

---

# NAVER Cloud Platform CI/CD Hands-on Lab Guide

2021-04-12

---

---

# 저작권

---

© NAVER Cloud Corp. All Rights Reserved.

이 문서는 NAVER Cloud(주)의 지적 자산이므로 NAVER Cloud(주)의 승인 없이 이 문서를 다른 용도로 임의 변경하여 사용할 수 없습니다. 이 문서는 정보제공의 목적으로만 제공됩니다. NAVER Cloud(주)는 이 문서에 수록된 정보의 완전성과 정확성을 검증하기 위해 노력하였으나, 발생할 수 있는 내용상의 오류나 누락에 대해서는 책임지지 않습니다. 따라서 이 문서의 사용이나 사용 결과에 따른 책임은 전적으로 사용자에게 있으며, NAVER Cloud(주)는 이에 대해 명시적 혹은 묵시적으로 어떠한 보증도 하지 않습니다. 관련 URL 정보를 포함하여 이 문서에서 언급한 특정 소프트웨어 상품이나 제품은 해당 소유자의 저작권법을 따르며, 해당 저작권법을 준수하는 것은 사용자의 책임입니다.

NAVER Cloud(주)는 이 문서의 내용을 예고 없이 변경할 수 있습니다.

---

# 목차

---

## 1. 사전 준비 사항

- 1) VPC, Subnet 생성
- 2) Server 생성
- 3) Server 접속
- 4) 툴 설치

## 2. SourceCommit 을 이용한 소스 형상관리

- 1) Sub Account 생성
- 2) SourceCommit 생성
- 3) Git Client 와 SourceCommit Repository 연동

## 3. Container Registry Image Push

- 1) ObjectStorage 버킷 생성
- 2) Node.js Container Image 생성
- 3) Container Registry 이미지 Push

## 4. SourceBuild 를 이용한 Container Image build

- 1) SourceCommit 을 이용한 nodejs 코드관리
- 2) SourceBuild 설정 및 컨테이너 이미지 빌드
- 3) kubectl 을 이용한 Deployment 생성

## 5. SourceDeploy 를 이용한 Kubernetes Deploy

---

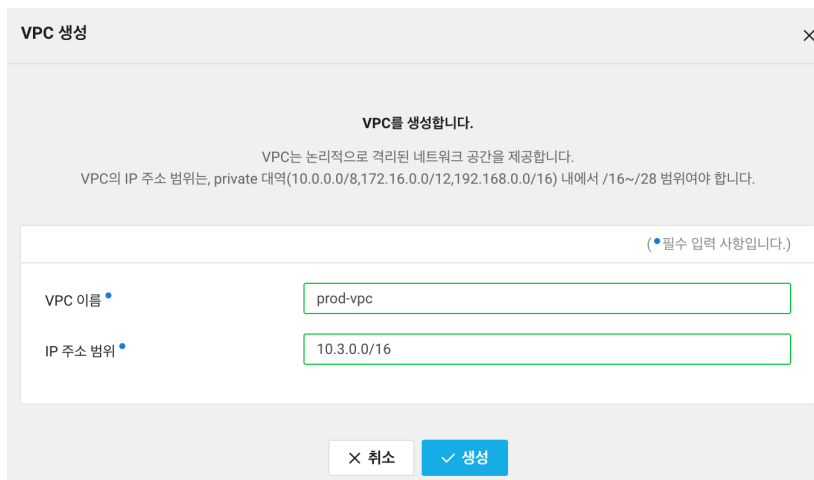
# 1. 사전 준비 사항

---

## 1.1 VPC, Subnet 생성

### 1.1.1 VPC 생성

- Product & Services > VPC > VPC Management
- VPC 생성 클릭
- VPC 이름 : prod-vpc
- IP 주소범위 : 10.3.0.0/16



**VPC 생성** [X]

**VPC를 생성합니다.**

VPC는 논리적으로 격리된 네트워크 공간을 제공합니다.  
VPC의 IP 주소 범위는, private 대역(10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16) 내에서 /16~/28 범위여야 합니다.

( \* 필수 입력 사항입니다. )

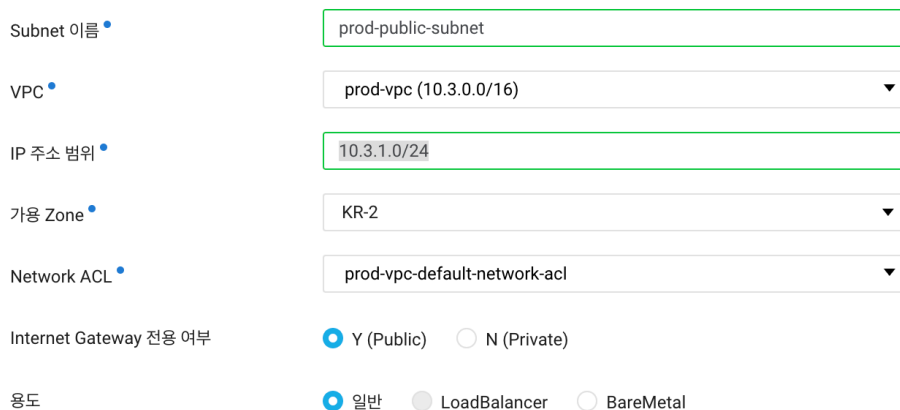
VPC 이름

IP 주소 범위

[X 취소] [✓ 생성]

