

# 네이버클라우드플랫폼 Expert 과정

Lab Guide

대외비

Cloud Tech Froniter | 2022.

## 문서정보

본 문서는 네이버클라우드플랫폼 공인 교육 - Professional 진행을 위한 실습 가이드입니다. 본 실습 가이드는 이론 교육과 더불어 실습 교육 시 활용됩니다.

## Lab 1

### 1. VPC 생성

All Product > VPC > VPC Management 선택 > +VPC 생성 선택

“VPC 이름”에 lab1-vpc 라고 입력

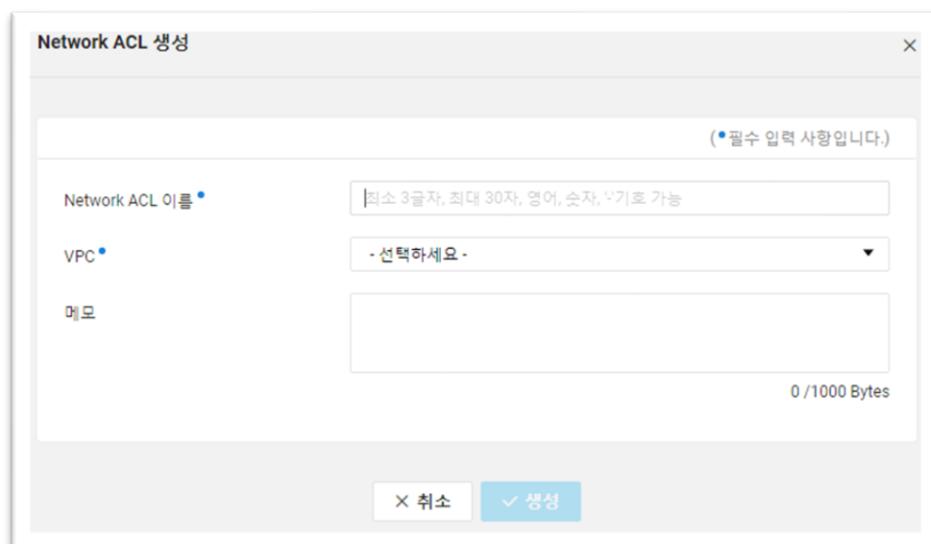
“IP 주소 범위”에 10.0.0.0/16 라고 입력 후 생성 클릭



All Product > VPC > Network ACL > +Network ACL 생성 선택

“Network ACL 이름”에 lab1-vpc-web-nacl 라고 입력

“VPC”는 lab1-vpc 선택 후 생성 클릭



## 2. Subnet 생성

Public Subnet을 생성합니다.

All Product > VPC > Subnet Management 선택 > +Subnet 생성 선택

다음과 같이 내용 설정 후 생성 클릭

\*Subnet 이름 : lab1-vpc-web-subnet

\*VPC : lab1-vpc

\*IP 주소 범위 : 10.0.1.0/24

\*가용 Zone : KR-2

\*Network ACL : lab1-vpc-web-nacl

\*Internet Gateway 전용 여부 : Y

\*용도 : 일반

**Subnet 생성**

×

**Subnet을 생성합니다.**

VPC 내에 세분화된 격리 공간을 제공합니다.  
 IP 주소 범위는 VPC 주소 범위 이하로만 지정이 가능하며,  
 private 대역(10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16) 내에서 /16~/28 범위여야 합니다.  
 생성 이후 Network ACL 만 변경이 가능하므로 생성시 주의해주시기 바랍니다.

(•필수 입력 사항입니다.)

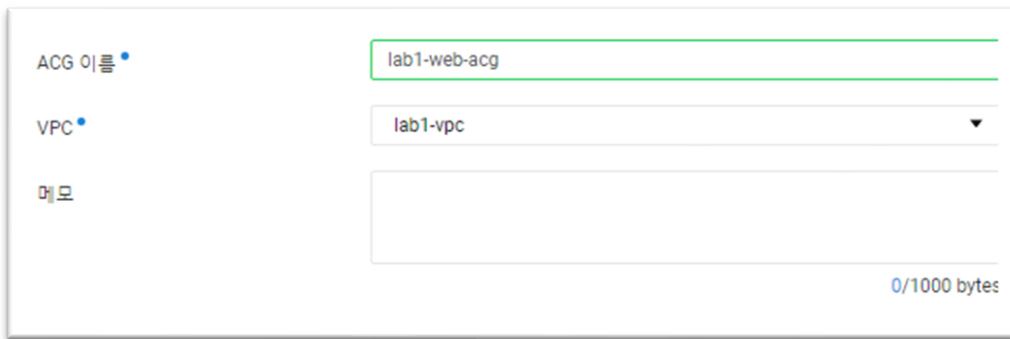
Subnet 이름 *	lab1-vpc-web-subnet
VPC *	lab-vpc (10.0.0.0/16)
IP 주소 범위 *	10.0.1.0/24
가용 Zone *	KR-2
Network ACL *	lab1-vpc-web-nacl
Internet Gateway 전용 여부	<input checked="" type="radio"/> Y (Public) <input type="radio"/> N (Private)
용도	<input checked="" type="radio"/> 일반 <input type="radio"/> LoadBalancer <input type="radio"/> BareMetal 일반서버에서만 사용 가능한 서브넷입니다.

### 3. ACG 만들기

All Product > Server > ACG 선택 > +ACG 생성 선택

ACG 이름에 lab1-web-acg 라고 입력

“VPC”에 lab1-vpc 선택 후 생성 클릭



\*Inbound 규칙 설정

프로토콜 : ICMP, 접근 소스 : 0.0.0.0/0

프로토콜 : TCP, 접근 소스 : 0.0.0.0/0 허용 포트 (서비스) : 1-65535

\*Outbound 규칙 설정

프로토콜 : ICMP, 접근 소스 : 0.0.0.0/0

프로토콜 : TCP, 접근 소스 : 0.0.0.0/0 허용 포트 : 1-65535

프로토콜 : UDP, 접근 소스 : 0.0.0.0/0 허용 포트 : 1-65535

#### 4. 스크립트 만들기

Server > Init Script를 선택합니다.

스크립트 이름 : ncp-script

OS 타입 : Linux

스크립트 내용

스크립트의 내용은 서버 부팅 후 아파치 웹서버와 PHP를 설치하고 테스트 페이지를 다운받은 후,

설정 내용을 수정 후 아파치 웹서버를 기동하는 스크립트입니다.

```
#!/bin/bash

yum install -y httpd
sleep 10
cd /var/www/html
wget http://demo2.ncloudedu.com/ncp.tgz
sleep 10
tar xvfz ncp.tgz
rm -rf /etc/yum.repos.d/*
cp -rf yum.repos.d/* /etc/yum.repos.d/
cat phpadd >> /etc/httpd/conf/httpd.conf
yum install -y redis php php-redis mariadb-server php-mysql mongodb-org php-mongodb
sleep 60

systemctl enable httpd
systemctl enable mariadb
systemctl enable redis
systemctl enable mongod

systemctl start httpd
systemctl start mariadb
systemctl start redis
systemctl start mongod
```

```
mysql < dbstep1.sql
```

```
mysql < dbstep2.sql
```

## 5. Inxsvr 서버 만들기

서버탭에서 서버생성을 눌러 서버를 만들 수 있습니다.

이미지는 **Centos-7.3-64**를 선택합니다.

서버 이미지 이름	설명
centos-7.3-64	CentOS 7.3 (64-bit) (커널 업데이트 시 서버의 정상적인 사용이 불가능할 수 있으며 이에 따른 복구는 지원하지 않습니다.)
centos-7.8-64	CentOS 7.8 (64-bit) (커널 업데이트 시 서버의 정상적인 사용이 불가능할 수 있으며 이에 따른 복구는 지원하지 않습니다.)
ubuntu-16.04-64-server	Ubuntu Server 16.04 (64-bit) (커널 업데이트 시 서버의 정상적인 사용이 불가능할 수 있으며 이에 따른 복구는 지원하지 않습니다.)

VPC : lab-vpc1

Subnet : lab1-vpc-web-subnet

서버 탑재은 vCPU 2개, 메모리 4GB, 디스크 50GB 를 선택합니다.

서버 개수는 1, 서버 이름은 Inxsvr1 입니다.

Network Interface의 IP는 10.0.1.101 을 입력합니다.

Script 선택에 ncp-script를 선택해 주세요.

VPC\* lab1-vpc

Subnet\* lab1-vpc-web-sub1 | KR-2 | 10.0.1.0/24 | Public

공인 IP 연결을 위해서는 반드시 Public Subnet을 선택해야 합니다.

스토리지 종류\* SSD  HDD

서버 타입\* Standard

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

요금제 선택\* 월요금제  시간 요금제 를 88,000원 (OS 제외)

서버 개수\* 1

서버 이름 lab1-org1  입력하신 서버 이름으로 hostname을 설정합니다.

Network Interface*	디바이스	Network Interface	Subnet	IP
	eth1	new interface	- select -	<input type="button" value="미입력시 자"/> <input type="button" value="+ 추가"/>
	eth0	new interface	lab1-vpc-web-sub1   KR-2   10.0.1.0/24   Public	10.0.1.101/32 <input type="button" value="X"/>

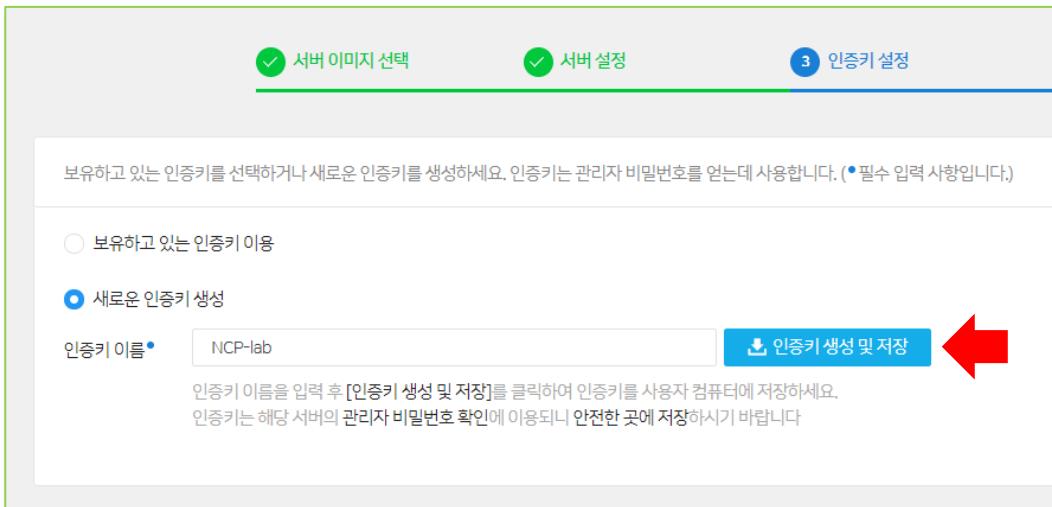
룰리 배치 그룹

반납 보호  설정  해제  
반납 보호를 설정하면 실수로 반납하는 사고를 방지할 수 있습니다.

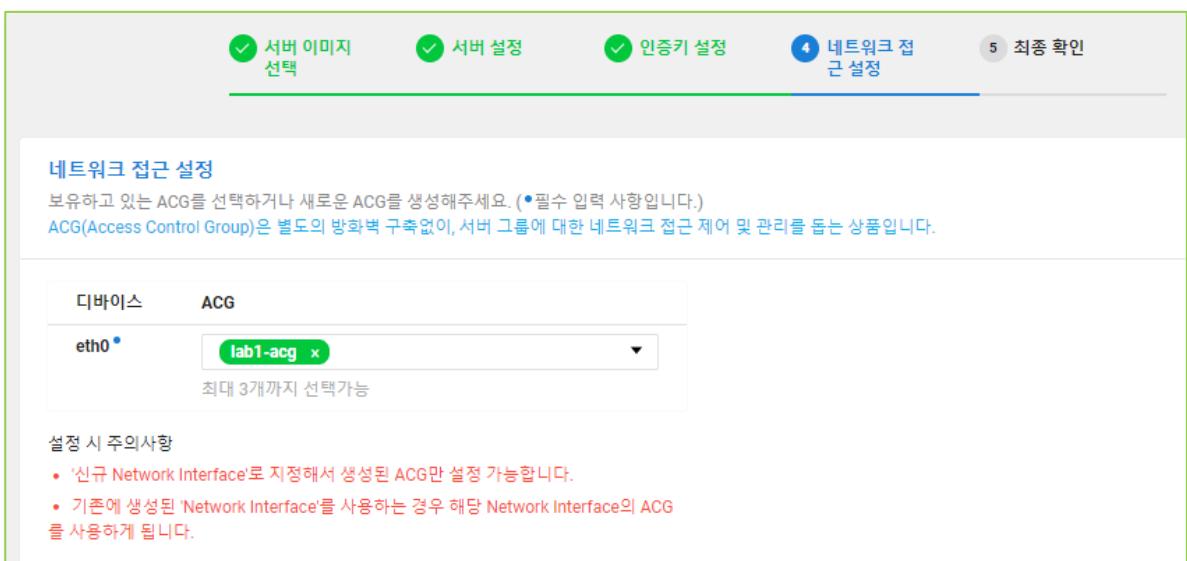
메모

Script 선택 lab-script

새로운 인증키 생성에서 인증키 이름에 NCP오늘날짜 (예, ncp20200325)을 입력하고 인증키 생성 및 저장을 클릭합니다.



네트워크 접근 설정에서 eth0 NIC에 lab1-web-acg 를 할당합니다.



### Public IP 생성 & 할당

- Server - Public IP - [공인 IP 신청]
- 공인 IP 생성 존 & 할당 서버 선택

Server / Public IP

**Public IP** 7

고객이 보유하고 있는 어떤 서버에도 연결될 수 있는 고정된 IP 주소를 저

+ 공인 IP 신청 상품 더 알아보기 다운로드

서버에 할당 서버에서 해제 공인 IP 반납

IP 주소

<input type="checkbox"/>	[REDACTED]

#### 관리자 패스워드 확인 (\*init script 패스워드가 적용 안된 경우)

- 초기 접속 ID(리눅스)는 root
- 서버 선택 > [서버 관리 및 설정 변경] > 관리자 비밀번호 확인 > 서버 생성 시 생성한 인증키 선택 > [비밀번호 확인]

시작 정지 재시작 반납 강제 정지 서버 접속 콘솔

모니터링 포트 포워딩 설정 서버 관리 및 설정 변경

서버 관리

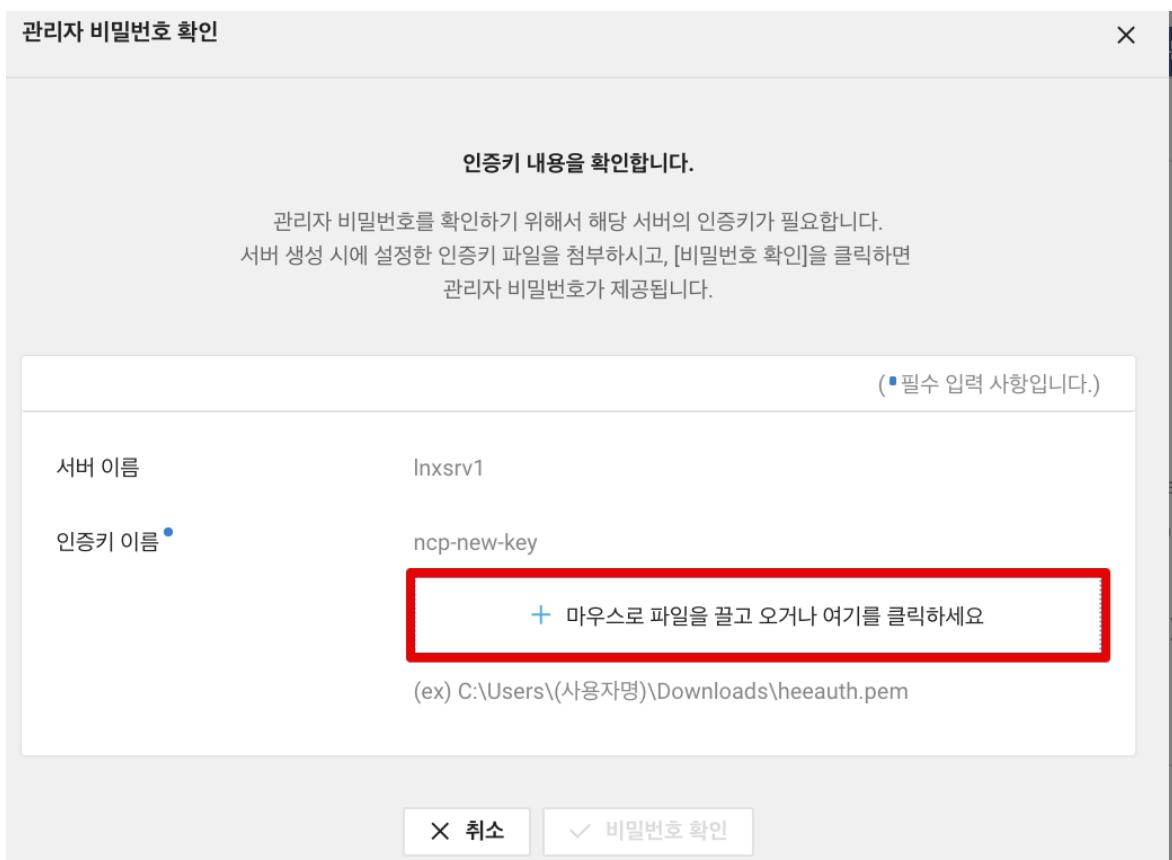
서버 이름	서버 상태
winsrv1	운영중
Inxsrv1	운영중

상세정보

이벤트 로그 확인 관리자 비밀번호 확인

내 서버 이미지 생성 유사 서버 생성

스토리지 생성



### Student 유저 생성

```
[root@lnxsvr-org ~]# useradd student  
[root@lnxsvr-org ~]# passwd student  
Changing password for user student.  
New password: ncp!@#123  
Retype new password: ncp!@#123  
passwd: all authentication tokens updated successfully.
```

### Root SSH 접속 제한 (Option)

/etc/ssh/sshd\_config 파일에서

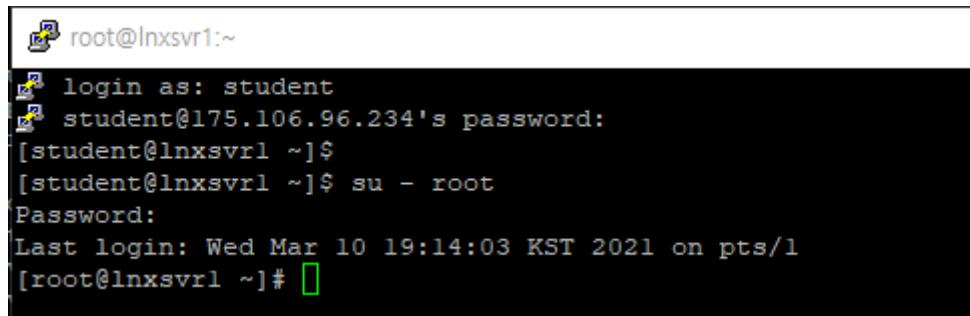
#PermitRootLogin yes 을

PermitRootLogin no 로 변경 후 저장

systemctl restart sshd.service 명령어로 sshd 서비스 재시작

이 후 root 계정으로는 서버에 접근할 수 없음을 확인할 수 있습니다.

Root 계정 접근을 위해서는 student 계정으로 로그인 한 후, su - root 명령어를 통해 계정을 변경합니다.



```
root@lnxsvr1:~  
[student@lnxsvr1 ~]$ login as: student  
[student@lnxsvr1 ~]$ student@175.106.96.234's password:  
[student@lnxsvr1 ~]$ [student@lnxsvr1 ~]$ su - root  
Password:  
Last login: Wed Mar 10 19:14:03 KST 2021 on pts/1  
[root@lnxsvr1 ~]#
```

### CLA 서비스 연동

NCP CLA (Cloud Log Analytics)를 이용하여 lnxsvr1 서버의 보안 로그를 수집합니다.

- 콘솔 Analytics > Cloud Log Analytics (CLA) “+이용 신청” 선택

Cloud Log Analytics / Subscription

## Cloud Log Analytics (CLA)

네이버 클라우드 플랫폼의 상품을 이용하면서 발생하는 다양한 로그를 손쉽게 저장하고 분석 데이터를 제공하는 시스템 [Region 통합 서비스](#)

서버를 비롯하여 네이버 클라우드 플랫폼이 제공하는 다양한 상품을 사용하면서 발생하는 로그 이력을 저장하고 분석할 수 있습니다.

다양한 검색 기능을 통해 손쉽게 데이터를 조회하고, 필요한 정보들을 간편하게 다운로드 할 수 있어 효과적인 로그 관리가 가능합니다.

- ✓ 쉽고 간편한 사용
- ✓ 실시간 로그 수집 및 검색
- ✓ 효율적인 로그 관리
- ✓ 다양한 부가 기능 제공 (기능 추가 예정)

+ 이용 신청      상품 더 알아보기 ▾

- Cloud Log Analytics > Management > Server > CLA 서비스를 신청할 서버(VM) 선택
  - [수집 설정] 선택

Cloud Log Analytics / Management

## Management

상품 더 알아보기    X 다운로드    새고고침

Server	CDB-MySQL	CDB-MSSQL	Bare Metal Server
<a href="#">수집 설정</a> <a href="#">수집 해제</a>			
<input type="checkbox"/> 서버 이름 <input checked="" type="checkbox"/> Inxsvr1	서버 이미지 이름 centos-6.6-64	상태 운영중	

- Log 수집 설정에서
  - Log Template : Custom Log 선택
  - Log Type : secure 입력
  - Log 경로 : "/var/log/secure\*" 입력
  - 추가 선택 / 적용



- 로그 수집 위한 CLA 에이전트 설치  
팝업창으로 뜨는 에이전트 설치 명령어를 로그 수집 대상 VM에서 실행

만약, 현재 사용중인 계정이 student 일 경우, root로 유저 변경 후에 실행. (명령어: su - root)

#### Cloud Log Analytics상품을 사용하기 위한 서버 내 agent 설치 방법 가이드

설정하신 로그 수집 정보의 등록이 완료 되었습니다.

로그 수집을 시작하기 위해서는 설정한 서버들에 다음과 같은 절차로 agent가 설치되어야 합니다.

콘솔 화면에서 수집 설정 등록을 완료 하였어도, 서버에 agent가 설치되지 않으면 로그가 수집되지 않으니 참고하시기 바랍니다.

1. 로그 수집 서버에 접속해 로그인을 합니다.

자세한 서버 접속 방법은 사용자 가이드의 "[서버 접속 가이드](#)"를 참고하시기 바랍니다.

2. 아래 표시된 "로그 수집 agent 설치 명령어"를 서버에서 실행합니다. 클립보드에 복사하기를 이용하시면 편리합니다.

로그 수집 agent 설치 명령어 :

Linux: [클립보드에 복사하기](#)

```
curl -s cm.cla.ncloud.com/setUpCla/619d44b0f9844458a7dfe4c042b78f0f | sudo sh
```

3. "로그 수집 agent 설치 명령어"를 서버에서 실행하면 자동으로 agent 다운로드 및 설치, 설정까지 진행됩니다.  
설정이 완료되면 아래와 같은 메시지가 출력됩니다.

로그 수집 agent 설치 명령어 실행 결과 ===== Finish Installation =====

4. Cloud Log Analytics 대시보드 화면에서 설정한 서버의 로그가 수집되고 있는지 확인합니다.

만약 수집이 안되는 경우 agent 설치 중 오류가 발생했는지 확인하고, 2번 단계를 다시 진행해 주시기 바랍니다.

```
curl -s cm.cla.ncloud.com/setUpCla/f8f3e1ee7a644fc083831ea86cd3ee5e | sudo sh  
===== Start Installation =====  
1. Check the Configuration for Installation  
    Configuration Success  
2. Connection Success for Installation  
    http_status_code: 200  
3. Remove Agent  
4. Download the Agent for Installation  
    Download Success  
5. Install and config the Agent  
    Installation Success  
Created symlink from /etc/systemd/system/multi-user.target.wants/filebeat.service to  
/usr/lib/systemd/system/filebeat.service.  
Configuration Success  
6. Run the Agent  
===== Finish Installation =====
```

```
[root@lnxsvrl ~]# curl -s http://cm.vcla.ncloud.com/setUpClaVPC/5bcbd064e27b4796  
8b597dbdac2c6391 | sudo sh  
===== Start Installation =====  
1. Check the Configuration for Installation  
    Configuration Success  
2. Connection Success for Installation  
    http_status_code: 200  
3. Remove Agent  
4. Download the Agent for Installation  
    Download Success  
5. Install and config the Agent  
    Installation Success  
Created symlink from /etc/systemd/system/multi-user.target.wants/filebeat.service to  
/usr/lib/systemd/system/filebeat.service.  
Configuration Success  
6. Run the Agent  
===== Finish Installation =====  
[root@lnxsvrl ~]# █
```

### Cloud DB for MySQL 생성

- All product > Cloud DB for MySQL 클릭
- + DB Server 생성 클릭

The screenshot shows the Naver Cloud Platform's 'Cloud DB for MySQL' service page. On the left, there's a sidebar with 'All Products' and a list of services: Dashboard, My Products (with 8 items), Cloud DB for MySQL (selected), DB Server, Monitoring, Backup, Event, IPsec VPN, Server, and Object Storage. The main content area has a title 'Cloud DB for MySQL' with a blue circular badge showing '0'. Below it, there's a section about managing databases and a row of checkmarks for features like automatic fail-over, convenient configuration, monitoring, and alerts. At the bottom right of the main area is a blue button labeled '+ DB Server 생성'.

