

Ch01. 개요 및 환경 설정



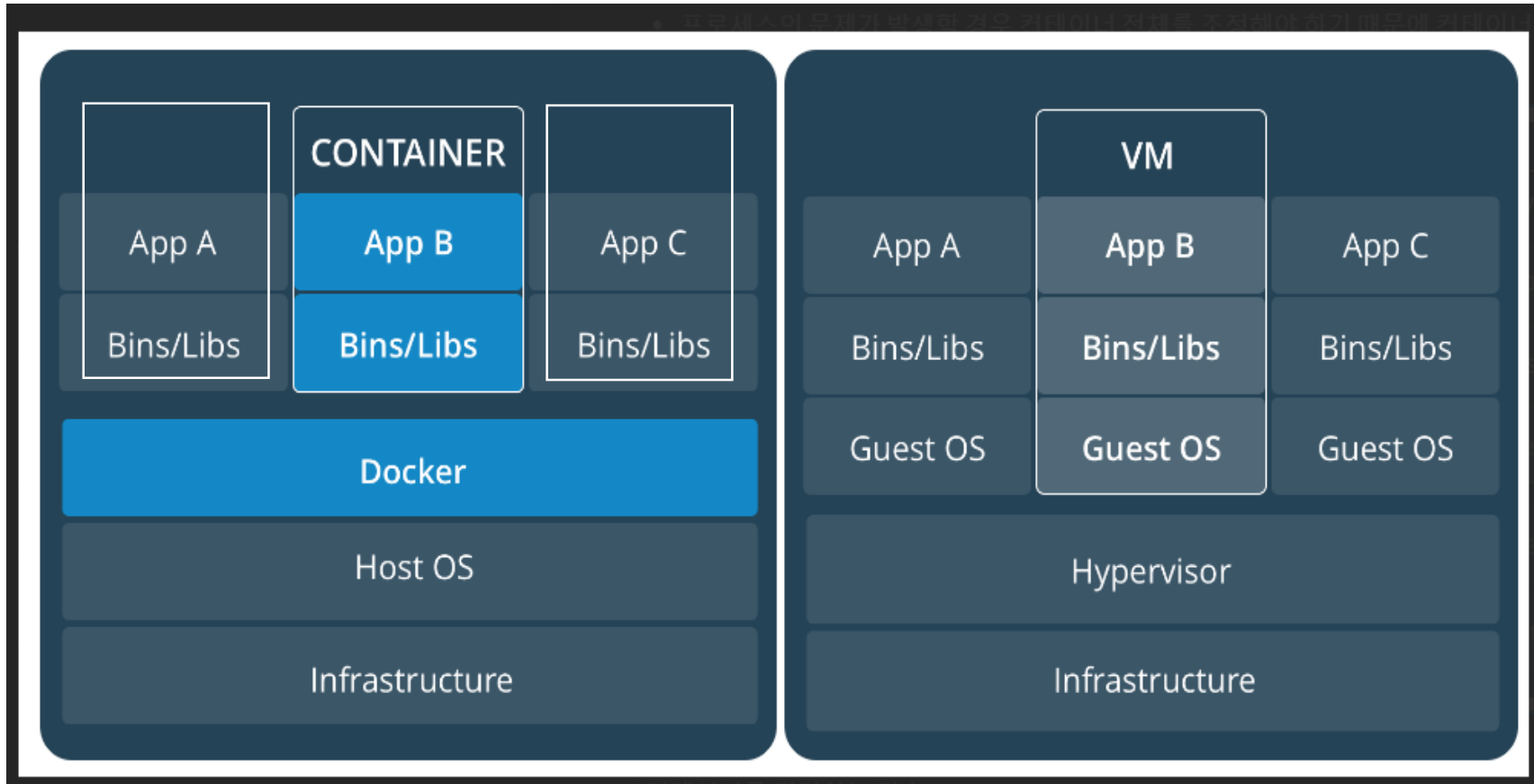
Introduction

- Docker란 무엇인가
 - 컨테이너를 사용하여 응용 프로그램을 더 쉽게 만들고, 배포하고 실행할 수 있는 도구이며,
 - 컨테이너 기반의 오픈소스 가상화 플랫폼임
- Container란 무엇인가
 - 컨테이너 안에 다양한 프로그램, 실행환경을 컨테이너로 추상화하고 동일한 인터페이스를 제공하여 프로그램의 배포 및 관리를 단순하게 해 줌
 - 프로그램을 손쉽게 이동 배포 관리를 할 수 있게 해 줌
 - 컨테이너는 코드와 모든 종속성을 패키징하여 응용 프로그램이 한 컴퓨팅 환경에서 다른 컴퓨팅 환경으로 빠르고 안정적으로 실행되도록 하는 소프트웨어의 표준단위임
 - 컨테이너 이미지는 코드, 런타임, 시스템, 도구, 시스템 라이브러리 및 설정과 같은 응용 프로그램을 실행하는데 필요한 모든 것을 포함하는 가볍고 독립적이며 실행 가능한 소프트웨어 패키지임.