### 1.1.2 Subnet 생성

- Product & Services > VPC > Subnet Management
- Subnet 생성 클릭
- Subnet 이름 : prod-public-subnet
- VPC : prod-vpc
- IP 주소 범위 : 10.3.1.0/24
- 가용 Zone : KR-2
- Internet Gateway 전용 여부 : Y(public)
- 용도 : 일반



Subnet 이름

VPC

IP 주소 범위

가용 Zone

Network ACL

Internet Gateway 전용 여부 ☒ Y (Public) ☐ N (Private)

용도 ☒ 일반 ☐ LoadBalancer ☐ BareMetal

일반서버에서만 사용 가능한 서브넷입니다.

## 1.2 Server 생성

- Product & Services > Server
- 서버 생성 버튼 클릭
- VPC : prod-vpc
- Subnet : prod-public-subnet
- 서버이름 : devopslab
- Network Interface : 추가
- 공인 IP : 새로운 공인 IP 할당

VPC prod-vpc VPC 생성

Subnet prod-public-subnet | KR-2 | 10.3.1.0/24 | Public Subnet 생성

공인 IP 연결을 위해서는 반드시 Public Subnet을 선택해야 합니다.

스토리지 종류 ☒ SSD ☐ HDD

서버 세대 g2

서버 타입 High CPU

[High CPU] vCPU 2개, 메모리 4GB, [SSD]디스크 50GB [g2]

스토리지 암호화 적용 ☐

암호화 기본 스토리지(OS)가 적용된 서버에는 암호화된 추가 스토리지만 연결할 수 있습니다.  
마찬가지로, 암호화 되지 않은 기본 스토리지가 적용된 서버는 암호화 적용되지 않은 추가 스토리지만 연결 가능합니다.

요금제 선택 ☒ 월요금제 ☐ 시간 요금제 월 72,000원 (OS 제외)

서버 개수 1

서버 이름 devopslab

☒ 입력하신 서버 이름으로 hostname을 설정합니다.

Network Interface

디바이스	Network Interface	Subnet	IP
eth1	<span>new interface</span>	<span>- select -</span>	<span>미입력시 자동!</span> <span>+ 추가</span>
eth0	new interface	prod-public-subnet   KR-2   10.3.1.0/24   Public	자동할당 <span>×</span>

공인 IP ☐ 미설정

☒ 새로운 공인 IP 할당 신청된 공인 IP는 보유하신 동안 요금이 과금되므로, 사용하지 않을 때는 반납하시기를 권장드립니다. (월 이용료: 4,032원)  
서버 생성시 공인 IP를 함께 생성하시려면 Subnet 타입은 Public Subnet, 서버 개수는 1개여야 합니다.

- 새로운 인증키 설정 : devopslab

☐ 보유하고 있는 인증키 이용

☒ 새로운 인증키 생성

인증키 이름 devopslab

인증키 생성 및 저장

인증키 이름을 입력 후 [인증키 생성 및 저장]를 클릭하여 인증키를 사용자 컴퓨터에 저장하세요.