- 아래와 같이 DB 서버 생성을 위한 관련 정보를 기입

- DB 엔진버전 : 5.7.32
- VPC : lab1-VPC
- Subnet : lab1-vpc-web-subnet
- DB Server 타입 : Standard, vCPU 2 개, Memory 8GB 선택
- 데이터스토리지타입 : HDD
- DB Server 이름 : edu
- DB 서비스 이름 : edu

DBMS 종류	MySQL		
DB 엔진 버전	mysql(5.7.29) <div style="float: right;">▼</div>		
DB 라이센스	General Public License		
고가용성 지원	<input checked="" type="checkbox"/> 고가용성을 선택하면 Standby DB Server를 포함하여 2대의 서버가 생성되며 추가 요금이 발생합니다.		
Multi Zone	<input type="checkbox"/> Master DB 2대를 서로 다른 Zone에 생성하여 더욱 높은 가용성을 제공합니다.		
VPC *	left <div style="float: right;">▼</div>		
Subnet *	pub1   KR-2   Public <div style="float: right;">▼</div>		
	<input type="button" value="Subnet 생성 □"/>		
DB Server 타입	Standard <div style="float: right;">▼</div>		
	vCPU 2개, 메모리 8GB <div style="float: right;">▼</div>		
데이터 스토리지 암호화 적용	<input type="checkbox"/> 암호화 적용시 DB 데이터는 암호화 되어 스토리지에 저장됩니다. DB 서버 생성 이후에는 스토리지 암호화 설정 변경이 불가능합니다.		
데이터 스토리지 타입	<input checked="" type="radio"/> SSD <input type="radio"/> HDD    설치 이후에 스토리지 타입은 변경되지 않습니다.		
데이터 스토리지 용량	기본 10GB    10GB 단위로 과금되며, 최대 6000GB 까지 자동 증가합니다.		
요금제	<a href="#">시간 요금제</a> <a href="#">요금 안내</a>		
DB Server 이름 *	<input type="text"/> <span style="float: right;">최소 3글자, 최대 20자</span> <small>호스트명 중복 방지를 위해 임의의 text가 추가로 포함되어 만들어 집니다.(-xxxx)</small>		
DB 서비스 이름 *	<input type="text"/> <span style="float: right;">최소 3글자, 최대 30자</span>		
ACG 설정	<small>Cloud DB 를 위한 ACG는 자동 생성됩니다.(예 : cloud-mysql-*)          DB Server 접근을 위한 ACG 설정은 사용자 가이드를 참고하세요.</small>		

● DB 설정을 위한 정보 입력

- USER\_ID: student
- HOST(IP): %
- USER 암호: 원하는 암호 입력

- DB 접속포트: 3306
- 기본 DB 명: sakila
- Back up 파일 보관: 1 일
- Back up 시간: 사용자 정의 후 오후 2 시 입력
- 위와 같이 입력 후, 다음 버튼 클릭

서버설정

2 DB 설정

3 최종확인

USER_ID *	student	최소 4글자, 최대 16자	
HOST(IP) *	%	DB 접근 IP 입력	
USER 암호 *	*****	최소 8글자, 최대 20자	
DB 접속 포트 *	3306	3306 또는 10000 ~ 20000만 입력 가능합니다.	
기본 DB 명 *	sakila	최소 1글자, 최대 20자	
DB Config 설정	naver-mysql-5.7-standard		
DB log수집	<input checked="" type="checkbox"/> DB log 수집 및 뷰어 기능을 제공합니다.		
<b>Backup 설정</b>			
<input checked="" type="checkbox"/> Mysql 의 Backup 설정을 사용합니다.			
Backup 파일 보관 기간	1일		
Backup 시간	사용자 정의	14시 00분	선택한 시간 + 15분 사이에 Backup이 시작됩니다.
<a href="#">&lt; 이전</a> <a href="#">다음 &gt;</a>			

- 마지막에 CLOUD DB 기입 정보 확인 후, 생성 버튼 클릭. 다음과 같이 생성됨을 확인

재시작	DB Server 삭제	Monitoring	DB 관리	DB Server 이름 검색			
				데이터 스토리지	Status	ZONE	
	DB 서비스 이름	DB Role	DB Server 이름	DB Server 타입			
<input type="checkbox"/>	edu	Master	edu-001	2vCPU, 4GB Mem	10 GB	<span style="color: green;">● 운영중</span>	KR-2
<input type="checkbox"/>	edu	Standby Master	edu-002	2vCPU, 4GB Mem			

그리고 DB 서버를 접근하기 위한 도메인을 확인합니다.

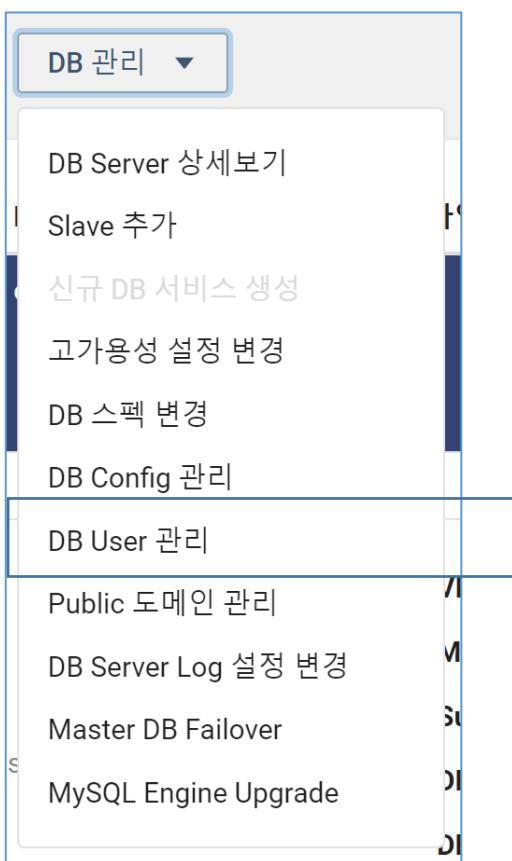
<b>Private 도메인</b>	db-66jdh.vpc-cdb.ntruss.com
<b>Public 도메인</b>	미할당

### ■ 사용자 추가 및 DB 추가

어플리케이션 실습을 위해 사용자와 DB 를 추가

사용자 추가

DB 서버 선택 후 DB 관리 선택 > DB User 관리



USER ID : ncpuser

HOST : %

DB권한 : DDL

암호 : ncp!@#12345

**DB 계정을 추가 및 삭제 할 수 있습니다.**

사용자가 변경한 DB 계정은 DB 서비스 전체에 적용됩니다.  
USER\_ID + HOST(IP) 단위로 계정 추가 및 권한 관리를 합니다.  
DB 권한에서 DDL 권한은 CRUD 권한을 포함합니다.  
(최대 1000개까지 계정을 추가 및 조회 할 수 있습니다.)

USER_ID	HOST(IP)	DB 권한	암호	설정
ncpuser	%	DDL	*****	+ DB User 추가
최소 4글자, 최대 16자	접근 IP 입력	최소 8글자, 최대 20자		

+ DB User 추가 버튼을 선택한 후 저장을 선택합니다.

DB 생성은 DB 선택 후 DB관리 > DB Server 상세보기에서 DB를 추가할 수 있습니다.



상단 탭에서 Database 관리를 선택합니다.

Process list	Variables	Status	Database 관리	DB Config 관리	DB User 관리	Backup 설정 관리	DB Server Logs
--------------	-----------	--------	-------------	--------------	------------	--------------	----------------

**Database를 생성 및 삭제 할 수 있습니다.**  
**Database를 삭제하면 선택한 Database의 모든 데이터가 지워집니다.**

사용자가 변경한 Database는 DB 서비스 전체에 적용됩니다.  
(Database 추가 및 삭제 작업은 한번에 10개만 가능하고, 최대 1000개까지 생성 및 조회 할 수 있습니다.)

Database Name	설정
<input type="text"/>	+ Database 추가

Database Name : application

+ Database 추가 를 선택한 후 하단의 저장을 선택합니다.

현재 생성된 edu 데이터서버의 ACG 를 확인해보면, 본인의 PC 가 접속할 수 있는 ACG 가 설정이 되어 있지 않은 상태이며 서버에서 접근하기 위해서는 ACG 설정이 필요합니다.

All Products> server> ACG 에서 Cloud DB for MySQL 에 해당하는 ACG 를 검색 후, 다음 정책 추가합니다.

Inbound : 프로토콜:TCP 접근소스: 0.0.0.0/0 에 대해 허용포트 3306 추가

Outbound : 프로토콜:TCP 접근소스: 0.0.0.0/0 주소에 대해 허용포트 1-65535 을 추가합니다.

Inbound	Outbound			설정
프로토콜*	접근 소스*	허용 포트*	메모	
TCP	myip			+ 추가
	예1) IP: 0.0.0.0/ 192.168.1.0/24, 192.168.1.7 예2) ACG 이름 : my-acg-1	예1) 단일포트 : 22 예2) 범위지정 : 1-65535		
	<b>Detail</b>			
TCP	0.0.0.0/0(전체)	3306		x
TCP	cloud-mysql-3vhkg(14017)	3306	(automatically created, don't delete it) for the DB service itself.	x

## 서버 설정

서버 실습을 위해 서버에 다음과 같은 설정 내용을 반영하여야 합니다.

### 변경 파일

```
/var/www/html/dbconnect.php
```

```
/var/www/html/key.php
```

필요한 정보는 다음과 같습니다.

Cloud DB for Mysql에서의 Private Domain  
API Key, Secret Key

/var/www/html/dbconnect.php 파일에서 다음 내용을 수정합니다.

```
<?php

$servername = "Cloud DB for Mysql에서의 Private Domain";
$username = "ncpuser";
$password = "ncp!@#12345";
$dbname = "application";
```

```
// Create connection  
  
$conn = new mysqli($servername, $username, $password, $dbname);  
  
// Check connection  
  
if ($conn->connect_error) {  
  
    die("Connection failed: " . $conn->connect_error);  
  
}  
  
?>
```

DB 설정을 위해 SQL 스크립트 실행합니다.

```
mysql -u ncpuser -p -h CDB도메인 < /var/www/html/dbstep3.sql
```

다음과 같이 데이터가 잘 들어갔는지 확인한다.

```
[root@test7-ncp html]# mysql -u ncpuser -p -h db-66jdh.vpc-cdb.ntruss.com
```

```
Enter password:
```

```
Welcome to the MariaDB monitor. Commands end with ; or \g.
```

```
Your MySQL connection id is 5911438
```

```
Server version: 5.7.32-log MySQL Community Server (GPL)
```

```
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
MySQL [(none)]> use application;
```

```
Reading table information for completion of table and column names
```

You can turn off this feature to get a quicker startup with -A

Database changed

```
MySQL [application]> show tables;
```

```
+-----+  
| Tables_in_application |  
+-----+  
| cfr |  
| cpv |  
| csr |  
| css |  
| geolocation |  
| guest_book |  
| papago |  
| ranking_board |  
| shorturl |  
+-----+
```

9 rows in set (0.00 sec)

```
MySQL [application]> select * from cfr;
```

```
+----+-----+-----+-----+  
| no | remote_ip | timestamp | filename |  
+----+-----+-----+-----+  
| 2 | 211.249.70.113 | 2021-04-06 11:32:26 | hyoju.jpg |  
| 3 | 211.249.40.20 | 2021-04-06 11:32:43 | |  
| 4 | 211.249.70.113 | 2021-04-28 10:59:52 | a1.jpg |  
+----+-----+-----+-----+
```

```
3 rows in set (0.00 sec)

MySQL [application]> exit

Bye

[root@test7-ncp html]#
```

Key.php 파일을 편집합니다. 이를 위해 다음 내용을 확인하여야 합니다.

[www.ncloud.com](http://www.ncloud.com)에서 마이페이지 > 인증키 관리에서 Access Key ID와 Secret Key를 확인합니다.

비밀번호 변경	회원정보 변경	파트너 관리	2차인증 관리	SNS 연동	계정 변경	인증키 관리	
API 인증키 관리							
Access Key ID	Secret Key	생성일자	상태	관리			
s5KIM9e0UcfwK1DrFOfJ	<button>보기</button>	2019년 08월 23일	사용 중	<button>사용 중지</button>			
Vzhx58O7NTkuCcOWzDYA	<button>보기</button>	2019년 12월 10일	사용 중	<button>사용 중지</button>			

인증키 정보를 key.php 파일(/var/www/html/key.php)에 추가합니다.

```
<?php

/* API Key */

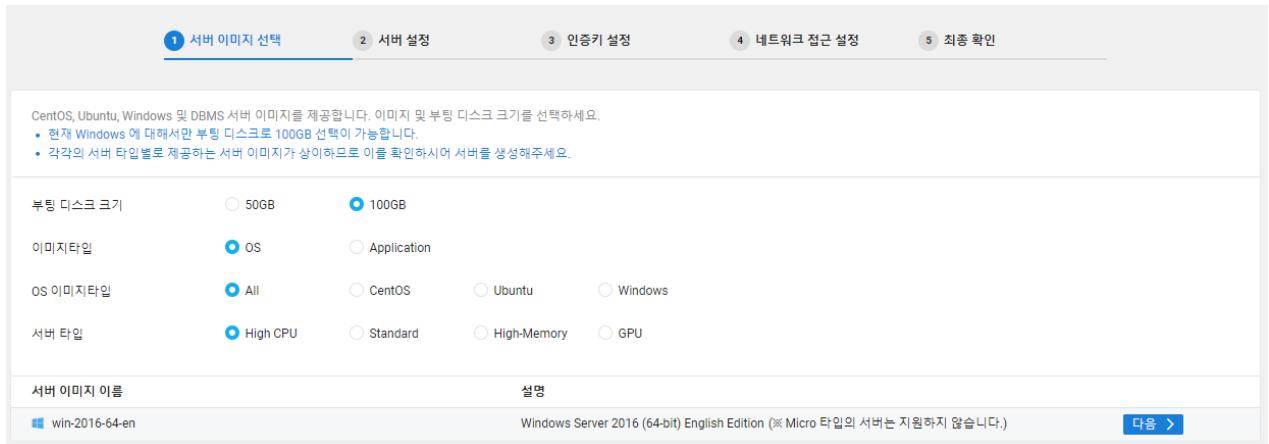
$client_id = "ACCESS_KEY";
$client_secret = "SECRET_KEY";

?>
```

### winsvr1 생성

Server > +서버 생성 버튼을 클릭합니다.

부팅 디스크 크기를 100GB로 변경하면, 하단에 Windows OS 서버 이미지가 보입니다.



Windows 2016 이미지 오른쪽 [다음] 버튼을 클릭합니다.

아래 환경에 맞춰 서버를 생성합니다.

VPC : lab1-vpc

Subnet : lab1-vpc-web-subnet

서버 타입은 vCPU 2개, 메모리 4GB, 디스크 100GB 를 선택합니다.

서버 개수는 1, 서버 이름은 winsvr1 입니다.

Network Interface의 IP는 10.0.1.102 을 입력합니다.

공인 IP > [새로운 공인 IP 할당] 클릭

서버가 생성되며 공인IP를 자동으로 할당 받게 됩니다.

### 관리자 패스워드 확인

윈도우 서버의 경우 생성 시 init script를 따로 적용시키지 않았습니다. 이에 관리자 계정 패스워드는 다음과 같이 인증키를 통해 확인할 수 있습니다.

- 초기 접속 윈도우 계정 : Administrator
- 서버 선택 > [서버 관리 및 설정 변경] > 관리자 비밀번호 확인 > 서버 생성 시 생성한 인증키 선택 > [비밀번호 확인]

The screenshot shows the NAVER Cloud Platform interface. At the top, there are tabs: '시작' (Start), '정지' (Stop), '재시작' (Restart), '반납' (Return), '강제 정지' (Force Stop), and '서버 접속 콘솔' (Server Console). Below these are buttons for '모니터링' (Monitoring), '포트 포워딩 설정' (Port Forwarding Settings), and '서버 관리 및 설정 변경' (Server Management and Configuration Changes). A dropdown menu is open under '서버 관리 및 설정 변경'. On the right, there are '필터' (Filter) and 'Zone' buttons.

The main area displays a list of servers. One server, 'winsrv1', is selected and highlighted with a red box. Another red box highlights the '관리자 비밀번호 확인' (Administrator Password Confirmation) link in the context menu for this server. The server details shown are: 서버 이름 (Server Name): winsrv1, 상태 (Status): 운영중 (Operational), 비공인 IP (Public IP): 10.41.7.145.

A modal dialog box titled '관리자 비밀번호 확인' (Administrator Password Confirmation) is open. It contains the following text: '인증키 내용을 확인합니다.' (Check the verifier key content). Below it, it says: '관리자 비밀번호를 확인하기 위해서 해당 서버의 인증키가 필요합니다.' (A verifier key for this server is required to confirm the administrator password). It also states: '서버 생성 시에 설정한 인증키 파일을 첨부하시고, [비밀번호 확인]을 클릭하면 관리자 비밀번호가 제공됩니다.' (If you have attached the verifier key file set when creating the server, click [Password Confirmation] to provide the administrator password). A note at the bottom left says '(필수 입력 사항입니다.)' (This is a required input item).

The dialog form fields are: '서버 이름' (Server Name): winsrv1, '인증키 이름' (Verifier Key Name): ncp-new-key. There is a large red box around the '(ex) C:\Users\사용자명\Downloads\heeauth.pem' placeholder text and the '마우스로 파일을 끌고 오거나 여기를 클릭하세요' (Drag and drop the file or click here) button.

At the bottom of the dialog are two buttons: '× 취소' (Cancel) and '✓ 비밀번호 확인' (Password Confirmation).

### 스토리지 생성 1

- Lnxsvr1 서버를 선택한 후 상단 메뉴의 서버 관리 및 설정 변경을 선택 후 “스토리지 생성” 선택
- 스토리지 종류 : HDD

- 스토리지 이름 : lnxsvr1-disk1
- 적용 서버 : lnxsvr1
- 크기 : 10GB

스토리지 생성

스토리지 이름과 크기를 입력해 주세요.

(\*필수 입력 사항입니다.)

서버 이름 *	target-linux
스토리지 종류 *	<input checked="" type="radio"/> SSD <input type="radio"/> HDD
기본 스토리지 암호화 적용	N
스토리지 이름 *	lnxsvr-disk1
스냅샷 선택	(선택 없음)
크기 *	10 GB
Max IOPS	4000
메모	0 / 1000 Bytes

취소

Lnxsvr1 서버에 접속하여 다음과 같이 작업 진행

마운트 하기 위한 폴더를 생성

```
mkdir /disk1
```

생성하여 Attach한 디스크의 파티션 생성을 진행

```
[root@lnxsvr-org ~]# fdisk /dev/xvdb
```

```
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 0xd7dfcbf5.

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): **1**

First sector (2048-20971519, default 2048):

Using default value 2048

Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519):

Using default value 20971519

Partition 1 of type Linux and of size 10 GiB is set

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

The partition table has been altered!

Calling ioctl() to re-read partition table.

Syncing disks.

[root@lnxsvr-org ~]#

생성한 /dev/xvdb1 파티션에 대해 포맷을 진행

[root@lnxsvr-org ~]# **mkfs.ext4 /dev/xvdb1**

```
mke2fs 1.42.9 (28-Dec-2013)

Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
655360 inodes, 2621184 blocks
131059 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2151677952
80 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

[root@lnxsvr-org ~]#
```

마운트 작업 진행하고 데이터를 생성

```
[root@lnxsvr-org ~]# mount /dev/xvdb1 /disk1
[root@lnxsvr-org ~]# df
Filesystem      1K-blocks    Used Available Use% Mounted on
```

/dev/xvda3	50305028	1957200	48347828	4%	/
devtmpfs	1926388	0	1926388	0%	/dev
tmpfs	1809260	0	1809260	0%	/dev/shm
tmpfs	1809260	8536	1800724	1%	/run
tmpfs	1809260	0	1809260	0%	/sys/fs/cgroup
tmpfs	361852	0	361852	0%	/run/user/0
/dev/xvdb1	10189076	36888	9611568	1%	/disk1
[root@lnxsvr-org ~]# cp -rf /etc/* /disk1/					

### 상세 모니터링 설정

- Server > Server로 이동
- 모든 서버를 선택하고 “서버 관리 및 설정 변경”에서 “상세 모니터링 설정 변경” 선택

### 이미지로 서버 생성 1

- Lnxsvr1 서버 선택 > 상단의 서버 관리 및 설정 변경 > 내 서버 이미지 생성 클릭
- 내서버 이미지 이름은 ‘lnxsvr-img1’ 입력
- Server> Server Image를 선택
- lnxsvr-image1을 선택하고 +서버생성을 선택
- VPC : lab1-VPC
- Subnet : lab1-vpc-web-subnet
- 스토리지 종류: HDD
- 서버 타입: 2vCPU, 4G Memory
- 서버 이름: lnxsvr2
- Network Interface : 추가 버튼 클릭 (IP가 자동으로 할당됩니다.)

하단의 다음 버튼 클릭 후, 보유하고 있는 인증키 사용 확인 후 다음 버튼 클릭

- ACG : lab1-web-acg

- 마지막 서버 설정 사항 확인 후 하단의 서버 생성 클릭

## Lab 2

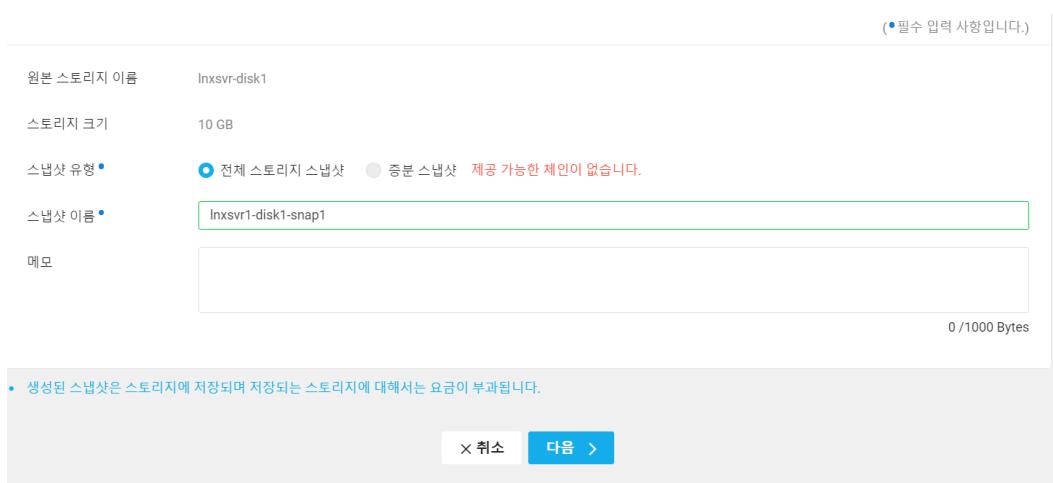
### 1. 추가 스토리지에 대해 스냅샷 생성

All Product > Compute > Server > Storage 선택

Lnxsvr-disk1 선택 후 스토리지 설정 > 스냅샷 생성 선택

전체 스토리지 스냅샷 선택

스냅샷 이름에 Lnxsvr1-disk1-snap1 입력



(•필수 입력 사항입니다.)

원본 스토리지 이름	Inxsvr-disk1
스토리지 크기	10 GB
스냅샷 유형*	<input checked="" type="radio"/> 전체 스토리지 스냅샷 <input type="radio"/> 증분 스냅샷 <small>제공 가능한 체인이 없습니다.</small>
스냅샷 이름*	Inxsvr1-disk1-snap1
메모	<div style="height: 40px; border: 1px solid #ccc; margin-top: 10px;"> </div>

0 /1000 Bytes

• 생성된 스냅샷은 스토리지에 저장되며 저장되는 스토리지에 대해서는 요금이 부과됩니다.

스토리지에 데이터 추가