Introduction

- Docker는 다양한 클라우드 서비스 모델과 같이 사용가능 (IaaS, PaaS, SaaS)
 - **이미지** : 필요한 프로그램과 라이브러리, 소스를 설치한 뒤 만든 하나의 파일
 - **컨테이너** : 이미지를 격리하여 독립된 공간에서 실행한 가상 환경
- 컨테이너가 해결하는 것들
 - 컨테이너는 가상 머신을 사용해 각 마이크로 서비스를 격리(isolate)하는 기술임
 - 컨테이너는 가상 머신처럼 하드웨어를 전부 구현하지 않기 때문에 매우 빠른 실행이 가능함

Introduction



- Hypervisor : 가상 머신(Virtual Machine, VM)을 생성하고, 구동하는 소프트웨어

쿠버네티스

■ 쿠버네티스란

- 오케스트레이션 (컨테이너 관리 및 운영을 자동화)하는 도구임
- 쿠버네티스는 구글에서 개발해 2014년 오픈소스로 공개함
- 쿠버네티스는 그리스어에서 유래, 영어로는 helmsman(조타수) 나 pilot(조종사)등을 의미함

■ 쿠버네티스에서 가능한 것

헬스 체크와
정지한 컨테이너 재시작

엔드포인트 제공과
로드 밸런싱

소비 리소스를 고려해
컨테이너의 배포

롤링 업데이트와
롤백

스토리지 관리

기밀 정보 및
구성 정보 저장

■ 쿠버네티스에서 불가능한 것

자동 빌드

로그의 기록 및 감시

미들웨어 기능
(메시지 브로커, DB등)

