킴

```
eevee@myserver01:~$ sudo ufw disable          ❶
Firewall stopped and disabled on system startup
eevee@myserver01:~$ sudo ufw status
Status: inactive
eevee@myserver01:~$ ssh myserver02

eevee@myserver02:~$ sudo ufw disable          ❷
[sudo] password for eevee:
Firewall stopped and disabled on system startup
eevee@myserver02:~$ sudo ufw status
Status: inactive
eevee@myserver02:~$ ssh myserver03
eevee@myserver03's password:

eevee@myserver03:~$ sudo ufw disable          ❸
[sudo] password for eevee:
Firewall stopped and disabled on system startup
eevee@myserver03:~$ sudo ufw status
Status: inactive
```

- 클러스터를 구성하는 모든 노드 myserver01, myserver02, myserver03에서 모두 ufw 정지

18 네트워크 설정

IPv4를 포워딩하여 iptables가 연결된 트래픽 확인하도록 수정

```
eevee@myserver01:~$ sudo -i                                          ❶
root@myserver01:~# cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf  ❷
overlay                                                              ❸
br_netfilter                                                         ❹
EOF                                                                  ❺
```

- Sudo vi /etc/modules-load.d/k8s.conf 명령으로 실행할 수도 있는데 굳이 cat <<EOF를 사용하는 이유 => 이런 형식이 스크립트를 통한 자동 설정에 적합하기 때문
- vi를 사용한다는 것은 수동 작업을 가정한 것임

19 네트워크 설정

IPv4를 포워딩하여 iptables가 연결된 트래픽 확인하도록 수정

```
eevee@myserver01:~$ sudo -i                                          ❶
root@myserver01:~# cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf  ❷
overlay                                                              ❸
br_netfilter                                                         ❹
EOF                                                                  ❺
```

- **Overlay**: 리눅스 커널의 네트워크 드라이버를 가리킴. 다른 호스트에 존재하는 파드 간의 네트워크 연결을 가능하게 하는 기술. 여러 개의 독립적인 네트워크 레이어를 겹쳐서 하나로 연결된 네트워크 생성. Overlay를 통해 서로 다른 호스트에 존재하는 파드가 동일한 네트워크에 존재하는 것처럼 통신 가능 => 시스템 부팅 시 overlay 네트워크 드라이버를 로드하도록 설정
- **Br_netfilte**: 네트워크 패킷 처리 관련 모듈. Iptables/netfilter 규틱 적용. 컨테이너와 호스트 간의 인터페이스 등에서 발생하는 트래픽에 대해 규칙을 적용해 트래픽 관리

- *Sudo vi /etc/modules-load.d/k8s.conf 명령으로 실행할 수도 있는데 굳이 cat <<EOF를 사용하는 이유 => 이런 형식이 스크립트를 통한 자동 설정에 적합하기 때문*
- *vi를 사용한다는 것은 수동 작업을 가정한 것임*

20 네트워크 설정

리눅스 커널 모듈 수동 로딩

```
root@myserver01:~# sudo modprobe overlay          ❶
root@myserver01:~# sudo modprobe br_netfilter     ❷
```