```
[root@lnxsvr-org ~]# cp -rf /var/log /disk1
[root@lnxsvr-org ~]#
```

### 2. 증분 스냅샷 생성

All Product > Compute > Server > Storage 선택

Lnxsvr1-disk1 선택 후 스토리지 설정 > 스냅샷 생성 선택

증분 스냅샷 선택

스냅샷 이름에 Inxsvr1-disk1-snap2 입력

(• 필수 입력 사항입니다.)

원본 스토리지 이름	Inxsvr-disk1
스토리지 크기	10 GB
스냅샷 유형 *	<input type="radio"/> 전체 스토리지 스냅샷 <input checked="" type="radio"/> 증분 스냅샷
스냅샷 이름 *	Inxsvr1-disk1-snap2
베이스 스냅샷	Inxsvr1-disk1-snap1 생성 일시 : 2021-06-21 오전 7:46 (UTC+09:00)
메모	
	0 / 1000 Bytes

• 생성된 스냅샷은 스토리지에 저장되며 저장되는 스토리지에 대해서는 요금이 부과됩니다.

[× 취소](#) [다음 >](#)

All Product &gt; Compute &gt; Server &gt; Snapshot 선택

## 생성된 스냅샷의 크기 비교

스냅샷 이름	스냅샷 유형	Depth (n)	상태	원본 스토리지 이름	크기
Inxsvr1-disk1-snap2	증분 스냅샷	1	생성됨	Inxsvr-disk1	0 GB
<b>Inxsvr1-disk1-snap1</b>	전체 스토리지 스냅샷	0	생성됨	Inxsvr-disk1	<b>10 GB</b>

상세정보

스냅샷 이름(Instance ID)	Inxsvr1-disk1-snap1(7030974)	생성 일시	2021-0
---------------------	------------------------------	-------	--------

## 3. 스토리지 암호화된 서버 생성

All Product &gt; Compute &gt; Server &gt; Server &gt; +서버 생성 선택

이미지는 **Centos-7.8-64**를 선택합니다.

서버 이미지 이름	설명
centos-7.3-64	CentOS 7.3 (64-bit) (커널 업데이트 시 서버의 정상적인 사용이 불가능할 수 있으며 이에 따른 복구는 지원하지 않습니다.)
centos-7.8-64	CentOS 7.8 (64-bit) (커널 업데이트 시 서버의 정상적인 사용이 불가능할 수 있으며 이에 따른 복구는 지원하지 않습니다.)
ubuntu-16.04-64-server	Ubuntu Server 16.04 (64-bit) (커널 업데이트 시 서버의 정상적인 사용이 불가능할 수 있으며 이에 따른 복구는 지원하지 않습니다.)

VPC : lab-vpc1

Subnet : lab1-vpc-web-subnet

서버 타입은 vCPU 2개, 메모리 4GB, 디스크 50GB 를 선택합니다.

서버 개수는 1, 서버 이름은 lnxsvr-en 입니다.

스토리지 암호화 적용을 선택합니다.

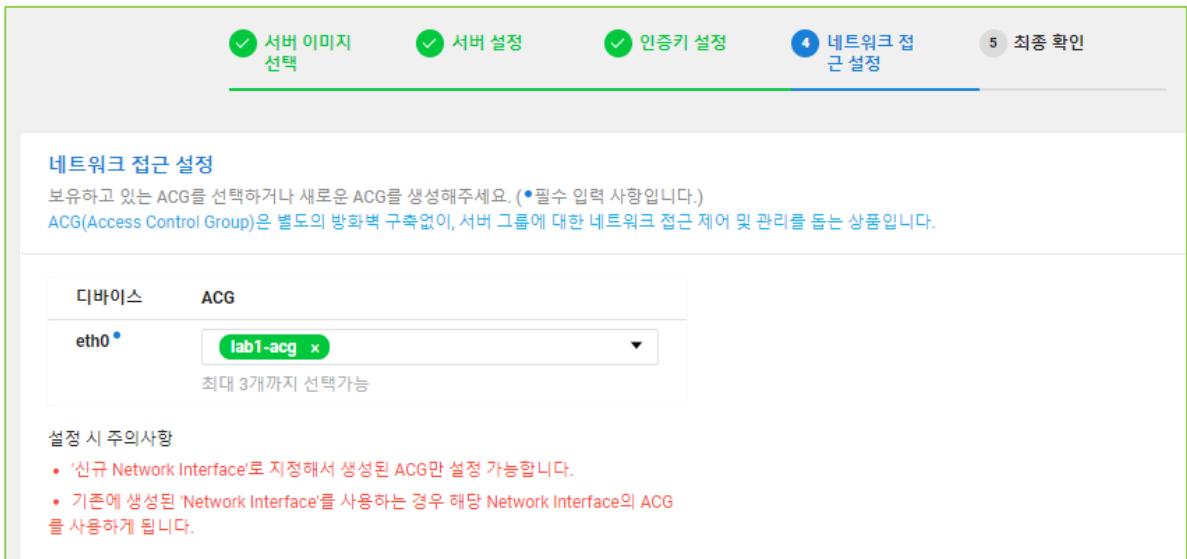
Network Interface의 IP는 10.0.1.181 을 입력합니다.

스토리지 종류 •	<input checked="" type="radio"/> SSD <input type="radio"/> HDD
서버 세대 •	g2
서버 타입 •	High CPU
	[High CPU] vCPU 2개, 메모리 4GB, [SSD]디스크 50GB [g2]
	C2-g2-s50 ▾
스토리지 암호화 적용	<input checked="" type="checkbox"/> 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 •	<input checked="" type="radio"/> 월요금제 <input type="radio"/> 시간 요금제    월 72,000원 (OS 제외)
서버 개수 •	1
서버 이름	최소 3글자, 최대 30자 (서버 이름을 작성하지 않으면 자동 생성됩니다.)
	<input checked="" type="checkbox"/> 입력하신 서버 이름으로 hostname을 설정합니다.

Init script는 미선택,

인증키는 lab1에서 만든 인증키를 사용합니다.

네트워크 접근 설정에서 eth0 NIC에 lab1-web-acg 를 할당합니다.



서버가 생성되면 공인 IP를 할당하고 서버에 SSH로 로그인합니다.

#### 4. 스토리지 암호화된 스토리지 생성

All Product > Compute > Server > Server > lnxsvr-en 선택

- lnxsvr-en 서버를 선택한 후 상단의 ‘정지’ 버튼 클릭
- 정지가 완료되면, 상단 메뉴의 서버 관리 및 설정 변경을 선택 후 “스토리지 생성” 선택
- 스토리지 종류 : HDD
- 스토리지 이름 : lnxsvr-en-disk1
- 적용 서버 : lnxsvr-en
- 크기 : 10GB
- 스토리지 추가가 완료되면 상단의 ‘시작’ 버튼 클릭

lnxsvr-en 서버에 접속하여 다음과 같이 작업 진행

- (해당 서버에 접속하기 위해서 이전 실습과 동일하게 Public IP 부여 필요)
- 마운트 하기 위한 폴더를 생성  
`mkdir /disk1`
- 생성하여 Attach한 디스크의 파티션 생성을 진행

```
[root@lnxsvr-org ~]# fdisk /dev/xvdb
```

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 0xd7dfcbf5.

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): 1

First sector (2048-20971519, default 2048):

Using default value 2048

Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519):

Using default value 20971519

Partition 1 of type Linux and of size 10 GiB is set

Command (m for help): w

The partition table has been altered!

Calling ioctl() to re-read partition table.

Syncing disks.

```
[root@lnxsvr-org ~]#
```

생성한 /dev/xvdb1 파티션에 대해 포맷을 진행, 포맷이 안되는 것 확인

```
[root@lnxsvr-org ~]# mkfs.ext4 /dev/xvdb1
mke2fs 1.39 (29-Oct-2015)
/dev/sdb2 is apparently in use by the system; will not make a filesystem here!
[root@lnxsvr-org ~]#
```

가비지 정보 업데이트

```
[root@lnxsvr-org ~]# dmsetup status
[root@lnxsvr-org ~]# dmsetup remove_all
[root@lnxsvr-org ~]# dmsetup status
```

포맷 재시도

```
[root@lnxsvr-org ~]# mkfs.ext4 /dev/xvdb1
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
655360 inodes, 2621184 blocks
131059 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2151677952
80 block groups
```

```

32768 blocks per group, 32768 fragments per group

8192 inodes per group

Superblock backups stored on blocks:

      32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done

Writing inode tables: done

Creating journal (32768 blocks): done

Writing superblocks and filesystem accounting information: done

[root@lnxsvr-org ~]#

```

마운트 작업 진행하고 데이터를 생성

```

[root@lnxsvr-org ~]# mount /dev/xvdb1 /disk1

[root@lnxsvr-org ~]# df

Filesystem      1K-blocks    Used Available Use% Mounted on
/dev/xvda3        50305028 1957200   48347828   4% /
devtmpfs          1926388     0   1926388   0% /dev
tmpfs            1809260     0   1809260   0% /dev/shm
tmpfs            1809260    8536   1800724   1% /run
tmpfs            1809260     0   1809260   0% /sys/fs/cgroup
tmpfs            361852      0   361852   0% /run/user/0
/dev/xvdb1        10189076  36888   9611568   1% /disk1

[root@lnxsvr-org ~]# cp -rf /etc/* /disk1/

```

## Lab 3

### 1. CLI 설치

Lnxsrv1 로그인 후 다음 명령 실행

```
[root@target-linux ~]# wget https://www.ncloud.com/api/support/download/5/65
Resolving www.ncloud.com (www.ncloud.com)... 49.236.142.51
Connecting to www.ncloud.com (www.ncloud.com)|49.236.142.51|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 177330843 (169M) [application/zip]
Saving to: ?5?

100%[=====] 177,330,843 103MB/s
in 1.6s

[root@target-linux ~]# ls
65
[root@target-linux ~]#
```

압축 해제

```
[root@target-linux ~]# unzip 65
```

환경 설정

```
[root@target-linux ~]# cd cli_linux/
[root@target-linux cli_linux]# chmod 755 -R *
[root@target-linux cli_linux]#
[root@target-linux cli_linux]# ./ncloud configure
set [DEFAULT]'s configuration.
Ncloud Access Key ID []: api-key
Ncloud Secret Access Key []: secret-key
Ncloud API URL (default:https://ncloud.apigw.ntruss.com) []:
```

```
[root@target-linux cli_linux]#
```

## VPC 조회

```
[root@target-linux cli_linux]# ./ncloud vpc getVpcList --regionCode KR --vpcStatusCode RUN
{"getVpcListResponse": {
    "requestId": "d9f05a65-a3ee-4d22-8f9f-18f23b5a4317",
    "returnCode": "0",
    "returnMessage": "success",
    "totalRows": 3,
    "vpcList": [
        {
            "vpcNo": "9052",
            "vpcName": "security-lab",
            "ipv4CidrBlock": "10.10.0.0/16",
            "vpcStatus": {
                "code": "RUN",
                "codeName": "run"
            },
            "regionCode": "KR",
            "createDate": "2021-06-18T13:52:40+0900"
        },
        {
            "vpcNo": "9006",
            "vpcName": "targ20et",
            "ipv4CidrBlock": "192.168.0.0/16",
            "vpcStatus": {
                "code": "RUN",
                "codeName": "run"
            },
            "regionCode": "KR",
            "createDate": "2021-06-17T13:16:51+0900"
        },
        {
            "vpcNo": "9007",
            "vpcName": "targ20et",
            "ipv4CidrBlock": "192.168.0.0/16",
            "vpcStatus": {
                "code": "RUN",
                "codeName": "run"
            },
            "regionCode": "KR",
            "createDate": "2021-06-17T13:16:51+0900"
        }
    ]
}}
```

```

    "vpcNo": "2511",
    "vpcName": "naksoo",
    "ipv4CidrBlock": "10.4.0.0/16",
    "vpcStatus": {
        "code": "RUN",
        "codeName": "run"
    },
    "regionCode": "KR",
    "createDate": "2020-11-17T14:54:44+0900"
}
]
}}

```

[root@target-linux cli\_linux]#

## Subnet 조회

```

[root@target-linux cli_linux]# ./ncloud vpc getSubnetList
{"getSubnetListResponse": {
    "requestId": "b8b6659f-ea0a-465d-9ec5-1e32c3b630db",
    "returnCode": "0",
    "returnMessage": "success",
    "totalRows": 2,
    "subnetList": [
        {
            "subnetNo": "18215",
            "vpcNo": "9006",
            "zoneCode": "KR-2",
            "subnetName": "target-private",
            "subnet": "192.168.250.0/24",
            "subnetStatus": {
                "code": "RUN",
                "codeName": "run"
            },
            "createDate": "2021-06-17T17:57:20+0900",
            "subnetType": {

```

```
        "code": "PRIVATE",
        "codeName": "Private"
    },
    "usageType": {
        "code": "GEN",
        "codeName": "General"
    },
    "networkAclNo": "12687"
},
{
    "subnetNo": "18181",
    "vpcNo": "9006",
    "zoneCode": "KR-2",
    "subnetName": "target-web",
    "subnet": "192.168.1.0/24",
    "subnetStatus": {
        "code": "RUN",
        "codeName": "run"
    },
    "createDate": "2021-06-17T13:18:08+0900",
    "subnetType": {
        "code": "PUBLIC",
        "codeName": "Public"
    },
    "usageType": {
        "code": "GEN",
        "codeName": "General"
    },
    "networkAclNo": "12687"
}
]
}}
```

[root@target-linux cli\_linux]#

```
[root@target-linux cli_linux]# ./ncloud vpc getNetworkAclList --regionCode KR -  
-networkAclStatusCode RUN --vpcNo 9006  
{  
    "getNetworkAclListResponse": {  
        "requestId": "268321cd-6a3c-4ddd-b6f5-8efba8119e7c",  
        "returnCode": "0",  
        "returnMessage": "success",  
        "totalRows": 2,  
        "networkAclList": [  
            {  
                "networkAclNo": "12687",  
                "networkAclName": "target-nacl",  
                "vpcNo": "9006",  
                "networkAclStatus": {  
                    "code": "RUN",  
                    "codeName": "run"  
                },  
                "networkAclDescription": "",  
                "createDate": "2021-06-17T13:17:42+0900",  
                "isDefault": false  
            },  
            {  
                "networkAclNo": "12686",  
                "networkAclName": "target-default-network-acl",  
                "vpcNo": "9006",  
                "networkAclStatus": {  
                    "code": "RUN",  
                    "codeName": "run"  
                },  
                "networkAclDescription": "VPC [target] default Network ACL",  
                "createDate": "2021-06-17T13:16:51+0900",  
                "isDefault": true  
            }  
        ]  
    }  
}  
[root@target-linux cli_linux]#
```

## NACL Rule 조회

```
[root@target-linux cli_linux]# ./ncloud vpc getNetworkAclRuleList --regionCode K
R --networkAclNo 12687
{"getNetworkAclRuleListResponse": {
    "requestId": "670ddb70-6e96-45c6-80ed-56d583edcb51",
    "returnCode": "0",
    "returnMessage": "success",
    "totalRows": 7,
    "networkAclRuleList": [
        {
            "networkAclNo": "12687",
            "priority": 2,
            "protocolType": {
                "code": "TCP",
                "codeName": "tcp"
            },
            "portRange": "1-65535",
            "ruleAction": {
                "code": "ALLOW",
                "codeName": "Allow"
            },
            "createDate": "2021-06-21T00:10:04+0900",
            "ipBlock": "116.124.188.0/24",
            "denyAllowGroupNo": "",
            "networkAclRuleType": {
                "code": "INBND",
                "codeName": "Inbound"
            },
            "networkAclRuleDescription": ""
        },
        {
            "networkAclNo": "12687",
            "priority": 41,
            "protocolType": {
```

```
        "code": "ICMP",
        "codeName": "icmp"
    },
    "portRange": "",
    "ruleAction": {
        "code": "ALLOW",
        "codeName": "Allow"
    },
    "createDate": "2021-06-18T14:33:07+0900",
    "ipBlock": "211.249.70.115/32",
    "denyAllowGroupNo": "",
    "networkAclRuleType": {
        "code": "INBND",
        "codeName": "Inbound"
    },
    "networkAclRuleDescription": ""
},
{
    "networkAclNo": "12687",
    "priority": 11,
    "protocolType": {
        "code": "TCP",
        "codeName": "tcp"
    },
    "portRange": "80",
    "ruleAction": {
        "code": "ALLOW",
        "codeName": "Allow"
    },
    "createDate": "2021-06-18T15:21:51+0900",
    "ipBlock": "0.0.0.0/0",
    "denyAllowGroupNo": "",
    "networkAclRuleType": {
        "code": "INBND",
        "codeName": "Inbound"
    },
}
```

```
        "networkAclRuleDescription": "",  
    },  
    {  
        "networkAclNo": "12687",  
        "priority": 12,  
        "protocolType": {  
            "code": "TCP",  
            "codeName": "tcp"  
        },  
        "portRange": "443",  
        "ruleAction": {  
            "code": "ALLOW",  
            "codeName": "Allow"  
        },  
        "createDate": "2021-06-18T15:21:51+0900",  
        "ipBlock": "0.0.0.0/0",  
        "denyAllowGroupNo": "",  
        "networkAclRuleType": {  
            "code": "INBND",  
            "codeName": "Inbound"  
        },  
        "networkAclRuleDescription": ""  
    },  
    {  
        "networkAclNo": "12687",  
        "priority": 1,  
        "protocolType": {  
            "code": "TCP",  
            "codeName": "tcp"  
        },  
        "portRange": "1-65535",  
        "ruleAction": {  
            "code": "ALLOW",  
            "codeName": "Allow"  
        },  
        "createDate": "2021-06-18T15:30:17+0900",  
    }
```

```
"ipBlock": "211.249.70.115/32",
"denyAllowGroupNo": "",
"networkAclRuleType": {
    "code": "INBND",
    "codeName": "Inbound"
},
"networkAclRuleDescription": ""
},
{
    "networkAclNo": "12687",
    "priority": 30,
    "protocolType": {
        "code": "ICMP",
        "codeName": "icmp"
    },
    "portRange": "",
    "ruleAction": {
        "code": "DROP",
        "codeName": "Drop"
    },
    "createDate": "2021-06-18T14:31:01+0900",
    "ipBlock": "0.0.0.0/0",
    "denyAllowGroupNo": "",
    "networkAclRuleType": {
        "code": "INBND",
        "codeName": "Inbound"
    },
    "networkAclRuleDescription": ""
},
{
    "networkAclNo": "12687",
    "priority": 11,
    "protocolType": {
        "code": "TCP",
        "codeName": "tcp"
    },
}
```

```

    "portRange": "3306",
    "ruleAction": {
        "code": "DROP",
        "codeName": "Drop"
    },
    "createDate": "2021-06-18T14:37:21+0900",
    "ipBlock": "0.0.0.0/0",
    "denyAllowGroupNo": "",
    "networkAclRuleType": {
        "code": "OTBND",
        "codeName": "Outbound"
    },
    "networkAclRuleDescription": ""
}
]
})
}

[root@target-linux cli_linux]#

```

## Initscript 조회

```

[root@target-linux cli_linux]# ./ncloud vserver getInitScriptList --regionCode K
R --osTypeCode LNX
{
    "getInitScriptListResponse": {
        "returnCode": "0",
        "returnMessage": "success",
        "totalRows": 4,
        "initScriptList": [
            {
                "initScriptNo": "3773",
                "initScriptName": "lab01-script",
                "createDate": "2021-03-24T11:55:03+0900",
                "initScriptDescription": "",
                "initScriptContent": "",
                "osType": {
                    "code": "LNX",

```

```
        "codeName": "LINUX"
    }
},
{
    "initScriptNo": "1429",
    "initScriptName": "lab-script",
    "createDate": "2020-11-17T15:12:24+0900",
    "initScriptDescription": "",
    "initScriptContent": "",
    "osType": {
        "code": "LNX",
        "codeName": "LINUX"
    }
},
{
    "initScriptNo": "1190",
    "initScriptName": "centos7-init",
    "createDate": "2020-11-05T17:23:58+0900",
    "initScriptDescription": "",
    "initScriptContent": "",
    "osType": {
        "code": "LNX",
        "codeName": "LINUX"
    }
},
{
    "initScriptNo": "907",
    "initScriptName": "centos7-passwd",
    "createDate": "2020-10-12T18:03:13+0900",
    "initScriptDescription": "",
    "initScriptContent": "",
    "osType": {
        "code": "LNX",
        "codeName": "LINUX"
    }
}
```

```
]
}

[root@target-linux cli_linux]#
```

## ACG 조회

```
[root@target-linux cli_linux]# ./ncloud vserver getAccessControlGroupList --regi
onCode KR --vpcNo 9006 --accessControlGroupStatusCode RUN
{"getAccessControlGroupListResponse": {
    "requestId": "ab6f0ce8-ccb2-4438-b8f3-ad9c6d1d5b5a",
    "returnCode": "0",
    "returnMessage": "success",
    "totalRows": 5,
    "accessControlGroupList": [
        {
            "accessControlGroupNo": "18671",
            "accessControlGroupName": "target-adminserver",
            "isDefault": false,
            "vpcNo": "9006",
            "accessControlGroupStatus": {
                "code": "RUN",
                "codeName": "run"
            },
            "accessControlGroupDescription": ""
        },
        {
            "accessControlGroupNo": "18602",
            "accessControlGroupName": "cloud-redis-45xx1",
            "isDefault": false,
            "vpcNo": "9006",
            "accessControlGroupStatus": {
                "code": "RUN",
                "codeName": "run"
            },
            "accessControlGroupDescription": "(automatically created, don't delete it)"
        }
    ]
}}
```

```
cloud-redis-desc"
},
{
    "accessControlGroupNo": "18601",
    "accessControlGroupName": "cloud-mysql-45xwo",
    "isDefault": false,
    "vpcNo": "9006",
    "accessControlGroupStatus": {
        "code": "RUN",
        "codeName": "run"
    },
    "accessControlGroupDescription": "(automatically created, don't delete it)
cloud-mysql-desc"
},
{
    "accessControlGroupNo": "18583",
    "accessControlGroupName": "target-acg",
    "isDefault": false,
    "vpcNo": "9006",
    "accessControlGroupStatus": {
        "code": "RUN",
        "codeName": "run"
    },
    "accessControlGroupDescription": ""
},
{
    "accessControlGroupNo": "18581",
    "accessControlGroupName": "target-default-acg",
    "isDefault": true,
    "vpcNo": "9006",
    "accessControlGroupStatus": {
        "code": "RUN",
        "codeName": "run"
    },
    "accessControlGroupDescription": "VPC [target] default ACG"
}
```

```
]
}

[root@target-linux cli_linux]#
```

## ACG 를 조회

```
[root@target-linux cli_linux]# ./ncloud vserver getAccessControlGroupRuleList --regionCode KR --accessControlGroupNo 18583
{"getAccessControlGroupRuleListResponse": {
    "requestId": "7a62b512-e554-4628-80b4-899f04e167b0",
    "returnCode": "0",
    "returnMessage": "success",
    "totalRows": 10,
    "accessControlGroupRuleList": [
        {
            "accessControlGroupNo": "18583",
            "protocolType": {
                "code": "TCP",
                "codeName": "tcp"
            },
            "ipBlock": "116.124.188.0/24",
            "accessControlGroupSequence": "",
            "portRange": "1-65535",
            "accessControlGroupRuleType": {
                "code": "INBND",
                "codeName": "Inbound"
            },
            "accessControlGroupRuleDescription": ""
        },
        {
            "accessControlGroupNo": "18583",
            "protocolType": {
                "code": "TCP",
                "codeName": "tcp"
            },
            "ipBlock": "116.124.188.0/24",
            "accessControlGroupSequence": "",
            "portRange": "1-65535",
            "accessControlGroupRuleType": {
                "code": "OUTBND",
                "codeName": "Outbound"
            },
            "accessControlGroupRuleDescription": ""
        }
    ]
}}
```

```
"ipBlock": "0.0.0.0/0",
"accessControlGroupSequence": "",
"portRange": "1-65535",
"accessControlGroupRuleType": {
    "code": "INBND",
    "codeName": "Inbound"
},
"accessControlGroupRuleDescription": ""

},
{
    "accessControlGroupNo": "18583",
    "protocolType": {
        "code": "TCP",
        "codeName": "tcp"
    },
    "ipBlock": "211.249.70.115/32",
    "accessControlGroupSequence": "",
    "portRange": "1-65535",
    "accessControlGroupRuleType": {
        "code": "INBND",
        "codeName": "Inbound"
    },
    "accessControlGroupRuleDescription": ""

},
{
    "accessControlGroupNo": "18583",
    "protocolType": {
        "code": "TCP",
        "codeName": "tcp"
    },
    "ipBlock": "",
    "accessControlGroupSequence": "18671",
    "portRange": "22",
    "accessControlGroupRuleType": {
        "code": "INBND",
        "codeName": "Inbound"
    }
}
```

```
        },
        "accessControlGroupRuleDescription": ""
    },
    {
        "accessControlGroupNo": "18583",
        "protocolType": {
            "code": "TCP",
            "codeName": "tcp"
        },
        "ipBlock": "",
        "accessControlGroupSequence": "18671",
        "portRange": "3389",
        "accessControlGroupRuleType": {
            "code": "INBND",
            "codeName": "Inbound"
        },
        "accessControlGroupRuleDescription": ""
    },
    {
        "accessControlGroupNo": "18583",
        "protocolType": {
            "code": "TCP",
            "codeName": "tcp"
        },
        "ipBlock": "0.0.0.0/0",
        "accessControlGroupSequence": "",
        "portRange": "80",
        "accessControlGroupRuleType": {
            "code": "INBND",
            "codeName": "Inbound"
        },
        "accessControlGroupRuleDescription": "HTTP"
    },
    {
        "accessControlGroupNo": "18583",
        "protocolType": {
```