인증키는 해당 서버의 관리자 비밀번호 확인에 이용되니 안전한 곳에 저장하시기 바랍니다

- 네트워크 접근 설정 : prod-vpc-default-acg
- 생성 버튼 클릭

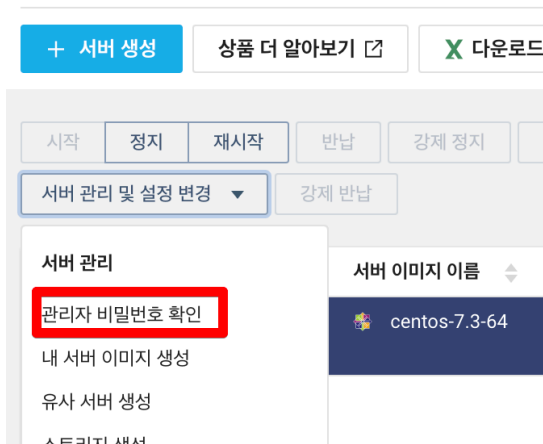
## 1.3 Server 접속

Putty, terminal 같은 ssh client 설치 필요

- 관리자 비밀번호 확인

### Server 6

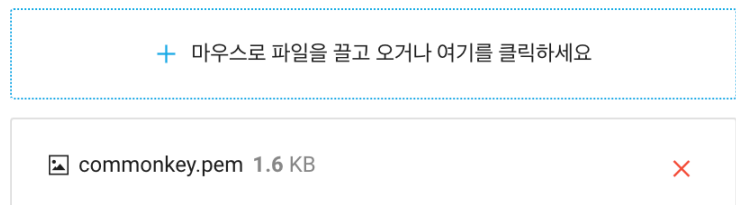
커널 업데이트 시 서버의 정상적인 사용이 불가능할 수 있으며 이에 따른



- 서버 생성시 생성한 .pem 파일을 선택

서버 이름 devopslab

인증키 이름 commonkey



- 서버 접속 (공인 IP)

```
user@AL01982495 ~ % ssh root@175.106.97.187
The authenticity of host '175.106.97.187 (175.106.97.187)' can't be established.
ED25519 key fingerprint is SHA256:ow2TXeFEAEYM9/T0drSweBCWFzHhMMJZ2rYLozKrC5s.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:7: 175.106.96.16
  ~/.ssh/known_hosts:8: 175.106.99.69
  ~/.ssh/known_hosts:9: 110.165.17.78
  ~/.ssh/known_hosts:10: 101.79.11.50
  ~/.ssh/known_hosts:17: 223.130.161.9
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '175.106.97.187' (ED25519) to the list of known hosts
```

## 1.4 Server 접속 후 툴 설치

### 1.4.1 Docker 설치

```
curl -s https://get.docker.com/ | sudo sh
systemctl start docker
docker -v
```

### 1.4.2 Npm node 설치

```
yum install epel-release
yum install -y npm nodejs
node -v
```

### 1.4.3 Kubectl 설치

```
curl -LO "https://dl.k8s.io/release/stable.txt)bin/linux/amd64/kubectl">https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)bin/linux/amd64/kubectl"
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
kubectl version
```



---

## 2. SourceCommit 을 이용한 소스 형상관리

---

## 2.1 Sub Account 생성

- Product & Services > Management > Sub Account

- 계정 생성

로그인 아이디 : devopslab

**서브 계정 정보**

( \* 필수 입력 사항입니다.)

로그인 아이디  최소 3자, 최대 20자

사용자 이름  최소 2자, 최대 30자

이메일

**로그인 비밀번호**

( \* 필수 입력 사항입니다.)

로그인 비밀번호

비밀번호는 8자 이상, 16자 이하의 영문자, 숫자 및 특수문자를 조합하여 사용해야 합니다

- 정책 설정

### Sub Accounts ?

+ 서브 계정 생성

상품 더 알아보기 [↗](#)

↻ 새로 고침

▼

삭제

일시 정지

일시 정지 해제

<input type="checkbox"/>	로그인 아이디	사용자 이름	Console 접근	API 접근	상태	Access Key 수명 <span>?</span>	비밀번호명
<input type="checkbox"/>	devopslab0411	홍길동	✓		● 사용 중		1일 미
<input type="checkbox"/>	databox	csa	✓		● 사용 중		47 일

### ■ 정책 : NCP\_SOURCECOMMIT\_MANAGER 추가

관리형 정책 (5)			사용자 정의 정책 (4)		
<input type="checkbox"/>	정책 이름	정책 설명	<input type="checkbox"/>	정책 이름	정책 유형
<input type="checkbox"/>	NCP_SOURCECOMMIT_VIEWER	Source Commit 서비스 내 조회 기능만 이용할 수 있는 권한	<input type="checkbox"/>		SYSTEM_MANAGED
<input type="checkbox"/>	NCP_SOURCECOMMIT_READ	SourceCommit에서 관리하는 READ 권한 template 입니다.	<input type="checkbox"/>		SYSTEM_MANAGED
<input type="checkbox"/>	NCP_SOURCECOMMIT_WRITE	SourceCommit에서 관리하는 WRITE 권한 template 입니다.	<input type="checkbox"/>		SYSTEM_MANAGED
<input type="checkbox"/>	NCP_SOURCECOMMIT_ADMIN	SourceCommit에서 관리하는 ADMIN 권한 template 입니다.	<input type="checkbox"/>		SYSTEM_MANAGED
<input checked="" type="checkbox"/>	NCP_SOURCECOMMIT_MANAGER	SourceCommit 서비스 내 모든 기능을 이용할 수 있는 권한	<input type="checkbox"/>		SYSTEM_MANAGED

## 2.2 SourceCommit 생성

- Product & Services > DevTools > SourceCommit
- 리파지토리 생성 버튼 클릭

### SourceCommit 8

+ 리파지토리 생성
+ 외부 리파지토리 복사
상품 더 알아보기 |

코드로 이동
설정 변경
삭제

- 리파지토리 이름 입력

리파지토리 이름 \*

devops-lab-220411

리파지토리 설명

최대 500자

0/500 Bytes

초기화 설정

☒ Add a README  

Add a .gitignore: None ▼

- 생성 버튼 클릭
- SubAccount 로그인
  - 서버계정 로그인 페이지 주소 복사 후 인터넷 주소창에 붙여 넣기

대시보드

서버계정 로그인 페이지 접속키	wassub	↻ 수정	✕ 삭제
서버계정 로그인 페이지	https://www.ncloud.com/nsa/wassub	📄 주소 복사	
미사용 세션 만료 설정 (만료시간)	OFF	변경	