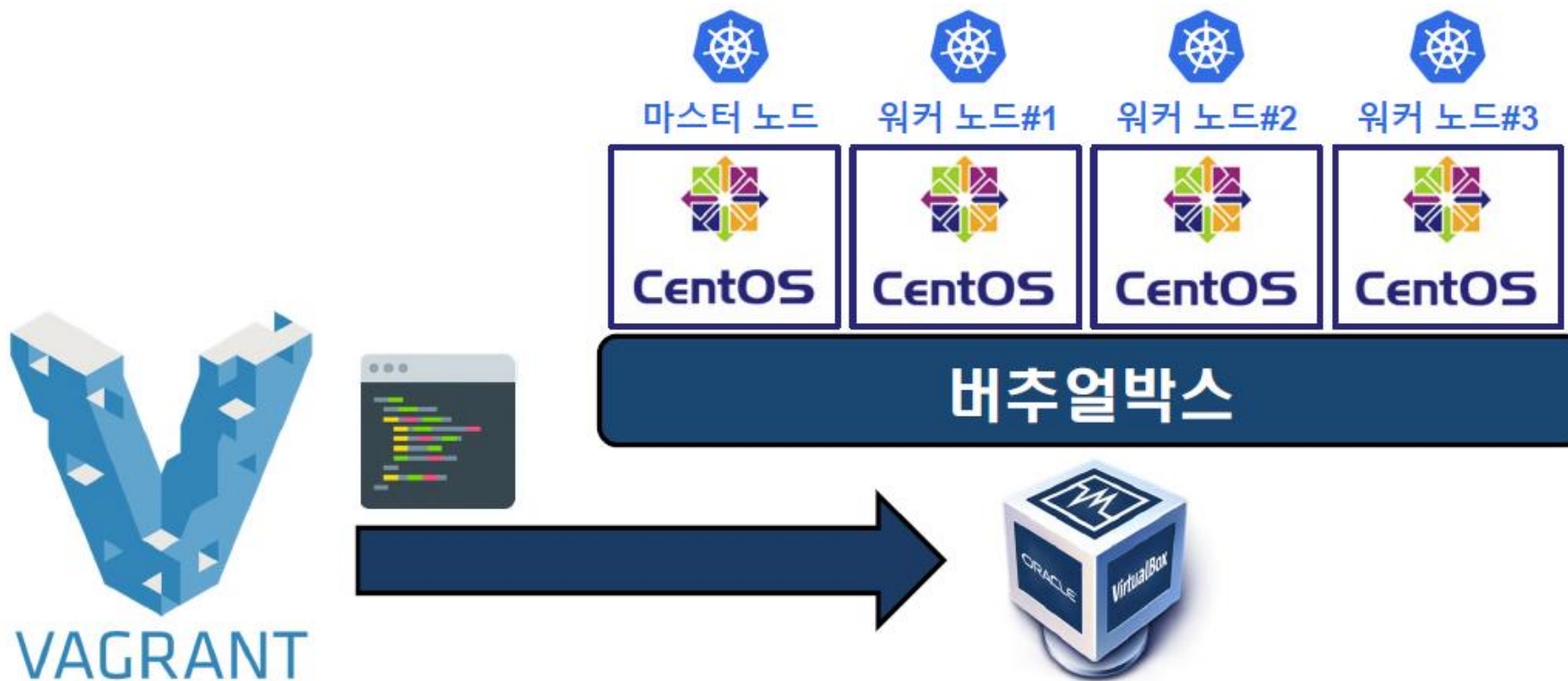
서버 자체의 관리

쿠버네티스

- 쿠버네티스는 컨테이너 오케스트레이션 도구의 사실상 표준
- 컨테이너 오케스트레이션 도구에는 여러가지 선택지가 있지만 (Marathon, Mesos등) 쿠버네티스가 2018년 7월에 출시된 도커 데스크탑과 더불어 컨테이너 오케스트레이션 도구의 표준이 되었음.
- 쿠버네티스는 대량의 컨테이너 및 서버로 구축된 서비스의 관리 및 운영에 대한 부하를 줄일 수 있다. 이러한 시스템의 집합을 **클러스터**라 하며,
- 쿠버네티스에서 관리하는 클러스터를 **쿠버네티스 클러스터**라고 하고, 클러스터를 구성하는 각 서버를 **노드**라고 함

쿠버네티스 테스트 환경

- vagrant는 개발 환경을 구축하고 관리하기 위한 소프트웨어 도구임.
 - 이것은 가상화 플랫폼위에 경량의, 재현 가능하며, 이식 가능한 개발환경을 쉽게 생성할 수 있게 해줌
 - vagrant에서 코드를 VM에 보내게 되면, virtual box를 통해서 CentOS를 설치하게 해주고, 쿠버네티스 환경을 구성해줌



쿠버네티스 테스트 환경

- Vagrant 설치 (<https://www.vagrant.com/>)
- VirtualBox 6.1 설치 (<https://virtualbox.org>)


Community

Self-managed | always free

Download

Download the Vagrant binary and run locally or within your environments.

Developer / Vagrant / Install



Install Vagrant

2.4.1 (latest) ▾





Install Vagrant

2.2.14 ▾

Windows

Binary download

<div>I686</div> <div>Version: 2.2.14</div> <div>Download </div>	<div>X86_64</div> <div>Version: 2.2.14</div> <div>Download </div>
--	--

쿠버네티스 테스트 환경

- Vagrant에서 사용하는 코드는 Vagrantfile(script 파일)을 이용함
 - /ch1/1.2/Vagrantfile
 - 이 파일의 의미는 교재 page 87 참고할 것

```
13 Vagrant.configure("2") do |config|
14
15   #=====#
16   # Master Node #
17   #=====#
18
19   config.vm.define "m-k8s-#{k8s_V[0..5]}" do |cfg|
20     cfg.vm.box = "sysnet4admin/CentOS-k8s"
21     cfg.vm.provider "virtualbox" do |vb|
22       vb.name = "m-k8s-#{k8s_V[0..5]}(github_SysNet4Admin)"
23       vb.cpus = 2
24       vb.memory = 1746