21 네트워크 설정

쿠버네티스 네트워킹을 위한 커널 설정

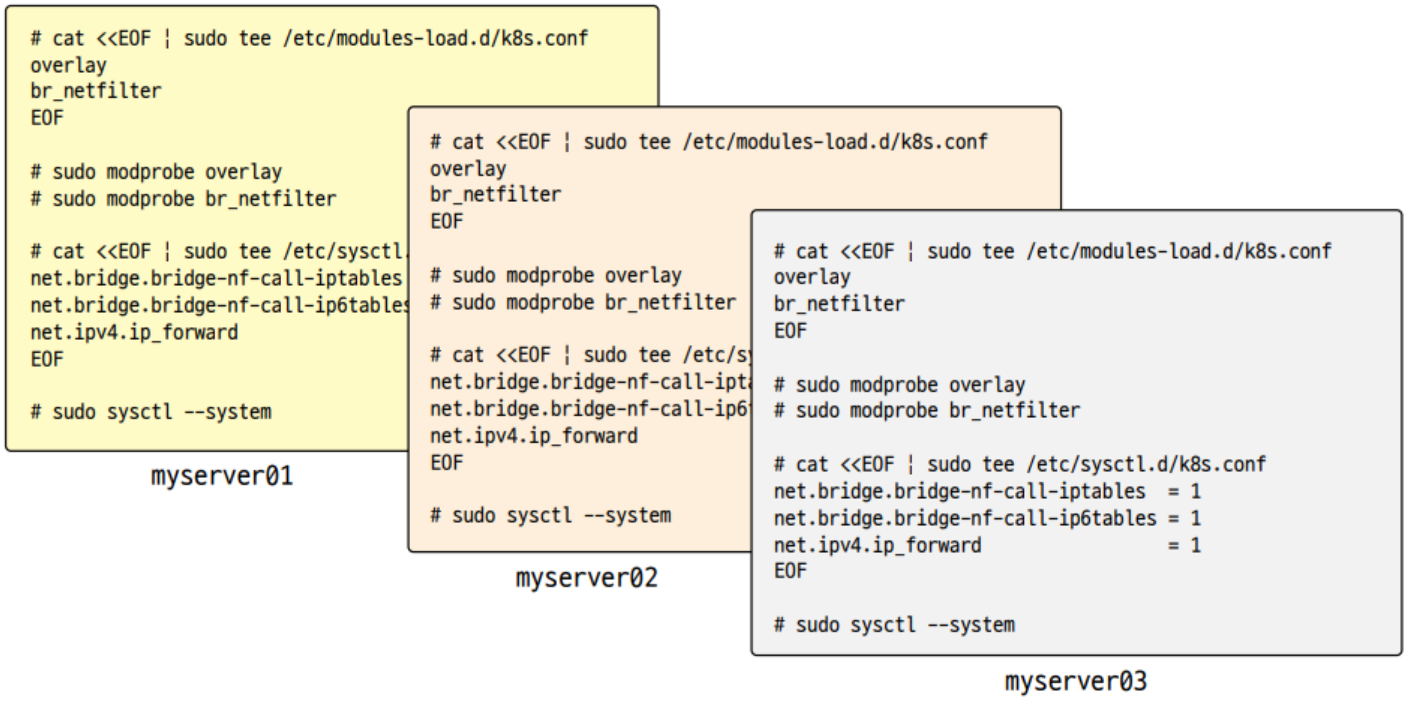
```
root@myserver01:~# cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf          ❶
net.bridge.bridge-nf-call-iptables  = 1                                  ❷
net.bridge.bridge-nf-call-ip6tables = 1                                  ❸
net.ipv4.ip_forward                 = 1                                  ❹
EOF                                                                      ❺

net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
```

- (2) Iptable의 규칙이 bridge 트래픽에 적용되도록 설정
- (4) 파드와 노드간 패킷 포워딩을 위한 설정

22 네트워크 설정

재부팅 전 sysctl 매개변수 적용

```
root@myserver01:~# sudo sysctl --system
```



- 동일한 방식으로 myserver01을 포함한 모든 가상머신에 대해 동일하게 작업

23 Containerd 설정

쿠버네티스가 컨테이너 런타임으로 containerd를 사용하도록 설정

```
eevee@myserver01:~$ sudo mkdir -p /etc/containerd      ❶
eevee@myserver01:~$ containerd config default ¦ sudo tee /etc/containerd/config.toml
> /dev/null                                            ❷
```

- Containerd의 설정값을 출력하면서 동시에 config.toml 파일로 저장

  (*) toml(Tom's Obvious, Minimal Language) simple human-readble 파일 포맷

```
eevee@myserver01:~$ sudo vim /etc/containerd/config.toml       ❶
…(중략)
  [plugins. " io.containerd.grpc.v1.cri " .containerd.runtimes.runc]
  …(중략)
    [plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc.options]
      SystemdCgroup = true                                     ❷
```