- 서버 계정으로 로그인

## 서브 계정으로 로그인

wassub

devopslab0411

.....|

🔑 ▼

- SourceCommit 메뉴 클릭 후 Git 계정 설정

## SourceCommit 14

+ 리포지토리 생성
+ 외부 리포지토리 복사
상품 더 알아보기
새로 고침

코드로 이동
설정 변경
삭제
**⚙ GIT 계정/GIT SSH 설정**

리포지토리 이름	리포지토리 URL	상태
devops-lab-220411	https://devtools.ncloud.com/2534636/devop...	●
devops-lab-220405	https://devtools.ncloud.com/2534636/devon...	●

- Git 계정 설정

### ■ 패스워드 입력

GIT 계정 설정
GIT SSH 설정

**Git client에서 사용할 계정을 설정합니다**

여기서 설정한 비밀번호는 Git client를 통한 리포지토리 접근시 사용되며,  
해당 비밀번호를 사용하여 포털, 콘솔 등 웹을 통한 로그인도 불가능합니다

(● 필수 입력 사항입니다)

User name ●

devopslab0411

Password ●

.....

Confirm Password ●

.....

🔑

## 2.3 Git Client 와 SourceCommit Repository 연동

```
# cd ~
# mkdir sourcecommit
# cd sourcecommit
# git init
# git config --global user.email "{email 주소}"
# git config --global user.name "{유저명}"
# touch readme.txt
# git status
[root@lab-20190722c sourcecommit]# git status
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
# git add readme.txt
# git status
[root@lab-20190722c sourcecommit]# git status
# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
# git commit -m "first commit"
[root@lab-20190722c sourcecommit]# git commit -m "first commit"
[master (root-commit) 6ca1e7a] first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 readme.txt
# git remote add origin {repository url}
# git pull --rebase origin master
[root@lab-20190722c sourcecommit]# git pull --rebase origin master
Username for 'https://devtools.ncloud.com': dev10
Password for 'https://dev10@devtools.ncloud.com':
warning: no common commits
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
From https://devtools.ncloud.com/2534636/lab-repo
* branch          master      -> FETCH_HEAD
First, rewinding head to replay your work on top of it...
Applying: first commit
```

```
# git push -u origin +master
```

```
[root@lab-20190722c sourcecommit]# git push -u origin +master
Username for 'https://devtools.ncloud.com': dev10
Password for 'https://dev10@devtools.ncloud.com':
Counting objects: 4, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 284 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://devtools.ncloud.com/2534636/lab-repo.git
    db5b577..a8d0b8a  master -> master
Branch master set up to track remote branch master from origin.
```

```
# git branch lab
```

```
# git branch
```

```
# git checkout lab
```

```
# git branch
```

```
# vi readme.txt
```

```
    branch test
```

```
# git add *
```

```
# git commit -m "branch test"
```

```
[root@lab-20190722c sourcecommit]# git commit -m "branch test"
[lab 69bc3a3] branch test
 1 file changed, 2 insertions(+)
[root@lab-20190722c sourcecommit]# git push origin lab
Username for 'https://devtools.ncloud.com': dev10
Password for 'https://dev10@devtools.ncloud.com':
Counting objects: 5, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 307 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://devtools.ncloud.com/2534636/lab-repo.git
 * [new branch]      lab -> lab
```

```
# git push origin lab
```

```
# git checkout master
```

```
# git merge lab
```

```
[root@lab-20190722c sourcecommit]# git merge lab
Updating bdb3325..69bc3a3
Fast-forward
 readme.txt | 2 ++
 1 file changed, 2 insertions(+)
```

```
# git push origin master
```

```
[root@lab-20190722c sourcecommit]# git push origin master
Username for 'https://devtools.ncloud.com': dev10
Password for 'https://dev10@devtools.ncloud.com':
Total 0 (delta 0), reused 0 (delta 0)
To https://devtools.ncloud.com/2534636/lab-repo.git
    bdb3325..69bc3a3  master -> master
```

---

## 3. Container Registry Image Push

---

### 3.1 ObjectStorage 버킷 생성

- ObjectStorage 에 결과물 저장을 위한 버킷 생성 및 폴더 생성

- Product & Services > Storage > Object Storage

- 버킷 생성

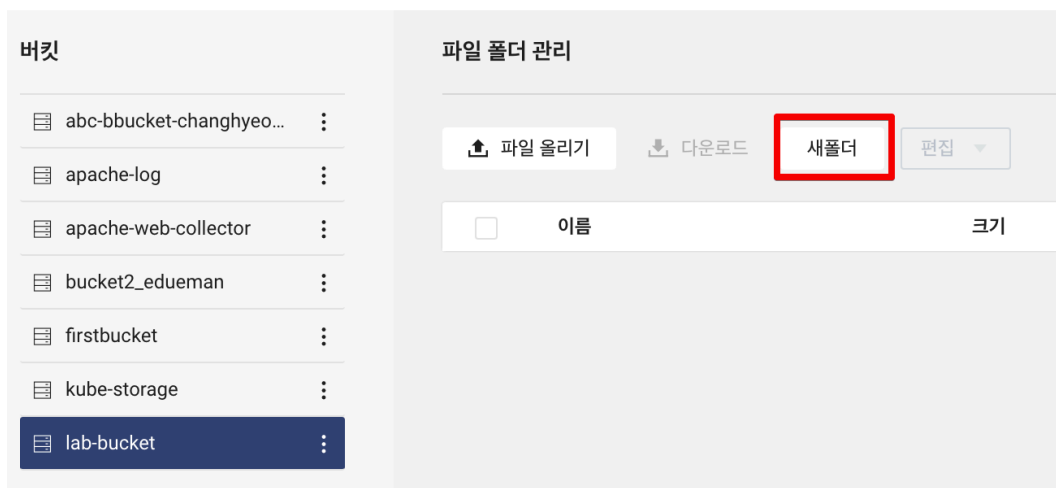
VPC / Object Storage / [Bucket Management](#)