```
        "code": "TCP",
        "codeName": "tcp"
    },
    "ipBlock": "0.0.0.0/0",
    "accessControlGroupSequence": "",
    "portRange": "443",
    "accessControlGroupRuleType": {
        "code": "INBND",
        "codeName": "Inbound"
    },
    "accessControlGroupRuleDescription": "HTTPS"
},
{
    "accessControlGroupNo": "18583",
    "protocolType": {
        "code": "TCP",
        "codeName": "tcp"
    },
    "ipBlock": "0.0.0.0/0",
    "accessControlGroupSequence": "",
    "portRange": "1-65535",
    "accessControlGroupRuleType": {
        "code": "OTBND",
        "codeName": "Outbound"
    },
    "accessControlGroupRuleDescription": ""
},
{
    "accessControlGroupNo": "18583",
    "protocolType": {
        "code": "UDP",
        "codeName": "udp"
    },
    "ipBlock": "0.0.0.0/0",
    "accessControlGroupSequence": "",
    "portRange": "1-65535",
```

```

"accessControlGroupRuleType": {
    "code": "OTBND",
    "codeName": "Outbound"
},
"accessControlGroupRuleDescription": ""
},
{
    "accessControlGroupNo": "18583",
    "protocolType": {
        "code": "ICMP",
        "codeName": "icmp"
    },
    "ipBlock": "0.0.0.0/0",
    "accessControlGroupSequence": "",
    "portRange": "",
    "accessControlGroupRuleType": {
        "code": "OTBND",
        "codeName": "Outbound"
    },
    "accessControlGroupRuleDescription": ""
}
]
}}
[root@target-linux cli_linux]#

```

## 기존 서버 조회

```

[root@target-linux cli_linux]# ./ncloud vserver getServerInstanceList --regionCo
de KR --vpcNo 9006 --serverInstanceStatusCode RUN
{"getServerInstanceListResponse": {
    "returnCode": "0",
    "returnMessage": "success",
    "totalRows": 3,
    "serverInstanceList": [
        {
            "serverInstanceNo": "7031114",
            "serverName": "s17a2baf9bf6",

```

```
"serverDescription": "",  
"cpuCount": 2,  
"memorySize": 4294967296,  
"platformType": {  
    "code": "LNX64",  
    "codeName": "Linux 64 Bit"  
},  
"loginKeyName": "ncp20210617",  
"publiclplnstanceNo": "",  
"publiclp": "",  
"serverlnstanceStatus": {  
    "code": "RUN",  
    "codeName": "Server run state"  
},  
"serverlnstanceOperation": {  
    "code": "NULL",  
    "codeName": "Server NULL OP"  
},  
"serverlnstanceStatusName": "running",  
"createDate": "2021-06-21T08:09:27+0900",  
"uptime": "2021-06-21T08:14:51+0900",  
"serverImageProductCode": "SW.VSVR.OS.LNX64.CNTOS.0708.B050",  
"serverProductCode":  
"SVR.VSVR.HICPU.C002.M004.NET.SSD.B050.G002",  
"isProtectServerTermination": false,  
"zoneCode": "KR-2",  
"regionCode": "KR",  
"vpcNo": "9006",  
"subnetNo": "18215",  
"networklnterfaceNoList": [  
    "250893"  
],  
"initScriptNo": "",  
"serverlnstanceType": {  
    "code": "HICPU",  
    "codeName": "High CPU"  
},  
"baseBlockStorageDiskType": {  
    "code": "NET",  
    "codeName": "Network Storage"  
},  
"baseBlockStorageDiskDetailType": {  
    "code": "SSD",  
    "codeName": "SSD"  
},
```

```
"placementGroupNo": "",  
"placementGroupName": "",  
"memberServerImageInstanceId": ""  
},  
{  
    "serverInstanceId": "6992753",  
    "serverName": "target-win",  
    "serverDescription": "",  
    "cpuCount": 2,  
    "memorySize": 4294967296,  
    "platformType": {  
        "code": "WND64",  
        "codeName": "Windows 64 Bit"  
    },  
    "loginKeyName": "ncp20210617",  
    "publicIpInstanceId": "6993648",  
    "publicIp": "175.106.98.217",  
    "serverInstanceState": {  
        "code": "RUN",  
        "codeName": "Server run state"  
    },  
    "serverInstanceOperation": {  
        "code": "NULL",  
        "codeName": "Server NULL OP"  
    },  
    "serverInstanceStateName": "running",  
    "createDate": "2021-06-17T13:53:41+0900",  
    "uptime": "2021-06-17T14:05:02+0900",  
    "serverImageProductCode":  
    "SW.VSVR.OS.WND64.WND.SVR2016EN.B100",  
    "serverProductCode":  
    "SVR.VSVR.HICPU.C002.M004.NET.SSD.B100.G002",  
    "isProtectServerTermination": false,  
    "zoneCode": "KR-2",  
    "regionCode": "KR",  
    "vpcNo": "9006",  
    "subnetNo": "18181",  
    "networkInterfaceNoList": [  
        "245713"  
    ],  
    "initScriptNo": "",  
    "serverInstanceType": {  
        "code": "HICPU",  
        "codeName": "High CPU"  
    },
```

```
"baseBlockStorageDiskType": {
    "code": "NET",
    "codeName": "Network Storage"
},
"baseBlockStorageDiskDetailType": {
    "code": "SSD",
    "codeName": "SSD"
},
"placementGroupNo": "",
"placementGroupName": "",
"memberServerImageInstanceId": ""
},
{
    "serverInstanceNo": "6992741",
    "serverName": "target-linux",
    "serverDescription": "",
    "cpuCount": 2,
    "memorySize": 4294967296,
    "platformType": {
        "code": "LNX64",
        "codeName": "Linux 64 Bit"
    },
    "loginKeyName": "ncp20210617",
    "publicIpInstanceId": "6992800",
    "publicIp": "175.106.97.47",
    "serverInstanceState": {
        "code": "RUN",
        "codeName": "Server run state"
    },
    "serverInstanceOperation": {
        "code": "NULL",
        "codeName": "Server NULL OP"
    },
    "serverInstanceStateName": "running",
    "createDate": "2021-06-17T13:53:07+0900",
    "uptime": "2021-06-17T13:56:51+0900",
    "serverImageProductCode": "SW.VSVR.OS.LNX64.CNTOS.0708.B050",
    "serverProductCode": "SVR.VSVR.HICPU.C002.M004.NET.SSD.B050.G002",
    "isProtectServerTermination": false,
    "zoneCode": "KR-2",
    "regionCode": "KR",
    "vpcNo": "9006",
    "subnetNo": "18181",
    "networkInterfaceNoList": [

```

```

        "245712"
    ],
    "initScriptNo": "",
    "serverInstanceType": {
        "code": "HICPU",
        "codeName": "High CPU"
    },
    "baseBlockStorageDiskType": {
        "code": "NET",
        "codeName": "Network Storage"
    },
    "baseBlockStorageDiskDetailType": {
        "code": "SSD",
        "codeName": "SSD"
    },
    "placementGroupNo": "",
    "placementGroupName": "",
    "memberServerImageInstanceId": ""
}
]
}}
[root@target-linux cli_linux]#

```

## Lnxsvr3 서버 생성

```

[root@target-linux cli_linux]# ./ncloud vserver createServerInstances --regionCode
KR --serverImageProductCode SW.VSVR.OS.LNX64.CNTOS.0708.B050 --vpcNo
9006 --subnetNo 18181 --serverProductCode
SVR.VSVR.HICPU.C002.M004.NET.SSD.B050.G002 --feeSystemTypeCode MTRAT
--serverCreateCount 1 --serverName lnxsvr3 --networkInterfaceList
"networkInterfaceOrder='0', accessControlGroupNoList=['18583']" --
isProtectServerTermination false --initScriptNo 1429 --associateWithPublicIp true
{"createServerInstancesResponse": {
    "requestId": "3d90bcfc-fb95-4bc3-ad1c-b7199e2a8235",
    "returnCode": "0",
    "returnMessage": "success",
    "totalRows": 1,
    "serverInstanceList": [
        {
            "serverInstanceId": "7031502",
            "serverName": "lnxsvr2",
            "serverDescription": "",
            "cpuCount": 2,
            "memorySize": 4294967296,

```

```
"platformType": {
    "code": "LNX64",
    "codeName": "Linux 64 Bit"
},
"loginKeyName": "ncp20210617",
"publiclplInstanceNo": "",
"publiclp": "",
"serverInstanceStatus": {
    "code": "INIT",
    "codeName": "Server init state"
},
"serverInstanceOperation": {
    "code": "NULL",
    "codeName": "Server NULL OP"
},
"serverInstanceStateName": "init",
"createDate": "2021-06-21T09:07:04+0900",
"uptime": "2021-06-21T09:07:04+0900",
"serverImageProductCode": "SW.VSVR.OS.LNX64.CNTOS.0708.B050",
"serverProductCode": "SVR.VSVR.HICPU.C002.M004.NET.SSD.B050.G002",
"isProtectServerTermination": false,
"zoneCode": "KR-2",
"regionCode": "KR",
"vpcNo": "9006",
"subnetNo": "18181",
"networkInterfaceNoList": [
    "250938"
],
"initScriptNo": "1429",
"serverInstanceType": {
    "code": "HICPU",
    "codeName": "High CPU"
},
"baseBlockStorageDiskType": {
    "code": "NET",
    "codeName": "Network Storage"
},
"baseBlockStorageDiskDetailType": {
    "code": "SSD",
    "codeName": "SSD"
},
"placementGroupNo": "",
"placementGroupName": "",
"memberServerImageInstanceId": ""
```

```
        }
    ]
}
[root@target-linux cli_linux]#
```

## 2. 서버에 추가ip 설정

- Product and services > server > network interface 선택
- Lnxsvr1 서버에 할당되어있는 네트워크인터페이스 선택 후 상단의 secondary IP 버튼 클릭
- Secondary IP 입력란에 '10.0.1.251' 입력 후 우측에 추가 버튼 클릭
- 하단의 설정 버튼 클릭

Lnxsvr1에 로그인 후 /etc/sysconfig/network-scripts/ifcfg-eth0:1파일 생성

파일 내용

```
DEVICE=eth0:1
BOOTPROTO=STATIC
IPADDR=10.0.1.251
NETMASK=255.255.255.0
ONBOOT=yes
```

인터페이스 활성화

```
[root@target-linux network-scripts]# ifup eth0:1
[root@target-linux network-scripts]#
```

## Lab 4

### 1. CLI로 NAS 생성

볼륨 생성

```
[root@target-linux cli_linux]# ./ncloud vnas createNasVolumeInstance --regionCode KR --zoneCode KR-1 --volumeName data --volumeAllotmentProtocolTypeCode NFS --volumeSize 500 --isReturnProtection false

{"createNasVolumeInstanceResponse": {

    "requestId": "f0868d6a-8073-4e55-bcc5-48b249745dc2",

    "returnCode": "0",

    "returnMessage": "success",

    "totalRows": 1,

    "nasVolumeInstanceList": [

        {

            "nasVolumeInstanceNo": "7031541",

            "nasVolumeInstanceState": {

                "code": "CREAT",

                "codeName": "NAS create"

            },

            "nasVolumeInstanceOperation": {

                "code": "NULL",

                "codeName": "NAS NULL OP"

            }

        }

    ]

}}
```

```
"createDate": "2021-06-21T09:13:54+0900",

"nasVolumeDescription": "",

"mountInformation": "169.254.82.17:/n871882_data",

"volumeAllotmentProtocolType": {

    "code": "NFS",

    "codeName": "NFS"

},

"volumeName": "n871882_data",

"volumeTotalSize": 536870912000,

"volumeSize": 536870912000,

"volumeUseSize": 344064,

"volumeUseRatio": 0.0,

"snapshotVolumeConfigurationRatio": 0.0,

"snapshotVolumeConfigTime": 0,

"snapshotVolumeSize": 0,

"snapshotVolumeUseSize": 0,

"snapshotVolumeUseRatio": 0.0,

"isSnapshotConfiguration": false,

"isEventConfiguration": false,

"regionCode": "KR",

"zoneCode": "KR-1",

"nasVolumeServerInstanceNoList": [],

"isEncryptedVolume": false,
```

```
        "nasVolumeInstanceCustomIpList": [],  
  
        "isReturnProtection": false  
  
    }  
  
]  
  
}}  
  
[root@target-linux cli_linux]#
```

## 볼륨 조회

```
[root@target-linux cli_linux]# ./ncloud vnas getNasVolumeInstanceList --regionCo  
de KR --zoneCode KR-1 --volumeAllotmentProtocolTypeCode NFS --  
isEventConfiguration false --isSnapshotConfiguration false  
  
{"getNasVolumeInstanceListResponse": {  
  
    "requestId": "aa985a35-9dae-41bf-ab75-d024b17d80ae",  
  
    "returnCode": "0",  
  
    "returnMessage": "success",  
  
    "totalRows": 1,  
  
    "nasVolumeInstanceList": [  
  
        {  
  
            "nasVolumeInstanceNo": "7031541",  
  
            "nasVolumeInstanceStatus": {  
  
                "code": "CREAT",  
  
                "codeName": "NAS create"  
  
            },  
  
        },  
  
    ]  
}
```

```
"nasVolumeInstanceOperation": {  
  
    "code": "NULL",  
  
    "codeName": "NAS NULL OP"  
  
},  
  
"nasVolumeInstanceStatusName": "",  
  
"createDate": "2021-06-21T09:13:54+0900",  
  
"nasVolumeDescription": "",  
  
"mountInformation": "169.254.82.17:/n871882_data",  
  
"volumeAllotmentProtocolType": {  
  
    "code": "NFS",  
  
    "codeName": "NFS"  
  
},  
  
"volumeName": "n871882_data",  
  
"volumeTotalSize": 536870912000,  
  
"volumeSize": 536870912000,  
  
"volumeUseSize": 344064,  
  
"volumeUseRatio": 0.0,  
  
"snapshotVolumeConfigurationRatio": 0.0,  
  
"snapshotVolumeConfigTime": 0,  
  
"snapshotVolumeSize": 0,  
  
"snapshotVolumeUseSize": 0,  
  
"snapshotVolumeUseRatio": 0.0,  
  
"isSnapshotConfiguration": false,
```

```
        "isEventConfiguration": false,  
  
        "regionCode": "KR",  
  
        "zoneCode": "KR-1",  
  
        "nasVolumeServerInstanceNoList": [],  
  
        "isEncryptedVolume": false,  
  
        "nasVolumeInstanceCustomIpList": [],  
  
        "isReturnProtection": false  
    },  
],  
}  
  
[root@target-linux cli_linux]#
```

## NAS ACL 생성

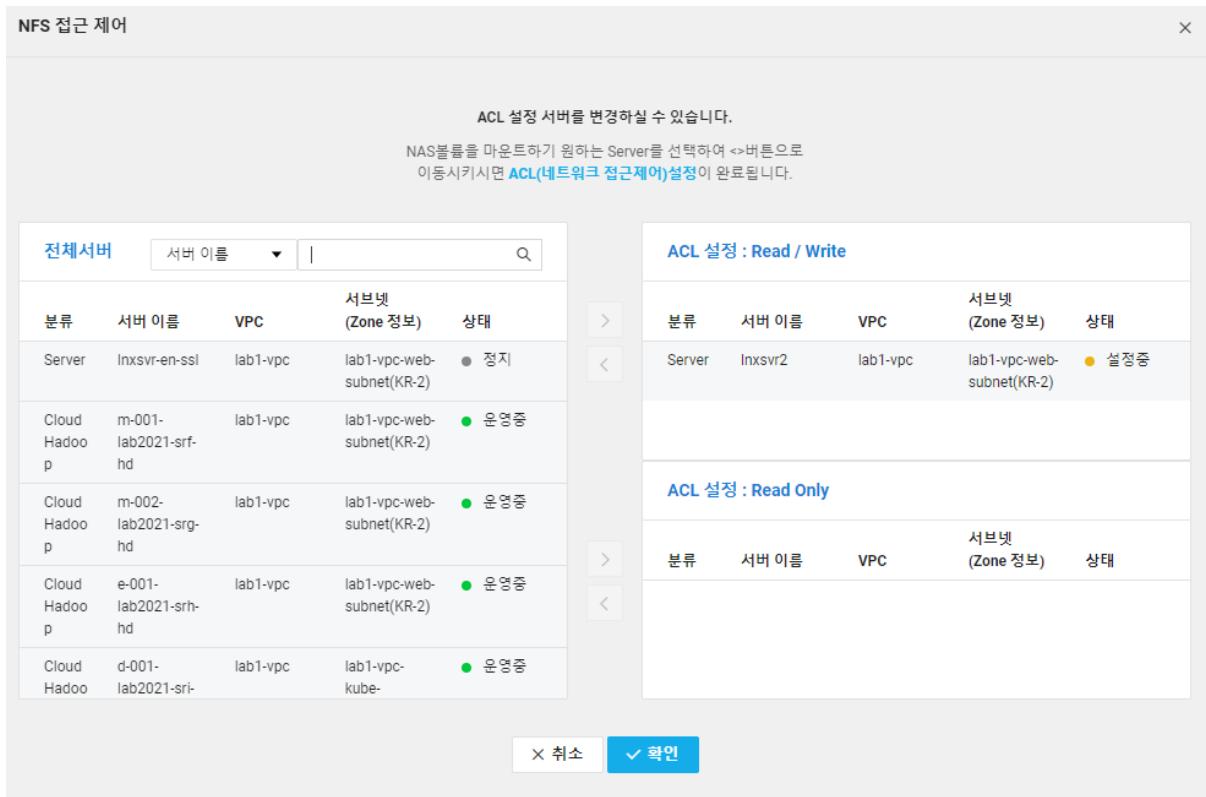
```
[root@target-linux cli_linux]# ./ncloud vnas addNasVolumeAccessControl --regionCode KR --nasVolumeInstanceNo 7031541 --serverInstanceNoList 6992741  
  
{"addNasVolumeAccessControlResponse": {  
  
    "requestId": "b4cae81b-dc70-4b7d-acd1-d0549f96ef90",  
  
    "returnCode": "0",  
  
    "returnMessage": "success",  
  
    "totalRows": 1,  
  
    "nasVolumeInstanceList": [  
  
        {  
  
            "nasVolumeInstanceNo": "7031541",  
        }  
    ]  
}
```

```
"nasVolumeInstanceStatus": {  
  
    "code": "CREAT",  
  
    "codeName": "NAS create"  
  
},  
  
"nasVolumeInstanceOperation": {  
  
    "code": "NULL",  
  
    "codeName": "NAS NULL OP"  
  
},  
  
"nasVolumeInstanceStatusName": "",  
  
"createDate": "2021-06-21T09:13:54+0900",  
  
"nasVolumeDescription": "",  
  
"mountInformation": "169.254.82.17:/n871882_data",  
  
"volumeAllotmentProtocolType": {  
  
    "code": "NFS",  
  
    "codeName": "NFS"  
  
},  
  
"volumeName": "n871882_data",  
  
"volumeTotalSize": 536870912000,  
  
"volumeSize": 536870912000,  
  
"volumeUseSize": 344064,  
  
"volumeUseRatio": 0.0,  
  
"snapshotVolumeConfigurationRatio": 0.0,  
  
"snapshotVolumeConfigTime": 0,
```

```
        "snapshotVolumeSize": 0,  
  
        "snapshotVolumeUseSize": 0,  
  
        "snapshotVolumeUseRatio": 0.0,  
  
        "isSnapshotConfiguration": false,  
  
        "isEventConfiguration": false,  
  
        "regionCode": "KR",  
  
        "zoneCode": "KR-1",  
  
        "nasVolumeServerInstanceNoList": [  
  
            "6992741"  
  
        ],  
  
        "isEncryptedVolume": false,  
  
        "nasVolumeInstanceCustomIpList": [],  
  
        "isReturnProtection": false  
    }  
  
]
```

}}

[root@target-linux cli\_linux]#



## 2. Object Storage 사용을 위한 fuse 구성

All Product > Storage > Object Storage 선택 > +버킷 생성 선택

버킷 이름에 ‘네이버클라우드플랫폼 아이디-fuse’ 버킷 생성

Inxsvr1 에 접속하여 패키지 설치

```
[root@target-linux cli_linux]# yum install automake fuse-devel gcc-c++ git libcurl-devel
libxml2-devel make openssl-devel -y
[root@target-linux cli_linux]# git clone https://github.com/s3fs-fuse/s3fs-fuse.git
[root@target-linux cli_linux]# cd s3fs-fuse
[root@target-linux cli_linux]# ./autogen.sh
[root@target-linux cli_linux]# ./configure
[root@target-linux cli_linux]# make
[root@target-linux cli_linux]# make install
```

마운트 포인트 생성

```
[root@target-linux cli_linux]# mkdir -p /objectstorage/아이디
```

환경 설정

```
echo ACCESS_KEY_ID:SECRET_ACCESS_KEY > /etc/passwd-s3fs
```

```
chmod 600 /etc/passwd-s3fs
```

마운트

```
s3fs          오브젝트스토리지버킷명      /objectstorage/아이디      -o  
url=https://kr.object.ncloudstorage.com
```

## Lab 5 Guide

**로드밸런서 CLI로 생성**

1. 로드밸런서 생성 전, 로드밸런서를 배치시킬 Private Subnet을 생성합니다. Prviate Sunbet 생성 후, Private subnet 번호를 확인합니다.

```
./ncloud vpc getSubnetList
```

2. Target Group은 콘솔에서 아래와 같이 생성합니다.

- Target 이름 : lab-tg
- Target 유형 : VPC Server
- VPC : lab1-vpc
- 프로토콜 : HTTP
- 포트 : 80
- 헬스 체크 설정
  - 프로토콜 : HTTP
  - 포트 : 80
  - HTTP Method : HEAD
  - 헬스체크 주기 : 30
  - 정상 임계값 : 2
  - 실패 임계값 : 2
- 적용 서버 : lnxsvr1

- 타겟 그룹 정보 조회

```
./ncloud vloadbalancer getTargetGroupList
```

- Lab1에서 확인한 VPC 번호와 서브넷 번호, 타겟 그룹 번호를 이용하여 다음과 같이 생성합니다.