- Config.toml 파일을 열어서 SystemdCgroup값을 true 로 변경 후 저장

24 Containerd 설정

Config.toml 파일의 SystemdCgroup값 true 로 변경

```
[plugins."io.containerd.grpc.v1.cri".containerd.runtimes]

  [plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc]
    base_runtime_spec = ""
    cni_conf_dir = ""
    cni_max_conf_num = 0
    container_annotations = []
    pod_annotations = []
    privileged_without_host_devices = false
    runtime_engine = ""
    runtime_path = ""
    runtime_root = ""
    runtime_type = "io.containerd.runc.v2"

    [plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc.options]
      BinaryName = ""
      CriuImagePath = ""
      CriuPath = ""
      CriuWorkPath = ""
      IoGid = 0
      IoUid = 0
      NoNewKeyring = false
      NoPivotRoot = false
      Root = ""
      ShimCgroup = ""
      SystemdCgroup = false

  [plugins."io.containerd.grpc.v1.cri".containerd.untrusted_workload_runtime]
    base_runtime_spec = ""
    cni_conf_dir = ""
    cni_max_conf_num = 0
    container_annotations = []
    pod_annotations = []
    privileged_without_host_devices = false
    runtime_engine = ""
    runtime_path = ""
    runtime_root = ""
    runtime_type = ""
```

```
[plugins."io.containerd.grpc.v1.cri".containerd.runtimes]

  [plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc]
    base_runtime_spec = ""
    cni_conf_dir = ""
    cni_max_conf_num = 0
    container_annotations = []
    pod_annotations = []
    privileged_without_host_devices = false
    runtime_engine = ""
    runtime_path = ""
    runtime_root = ""
    runtime_type = "io.containerd.runc.v2"

    [plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc.options]
      BinaryName = ""
      CriuImagePath = ""
      CriuPath = ""
      CriuWorkPath = ""
      IoGid = 0
      IoUid = 0
      NoNewKeyring = false
      NoPivotRoot = false
      Root = ""
      ShimCgroup = ""
      SystemdCgroup = true

  [plugins."io.containerd.grpc.v1.cri".containerd.untrusted_workload_runtime]
    base_runtime_spec = ""
    cni_conf_dir = ""
    cni_max_conf_num = 0
    container_annotations = []
    pod_annotations = []
    privileged_without_host_devices = false
    runtime_engine = ""
    runtime_path = ""
    runtime_root = ""
    runtime_type = ""
```