## Bucket Management 17



- 생성된 버킷 클릭 후 새폴더 생성

- ◆ 폴더명 : build





## 3.2 Node.js Container Image 생성

- 디렉터리 생성

```
# cd ~  
# cd sourcecommit  
# npm init -y  
# npm i fastify --save
```

- Nodejs 코드 생성 및 실행  
# vi app.js

```
// Require the framework and instantiate it  
  
const fastify = require('fastify')({  
  
  logger: true  
  
})  
  
// Declare a route  
  
fastify.get('/', function (request, reply) {  
  
  reply.send({ hello: 'world' })  
  
})  
  
// Run the server!  
  
fastify.listen(3000, '0.0.0.0', function (err, address) {  
  
  if (err) {  
  
    fastify.log.error(err)  
  
    process.exit(1)  
  
  }  
  
  fastify.log.info(`server listening on ${address}`)  
  
  })  
  
# node app.js
```

```
- Docker Image 생성

# vi .dockerignore
node_modules/*

vi Dockerfile

# 1. node 이미지 사용
FROM node:12-alpine

# 2. 패키지 우선 복사
COPY ./package* /usr/src/app/
WORKDIR /usr/src/app
RUN npm install

# 3. 소스 복사
COPY . /usr/src/app

# 4. WEB 서버 실행 (Listen 포트 정의)
EXPOSE 3000
CMD node app.js

# docker build -t devopsweb .
# docker images
# docker run -d -p 3000:3000 devopsweb

# docker ps -a
# docker stop {container id}
```

### 3.3 Container Registry 이미지 Push

- Product & Services > Compute > Container Registry
- Container Registry 생성

#### Container Registry <sup>2</sup>

+ 레지스트리 생성
상품 더 알아보기

삭제

<input type="checkbox"/>	레지스트리 이름	버킷 이름	사용량
<input type="checkbox"/>	devopslab0411	devopslab0411	0B
<input type="checkbox"/>	devops-lab	devops-lab	34.51MB

새로운 레지스트리 추가

(필수 입력 사항입니다.)

레지스트리 이름

devopslab

버킷

labs

- Container Registry 요금은 무료이나, Private 도커 컨테이너 이미지는 Object Storage에 저장되므로 사용량에 따른 Object Storage 요금이 발생할 수 있습니다.

- Private Endpoint 확인 및 복사

상태

이미지 리스트

● 운영중

이동 >

Public Endpoint

devopslab0411.kr.ncr.ntruss.com

Private Endpoint

dm5xoi1l.kr.private-ncr.ntruss.com

사용량

0B

오래된 태그 정리

이동 >

생성 일시

2022-04-12 00:46:27 (UTC+09:00)

### - Container Registry 이미지 Push

```
# docker login -u {AccessKey} {Container Registry Endpoint URL}
# docker image tag {image name} {Container Registry Endpoint URL}/{imagename}
# docker push {Container Registry Endpoint URL}/{imagename}
```

\* AccessKey : 마이페이지 > 인증키 관리 > AccessKey

\* Password : 마이페이지 > 인증키 관리 > SecretKey

예)

```
docker login -u Mu2Z9upXcenwRX9gZmT2 k1kozuoo.kr.private-ncr.ntruss.com
```

```
[root@devopslab web]# docker login -u Mu2Z9upXcenwRX9gZmT2 dm5xoi11.kr.private-ncr.ntruss.com
[Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

```
docker image tag devopsweb k1kozuoo.kr.private-ncr.ntruss.com/devopsweb
```

```
docker push k1kozuoo.kr.private-ncr.ntruss.com/devopsweb
```

```
[root@devopslab web]# docker image tag devopsweb dm5xoi11.kr.private-ncr.ntruss.com/devopsweb
[root@devopslab web]# docker push dm5xoi11.kr.private-ncr.ntruss.com/devopsweb
Using default tag: latest
The push refers to repository [dm5xoi11.kr.private-ncr.ntruss.com/devopsweb]
cc3ae23a8852: Pushed
86e39e60090c: Pushed
fee8a536ea34: Pushed
a2fd151d8ce8: Pushed
6ede17d765cd: Pushed
d5d9b80a0fda: Pushed
4fc242d58285: Pushed
latest: digest: sha256:a0f7eb4c9f158fe4dd4f8dd0293b0a0ff6be89b966ae35ab3e633df291cab514 size: 1785
```

### - Container Registry 이미지 확인

<input checked="" type="checkbox"/> 이미지 이름	Short Description	최근 변경일
<input checked="" type="checkbox"/> devopsweb		2022-04-12 01:00:24 (UTC+09:00)

Details	Tags
---------	------

Tag 삭제

<input type="checkbox"/> 태그 이름	압축 사이즈	최근 변경일	Digest	보안 취약점
<input type="checkbox"/> latest	31.25MB	2022-04-12 01:00:24 (UTC+09:00)	a0f7eb4	PENDING

<< < 1 > >>

---

## 4. SourceBuild 를 이용한 Container Image build

---

## 4.1 SourceCommit 에 nodejs 코드 Push