```
./ncloud vloadbalancer createLoadBalancerInstance --regionCode KR --  
loadBalancerTypeCode APPLICATION --loadBalancerName lab-lb-cli --  
loadBalancerNetworkTypeCode PRIVATE --throughputTypeCode SMALL --idleTimeout  
60 --vpcNo [VPC 번호] --subnetNoList [서브넷 번호] --loadBalancerListenerList
```

```
"protocolTypeCode='HTTP', port='80', targetGroupNo='[타겟그룹번호]"
```

### 라이브 스트리밍 영상 저장용 Object Storage 생성

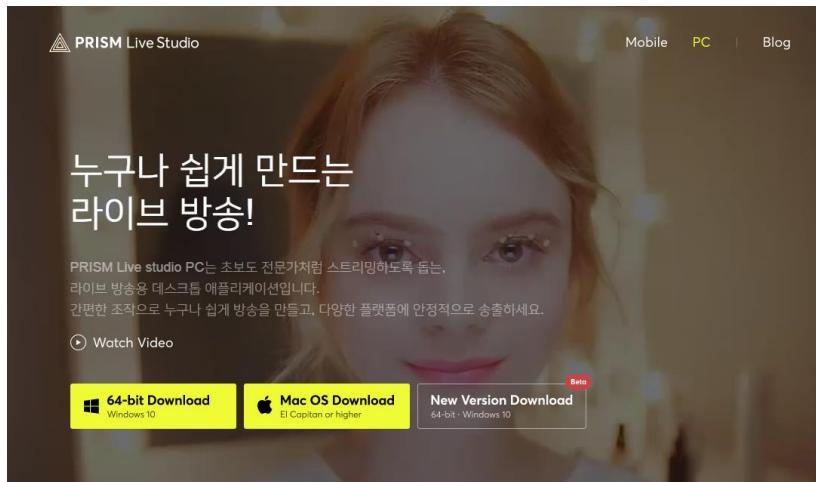
- Storage > Object Storage 선택 후 +버킷 생성 선택
- 버킷 이름 : nce-live-(계정명)
- 전체공개 : 공개

### 라이브 스트리밍 구성

- Media > Live Station 선택 후 +채널 생성 선택
- 채널 명 : ncels
- Output Protocol 설정 : HLS
- CDN 생성 : 신규 생성, CDN+
- 채널 타입 : REAL
- System 화질 설정
  - 480p-set
- 타임머신 설정 : 설정
- 녹화 저장 설정 : 자동저장
  - 저장 버킷은 nce-live-(계정명) 선택
- 자동 녹화 설정 : 자동
- 자동 저장 녹화 타입 : MP4
- 하단의 ‘다음’ 버튼 클릭
- 하단의 ‘채널 생성’ 클릭

### NAVER Prism Studio 설치

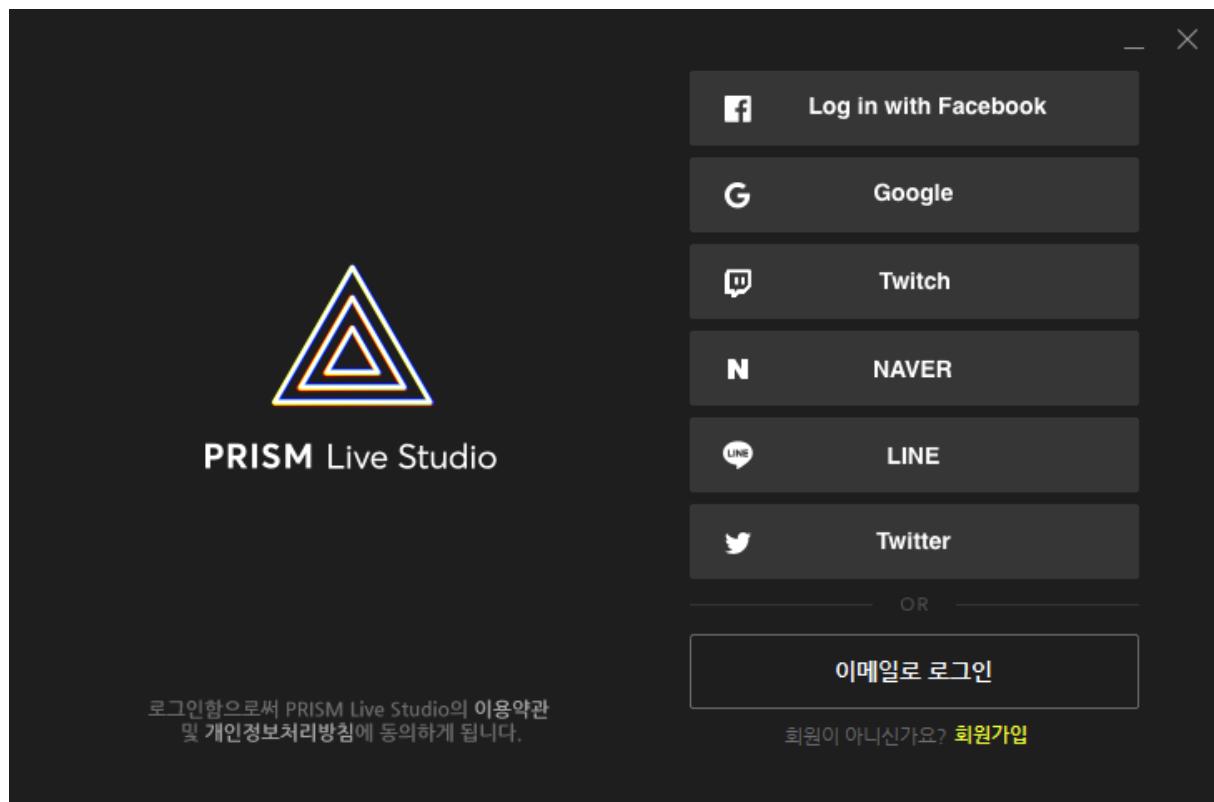
- [http://prismlive.com/ko\\_kr/pcapp/](http://prismlive.com/ko_kr/pcapp/)에서 설치 파일 설치



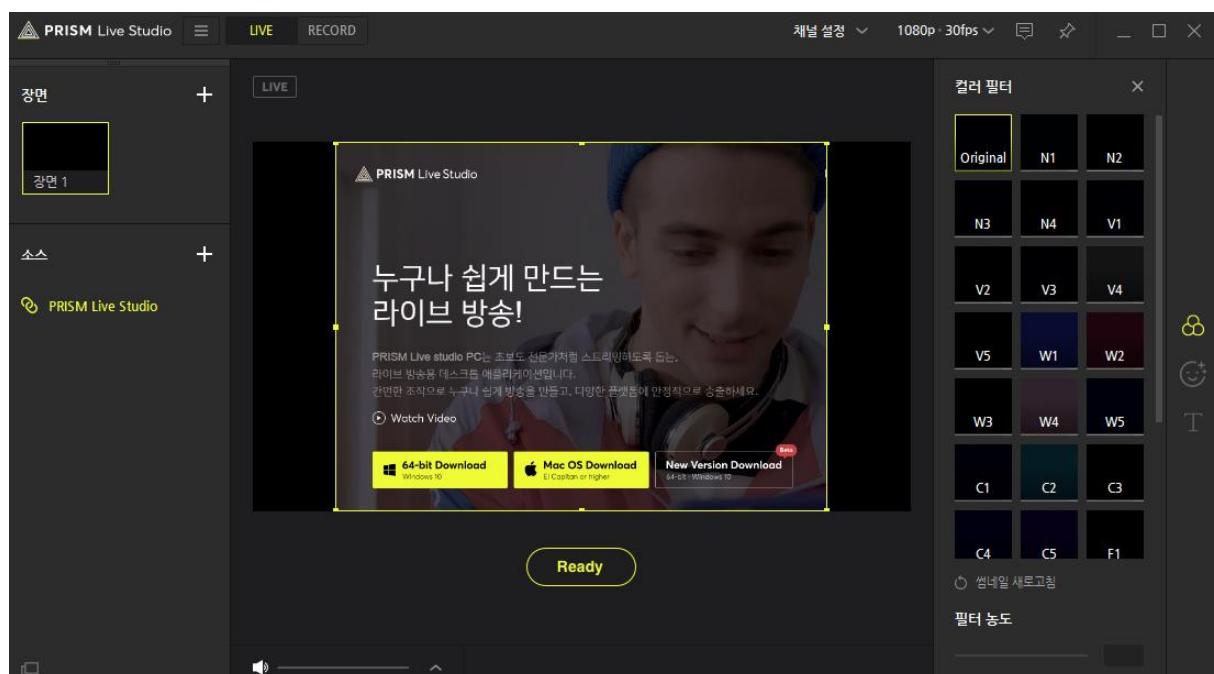
- Media > Live Station > Channel Management 선택, ncels 선택
- 상세정보에서 스트림 정보 선택하여 스트림 정보 확인

A screenshot of the Ncloud Media Service Channel Management interface. On the left, there's a list of channels. In the center, a modal window titled '스트림 정보' (Stream Information) is open, showing the URL 'rtmp://rtmp-ls2-k1.video.media.ntruss.com:8080/re...' and Stream Key 'lgz7gbjhcvgtwf7u8s3dl3'. There are '복사' (Copy) and '확인' (Confirm) buttons at the bottom. The background shows other channel details like '채널 ID: ls-20200810122105-GKQvd', '스트림 정보: lgz7gb...', and '화질 설정: 480p-s'.

- 네이버 프리즘 라이브 스튜디오 설정
  - 알맞은 인증 방식 선택 후 로그인

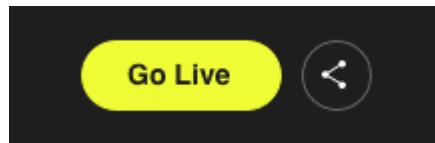


■ 나의 채널 > 채널 추가



- RTMP 수동 입력 선택
- URL 직접 입력 선택, URL 및 스트리밍키 입력

- 채널명 : NCELS
- 라이브 스테이션의 스트림 정보 입력
- Reday > Go Live 클릭



### 라이브 스테이션 송출 확인

- Media > Live Station > Channel Management 선택, ncls 선택
- Service URL에서 ABR URL 복사

화질명	Play URL(Full URL)
480p-16-9	<a href="https://romoffyxbfmt4920656.cdn.ntruss.com/live/video/ls-20200810122105-GKQvd/480p-16-9/playlist.m3u8">https://romoffyxbfmt4920656.cdn.ntruss.com/live/video/ls-20200810122105-GKQvd/480p-16-9/playlist.m3u8</a>
360p-16-9	<a href="https://romoffyxbfmt4920656.cdn.ntruss.com/live/video/ls-20200810122105-GKQvd/360p-16-9/playlist.m3u8">https://romoffyxbfmt4920656.cdn.ntruss.com/live/video/ls-20200810122105-GKQvd/360p-16-9/playlist.m3u8</a>
audio-192k	<a href="https://romoffyxbfmt4920656.cdn.ntruss.com/live/video/ls-20200810122105-GKQvd/audio-192k/playlist.m3u8">https://romoffyxbfmt4920656.cdn.ntruss.com/live/video/ls-20200810122105-GKQvd/audio-192k/playlist.m3u8</a>
ABR	<a href="https://romoffyxbfmt4920656.cdn.ntruss.com/live/video/ls-20200810122105-GKQvd/playlist.m3u8">https://romoffyxbfmt4920656.cdn.ntruss.com/live/video/ls-20200810122105-GKQvd/playlist.m3u8</a>

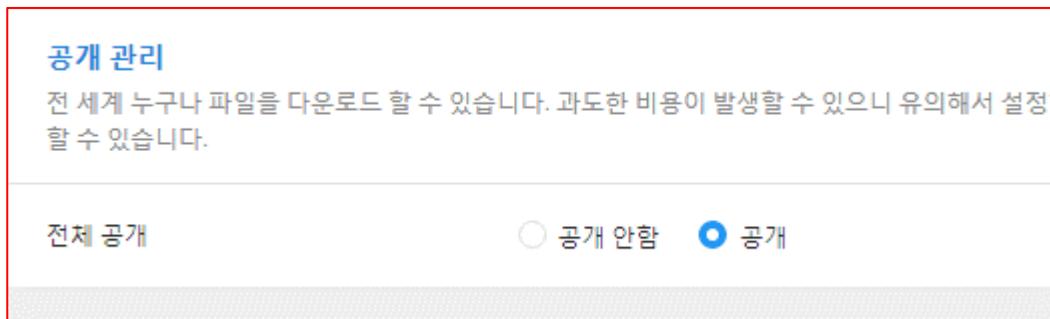
  

화질명	Play URL(Full URL)
480p-16-9	<a href="https://romoffyxbfmt4920656.cdn.ntruss.com/dvr/video/ls-20200810122105-GKQvd/480p-16-9/playlist.m3u8">https://romoffyxbfmt4920656.cdn.ntruss.com/dvr/video/ls-20200810122105-GKQvd/480p-16-9/playlist.m3u8</a>
360p-16-9	<a href="https://romoffyxbfmt4920656.cdn.ntruss.com/dvr/video/ls-20200810122105-GKQvd/360p-16-9/playlist.m3u8">https://romoffyxbfmt4920656.cdn.ntruss.com/dvr/video/ls-20200810122105-GKQvd/360p-16-9/playlist.m3u8</a>
audio-192k	<a href="https://romoffyxbfmt4920656.cdn.ntruss.com/dvr/video/ls-20200810122105-GKQvd/audio-192k/playlist.m3u8">https://romoffyxbfmt4920656.cdn.ntruss.com/dvr/video/ls-20200810122105-GKQvd/audio-192k/playlist.m3u8</a>
ABR	<a href="https://romoffyxbfmt4920656.cdn.ntruss.com/dvr/video/ls-20200810122105-GKQvd/playlist.m3u8">https://romoffyxbfmt4920656.cdn.ntruss.com/dvr/video/ls-20200810122105-GKQvd/playlist.m3u8</a>

- 브라우저에 붙여넣기
  - 단, 크롬의 경우 확장팩 설치 필요
  - <https://chrome.google.com/webstore/detail/native-mpeg-dash-%20-hls-pl/cjfbmleiaobegagekpmlhmaadepdeedn>
- 송출 된 걸 확인 후, 프리즘 라이브 스튜디오에서 라이브종료

### DVR 저장

- Storage > Object Storage > nce-live-[계정명]에서 녹화본 확인
- 하위 디렉토리의 mp4 확장자를 가진 영상 파일에 대해 권한 관리에서 공개 관리를 공개로 변경



### VOD 스테이션을 이용하여 라이브 스테이션 영상 송출

- Media > VOD Station 선택
- '+카테고리 생성' 클릭
  - Category 명 : ncevs
  - 인코딩 설정 : 나중에 설정
  - 썸네일 설정 : 설정 안함
  - 아웃풋 파일 경로 : 오브젝트 스토리지 > nce-live-[계정명] 선택
  - 고급 설정 : 나중에 설정
- 하단의 '생성' 버튼 클릭
- 카테고리 생성 후 'channel' 메뉴로 이동
  - 채널 이름 : ncevs
  - 버킷 : nce-live-[계정명]
  - Object storage 비공개 파일 접근 : 제한
  - Protocol : HLS
  - Segment Duration : 5초
  - Segment duration option : basic
  - CDN 생성 : 신규 생성
  - 하단의 '채널 생성' 클릭

**채널 생성**

VOD Streaming을 구성하기 위해 새로운 채널을 생성합니다. (필수 입력 사항입니다.)

**채널 이름\*** ncevs  
파일경로에 지정한 Object storage bucket이 가지고 있는 영의 풀더가 자동으로 생성됩니다.

**Object Storage Bucket\*** nclive-edu50  
Object Storage 버킷에 파일 접근 \*

**Protocol\*** 제한 (선택)  
 HLS  DASH

**Segment Duration\*** 5초

**CDN 설정\***  
 신규 생성  생성 안함

**CDN 선택\***  
 CDN+  
VOD Station은 CDN 연동이 필수인 상품입니다.  
신규 생성을 통해 VOD Station에 최적화된 CDN을 생성하거나,  
기존 CDN 연동 또는 별도 CDN을 생성해 VOD Station을 이용하실 수 있습니다.

**콘텐츠 보호 설정**  
콘텐츠 보호는 VOD Station을 위한 콘텐츠 보호(암호화) 설정 단계입니다. 필수 사항은 아니며, 콘텐츠 보호 기능이 필요 없다면 생략할 수 있습니다.  
 일부 기능은 사용에 DRM 라이선스 구매가 필요하며, 보다 자세한 사항은 '콘텐츠 보호 설정 가이드'를 참고하세요. (필수 입력 사항입니다.)

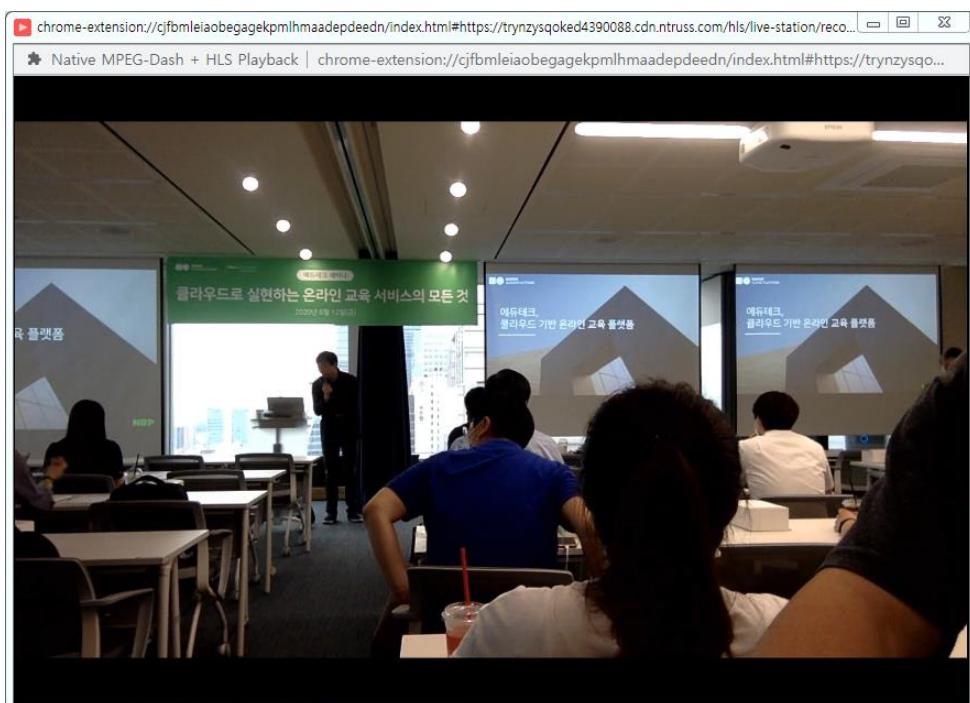
DRM 유형	암호화 유형	Protocol
Media Encryption		
DRM		

**취소** **채널 생성**

(object storage 내 버킷에 있는 영상을 ‘전체 공개’로 수정해야 함)

### VOD 재생

- Media > VOD Station > 채널 선택
- Ncevs 채널 우측 ‘HLS URL생성’ 클릭



### Image Optimizer 이미지 저장용 Object Storage 생성

- Storage > Object Storage 선택 후 +버킷 생성 선택

- 버킷 이름 : nce-image-[계정명]
- 전체공개 : 공개
- 버킷이 생성되면 보유하고 있는 이미지를 업로드

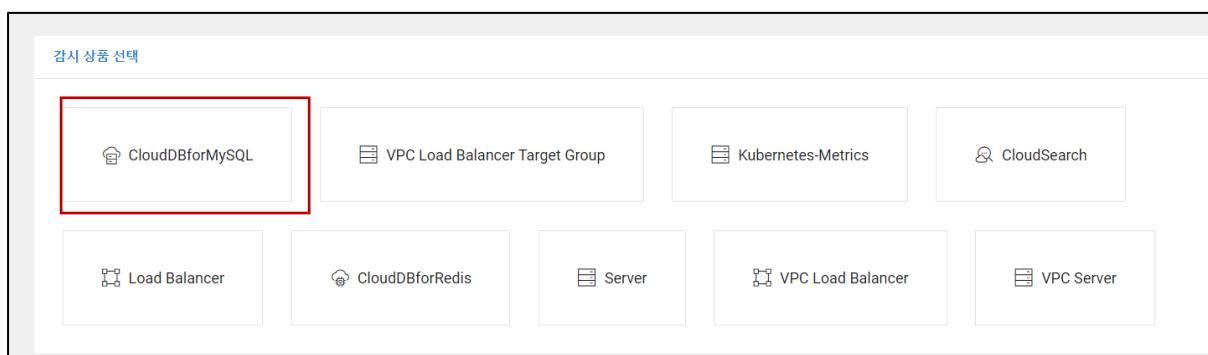
#### Image Optimizer로 변환 룰 적용

- Media > Image Optimizer > +Project 생성 선택
- 프로젝트 이름 : nce-io
- 원본 이미지 저장소 : nce-image-[계정명] 버킷 선택
- 서비스 프로토콜 : HTTP
- CDN 도메인 사용
- 캐시 만료 : 1시간
- Custom header : 사용안함
- CLA 서비스 연동 신청(필수)
  
- Media > Image Optimizer > Project Management > nce-io 선택
- 변환 Rule 설정에서 쉬운 입력 선택
  - 리사이즈 & 크롭 선택
  - 가로길이 800
  - 세로길이 400
  - 크롭위치 4
  - 얼굴인식 예
  - 품질 90
  - 자동회전 예
  - 입력 선택
  - + 저장 선택
  - 미리보기에서 이미지 확인
- 변환 Rule 설정에서 쉬운 입력 선택

- 가로 기준 변경 선택
- 가로길이 800
- 품질 90
- 자동회전 예
- 필터에서 블러 선택
- Blur\_rad에 10, Blur\_sig에 10 입력
- 입력 선택
- + 저장 선택
- 미리보기에서 이미지 확인
- 변환 Rule 설정에서 쉬운 입력 선택
  - 강제 변경 선택
  - 가로길이 800
  - 세로길이 400
  - 자동회전 예
  - 필터에서 흑백 선택
  - 입력 선택
  - + 저장 선택
  - 미리보기에서 이미지 확인
- 미리보기에서 이미지 선택 후 이미지 변환 링크 복사
- 복사된 링크를 웹 브라우저에 입력하여 확인

**Lab 6 Guide****Cloud DB for MySQL 이벤트 설정하기**

- 1) Cloud DB for MySQL 메뉴에 Event 메뉴를 클릭.
- 2) Event Rule(Cloud Insight) 설정 클릭
- 3) Configuration > Event Rule 생성 클릭
- 4) 감시 상품 선택 > Cloud DB For MySQL 선택



- 5) 감시 대상 설정 > 그룹 생성 클릭

그룹 이름 : edu-cdb

Edu CDB 서버 선택 후, 하단의 생성 클릭

- 6) Target group에서 edu-cdb 선택 후 하단의 '다음'버튼 클릭
- 7) 감시 항목 설정 > + 템플릿 생성 클릭
  - A. 템플릿 이름 : edu-cdb-template 입력
  - B. Metric : cpu\_iowait / cpu\_load1 / disk\_mysql\_used / mysql\_slavedelay / mysql\_slow 선택 후, 하단의 다음버튼 클릭
  - C. 아래와 같이 임계치 설정 후, 하단의 설정 버튼 클릭

Cpu\_load\_1 / level : critical / condition = 30 / method = avg / duration : 1

mysql-slow / level : critical / condition = 1 / method = avg / duration : 1

템플릿 생성

① 메트릭 항목 선택

( \* 필수 입력 사항입니다.)

감시 대상 분류 : CloudDBforMySQL

템플릿 이름 : edu-cdb-template

설명 :

0/300 bytes

Metric	Description	Dimension ⓘ	Level	Condition	Method	Duration	+	X
cpu_iowait	CPU iowait(%)		INFO	= ▼ 0	Avg	1 min		
disk_mysql_...	mysql disk us...		INFO	= ▼ 0	Avg	1 min		
mysql_slave_...	mysq replicati...		INFO	= ▼ 0	Avg	1 min		
cpu_load_1	Load avg 1min		CRITICAL	= ▼ 30	Avg	1 min		
mysql_slow	mysql slow q...		CRITICAL	= ▼ 1	Avg	1 min		

- 8) Edu\_cdb\_template 선택 후, 다음 클릭
- 9) 상단의 ‘통보 대상자 그룹 생성’ 버튼 클릭
  - A. 전체 대상자 옆 ‘+’ 클릭하여 통보 대상자 그룹 생성
  - B. 대상자 그룹 명 : edu\_cdb\_관리자 입력

VPC / Cloud Insight(Monitoring) / Notification Recipient

## Notification Recipient

통보대상자 정보를 등록해 두시면, 모니터링 등 이벤트 발생 시 통보 대상으로 설정하실 수 있습니다.

+ 대상자 추가 ▾

대상자 그룹	전체 대상자 6									
전체 대상자 +	<input type="button" value="삭제"/> <input type="button" value="이동/복사"/> <input type="button" value="할당"/> <input type="button" value="수정"/> <table border="1"> <thead> <tr> <th>대상자 이름</th> <th>설명</th> <th>휴대폰</th> </tr> </thead> <tbody> <tr> <td>강지나</td> <td></td> <td></td> </tr> <tr> <td>강지나</td> <td></td> <td></td> </tr> </tbody> </table>	대상자 이름	설명	휴대폰	강지나			강지나		
대상자 이름	설명	휴대폰								
강지나										
강지나										

- 10) 대상자 그룹에 통보 알림을 받을 대상자 추가를 위해 상단의 대상자 추가 클릭 → 대상자 이름과 알람을 통보 받을 이메일 주소 입력 후 하단의 등록 버튼 클릭

Recipient Add

(필수 입력 사항입니다.)

Recipient Name\*: 강지나  
Description: 최대 30자

**V [개인정보 수집, 이용에 대한 안내]**  
네이버 클라우드 플랫폼에서는 개인정보 수집, 이용 등 처리에 있어 아래의 사항을 정보주체에게 안내합니다.

1. 수집/이용 목적: 주요 이벤트(장애 등) 발생 시 통보  
2. 항목: (필수항목)성명, 메일 주소, (선택항목)휴대폰번호  
3. 보유/이용기간: 일정 대상자 삭제시 또는 필요시까지

위 개인정보 수집 및 이용에 대한 안내사항에 동의 합니다.

Email Address\*:  @ naver.com naver.com  
Email address input field.  
Phone Number: 한국 (+82)  인증번호 전송  
Phone number input field.  
Authentic Number Input: 인증번호 입력  
Authentic number input field.  
Group Selection: 전체 대상자  
Group selection dropdown.

x 취소 ✓ 등록

11) 등록된 대상자를 edu\_cdb 관리자 그룹에 추가

A. 등록한 대상자를 선택 후, 상단의 할당 클릭

+ 대상자 추가 ▾

대상자 그룹		전체 대상자 7								
전체 대상자 7	+	<input type="button" value="삭제"/> <input type="button" value="이동/복사"/> <input style="border: 2px solid red;" type="button" value="할당"/> <input type="button" value="수정"/>								
edu_cdb 관... 0		<table border="1"> <thead> <tr> <th>대상자 이름</th> <th>설명</th> <th>휴대폰 번호</th> </tr> </thead> <tbody> <tr> <td><input checked="" type="checkbox"/> 강지나</td> <td></td> <td></td> </tr> </tbody> </table>			대상자 이름	설명	휴대폰 번호	<input checked="" type="checkbox"/> 강지나		
대상자 이름	설명	휴대폰 번호								
<input checked="" type="checkbox"/> 강지나										

B. 대상자 그룹 리스트 중, edu\_cdb 관리자 선택 후 하단의 할당 버튼 클릭

12) 다시 cloud insight 설정하는 화면으로 이동하여 통보 대상자 그룹에서 edu\_cdb 관리자 선택 후, 다음 버튼 클릭

13) 규칙 이름 : edu\_cdb\_rule 입력 후 하단의 생성 버튼 클릭

### Cloud DB for MySQL 부하 테스트(데모)

1) CDB에 접속할 서버에 Sysbench 설치