25       vb.customize ["modifyvm", :id, "--groups", "/k8s-C#{k8s_V[0..5]}-ctrd-#{ctrd_V[0..5]}"]
26     end
27     cfg.vm.host_name = "m-k8s"
28     cfg.vm.network "private_network", ip: "192.168.1.10"
29     cfg.vm.network "forwarded_port", guest: 22, host: 60010, auto_correct: true, id: "ssh"
30     cfg.vm.synced_folder "../data", "/vagrant", disabled: true
31     cfg.vm.provision "shell", path: "k8s-env-build.sh", args: N
32     cfg.vm.provision "shell", path: "k8s-pkg-cfg.sh", args: [k8s_V, docker_V, ctrd_V]
33     cfg.vm.provision "shell", path: "master-node.sh"
34   end
```

```
35
36   #=====#
37   # Worker Nodes #
38   #=====#
39
40   (1..N).each do |i|
41     config.vm.define "w#{i}-k8s-#{k8s_V[0..5]}" do |cfg|
42       cfg.vm.box = "sysnet4admin/CentOS-k8s"
43       cfg.vm.provider "virtualbox" do |vb|
44         vb.name = "w#{i}-k8s-#{k8s_V[0..5]}(github_SysNet4Admin)"
45         vb.cpus = 1
46         vb.memory = 1024
47         vb.customize ["modifyvm", :id, "--groups", "/k8s-C#{k8s_V[0..5]}-ctrd-#{ctrd_V[0..5]}"]
48       end
49       cfg.vm.host_name = "w#{i}-k8s"
50       cfg.vm.network "private_network", ip: "192.168.1.10#{i}"
51       cfg.vm.network "forwarded_port", guest: 22, host: "6010#{i}", auto_correct: true, id: "ssh"
52       cfg.vm.synced_folder "../data", "/vagrant", disabled: true
53       cfg.vm.provision "shell", path: "k8s-env-build.sh", args: N
54       cfg.vm.provision "shell", path: "k8s-pkg-cfg.sh", args: [k8s_V, docker_V, ctrd_V]
55       cfg.vm.provision "shell", path: "work-nodes.sh"
56     end
57   end
58
59 end
```