```
# cd ~
# cd sourcecommit
# git add -A
# git status
# git commit -m "node code push"
# git push -u origin +master
# git status
```

```
[root@devopslab sourcecommit]#
[root@devopslab sourcecommit]# git status
# On branch master
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       .dockerignore
#       Dockerfile
#       app.js
#       node_modules/
#       package-lock.json
#       package.json
nothing added to commit but untracked files present (use "git add" to track)
[root@devopslab sourcecommit]# git add -A
[root@devopslab sourcecommit]# git pull --rebase origin master
Cannot pull with rebase: Your index contains uncommitted changes.
Please commit or stash them.
[root@devopslab sourcecommit]# git commit -m "node commit"
[master 951f37f] node commit
1502 files changed, 177666 insertions(+)
create mode 100644 .dockerignore
```

```
create mode 100644 package.json
[[root@devopslab sourcecommit]# git push -u origin +master
[Username for 'https://devtools.nccloud.com': devopslab0411
[Password for 'https://devopslab0411@devtools.nccloud.com':
Counting objects: 1666, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (1617/1617), done.
Writing objects: 100% (1665/1665), 1.50 MiB | 0 bytes/s, done.
Total 1665 (delta 191), reused 0 (delta 0)
To https://devtools.nccloud.com/2534636/devops-lab-220411.git
   dacd166..951f37f  master -> master
Branch master set up to track remote branch master from origin.
[[root@devopslab sourcecommit]#
[[root@devopslab sourcecommit]#
```

## 4.2 SourceBuild 설정 및 컨테이너 이미지 빌드

### - 기본설정

- Object Storage : 사용중
- 빌드 프로젝트 이름/설명 : 각자 입력
- 빌드 대상 : SourceCommit
- 빌드 대상 리파지토리 선택 : 앞에서 생성한 리파지토리
- 브랜치 선택 : master

#### 기본 설정

빌드 프로젝트의 이름 및 빌드 할 소스 코드의 리파지토리를 선택합니다. (\*필수 입력 사항입니다.)

SourceBuild 요금은 시작된 시점부터 종료될 때까지 사용자가 사용한 인스턴스 리소스에 대해서 과금합니다.

빌드 프로젝트 이름	<input type="text" value="devopslab0412"/>
빌드 프로젝트 설명	<div><div></div></div> 0/500 자
빌드 대상	<div>SourceCommit ▼</div>
빌드 대상 리파지토리 선택	<div>devops-lab-220411 ▼</div>
브랜치 선택	<div>master ▼</div>

### - 빌드 환경 설정

- 빌드 환경 이미지 : Container Registry 이미지
- 레지스트리 : Container Registry 에 생성한 Repository
- 이미지 : 컨테이너 이미지
- 도커 이미지 빌드 : 체크
- 도커 엔진 버전 : Docker:18.09.1
- 컴퓨팅 유형 : 2vCpu 4GB 메모리
- 타임 아웃 : 10 분

#### 빌드 환경 설정

빌드를 진행할 환경을 설정합니다. (\*필수 입력 사항입니다.)

빌드 환경 이미지	<input type="radio"/> SourceBuild에서 관리되는 이미지 <input checked="" type="radio"/> Container Registry 의 이미지 <input type="radio"/> Public
운영 체제	Linux 이미지만 사용 가능합니다.
레지스트리	<div>devopslab0411 ▼</div>
이미지	<div>devopsweb ▼</div>
태그	<div>latest ▼</div>
도커 이미지 빌드	<input checked="" type="checkbox"/> docker build 명령어를 사용하는 경우 체크 합니다. <span>?</span>
도커 엔진 버전	<div>Docker:18.09.1 ▼</div>
컴퓨팅 유형	<div>2vCpu 4GB 메모리 ▼</div>
타임 아웃	<div>60 분 <span>?</span></div> (최소 5분, 최대 540분)

- 빌드 명령어 설정
  - 빌드전 명령어  
docker login --username {AccessKey} --password {SecretKey } {Container Registry Endpoint URL}
  - 도커 이미지 빌드 설정 : 사용
  - Dockerfile 경로 : Dockerfile
  - Container Registry : Build 내용이 포함된 이미지를 업로드 할 레지스트리
  - 이미지 이름 : Build 내용이 포함된 이미지 이름 각자 입력
  - 이미지 태그 : 1.0#
  - Latest 로 설정 : 체크

**빌드 명령어 설정**

빌드를 진행할 명령어를 작성합니다. 여러 개의 명령어는 줄바꿈(엔터)으로 구분합니다.

빌드 전 명령어	<code>docker login --username Mu279uXcenwRX9gZmT2 --password r03KtoFcdtG3Nd3n82nIPtAshJ3uHj2YJjm48Ltv dm5xoi1l.kr.private-ncr.ntruss.com</code>	?
빌드 명령어		?
빌드 후 명령어		?

도커 이미지 빌드 설정 ☒ 사용 ☐ 사용 안함 ?

도커 이미지를 빌드 하여 Container Registry에 저장합니다, 사용량에 따라 요금이 발생할 수 있습니다. ([Container Registry 요금안내 바로가기](#))

**Dockerfile 경로**  ?

**Container Registry**

**이미지 이름**

**이미지 태그**  ?

☒ latest 로 설정

- 생성 버튼 클릭
- SourceBuild 목록에서 빌드로 이동 버튼 클릭

+ 빌드 프로젝트 생성
상품 더 알아보기
새로 고침
▼

빌드로 이동
설정 변경
삭제

프로젝트 이름

devopslab0412

프로젝트 설명

상세정보



- 빌드 시작하기 버튼 클릭

빌드

작업결과

빌드 프로젝트 시작

[devopslab0412] 빌드를 시작합니다. 빌드 설정을 변경할 수 있으며, 변경된 설정은 이번 빌드에만 적용됩니다.

빌드 시작하기

프로젝트 정보

빌드 프로젝트 이름

devopslab0412

빌드 리파지토리

devops-lab-220411

브랜치 선택

master

커밋 해시

commit hash

빌드 설정

- 빌드 결과 확인

#### 빌드 로그

빌드 준비

빌드 전 명령어

빌드 명령어

빌드 후 명령어

결과물 업로드

결과물을 업로드 합니다. (로그는 최대 4000 라인까지 출력됩니다.)

수집된 로그가 없습니다.

- 빌드 이미지 Container Registry 확인

<input checked="" type="checkbox"/> 이미지 이름	Short Description	최근 변경일
<input checked="" type="checkbox"/> devopsweb		2022-04-12 15:02:25 (UTC+09:00)

Details

Tags

Tag 삭제

<input type="checkbox"/> 태그 이름	압축 사이즈	최근 변경일	Digest	보안 취약점
<input type="checkbox"/> latest	32.86MB	2022-04-12 15:02:25 (UTC+09:00)	40b315a	not scanned
<input type="checkbox"/> 1.01	32.86MB	2022-04-12 15:02:24 (UTC+09:00)	40b315a	not scanned

### 4.3 kubectl 을 이용한 Deployment 생성