```
# yum install curl

# curl https://packagecloud.io/install/repositories/akopytov/sysbench/script.rpm.sh|sudo bash

# yum -y install sysbench

# sysbench --version
```

## 2) Sysbench로 부하 테스트

- 부하테스트에서 사용할 labdb 생성
- Services > Cloud DB for MySQL에서 edu데이터베이스 선택 후 상단의 DB관리 > DB Server상세 > Database 관리
- labdb 추가
- Inxsvr1서버에서 아래 명령어를 통한 부하 발생
  - [root@benchmark ~]# sysbench /usr/share/sysbench/oltp\_read\_write.lua --mysql-host=*private\_domain* --mysql-port=3306 --mysql-user=student --mysql-password='*password*' --mysql-db=labdb --db-driver=mysql --tables=3 --table-size=10000 prepare

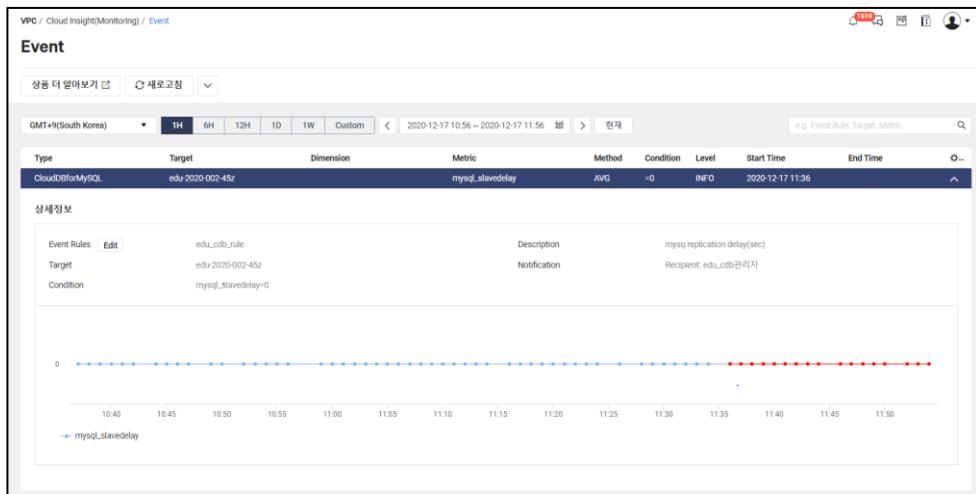
## 3) 모니터링 알람 확인

Cloud Insight 이벤트 알림

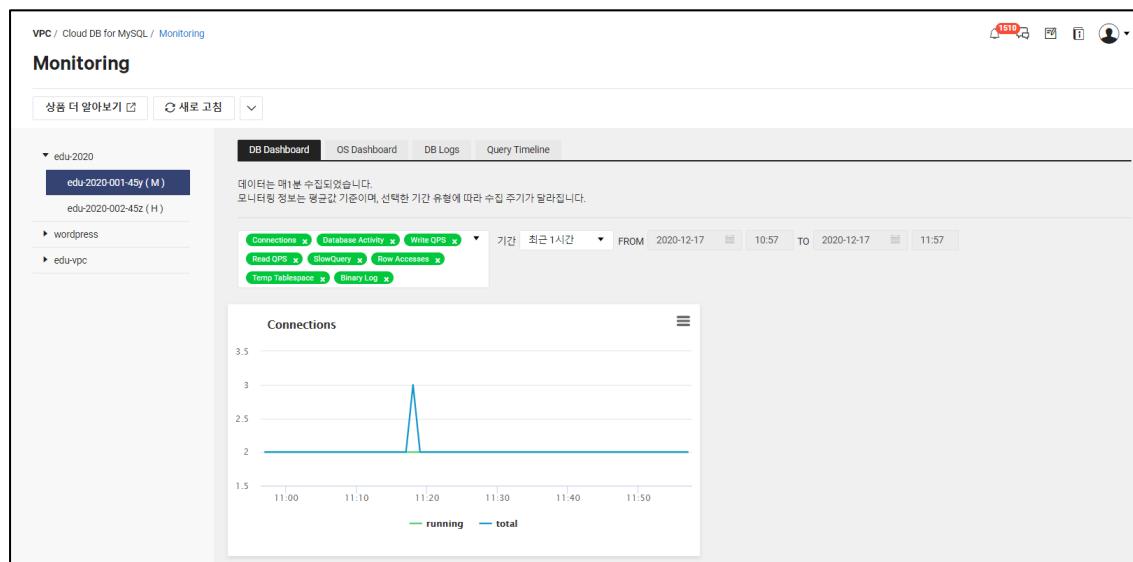
안녕하세요. 강지나 고객님  
고객님이 이용중이신 Cloud Insight 상품에서 이벤트가 발생했습니다.  
자세한 내용은 네이버클라우드플랫폼 Cloud Insight에 접속하여서 확인하시기 바랍니다.

시작시간	2020-12-17 11:38:05
레벨	INFO
규칙 이름	edu_cdb_rule
설명	mysq replication delay(sec)
서비스	CloudDBforMySQL
인스턴스 이름	edu-2020-002-45z

## 4) Cloud Insight > Event에서 모니터링 지표 확인



### 5) CDB For MySQL에 내장되어 있는 모니터링 기능을 통해서도 모니터링 지표 확인 가능



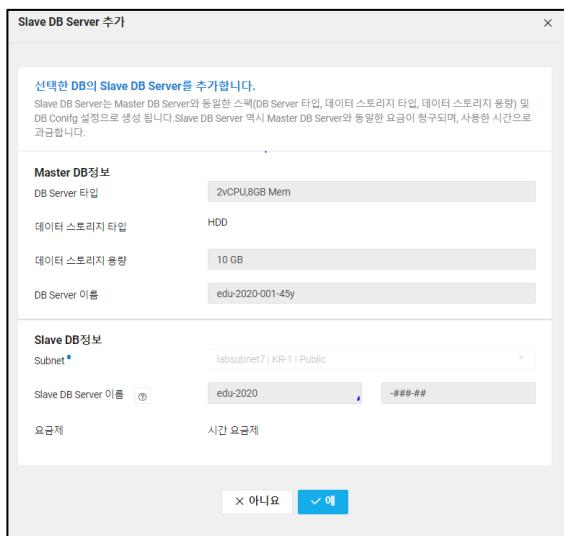
#### 테스트에 사용한 테이블 삭제

```
■ sysbench /usr/share/sysbench/oltp_read_write.lua --mysql-host=db-ann9o.vpc-cdb.ntruss.com --mysql-port=3306 --mysql-user=student --mysql-password=hoony@0406 --mysql-db=labdb --db-driver=mysql --tables=3 --table-size=10000 cleanup
```

#### Lab 10. DB for MySQL Slave DB를 추가하여 읽기 부하 분산

##### 1) Slave DB 추가

- 콘솔 > Cloud DB for MySQL 이동
- 금일 생성한 Cloud DB for MySQL 선택
- 상단의 DB관리 > Slave 추가 클릭



- 하단의 예 버튼 클릭
- 동일한 작업을 한번 더 진행하여 Slave DB 1대 더 추가

## 2) Private LB 생성을 통한 읽기 부하분산 환경 구축

- 콘솔 > 로드밸런서 > 로드밸런서 생성 클릭
- 네트워크 프록시 로드밸런서 선택

로드밸런서 이름 : edu-cdb-lb

Network : private ip

부하성능 처리 : small

대상 VPC : lab-VPC

서브넷 선택 : KR-2, 실습 때 생성한 lb 서브넷 선택 후 하단의 다음버튼 클릭

VPC / Load Balancer / Load Balancer

## 로드 밸런서 생성

1 로드밸런서 생성      2 로드밸런서 생성      3 Target Group 선택

**로드밸런서 생성**

생성할 로드밸런서 이름을 입력하고 로드밸런서가 활성화될 VPC 및 서브넷을 선택해주세요.  
로드밸런서를 생성하시면 로드밸런서 이용 시간과 트래픽 사용량에 따라 요금이 부과됩니다. (필수 입력 사항입니다.)

유형	네트워크 프록시
로드밸런서 이름*	edu-cdb-lb
Network*	<input checked="" type="radio"/> Private IP <input type="radio"/> Public IP
부하 처리 성능*	<input checked="" type="radio"/> Small <input type="radio"/> Medium <input type="radio"/> Large
대상 VPC*	lab-vpc (10.0.0.0/16)
서브넷 선택*	<input checked="" type="checkbox"/> KR-2    lb-subnet   10.0.8.0/24 <input type="checkbox"/> KR-1    -선택하세요-

각 서비스 Zone에 로드밸런서를 배치할 전용 서브넷을 생성하셔야 합니다. 전용 서브넷에 서버 인스턴스를 위치시키면  
로드밸런싱이 동작하지 않습니다. 각각의 로드밸런서마다 서브넷을 생성할 필요는 없으나 가급적 C클래스(255.255.255.0) 이상을 권고합니다.

**다음 >**

### ■ 리스너 설정

프로토콜 : TCP / 로드밸런서 포트 : 3306 추가 후, 하단의 다음 버튼 클릭

### ■ Target Group 생성

Target Group 이름 : edu-cdb-group

Target 유형 : VPC-Server

VPC : 실습 시 생성한 VPC

프로토콜 : PROXY\_TCP

포트 : 3306

VPC / Load Balancer / Target Group

## Target Group 생성

1 Target Group 생성    2 Health Check 설정    3 Target 추가    4 설정 정

**Target Group 생성**

생성할 Target Group의 이름을 입력하고 Target 유형과 포함될 Target의 VPC를 선택해주세요  
프로토콜에 따라 연결 가능한 로드밸런서 유형이 다릅니다. (• 필수 입력 사항입니다.)

Target Group 이름\* : edu-cdb-group

Target 유형\* : VPC Server

VPC\* : lab-vpc (10.0.0.0/16)

프로토콜\* : PROXY\_TCP

포트\* : 3306

메모

0 /1000 Bytes

다음 >

### ■ Health-check 설정

프로토콜 : TCP

포트 : 3306

Health check 주기 : 30

정상 임계치 : 2

실패 임계치 : 2

VPC / Load Balancer / Target Group

### Target Group 생성

1 Target Group 생성 2 Health Check 설정 3 Target 추가 4 설정 정보 보기

**Health Check 설정**

Target에 대한 Health Check를 위한 정보를 입력하세요.  
Health Check에 실패한 서버는 로드밸런싱 대상에서 제외됩니다.(•필수 입력 사항입니다.)

프로토콜*	TCP
포트*	3306
Health Check 주기 (초) *	30
정상 임계값*	2
실패 임계값*	2

< 이전 **다음 >**

■ Target 추가 메뉴에선느 서버 역할이 slave인걸 선택

VPC / Load Balancer / Target Group

### Target Group 생성

1 Target Group 생성 2 Health Check 설정 3 Target 추가 4 설정 정보 보기

**Target 추가**

Target Group에 포함시킬 대상을 적용 Target으로 이동해 주세요.

전체 Target					
선택	서버 이름	서브넷	서버타입	서버역할	상태
<input type="checkbox"/>	nks-pool-190-w-45m	labsubnet5 (10.0.5.0/24)	Kubernetes Server	Worker	<span style="color: green;">running</span>
<input type="checkbox"/>	nks-pool-190-w-40y	labsubnet5 (10.0.5.0/24)	Kubernetes Server	Worker	<span style="color: green;">running</span>
<input type="checkbox"/>	edu-nee	labsubnet7 (10.0.7.0/24)	-	-	<span style="color: green;">running</span>
<input type="checkbox"/>	s175d96aa74b	labsubnet6 (10.0.6.0/24)	-	-	<span style="color: green;">running</span>
<input type="checkbox"/>	lab-m-2bi	labsubnet7 (10.0.7.0/24)	Elasticsearch Server	Master	<span style="color: green;">running</span>
<input type="checkbox"/>	lab-m-2bh	labsubnet7 (10.0.7.0/24)	Elasticsearch Server	Master	<span style="color: green;">running</span>
<input type="checkbox"/>	lab1-org2	labsubnet1 (10.0.1.0/24)	-	-	<span style="color: green;">running</span>

작용 Target					
선택	서버 이름	서브넷	서버타입	서버역할	상태
<input type="checkbox"/>	edu-2020-003-46a	labsubnet7 (10.0.7.0/24)	MySQL Server	Slave	<span style="color: green;">running</span>
<input type="checkbox"/>	edu-2020-004-46b	labsubnet7 (10.0.7.0/24)	MySQL Server	Slave	<span style="color: green;">running</span>

< 이전 **다음 >**

LB 생성 메뉴로 다시 돌아서 edu-cdb-group선택 후 다음 버튼 클릭

**Target Group 선택**  
적용할 Target Group을 선택해주세요.  
로드밸런서 타입에 따라 선택할 수 있는 프로토콜이 제한됩니다.

**Target Group 설정**

Target Group 이름	edu-cdb-group	Target 유형	VPC Server
프로토콜	PROXY_TCP	포트	3306

**Health Check 설정**

프로토콜	TCP	포트	3306
URL Path		검사 주기	30

**적용 Target**

Target 태입	서버 이름	서브넷	서버역할	상태
edu-2020-004-46b	labsubnet7 (10.0.7.0/24)   KR-1	MySQL Server	Slave	● running
edu-2020-003-46a	labsubnet7 (10.0.7.0/24)   KR-1	MySQL Server	Slave	● running

마지막 확인 페이지 내용 살펴 본 후 하단의 로드밸런서 생성 클릭

## ■ 사설 로드밸런서 상태 확인

**Load Balancer**

로드밸런서 이름	상태	네트워크	VPC	유형	접속 정보	메모
<input checked="" type="checkbox"/> edu-cdb-lb	● 운영중	사설	lab-vpc	NETWORK_PROXY	edu-cdb-lb-5672802-18b6f5d14432.kr.lb.naverncp.com	

**상세정보**

로드밸런서 이름 (Instance ID)	edu-cdb-lb (5672802)	접속 정보	edu-cdb-lb-5672802-18b6f5d14432.kr.lb.naverncp.com
생성일시	2020-12-17 오후 1:36 (UTC+09:00)	부하 처리 성능	10.0.8.14
VPC	lab-vpc (10.0.0.0/16)		Small
로드밸런서 서브넷	lb-subnet (10.0.8.0/24)   KR-2		
네트워크	사설		
Idle Timeout	60		

## Cloud DB for MySQL 백업 파일로 복구

- 콘솔 > Cloud DB for MS-SQL > Back up 선택
- 백업 파일 중, 금일 생성한 Cloud DB > 상세 내역 클릭

Cloud DB for MySQL / Backup

### Backup

상품 더 알아보기  새로 고침

Cloud DB는 안정적인 운영을 위하여 최근 1회 백업을 항상 유지 합니다.  
최기 DB 백업은 로그 및 테이블스페이스를 포함하여 최소 500 MB 백업 스토리지를 사용합니다.

DB 서비스 이름	Backup 보관일	Backup 시작시간	Backup 데이터 크기	마지막 Backup 일시	상세정보 보기
ncpedu	1DAY	02:00	544.8 MB	2020-08-11 02:00:02 (UTC+09:00)	<input type="button"/> 상세내역
edu	1DAY	00:00	570.9 MB	2020-08-11 00:15:03 (UTC+09:00)	<input type="button"/> 상세내역
upgrade	1DAY	09:45	544.8 MB	2020-08-11 09:45:02 (UTC+09:00)	<input type="button"/> 상세내역
alone22	1DAY	06:30	544.8 MB	2020-08-11 06:30:06 (UTC+09:00)	<input type="button"/> 상세내역
alone	1DAY	12:30	544.8 MB	2020-08-11 12:30:08 (UTC+09:00)	<input type="button"/> 상세내역
edu2020	1DAY	00:00	566.8 MB	2020-08-11 00:15:03 (UTC+09:00)	<input type="button"/> 상세내역

#### ■ 백업 파일 복원 클릭

Cloud DB for MySQL / Backup

### ncpedu Backup 리스트

< Back

Backup 파일 복원  시점 복원  Object Storage로 보내기

Backup 파일로 백업 시간으로 데이터를 복원하거나 원하는 시간으로 시점 복원(최대 7일)이 가능합니다.  
Backup은 24시간 기준으로 사용자가 지정한 Backup 파일 보관일 만큼 보관되며 이후 백업데이터는 자동 삭제됩니다.

Backup 날짜	Backup 시작시간	Backup 완료시간
2020-08-11	02:00 (UTC+09:00)	02:00 (UTC+09:00)

#### ■ Backup 파일로 복원할 데이터베이스 이름 설정 및 복원 클릭

Backup 파일 복원

복원 요청시 신규 DB Server가 생성되며, 선택한 Backup파일로 복원됩니다.  
생성된 DB Server는 Recovery 모드로 복원됩니다.(데이터 조회만 가능)  
Backup 파일 복원은 Backup이 완료된 시간으로 복원 합니다.

Backup 시간	2020-08-11 02:00:02 (UTC+09:00)
Backup 완료 시간	2020-08-11 02:00:10 (UTC+09:00)
DB Server 타입	2vCPU, 4GB Mem, 10 GB스토리지
DB Server 이름	<input type="text" value="backup"/> -###

취소  복원하기

복원 파일로 데이터베이스 서버가 생성되는지 확인

**Lab 7****MSSQL 서버 생성**

Cloud DB for MSSQL 서버는 위 lab에서 생성한 lab1-vpc-public-subnet 에 생성합니다.

이번 lab에서는 콘솔이 아닌 CLI를 통해 DB를 생성하고, Slave 서버를 연결하는 작업을 진행합니다.

1. Lnxsvr1 서버 접속 후 아래 명령어를 통해 파라미터 값 확인

```
[root@lnxsvr1 cli_linux]# ./ncloud vmssql getCloudMssqlImageProductList --regionCode KR

{"getCloudMssqlImageProductListResponse": {

    "requestId": "005c4d10-bd51-4344-96b3-8695642b919a",

    "resultCode": "0",

    "returnMessage": "success",

    "totalRows": 1,

    "productList": [

        {

            "productCode": "SW.VDBAS.VMSSL.WND64.WINNT.2016.MSSQL.2019.B100",

            "productName": "MSSQL 2019 standard edition",

            "productType": {

                "code": "WINNT",

                "codeName": "Windows NT"

            },

            "productDescription": "Windows Server 2016 with MSSQL 2019 standard edition"
        }
    ]
}}
```

```

"infraResourceType": {

    "code": "SW",

    "codeName": "Software"

},

"baseBlockStorageSize": 107374182400,

"platformType": {

    "code": "WND64",

    "codeName": "Windows 64 Bit"

},

"osInformation": "Windows Server 2016 with MSSQL 2019 standard edition"

}

]

}}


[root@lnxsvr1 cli_linux]# ./ncloud vmssql getCloudMssqlProductList --regionCode KR --cloudMssqlImageProductCode
SW.VDBAS.VMSSL.WND64.WINNT.2016.MSSQL.2019.B100

{"getCloudMssqlProductListResponse": {

    "requestId": "4ec2df47-b50c-432d-a99d-525c545e7385",

    "returnCode": "0",

    "returnMessage": "success",

    "totalRows": 14,

    "productList": [

        {

            "productCode": "SVR.VMSSL.HICPU.C002.M004.NET.HDD.B100.G002",

            "productName": "vCPU 2EA, Memory 4GB",

```

```

"productType": {

    "code": "HICPU",
    "codeName": "High CPU"
},

"productDescription": "vCPU 2개, 메모리 4GB",
"infraResourceType": {

    "code": "VMSSL",
    "codeName": "Cloud DB For MSSQL (VPC)"
},
"cpuCount": 2,
"memorySize": 4294967296,
"diskType": {

    "code": "NET",
    "codeName": "Network Storage"
}
},
(이하 생략)

[root@lnxsvr1 cli_linux]#./ncloud vmssql createCloudMssqlInstance --regionCode KR --
vpcNo 9006 --subnetNo 18181 --cloudMssqlServiceName edu-mssql --
cloudMssqlImageProductCode
SW.VMSSL.OS.WND64.WINNT.SVR2016.MSSQL.15020005.SE.B100
--cloudMssqlProductCode SVR.VMSSL.HICPU.C002.M004.NET.HDD.B100.G002 --
dataStorageTypeCode SSD --isHa true --cloudMssqlUserName student --
cloudMssqlUserPassword abcde12# --cloudMssqlPort 1433

{"createCloudMssqlInstanceResponse": {

    "requestId": "769911bb-7c47-4ca0-9dda-f9bb7e235b12",
}

```

```
"returnCode": "0",  
"returnMessage": "success",  
"totalRows": 1,  
"cloudMssqlInstanceList": [  
{  
    "cloudMssqlInstanceNo": "7241902",  
    "cloudMssqlServiceName": "edu-mssql",  
    "cloudMssqlInstanceStatusName": "creating",  
    "cloudMssqlInstanceStatus": {  
        "code": "INIT",  
        "codeName": "CLOUD DATABASE(VPC) Init State"  
    },  
    "cloudMssqlInstanceOperation": {  
        "code": "CREAT",  
        "codeName": "CLOUD DATABASE(VPC) Creat OP"  
    },  
    "cloudMssqlImageProductCode":  
    "SW.VDBAS.VMSSL.WND64.WINNT.2016.MSSQL.2019.B100",  
    "isHa": true,  
    "license": {  
        "code": "SPLA",  
        "codeName": "Service Provider License Agreement"  
    },  
    "cloudMssqlPort": 1433,  
    "backupFileRetentionPeriod": 1,
```

```
"backupTime": "05:15",
"configGroupNo": "0",
"engineVersion": "MSSQL2019",
"createDate": "2021-07-08T16:15:04+0900",
"dbCollation": "Korean_Wansung_CI_AS",
"cloudMssqlServerInstanceList": [
{
  "cloudMssqlServerName": "m-7241902-001",
  "cloudMssqlServerRole": {
    "code": "M",
    "codeName": "Principal"
  },
  "cloudMssqlServerInstanceStatusName": "creating",
  "cloudMssqlServerInstanceState": {
    "code": "PEND",
    "codeName": "CLOUD DATABASE(VPC) Server Pending State"
  },
  "cloudMssqlServerInstanceOperation": {
    "code": "CREAT",
    "codeName": "CLOUD DATABASE(VPC) Server Create OP"
  },
  "regionCode": "KR",
  "zoneCode": "KR-2",
  "vpcNo": "3651",
```

```
"subnetNo": "6131",
  "dataStorageSize": 107374182400,
  "cpuCount": 2,
  "memorySize": 4294967296,
  "isPublicSubnet": true,
  "cloudMssqlProductCode":
  "SVR.VMSSL.HICPU.C002.M004.NET.HDD.B100.G002",
  "createDate": "2021-07-08T16:15:04+0900",
  "dataStorageType": {
    "code": "SSD",
    "codeName": "SSD"
  },
  "cloudMssqlServerName": "m-7241902-002",
  "cloudMssqlServerRole": {
    "code": "H",
    "codeName": "Mirror"
  },
  "cloudMssqlServerInstanceStatusName": "creating",
  "cloudMssqlServerInstanceState": {
    "code": "PEND",
    "codeName": "CLOUD DATABASE(VPC) Server Pending State"
  },
  "cloudMssqlServerInstanceOperation": {
```

```
        "code": "CREAT",

        "codeName": "CLOUD DATABASE(VPC) Server Create OP"

    },

    "regionCode": "KR",

    "zoneCode": "KR-2",

    "vpcNo": "3651",

    "subnetNo": "6131",

    "dataStorageSize": 107374182400,

    "cpuCount": 2,

    "memorySize": 4294967296,

    "isPublicSubnet": true,

    "cloudMssqlProductCode": "SVR.VMSSL.HICPU.C002.M004.NET.HDD.B100.G002",

    "createDate": "2021-07-08T16:15:04+0900",

    "dataStorageType": {

        "code": "SSD",

        "codeName": "SSD"

    }

}

]

}

}

}}
```

2. 서버가 생성되기 시작합니다. 상태가 운영중으로 바뀔 때 까지 대기

DB 서비스 이름	DB Role	DB Server 이름	DB Server 타입
edu-mssql	Principal	m-7241902-001	2vCPU, 4GB Mem
edu-mssql	Mirror	m-7241902-002	2vCPU, 4GB Mem

### 3. CLI 명령어를 이용하여 Slave 노드 추가

```
[root@lnxsvr1 cli_linux]#./ncloud vmssql createCloudMssqlSlaveInstance --regionCode KR --cloudMssqlInstanceNo 7241902 --privateDomainPostfix 001