쿠버네티스 테스트 환경

- Vagrant 설치 후 Command 창에서 `vagrant --version` [enter] 확인 (2.2.14)
- https://github.com/sysnet4admin/_Lecture_k8s.starterkit 에서 code Download 함

쿠버네티스 테스트 환경

■ 쿠버네티스 설치 및 확인

- 해당 디렉토리 내에 vagrantfile을 이용하여 쿠버네티스 환경 구축
- vagrant up을 실행하면 아래와 같이 CentOS vm이 생성이 진행되고,
- 이후에 kubectl을 이용해서 실제 쿠버네티스 환경을 가상머신인 CentOS에 배포됨.
- 가상머신을 삭제하고 싶을 땐 (vagrant destroy -f [enter])

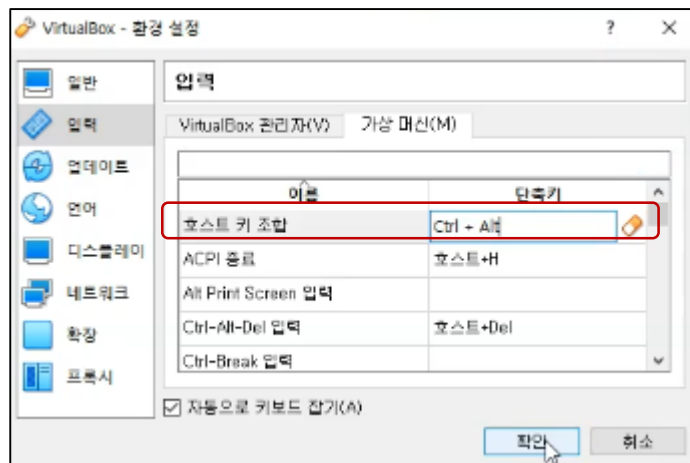
```
C:\ 명령 프롬프트

d:\Program Files\HashiCorp\Lecture_k8s_starter.kit-main\ch1\1.2\k8s-min-5GiB>
d:\Program Files\HashiCorp\Lecture_k8s_starter.kit-main\ch1\1.2\k8s-min-5GiB>
d:\Program Files\HashiCorp\Lecture_k8s_starter.kit-main\ch1\1.2\k8s-min-5GiB>
d:\Program Files\HashiCorp\Lecture_k8s_starter.kit-main\ch1\1.2\k8s-min-5GiB>
d:\Program Files\HashiCorp\Lecture_k8s_starter.kit-main\ch1\1.2\k8s-min-5GiB>vagrant up
```