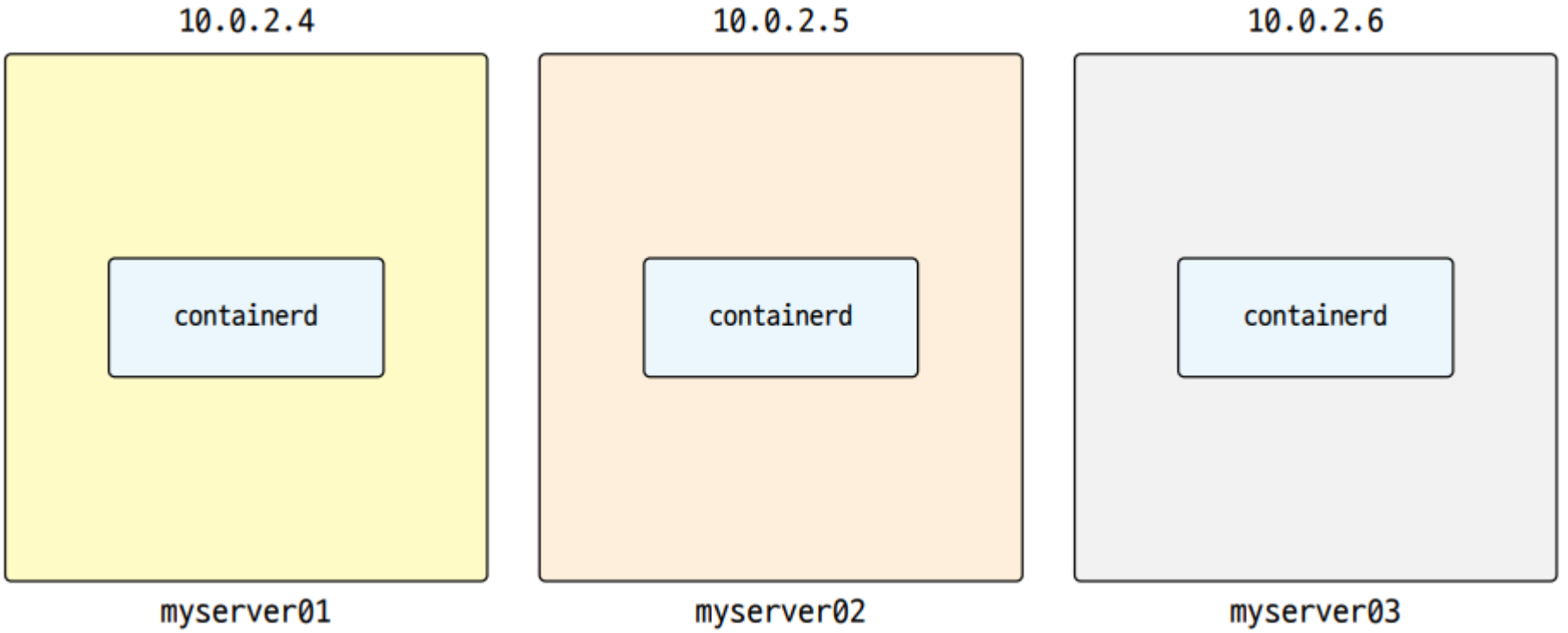
25 네트워크 설정

Containerd 재가동 및 작동 확인

```
eevee@myserver01:~$ sudo systemctl restart containerd          ❶
eevee@myserver01:~$ sudo systemctl enable containerd           ❷

eevee@myserver01:~$ sudo systemctl status containerd           ❸
● containerd.service - containerd container runtime
     Loaded: loaded (/lib/systemd/system/containerd.service; enabled; vendor preset: e>
     Active: active (running) since Sun 2023-11-05 05:11:35 UTC; 11s ago
       Docs: https://containerd.io
   Main PID: 114339 (containerd)
      Tasks: 10
     Memory: 13.4M
        CPU: 113ms
     CGroup: /system.slice/containerd.service
             └─114339 /usr/bin/containerd
```

26 네트워크 설정

Containerd 재가동 및 작동 확인



- 동일한 방식으로 myserver01, myserver02, myserver03 모두에 containerd 설정 변경

27 Swap 메모리 비활성화

원활한 컨테이너 관리를 위한 swap 메모리 영역 비활성화

```
eevee@myserver01:~$ free -h                                    ❶
               total        used        free      shared  buff/cache   available
Mem:           7.8Gi       389Mi       1.2Gi        16Mi       6.1Gi       7.1Gi
Swap:          4.0Gi          0B       4.0Gi                   ❷
```

- Free –h 명령어로 메모리 공간 확인
- Swap 영역이 0으로 표시된다면 swap 영역이 이미 비 활성화 되어 있는 것임

```
eevee@myserver01:~$ cat /proc/swaps                                    ❶
Filename                        Type            Size            Used            Priority
/swap.img                       file            4194300         0               -2          ❷
```

- /proc/swaps 의 내용을 확인하는 방식도 있음

28 Swap 메모리 비활성화

원활한 컨테이너 관리를 위한 swap 메모리 영역 비활성화

```
eevee@myserver01:~$ sudo -i                ❶
[sudo] password for eevee:
root@myserver01:~# swapoff --all           ❷
root@myserver01:~#
```

- 관리자 모드로 변경 후 swapoff –all 명령으로 swap 메모리 비활성화

29 Swap 메모리 비활성화

Swap 메모리 비활성화 작업 결과 확인

```
root@myserver01:~# free -h                          ❶
               total        used        free      shared  buff/cache   available
Mem:           7.8Gi       385Mi       1.2Gi        16Mi       6.1Gi       7.1Gi
Swap:             0B          0B          0B

root@myserver01:~# cat /proc/swaps                  ❷
Filename                Type            Size            Used            Priority
```

```
root@myserver01:~# vim /etc/fstab                       ❶
#/swap.img      none    swap    sw      0       0       ❷
```

- Fstab 파일의 내용 중 /swap.img 라인 주석 처리 => 재부팅시 swap 메모리 재활성화 방지

```
root@myserver01:~# shutdown -r now
```

- 시스템 재부팅

30 Swap 메모리 비활성화

3개 호스트 swap 메모리 비활성화

```
eevee@myserver01:~$ sudo -i
root@myserver01:~# swapoff --all

root@myserver01:~# vim /etc/fstab
#/swap.img      none    swap    sw

root@myserver01:~# shutdown -r now
```

myserver01

```
eevee@myserver03:~$ sudo -i
root@myserver03:~# swapoff --all

root@myserver03:~# vim /etc/fstab
#/swap.img      none    swap    sw

root@myserver03:~# shutdown -r now
```

myserver03

```
eevee@myserver02:~$ sudo -i
root@myserver02:~# swapoff --all

root@myserver02:~# vim /etc/fstab
#/swap.img      none    swap    sw

root@myserver02:~# shutdown -r now
```

myserver02

- 동일한 방식으로 myserver01, myserver02, myserver03 모두에 swap 메모리 비활성화

서로에게, 자신에게 친절합시다 - 허준이

DAEJEON UNIVERSITY
SINCE 1980
대전대학교