{"createCloudMssqlSlaveInstanceResponse": {

    "requestId": "8248b1fa-d33f-4058-80dc-521775d91770",

    "resultCode": "0",

    "returnMessage": "success",

    "totalRows": 1,

    (이하생략)
}}
```

### 4. Slave 노드가 생성되는 것을 확인

DB 서비스 이름	DB Role	DB Server 이름	DB Server 타입
<input type="checkbox"/> edu-mssql	Principal	m-7241902-001	2vCPU, 4GB Mem
edu-mssql	Mirror	m-7241902-002	2vCPU, 4GB Mem
edu-mssql	Slave	m-7241902-003	2vCPU, 4GB Mem

## Redis 서버 생성

- 먼저 Private Subnet을 생성합니다.
- Subnet 이름 : lab1-vpc-redis-subnet
- IP 대역 : 10.1.4.0/24
- 속성 : Private

All product > Cloud DB for Redis > Config Group > Config Group 생성 클릭

Config Group 생성

Redis에서 사용할 config group을 생성 합니다.

Version \* ① 4.0.14

이름 \* lab1-config 최소 3글자 최대 15

설명

0/255

x 아니요 ✓ 예

All product > Cloud DB for Redis > + Redis Cluster 생성 클릭

Redis Version: Redis(4.0.14)

라이센스: BSD(Berkeley Software Distribution)

VPC\*: left VPC 생성

Subnet\*: pri1 | KR-1 | Private Subnet 생성

Cloud DB 상품은 Private Subnet에서만 생성 가능합니다.

노드당 Memory: 메모리 1.5GB

사드 수: 3

사드당 복제본: 0 만약 사드당 복제본을 0으로 설정한다면 고가용성 지원이 되지 않습니다.

Config Group\*: - select - Config group 바로가기

요금제: 시간 요금제 (Cloud DB for Redis 상품은 시간 요금제만 지원합니다.)

Redis Server 이름\*: edu -001 최소 3글자, 최대 25자

Redis 서비스 이름\*: edu 최소 3글자, 최대 15자

ACG 설정: Cloud DB for Redis ACG는 자동 생성 됩니다. (예: cloud-redis-\*)  
Redis 접근을 위한 ACG 설정은 사용자 가이드 -> 블록체인 가이드를 참고하세요

Redis 접속포트\*: 6379 6379 또는 10000 ~ 20000만 입력 가능 합니다.

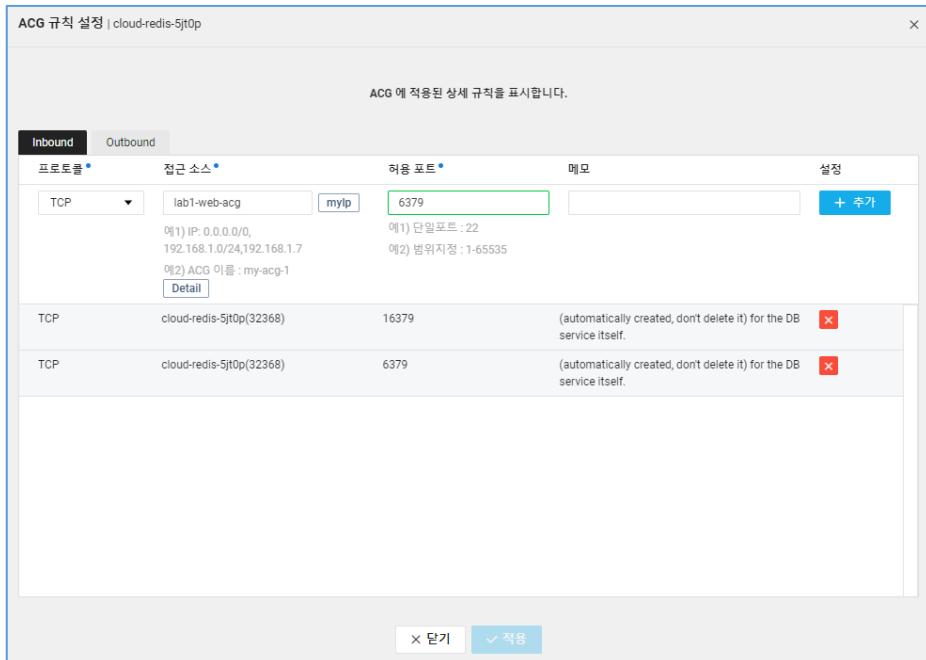
- Redis 서버 타입은 Standard 메모리 1.5GB
- Config Group은 lab1-config 선택
- Redis 서버 이름은 lab
- Redis 서비스 이름은 lab
- Redis 접속 포트는 6379로 입력
- 백업 설정 Check 후, 백업 파일 보관 기간은 7일, 백업 수행 시간은 자동으로 설정합니다.
- 모든 설정값 기입 후, 다음 버튼을 클릭하여 마지막 확인을 수행합니다.

서버정보	
Redis 버전:	Redis(4.0.14)
라이센스:	BSD
노드당 Memory:	1.5 GB
고가용성 지원:	아니오
Redis Server 이름:	lab
Redis 서비스 이름:	lab
ACG설정:	자동
VPC:	left
Subnet:	pri1   KR-1   Private
Cluster 정보	
shard 수:	3
shard당 복제본:	0
Config Group:	lab1-config
총 가용 메모리:	4.5GB
Redis 접속포트:	6379

마지막으로 생성 버튼을 클릭하여 Redis 서버를 생성합니다.

ACG에서 6379 포트의 허용 정책을 적용합니다.

- Product and services > server > acg에 가면, 'redis'라는 이름을 가지고 있는 ACG가 자동 생성되어 있습니다.
- 해당 ACG에 lab1-web-acg에 대하여 6379포트를 허용해주는 룰을 추가합니다.



## Redis 서버 접속

생성한 Cloud DB for Redis 서버에 접속을 해보겠습니다.

Redis의 DNS 정보는 Redis Cluster를 선택한 후, 관리 > Redis 설정에서 노드 별로 확인할 수 있습니다.

Hostname	Memory	Status	Role	Slot	Dns
lab-001-001-crq	1.5GB	● 운영중	Master	0-5460	redis-6dq1s.vpc-cdb.ntruss.com
lab-002-001-crr	1.5GB	● 운영중	Master	5461-10922	redis-6dq1v.vpc-cdb.ntruss.com
lab-003-001-crs	1.5GB	● 운영중	Master	10923-16383	redis-6dq22.vpc-cdb.ntruss.com

아래 명령어를 통해 Redis Client를 설치합니다

//6.2.6 버전 기준

```
[root@redis-web ~]# wget https://download.redis.io/releases/redis-6.2.6.tar.gz
[root@redis-web ~]# tar xvfz redis-6.2.6.tar.gz
[root@redis-web ~]# cd redis-6.2.6
[root@redis-web redis-6.2.6]# make
```

아래 명령어를 통해 Redis 노드에 접속합니다.

```
[root@redis-web ~]# cd src
[root@redis-web src]# ./redis-cli -h ①DNS 명 -p ② Redis 접속포트
[root@redis-web src]# ./redis-cli -c -h redis-fk5o.beta-cdb.ntruss.com -p 6379
redis-fk5o.beta-cdb.ntruss.com:6379>
(참고 페이지 : https://guide.ncloud-docs.com/docs/clouddbforredis-start )
```

```
[root@lnxsvr1 redis]# redis-cli -c -h redisc-6mq4h.vpc-cdb.ntruss.com -p 6379
redisc-6mq4h.vpc-cdb.ntruss.com:6379> zrevrange score 0 -1 withscores
```

## Lab 8 Guide

### Cloud Hadoop 구성

#### 1. (사전 작업) Hadoop cloud Object storage 구성

Product & Service > Object Storage > Bucket Management > + 버킷생성 선택

기본정보> “버킷이름”에 labhadoop아이디 입력

VPC / Object Storage / Bucket Management

◀ 버킷 생성 파일과 폴더를 저장하는 상위 단위인 버킷을 생성하세요.

1 기본 정보 2 권한 관리 3 잡금 관리 4 최종 확인

기본 정보 입력  
버킷은 파일과 폴더를 저장하는 상위 단위입니다.  
리전 내에서 유일하게 사용될 버킷의 이름을 입력하세요.  
Object Storage 요금은 저장된 데이터 양, API 요청수, 네트워크 전송 요금을 합산하여 부과합니다.

버킷 이름 : labhadoop-edu50

다음 >

권한관리, 최종확인 단계를 거쳐 “버킷생성” 클릭 하면 Object storage 생성 완료

#### 2. Hadoop 클러스터 설정

- Product & Service > Big data & Analytics > Cloud Hadoop > + 클러스터 생성 선택
- “클러스터 이름”에 lab 이라고 입력
- 클러스터 버전은 Cloud Hadoop 1.8
- 클러스터 타입은 Core Hadoop with Spark 선택
- 클러스터 add-on은 선택 안함
- 커버로스 인증 구성은 비활성화
- VPC는 lab1-vpc 입니다.
- 클러스터 관리자 계정 : hadoop
- 클러스터 관리자 계정 패스워드 : ncpNCP!@#123

**클러스터 설정**  
(필수 입력 사항입니다)

클러스터 이름\* : lab

클러스터 버전\* : Cloud Hadoop 1.4

클러스터 Type\* : Spark : HDFS(3.1.1), YARN(3.1.1), spark(2.3.2), HIVE(3.1.1), Hue(4.3.0), Zeppelin Notebook(0.8.0), Ranger(1.2.0)

커버로스 인증 구성

VPC\* : lab-vpc

ACG 설정\* : Cloud Hadoop ACG는 자동 생성 됩니다.  
(예: cloud-Hadoop-acg)  
hadoop 접근을 위한 ACG 설정은 사용자 가이드 > ACG 사용 가이드를 참고 하세요.

< 이전      다음 >

- Object Storage에 미리 생성한 버킷을 지정합니다.
- 나머지는 Default로 구성합니다.

**스토리지 & 서버 설정**  
(필수 입력 사항입니다)

Object Storage 버킷	labhadoop-edu50	<input type="button" value="새로고침"/>
고가용성 지원	<input checked="" type="checkbox"/> 기본적으로 2대의 마스터 노드가 생성됩니다.	
엣지노드 Subnet*	demo-subnet   KR-2   Public	<input type="button" value="Subnet 생성"/>
엣지노드 서버 타입	vCPU 4개, 메모리 8GB, 디스크 50GB	
엣지노드 개수	1 <input type="button" value="1개로 고정"/>	
마스터노드 Subnet*	demo-subnet   KR-2   Public	<input type="button" value="Subnet 생성"/>
마스터노드 서버타입	vCPU 4개, 메모리 16GB, 디스크 50GB	<input checked="" type="checkbox"/> 기본적으로 50G크기의 디스크(HDD)가 제공됩니다.
마스터노드 개수	2 <input type="button" value="고가용성 지원으로 2대의 마스터 노드가 생성되며 변경은 불가합니다."/>	
마스터노드 스토리지 타입	<input checked="" type="radio"/> SSD <input type="radio"/> HDD <input type="checkbox"/> 설치 이후에 스토리지 타입은 변경되지 않습니다.	
마스터노드 스토리지 용량	100 GB, 최소 100GB, 최대 2000GB 까지 10GB단위 선택 가능합니다.	
작업자노드 Subnet*	lab1-vpc-db-sub1   KR-2   Private	<input type="button" value="Subnet 생성"/>
작업자노드 서버타입	vCPU 4개, 메모리 32GB, 디스크 50GB	<input checked="" type="checkbox"/> 기본적으로 50G크기의 디스크(HDD)가 제공됩니다.
작업자노드 개수	2 <input type="button" value="작업자 노드는 최소 2개이상 되어야 합니다.(default 2)"/>	
작업자노드 스토리지 타입	<input checked="" type="radio"/> SSD <input type="radio"/> HDD <input type="checkbox"/> 설치 이후에 스토리지 타입은 변경되지 않습니다.	
작업자노드 스토리지 용량	100 GB, 최소 100GB, 최대 2000GB 까지 10GB단위 선택 가능합니다.	
요금제	<input type="checkbox"/> 시간 요금제 <input checked="" type="checkbox"/> 요금 안내(Storage 비용은 별도 부과됩니다.)	

- 엣지 노드, 마스터 노드는 Public Subnet에 배치, 작업자 노드는 Private Subnet 배치
- 나머지는 Default 설정을 유지합니다.
- 하단의 다음 버튼을 클릭합니다.
- 인증키 설정 부분은 ‘보유하고 있는 인증키 이용’을 선택 후, 서버 생성 시 다운로드 받았던 인증키를 선택합니다.
- 하단의 다음 버튼을 클릭합니다.
- 마지막 최종 확인 후, 생성 버튼을 클릭합니다.

### 3. Zeppelin Notebook 접속

- Zeppelin notebook에 접속하기 위해서 cloud Hadoop ACG를 업데이트합니다.
- Product and service > Server > ACG 항목에서 cloud-hadoop-acg-\*로 시작하는 acg를 클릭후, myip에 대해서 9996포트를 허용해줍니다.

ACG 규칙 설정 | cloud-hadoop-acg-49k7o

ACG에 적용된 상세 규칙을 표시합니다.

Inbound	Outbound			
프로토콜 *	접근 소스 *	허용 포트 *	메모	설정
TCP	myip			+ 추가
예1) IP: 0.0.0.0/0, 192.168.1.0/24,192.168.1.7 예2) ACG 이름 : my-acg-1		예1) 단일포트 : 22 예2) 범위지정 : 1-65535		
<a href="#">Detail</a>				
TCP	182.216.195.2/32	9996		<a href="#">X</a>
TCP	cloud-hadoop-acg-49k7o(19854)	1-65535	(automatically created, don't delete it) cloud-hadoop-acg-lab-hadoop	<a href="#">X</a>
ICMP	211.249.71.0/24		(automatically created, don't delete it) cloud-hadoop-acg-lab-hadoop	<a href="#">X</a>
ICMP	211.249.70.0/24		(automatically created, don't delete it) cloud-hadoop-acg-lab-hadoop	<a href="#">X</a>
ICMP	211.249.69.128/25		(automatically created, don't delete it) cloud-hadoop-acg-lab-hadoop	<a href="#">X</a>
TCP	211.249.71.0/24	8080	(automatically created, don't delete it) cloud-hadoop-acg-lab-hadoop	<a href="#">X</a>
TCP	211.249.70.0/24	8080	(automatically created, don't delete it) cloud-hadoop-acg-lab-hadoop	<a href="#">X</a>

[X 닫기](#) [✓ 적용](#)

- lab-hadoop선택 후, 상단의 ‘application별 보기’ 클릭 후 zeppelin notebook 접속 용 클릭

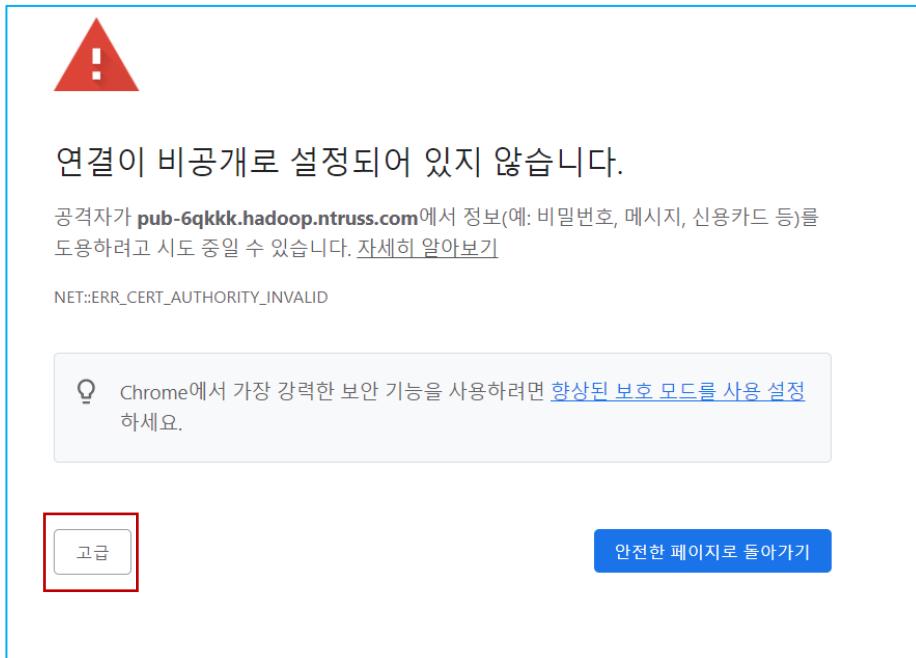
Application web UI 접속을 위한 사전 작업

Application web UI 접속을 위해서는 사전에 아래 모든 규칙이 해당 클러스터의 ACG에 필수로 설정되어 있어야 합니다.  
ACG 설정 변경은 몬술 > 클러스터 상세 정보 > ACG에서 가능합니다.

프로토콜	접근소스	허용포트	비고
TCP	도메인	8443	Ambari Web Console 접속용
TCP	도메인	8081	Hue Admin 접속용
TCP	도메인	9996	<a href="#">Zeppelin Notebook 접속용</a>
TCP	도메인	6182	Ranger 접속용

다시 보지 않음 [✓ 확인](#)

- 아래와 같이 페이지가 노출되면, 고급 버튼을 클릭합니다.



- \*\*(안전하지 않음) 링크를 클릭합니다.



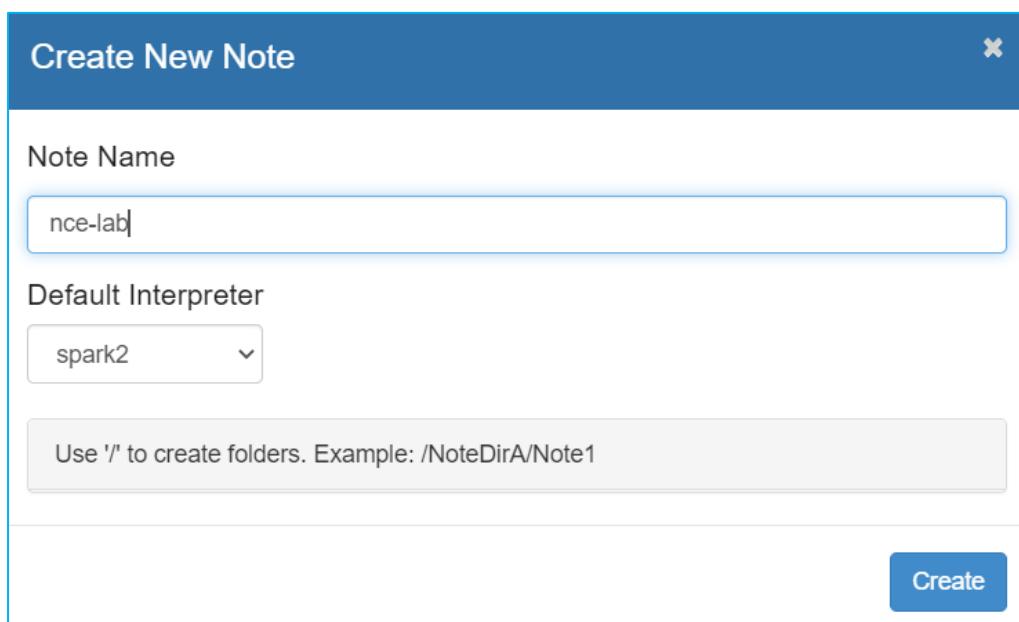
- 아래와 같이 zeppelin화면이 노출되면 정상 접속이 된 것입니다.
- Zeppelin 메인 화면 우측 상단의 로그인 버튼을 클릭한 후 클러스터 생성 시 기입한 클러스터 관리자 계정 정보로 접속을 합니다.



- ✓ Username : hadoop
- ✓ Password : ncpNCP!@#123

#### 4. Zeppelin notebook 생성 및 데이터 확인

- Zeppelin 상단의 노트북 클릭 후, 상단의 '+create new note'를 클릭합니다.
- Note name : nce-lab 입력
- Default interpreter : spark2 선택 후 하단의 create 버튼 클릭



Bank.csv파일을 bank테이블에 로드하는 샘플코드를 복사해서 붙여넣습니다.

```
%spark2.spark

import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset
import spark.implicits._

// Zeppelin creates and injects sc (SparkContext) and sqlContext (HiveContext or SqlContext)
```

```
// So you don't need create them manually

// load bank data

val bankText = sc.parallelize(
    IOUtils.toString(
        new URL("https://s3.amazonaws.com/apache-zeppelin/tutorial/bank/bank.csv"),
        Charset.forName("utf8")).split("\n"))

case class Bank(age: Integer, job: String, marital: String, education: String, balance: Integer)

val bank = bankText.map(s => s.split(";")).filter(s => s(0) != "age").map(
    s => Bank(s(0).toInt,
              s(1).replaceAll("\\", ""),
              s(2).replaceAll("\\", ""),
              s(3).replaceAll("\\", ""),
              s(5).replaceAll("\\", "").toInt
    )
).toDF()

bank.registerTempTable("bank")
```

- 우측 상단의 ▶ 버튼을 클릭하여 실행합니다.

```
%spark2.spark
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset

// Zeppelin creates and injects sc (SparkContext) and sqlContext (HiveContext or SqlContext)
// So you don't need create them manually

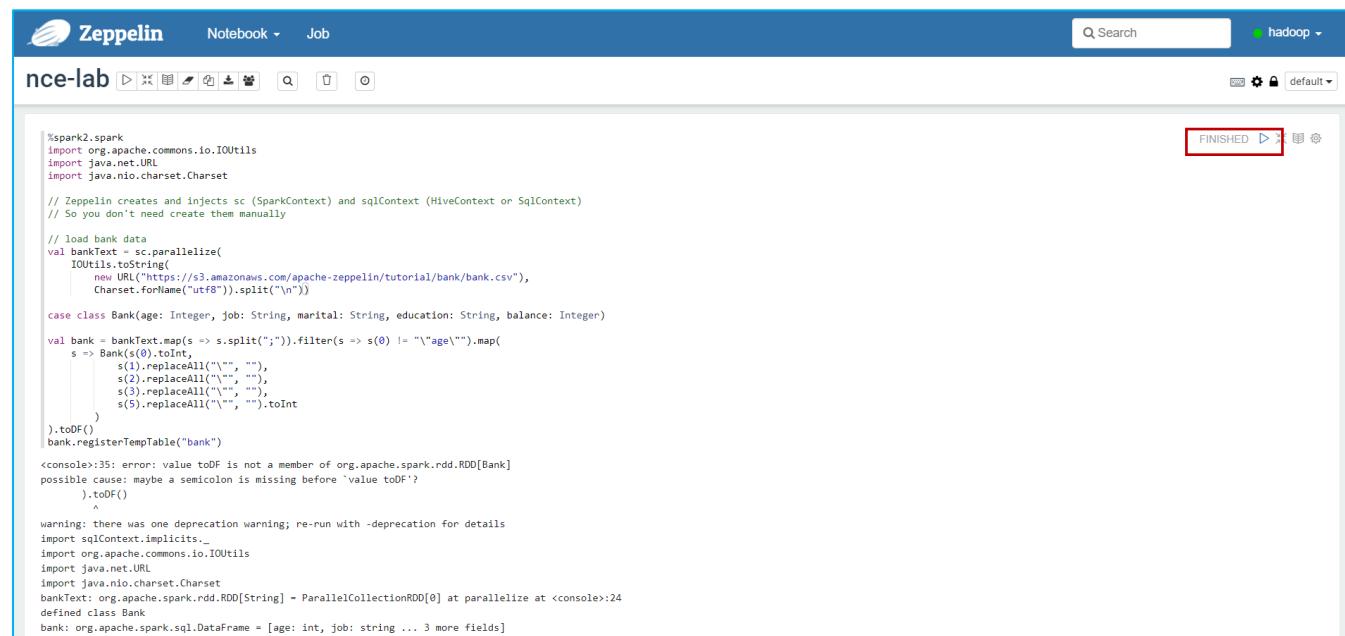
// load bank data
val bankText = sc.parallelize(
  IOUtils.toString(
    new URL("https://s3.amazonaws.com/apache-zeppelin/tutorial/bank/bank.csv"),
    Charset.forName("utf8")).split("\n"))

case class Bank(age: Integer, job: String, marital: String, education: String, balance: Integer)

val bank = bankText.map(s => s.split(",")).filter(s => s(0) != "\age\").map(
  s => Bank(s(0).toInt,
    s(1).replaceAll("\\", ""),
    s(2).replaceAll("\\", ""),
    s(3).replaceAll("\\", ""),
    s(5).replaceAll("\\", "").toInt
  )
).toDF()
bank.registerTempTable("bank")
```

READ   

실행 후, 우측 상단의 status가 finished로 변경되면 성공적으로 데이터가 입력된 것입니다.



```
%spark2.spark
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset

// Zeppelin creates and injects sc (SparkContext) and sqlContext (HiveContext or SqlContext)
// So you don't need create them manually

// load bank data
val bankText = sc.parallelize(
  IOUtils.toString(
    new URL("https://s3.amazonaws.com/apache-zeppelin/tutorial/bank/bank.csv"),
    Charset.forName("utf8")).split("\n"))

case class Bank(age: Integer, job: String, marital: String, education: String, balance: Integer)

val bank = bankText.map(s => s.split(",")).filter(s => s(0) != "\age\").map(
  s => Bank(s(0).toInt,
    s(1).replaceAll("\\", ""),
    s(2).replaceAll("\\", ""),
    s(3).replaceAll("\\", ""),
    s(5).replaceAll("\\", "").toInt
  )
).toDF()
bank.registerTempTable("bank")

<console>:35: error: value toDF is not a member of org.apache.spark.RDD[Bank]
possible cause: maybe a semicolon is missing before 'value toDF'?
  ).toDF()
  ^
warning: there was one deprecation warning; re-run with -deprecation for details
import sqlContext.implicits._
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset
bankText: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[0] at parallelize at <console>:24
defined class Bank
bank: org.apache.spark.sql.DataFrame = [age: int, job: string ... 3 more fields]
```

FINISHED   

이번에는 데이터를 조회해보고, 그래프로 결과를 확인해보겠습니다.

```
%spark2.sql

select age, count(1) value
from bank
where age < 30
group by age
order by age
```

```
%spark2.sql  
select age, count(1) value  
from bank  
where age < 30  
group by age  
order by age
```

READY ▶ X ⌂ ⌂

아래와 같이 표 형태로 결과를 확인해 볼 수 있습니다.