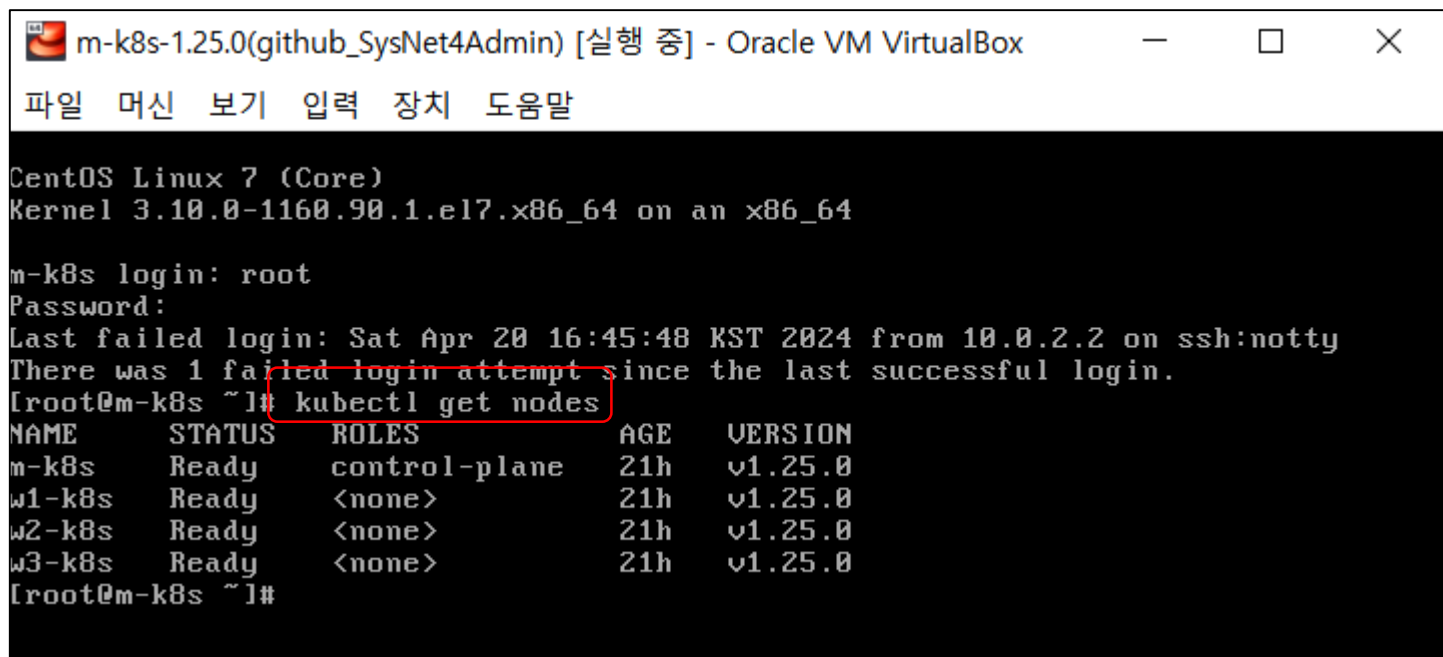
쿠버네티스 테스트 환경

- Virtual Box에 CentOS에서 쉽게 빠져 나오는 셋팅
 - centOS에 진입하면 쉽게 빠져 나올 수 없게 되는데, 이를 쉽게하기 위해 아래와 같이 설정함.
 - File > 입력 > 호스트 키 조합에서 'Ctrl + Alt' 키를 동시에 클릭함.



쿠버네티스 테스트 환경

- 쿠버네티스 환경 구축 확인
 - 마스터 노드를 클릭하여 root/vagrant로 로그인 한 후
 - 노드들이 생성되어 있는지 'kubectl get nodes' [enter]



The screenshot shows a terminal window titled "m-k8s-1.25.0(github_SysNet4Admin) [실행 중] - Oracle VM VirtualBox". The terminal output shows the user logging in as root, followed by the command "kubectl get nodes" which returns a table of nodes.

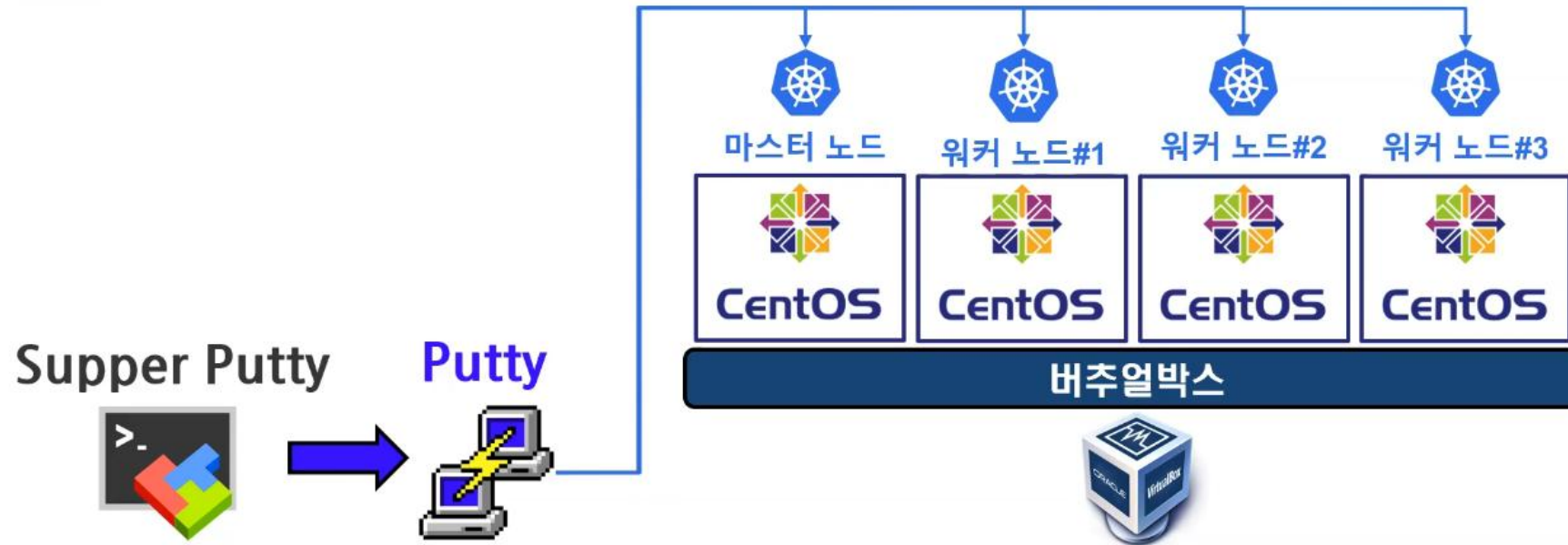
```
m-k8s login: root
Password:
Last failed login: Sat Apr 20 16:45:48 KST 2024 from 10.0.2.2 on ssh:notty
There was 1 failed login attempt since the last successful login.
[root@m-k8s ~]# kubectl get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
m-k8s	Ready	control-plane	21h	v1.25.0
w1-k8s	Ready	<none>	21h	v1.25.0
w2-k8s	Ready	<none>	21h	v1.25.0
w3-k8s	Ready	<none>	21h	v1.25.0