```
# cd ~
```

```
# vi deploy-node.yaml
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: build-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: sourcebuild
  template:
    metadata:
      labels:
        app: sourcebuild
    spec:
      containers:
        - name: sourcebuild
          image: {Container Registry url}/{image name} k1kozuoo.kr.private-ncr.ntruss.com/devopsweb
          ports:
            - containerPort: 3000
          imagePullSecrets:
            - name: regcred
      ---
kind: Service
apiVersion: v1
metadata:
  name: testweb
spec:
  ports:
    - port: 3000
      targetPort: 3000
  selector:
    app: sourcebuild
  type: LoadBalancer
```

```
# kubectl --kubeconfig=kubeconfig.yaml create namespace {namespace}
# kubectl --kubeconfig=kubeconfig.yaml create secret docker-registry regcred --docker-
server={container registry url} --docker-username={AccessKey} --docker-password={SecretKey} -
docker-email={email} -n {namespace}
# kubectl --kubeconfig=kubeconfig.yaml create -f deploy-node.yaml -n {namespace}
```

예)


```
# kubectl --kubeconfig=kubeconfig.yaml create namespace devopslab0411
# kubectl --kubeconfig=kubeconfig.yaml create secret docker-registry regcred --docker-
server=dm5xoi1l.kr.private-ncr.ntuoss.com --docker-username=MU2Z9upXcenwRX9gZmT2 --docker-
password=r03KtoFcdtG3Nd3n82nIPtAshJ3uHj2YJm48Ltv --docker-
email=changhyeon.heo@navercorp.com -n devopslab0411
# kubectl --kubeconfig=kubeconfig.yaml create -f deploy-node.yaml -n devopslab0411
```

```
# kubectl --kubeconfig=kubeconfig.yaml get pods -n devopslab0411
```

```
[root@devopslab ~]# kubectl --kubeconfig=kubeconfig.yaml get pods -n devopslab0411
NAME                                READY   STATUS    RESTARTS   AGE
build-deployment-566c4d7887-bz2dp   1/1     Running   0           7s
build-deployment-566c4d7887-h24hp   1/1     Running   0           7s
build-deployment-566c4d7887-vvtsg   1/1     Running   0           7s
```