```
[root@m-k8s ~]#
```

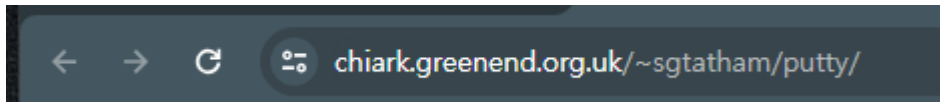
쿠버네티스 테스트 환경

- 터미널 프로그램으로 가상 머신 접속하기

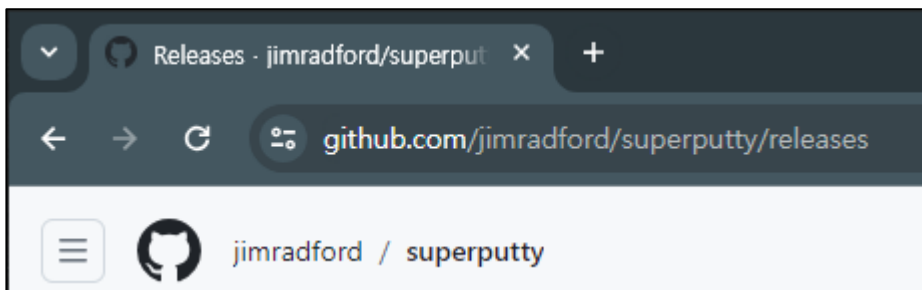


쿠버네티스 테스트 환경

- 터미널 프로그램으로 가상 머신 접속하기
 - putty 0.78을 아래의 URL에서 다운로드 함.



- SuperPutty 설치



Stable SuperPuTTY 1.4.0.9 Release


1.4.0.9

- Fixes several bugs and feature updates since 1.4.0.8 See [1.4.0.8...master](#) for all 79 commits

▼ Assets 4

 [SuperPuTTY-1.4.0.9.zip](#)

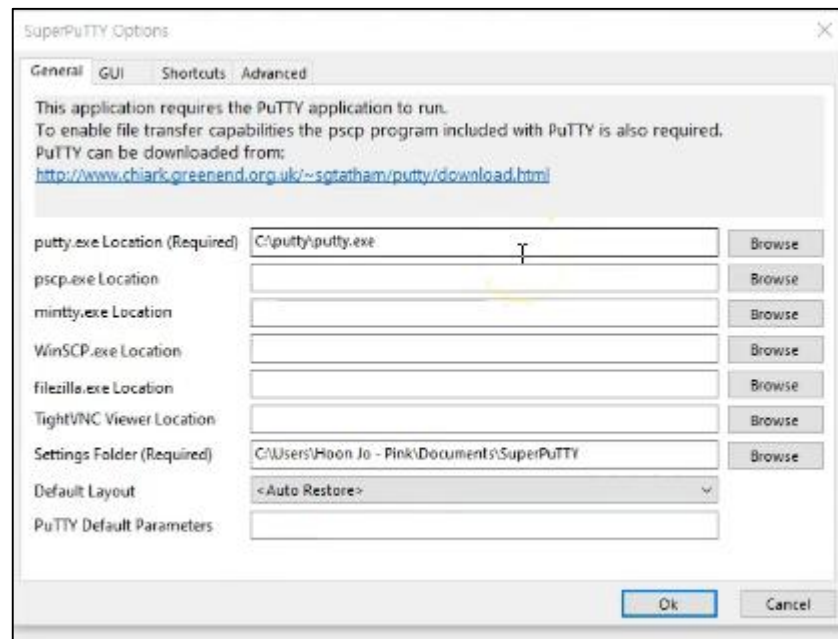
 [SuperPuttySetup-1.4.0.9.msi](#)