```
# kubectl --kubeconfig=kubeconfig.yaml get services -n devopslab0411
```

```
[root@devopslab ~]# kubectl --kubeconfig=kubeconfig.yaml get services -n devopslab0411
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
testweb   LoadBalancer  198.19.182.119 <pending>     3000:30306/TCP   13s
[root@devopslab ~]# kubectl --kubeconfig=kubeconfig.yaml get services -n devopslab0411
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
testweb   LoadBalancer  198.19.182.119 <pending>     3000:30306/TCP   16s
[root@devopslab ~]# kubectl --kubeconfig=kubeconfig.yaml get services -n devopslab0411
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)
testweb   LoadBalancer  198.19.182.119 devopslab04-testweb-c8aeb-10656383-182e2ccb6abd.kr.lb.naverncp.com 3000:30306/TCP
```



```
{"hello": "world"}
```

\*

삭제 필요 시

예)

```
# kubectl --kubeconfig=kubeconfig.yaml delete -f deploy-node.yaml -n {namespace}
```

---

## 5. SourceDeploy 를 이용한 Kubernetes Deploy

---

## 5.1 SourceDeploy 를 이용한 Kubernetes Deploy

- Deploy 메니페스트 파일 설정

```
# cd ~
```

```
# cd sourcecommit
```

```
# vi deploy.yaml
```

```
apiVersion: apps/v1
```

```
kind: Deployment
```

```
metadata:
```

```
  name: build-deployment
```

```
  namespace: {namespace} devopslab0411
```

```
spec:
```

```
  replicas: 3
```

```
  selector:
```

```
    matchLabels:
```

```
      app: sourcebuild
```

```
  template:
```

```
    metadata:
```

```
      labels:
```

```
        app: sourcebuild
```

```
    spec:
```

```
      containers:
```

```
        - name: sourcebuild
```

```
          image: {Container Registry url}/{image name} k1kozuoo.kr.private-ncr.ntruss.com/devopsweb
```

```
          ports:
```

```
            - containerPort: 3000
```

```
          imagePullSecrets:
```

```
            - name: regcred
```

## - SourceCommit Push

```
# git status
# git add *
# git commit -m "deploy yaml"
# git push origin +master
```

## - SourceDeploy 생성 및 설정

- 프로젝트 이름 : src-deploy
- 배포 Stage : real
- 배포 타겟 : Kubernetes Service
- Cluster : Kubernetes Cluster 선택

**배포 환경 설정**  
배포 Stage를 생성하고 배포 환경을 설정하세요. (\* 필수 입력 사항입니다.)

배포 Stage	+	dev stage 설정
dev		dev stage 설정
test		에이전트 설치 필수 배포 서버에는 SourceDeploy 용 에이전트가 설치되어있어야 합니다. <a href="#">(에이전트 설치 방법 바로 가기)</a>
real		<div>dev stage <span>•</span> <input checked="" type="radio"/> 설정 <input type="radio"/> 설정 안함</div> <div>배포 타겟 <span>•</span> <input type="radio"/> Server <input type="radio"/> Auto Scaling <input checked="" type="radio"/> <b>Kubernetes Service</b></div> <div><input type="radio"/> Object Storage</div> <div>Cluster <span style="border: 1px solid red; padding: 2px;">nks-ch</span> ▼</div>

## - 배포 시나리오 설정

- 배포 시나리오 생성

프로젝트 이름	최종 실행 결과	최종 실행 시간	최종 실행자
<span style="color: blue;">•</span> deploy0411	-	-	-

배포 Stage	+	배포 환경 <span>ⓘ</span>
dev	⋮	<div>설정 변경 삭제</div> <div>배포 타겟 Kubernetes Service (nks-ch)</div>

배포 시나리오 <span>ⓘ</span>		
<span style="border: 1px solid red; padding: 2px;">생성</span>	설정 변경	삭제
배포 시나리오 이름	Stage	배포 전략

- 배포 시나리오 이름 : deploy

## ■ 매니페스트 파일 위치 : deploy.yaml

### 매니페스트 설정

쿠버네티스 클러스터에 배포를 위한 매니페스트 파일을 설정합니다. (\* 필수 입력 사항입니다.)

매니페스트 파일 저장소 ☒ SourceCommit

리파지토리 선택  devops-lab-220411  master

매니페스트 파일 위치  deploy.yaml

## ■ 배포 전략 : Rolling

### 배포 전략 설정

컨테이너 배포 전략을 설정합니다.

배포 전략 ☒ Rolling ☐ 블루/그린 ☐ Canary

## - 배포 진행

### ■ 배포로 이동

프로젝트 이름	최종 실행 결과	최종 실행 시간	최종 실행자
<input checked="" type="radio"/> deploy0411	-	-	-

배포 Stage + 배포 환경 ?

dev

## ■ 배포 시작하기

배포 작업 결과

배포 시나리오 이름	stage	최종 실행자	최종 실행시간
deploy	dev	-	-

배포 프로젝트 시작

deploy 배포를 시작합니다. 배포 설정을 변경할 수 있으며, 변경된 설정은 이번 배포에만 적용됩니다.

## ■ Rolling 확인

```
kubectl --kubeconfig=kubeconfig.yaml get pods -n {namespace}
```