 [Source code \(zip\)](#)

 [Source code \(tar.gz\)](#)

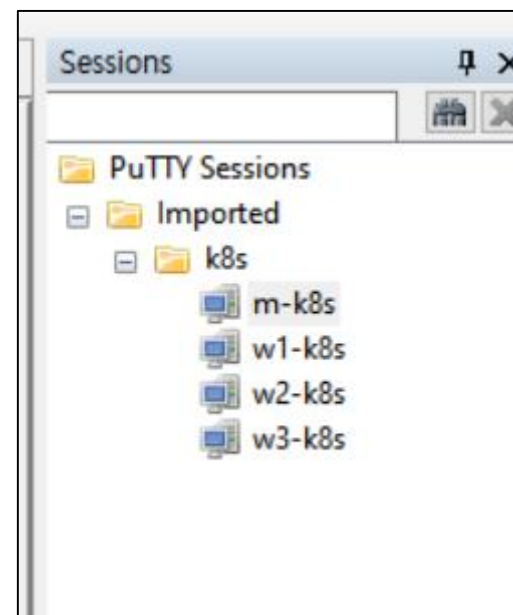
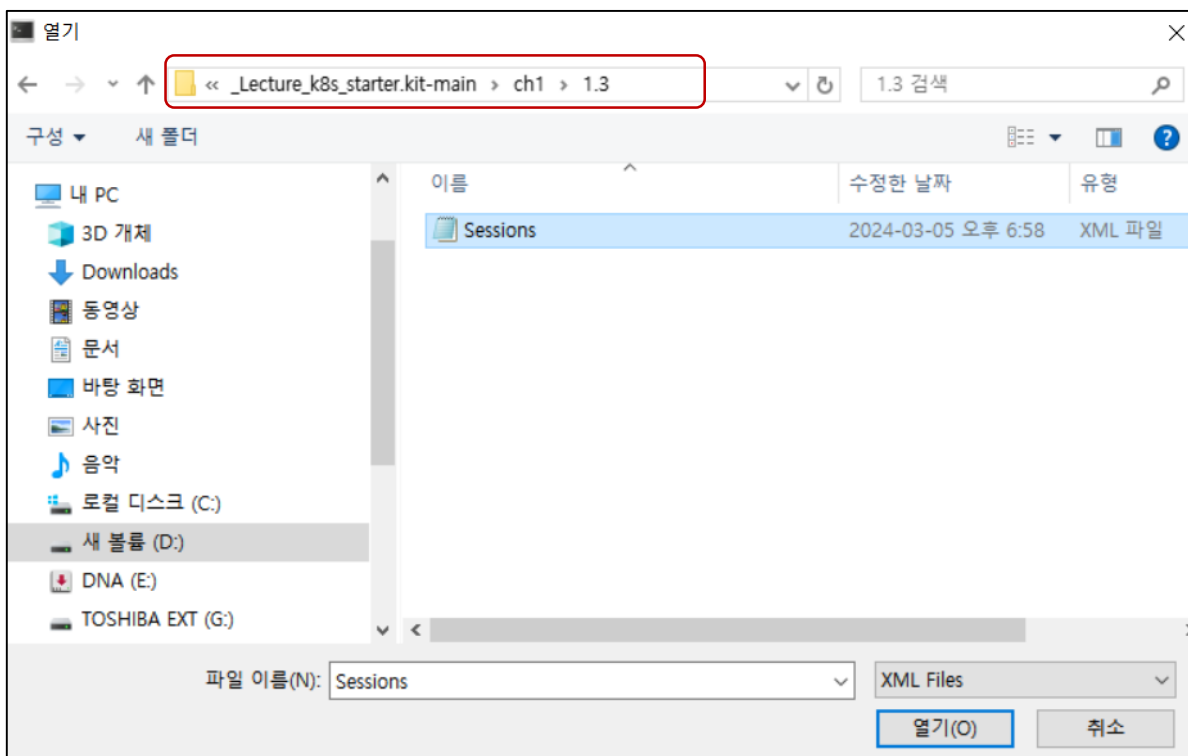
쿠버네티스 테스트 환경

- 터미널 프로그램으로 가상 머신 접속하기
 - SuperPutty 설정 (putty 위치 설정하기)



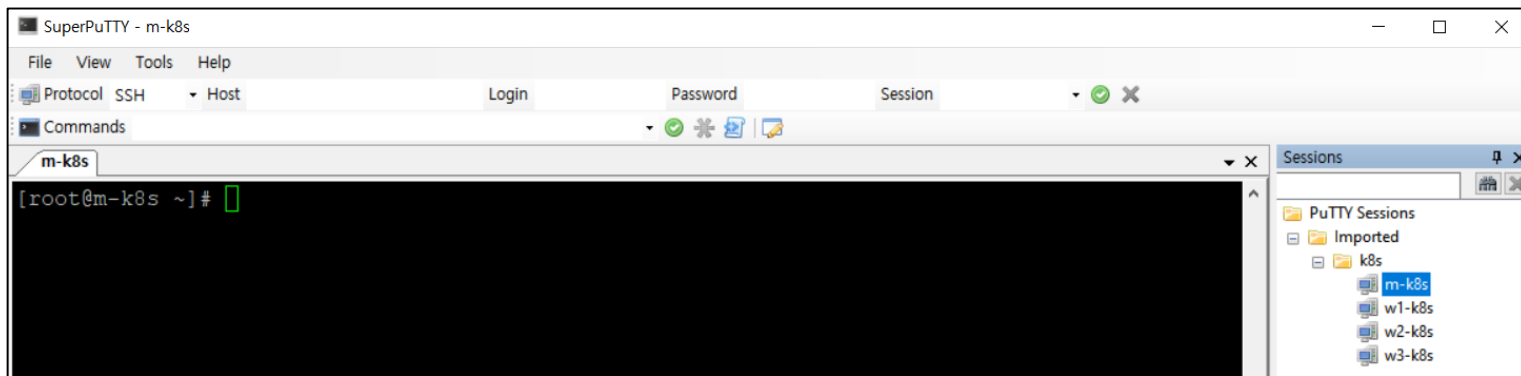
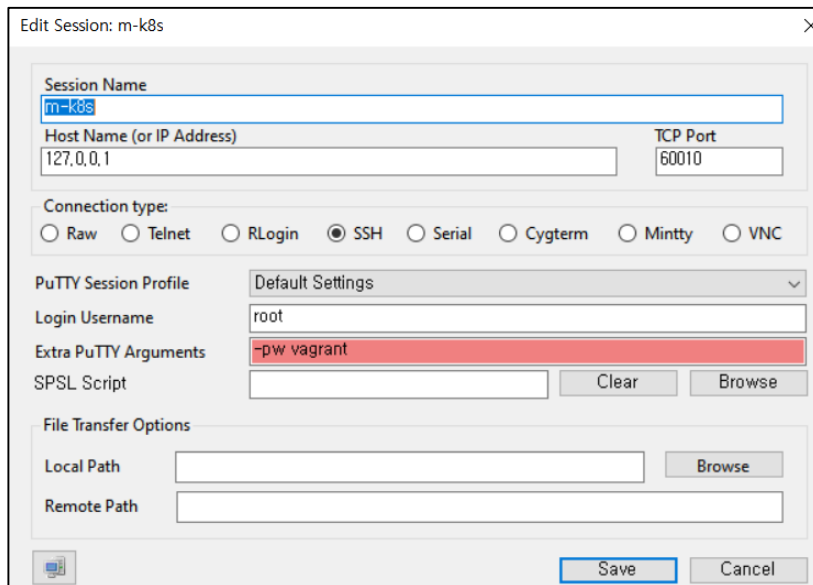
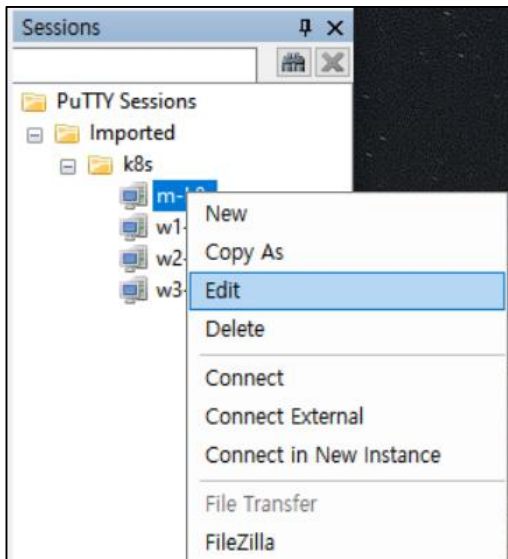
쿠버네티스 테스트 환경

- 터미널 프로그램으로 가상 머신 접속하기
 - SuperPutty로 다수의 가상 머신 접속하기
 - File > import Session > from File



쿠버네티스 테스트 환경

- 터미널 프로그램으로 가상 머신 접속하기
 - SuperPutty로 다수의 가상 머신 접속하기



수고했어요!!!