```
%spark2.sql  
select age, count(1) value  
from bank  
where age < 30  
group by age  
order by age
```

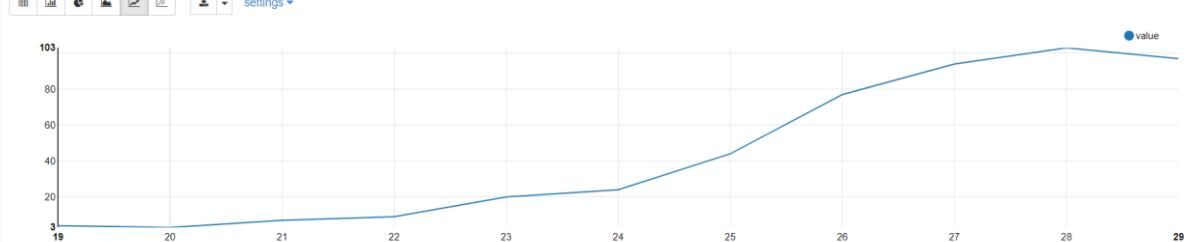
SPARK JOB FINISHED ▶ X ⌂ ⌂

age	value
19	4
20	3
21	7
22	9
23	20
24	24
25	44
26	77

도표 클릭 시, 아래와 같이 그래프 형태로도 데이터 결과를 확인할 수 있습니다.

```
%spark2.sql  
select age, count(1) value  
from bank  
where age < 30  
group by age  
order by age
```

SPARK JOB FINISHED ▶ X ⌂ ⌂



Took 6 sec. Last updated by hadoop at July 02 2021, 1:40:24 PM. (outdated)

**Lab 9 Guide****Cloud Data Streaming Service 생성**

- Product and service > big data & analytics > Cloud Data Streaming service 클릭
- 메뉴 중 'Config Group'클릭
  - Version은 Kafka-2.4.0, CMAK 3.0.0.5
  - Config Group Name은 'nce-config-group'으로 기입 후 하단의 확인 버튼 클릭
- 상단의 클러스터 메뉴로 이동 및 상단의 클러스터 생성 클릭
  - 클러스터 이름 : nce-kafka
  - Application 버전 : kafka 2.4.0, CMAK 3.0.0.5
  - Config group 설정 : nce-config-group
  - CMAK 접속 ID : student
  - CMAK 접속 Password : (수강생이 원하는 암호 입력)
  - CMASK 접속 Password 확인 : (암호 재입력)
  - 하단의 다음버튼 클릭

VPC / Cloud Data Streaming Service / Cluster

**Cluster 생성** 신규 Kafka 클러스터를 생성합니다.

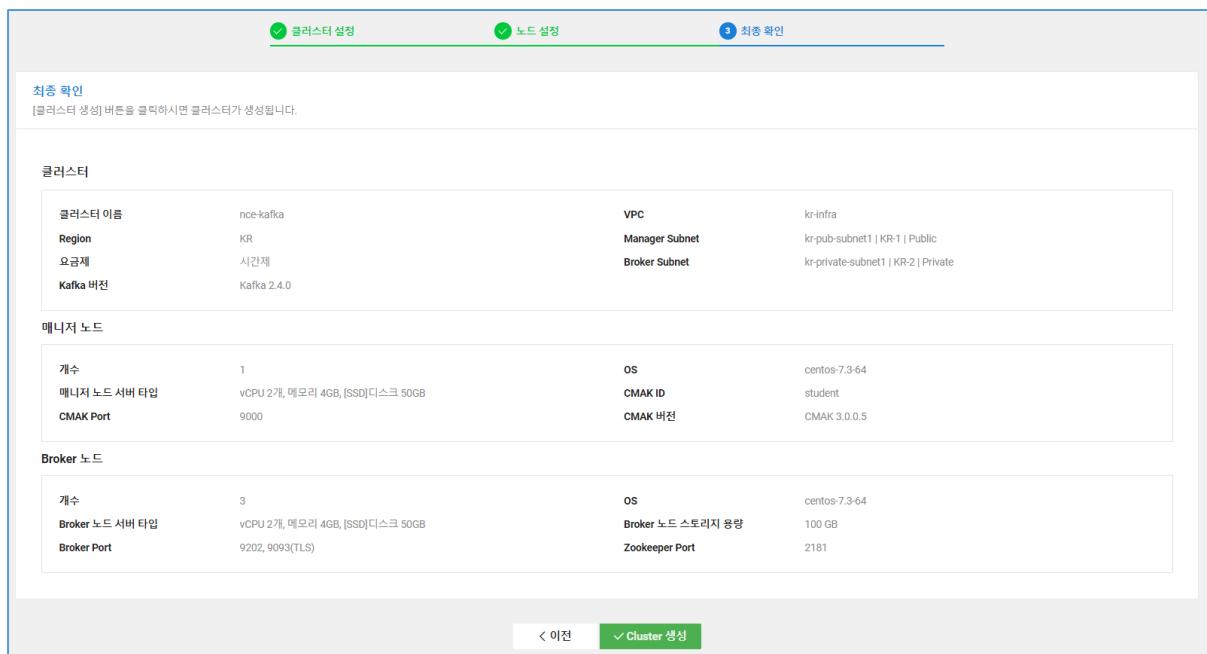
① 클러스터 설정      ② 노드 설정      ③ 최종 확인

**클러스터 설정**  
클러스터의 주요 항목을 입력합니다. (\* 필수 입력 사항입니다.)

클러스터 이름*	nce-kafka	입력하신 클러스터 이름으로 노드 서버의 이름과 hostname을 설정합니다. (예: 클러스터 이름-XX)
Application 버전*	Kafka 2.4.0, CMAK 3.0.0.5	Cloud Data Streaming Service의 ACG는 자동으로 생성됩니다. (예: cdss-클러스터 코드)
ACG 설정*	Broker와의 통신을 위한 포트입니다. 9092으로 고정되어 있습니다.	
Kafka Broker Port*	9092	Broker와의 통신을 위한 포트입니다. 9092으로 고정되어 있습니다.
Kafka Broker TLS Port*	9093	Broker와의 통신을 암호화 할 수 있는 포트입니다. 9093으로 고정되어 있습니다. 암호화를 이용한 통신은 가이드를 참고하여 사용하여 주세요.
Zookeeper Port*	2181	Zookeeper Client와의 통신을 위한 포트입니다. 2181으로 고정되어 있습니다.
CMAK Port*	9000	Public Domain의 9000번 port로 접근할 수 있습니다.
CMAK 접속 권한*	CMAK 접속 ID <input type="text" value="student"/>	ID는 향후 변경이 불가능합니다.
	CMAK 접속 Password <input type="password" value="*****"/>	
	CMAK 접속 Password 확인 <input type="password" value="*****"/>	

**다음 >**

- OS 타입 : centos-7.8-64
- VPC : lab1-vpc
- 매니저 노드 subnet : lab1-vpc-web-subnet 선택
- 매니저 노드 서버 타입 : CPU 2개, 메모리 4GB, 디스크 50GB
- Broker 노드 subnet : (실습 시 생성한 private subnet선택)
- Broker 노드 개수 : 3
- Broker 노드 서버 타입 : CPU 2개, 메모리 4GB, 디스크 50GB
- Broker 노드 스토리지 용량 : 100GB
- 하단의 다음 버튼 클릭



- 하단의 Cluster 생성 클릭

### 서버에 Apache Log-generator 설치

실습 시 생성했던 Inxsrv1서버에 접속 후 아래 명령어 실행

```
# wget https://github.com/kiritbasu/Fake-Apache-Log-Generator/archive/refs/heads/master.zip
# unzip master.zip
```

### 웹 서버 로그를 Cloud Data Streaming service로 전송하기 위해 Logstash 설치

(설치 전 python pip 및 java 설치 진행)

```
# yum -y install epel-release  
  
# yum -y install python-pip  
  
# yum -y install java-1.8.0-openjdk  
  
# wget https://artifacts.elastic.co/downloads/logstash/logstash-7.9.2.tar.gz  
  
# tar -zxvf logstash-7.9.2.tar.gz  
  
# cd logstash-7.9.2
```

### Logstash-apache.conf 파일을 통해 Cloud Data Streaming service 연동 정보 입력

```
# vi logstash-apache.conf  
  
input {  
    file {  
        path => "/tmp/access_log/*"  
        start_position => "beginning"  
    }  
}  
  
filter {  
    grok {  
        match => { "message" => "%{COMBINEDAPACHELOG}" }  
    }  
    date {  
        match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ]  
    }  
}
```

```
    }

}

output {
    kafka {
        bootstrap_servers => "10.0.8.7:9092, 10.0.8.8:9092, 10.0.8.13:9092"
        topic_id => "apache-access-log"
        codec => plain {
            format => "%{message}"
        }
    }
    stdout { codec => rubydebug }
}
```

\* Bootstrap\_servers 정보에는 Clou Data Streaming Service의 Broker 노드에 대한 내부 IP 정보를 기입

- ✓ Product and services > Analytics > Cloud Data Streaming Service > 클러스터 노드 목록에서 상세보기 클릭

클러스터 노드 서버 목록 보기

클러스터에 포함된 노드 서버의 목록을 표시합니다.

클러스터 이름: nce-kafka

서버이름	내부 IP	서버 상태	노드 타입	Subnet
nce-kafka-m-fvd	10.0.1.7	● 운영중	Manager	kr-pub-subnet1   KR-1
nce-kafka-b-fve	10.0.8.7	● 운영중	Broker	kr-private-subnet1   KR-2
nce-kafka-b-fvf	10.0.8.8	● 운영중	Broker	kr-private-subnet1   KR-2
nce-kafka-b-fvg	10.0.8.13	● 운영중	Broker	kr-private-subnet1   KR-2

확인

Logstash가 설치된 서버에서 Cloud Data Streaming Service로 통신이 가능하도록 Cloud Data Streaming Service의 Broker 노드 ACG 업데이트

■ Product and services > analytics > Cloud data streaming service 클릭

■ 하단의 Broker 노드 ACG 이름 확인

VPC / Cloud Data Streaming Service / Cluster

Cloud Data Streaming Service

+ Cluster 생성 상품 더 알아보기 새롭고 첨  
 클러스터 관리

클러스터 이름	Broker 노드 타입	노드 수	서버 상태	클러스터 상태	Kafka 버전	VPC
nce-kafka	2vCPU, 4GB Mem	3	● 운영중	상태 보기	2.4.0	kr-infra

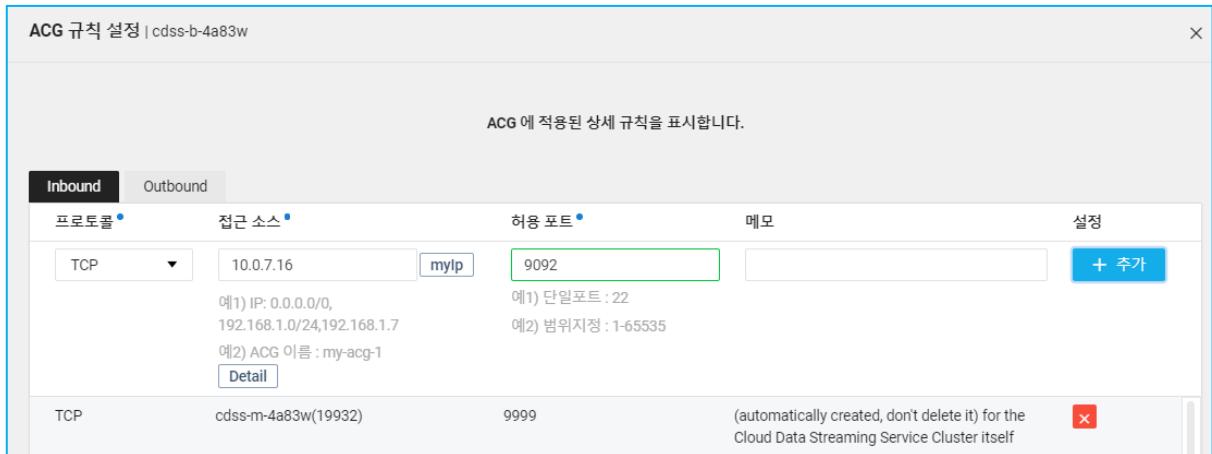
클러스터 이름: nce-kafka(7195532)  
OS: CentOS 7.3 (64-bit)  
Kafka 버전: 2.4.0  
Broker 노드 스토리지 용량(SSD): 300 GB  
Broker 노드 ACG: cdss-b-4a83w(19931) [규칙 보기]

Broker Port: 9092  
Zookeeper Port: 2181  
클러스터 노드 목록: 상세 보기  
Broker 노드 Subnet: kr-private-subnet1 | KR-2

생성일시: 2021-07-05 11:43:16  
CMAK 버전: 3.0.0.5  
매니저 노드 서버 타입: 2vCPU, 4GB Mem  
Public 도메인: 미활성화  
매니저 노드 ACG: cdss-m-4a83w(19932) [규칙 보기]  
CMAK Port: 9000 [바로가기]  
인증서 관리: [다운로드]  
매니저 노드 Subnet: kr-pub-subnet1 | KR-1  
Broker 노드 정보: 상세 보기

■ Product and services > Server > ACG에서 Broker 노드 ACG 클릭 후 상단의 'ACG 설정' 클릭

■ 프로토콜 : TCP, 접근 소스 : Lnxsvr1서버의 내부IP, 허용 포트 : 9092 추가



### ■ Logstash 실행

```
# bin/logstash -f logstash-apache.conf
```

[2021-07-05T16:02:57,233][INFO ][logstash.agent] [Successfully started Logstash API endpoint {:port=>9600}] → 성공적으로 연결된 것을 확인 후 종료

### ■ 로그 생성 수행

Fake-Apache-Log-Generator-master 폴더로 이동

```
# cd ~/Fake-Apache-Log-Generator-master
# pip install -r requirements.txt
# mkdir -p /tmp/access_log/sample
# python apache-fake-log-gen.py -n 0 -o LOG -p /tmp/access_log/sample
```

### ■ 해당 로그를 로그스태시로 전송

```
# cd ~/logstash-7.9.2
# bin/logstash -f logstash-apache.conf
```

### ■ 아래와 같이 로그가 전송이 완료되면, 카프카 클러스터에 실제 토픽이 생성되었는지 확인해보

도록 하겠습니다.

```

"ident" => "-",
"clientip" => "211.80.95.151",
"httpversion" => "1.0"
}

{
  "request" => "/explore",
  "auth" => "-",
  "timestamp" => "17/Dec/2021:00:07:15 +0900",
  "host" => "kr-nce",
  "agent" => "\"Mozilla/5.0 (Macintosh; U; PPC Mac OS X 10_5_7; rv:1.9.5.20) Gecko/2013-08-22 03:57:07 Firefox/13.0\"",
  "message" => "230.60.225.215 - - [17/Dec/2021:00:07:15 +0900] \"GET /explore HTTP/1.0\" 404 5036 \"http://www.franklin-bruce.com/explore/author.html\" \"Mozilla/5.0 (Macintosh; U; PPC Mac OS X 10_5_7; rv:1.9.5.20) Gecko/2013-08-22 03:57:07 Firefox/13.0\"",
  "path" => "/tmp/access_log/sample_access_log_20210705-170150.log",
  "response" => "404",
  "bytes" => "5036",
  "@version" => "1",
  "@timestamp" => 2021-12-16T15:07:15.000Z,
  "verb" => "GET",
  "referrer" => "\\"http://www.franklin-bruce.com/explore/author.html\\\"",
  "ident" => "-",
  "clientip" => "230.60.225.215",
  "httpversion" => "1.0"
}

```

- Product and services > Analytics > Cloud Data Streaming Service 선택
- 생성한 Kafka cluster선택 후, 상단의 클러스터 관리 클릭 > CMAK 접속 도메인 설정 변경을 클릭하여 public domain 활성화



- Public domain으로 접속
  - 클러스터 상세 정보 내 'CMAK Public 도메인' 옆 아이콘 클릭
  - 도메인 활성화 > 확인 클릭
  - ID : Student paaswd : (사용자가 설정한 패스워드)

클러스터 이름	Broker 노드 타입	노드 수	서버 상태	클러스터 상태	Kafka 버전	VPC
nce-kafka	2vCPU, 4GB Mem	3	● 운영중	상태 보기	2.4.0	kr-infra
클러스터 이름	nce-kafka(7195532)	생성일시	2021-07-05 11:43:16			
OS	CentOS 7.3 (64-bit)	CMAK 버전	3.0.0.5			
Kafka 버전	2.4.0	매니저 노드 서버 타입	2vCPU, 4GB Mem			
Broker 노드 스토리지 용량(SSD)	300 GB	Public 도메인	nce-kafka-1605642251.kr.cdss.navercp.com:9000			
Broker 노드 ACG	cdss-b-4a83w(19931)	매니저 노드 ACG	cdss-m-4a83w(19932)	규칙 보기	규칙 보기	
Broker Port	9092	CMAK Port	9000	바로가기		
Zookeeper Port	2181	인증서 관리		다운로드		
클러스터 노드 목록	상세 보기	매니저 노드 Subnet	kr-pub-subnet1   KR-1			
Broker 노드 Subnet	kr-private-subnet1   KR-2	Broker 노드 정보	상세 보기			

아래와 같이 CMAK 페이지가 뜨면 카프카 클러스터 클릭

The screenshot shows the CMAK Cluster management interface. At the top left, there is a cluster icon and the text "CMAK Cluster ▾". Below this, a "Clusters" button is visible. The main area displays a table titled "Clusters" with three columns: "Active", "Operations", and "Version". There is one row in the table, representing the "nce-kafka" cluster. The "Active" column contains the cluster name "nce-kafka" (which is highlighted with a red box). The "Operations" column contains two buttons: "Modify" (blue) and "Disable" (orange). The "Version" column shows "2.4.0".

Active	Operations	Version
nce-kafka	Modify Disable	2.4.0

Topics를 클릭

**Clusters / nce-kafka / Summary**

## Cluster Information

Zookeepers	10.0.8.7:2181 10.0.8.8:2181 10.0.8.13:2181
Version	2.4.0

## Cluster Summary

Topics	1	Brokers	3
--------	---	---------	---

Apache access log가 정상적으로 전송된 것을 확인해볼 수 있습니다.

**CMAK nce-kafka Cluster ▾ Brokers Topic ▾ Preferred Replica Election Schedule Leader Election Reassign Partitions Consumers**

**Clusters / nce-kafka / Topics**

### Operations

Generate Partition Assignments Run Partition Assignments Add Partitions

### Topics

Show 10 entries Search:

Topic	# Partitions	# Brokers	Brokers Spread %	Brokers Skew %	Brokers Leader Skew %	# Replicas	Under Replicated %	Producer Message/Sec	Summed Recent Offsets
apache-access-log	3	3	100	0	0	1	0	0.00	0

Showing 1 to 1 of 1 entries Previous 1 Next

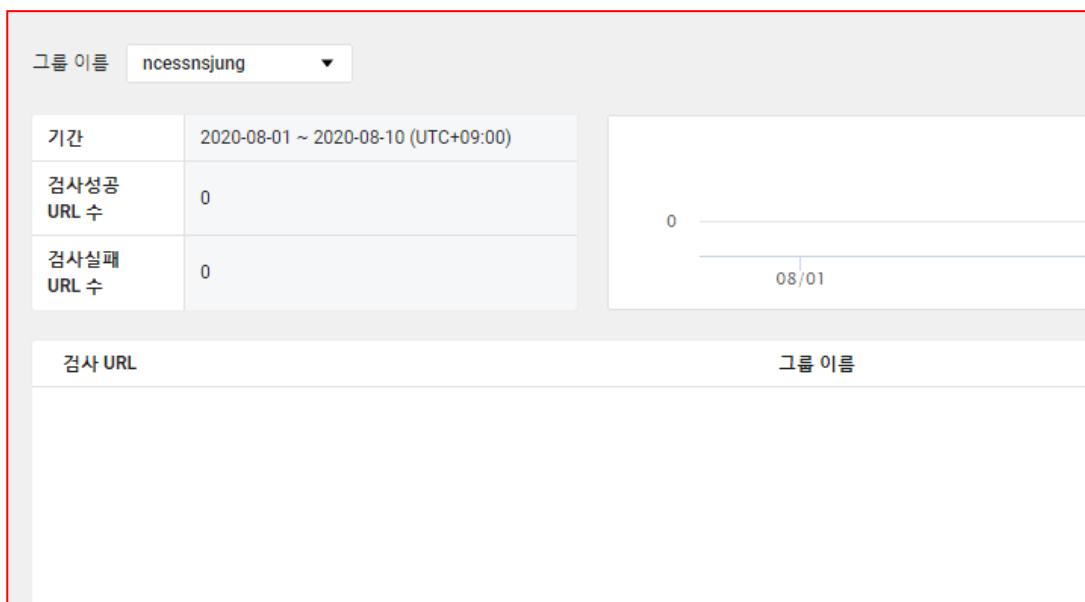
## Lab 10 Guide(데모로만 진행합니다)

### Site Safer 설정

- Security > Site Safer > Settings에서 +Setup Wizard 선택
- 그룹 이름 : ncess계정명
- 검사 URL 설정 : URL 자동 수집 선택
- <http://lnxsvr1> 서버의 공인IP 입력 후 URL 수집 버튼 클릭
- 최상위 URL 선택 후 완료 선택
- 하단의 다음버튼 클릭
- 한번 검사하기 선택
- 현재 시간에서 가장 가까운 시간 선택 후 추가 선택
- 통보 대상 설정

### Site Safer 확인

- Security > Sate Safer > URL Scan Results 선택
- 그룹 이름 : ncsss계정명으로 변경
- 내용 확인



### OS Security Checker 및 WAS Security Checker 설정

- Lnxsvr1 접속
- 다음 명령어 실행

```
[root@krweb001 ~]# wget http://ossc.ncloud.com:10080/download/ncp_secuagent.tar.gz
--2020-08-10 17:50:32-- http://ossc.ncloud.com:10080/download/ncp_secuagent.tar.gz
Resolving ossc.ncloud.com (ossc.ncloud.com)... 10.101.86.17
Connecting to ossc.ncloud.com (ossc.ncloud.com)|10.101.86.17|:10080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2543027 (2.4M) [application/x-gzip]
Saving to: ?cp_secuagent.tar.gz?

100%[=====] 2,543,027 --.-K/s in 0.02s

2020-08-10 17:50:32 (104 MB/s) - ?cp_secuagent.tar.gz?saved [2543027/2543027]

[root@krweb001 ~]# tar xvfz ncp_secuagent.tar.gz
ncp_secuagent
[root@krweb001 ~]# ./ncp_secuagent
[Project : linux] => Success
[root@krweb001 ~]# ./ncp_secuagent -p apache
[Project : apache] => Success
[root@krweb001 ~]#
```

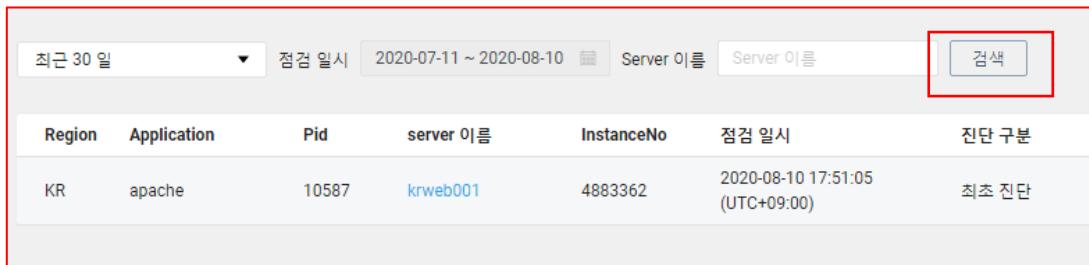
### 점검 내용 확인

- Security > System Security Checker > OS Security Checker 선택 후 검색 버튼 클릭

Region	server 이름	InstanceNo	Check list	점검 일시	OS version
KR	krweb001	4883362	Linux	2020-08-10 17:50:52 (UTC+09:00)	CentOS Linux release 7.3.1611 (Core)

- 우측의 리포트 클릭하여 내용 확인
- Security > System Security Checker > WAS Security Checker 선택 후 검색 버튼 클릭

릭



Region	Application	Pid	server 이름	InstanceNo	점검 일시	진단 구분
KR	apache	10587	krweb001	4883362	2020-08-10 17:51:05 (UTC+09:00)	최초 진단

- 우측의 리포트 클릭하여 내용 확인
- 리포트의 내용을 기반으로 보안 강화 작업 수행

## Lab 11 Guide

## 인증서 발급용 로드밸런서 생성

- Target Group 생성
  - 이름 : nce-cm-lab
  - Target 유형 : VPC Server
  - VPC : lab1-vpc
  - 프로토콜 : HTTP
  - 포트 : 80

**Target Group 생성**

생성할 Target Group의 이름을 입력하고 Target 유형과 포함될 Target의 VPC를 선택해주세요  
프로토콜에 따라 연결 가능한 로드밸런서 유형이 다릅니다. (•필수 입력 사항입니다.)

Target Group 이름 *	nce-cm-lab
Target 유형 *	VPC Server
VPC *	lab1-vpc (10.0.0.0/16)
프로토콜 <small>(?)</small> *	HTTP
포트 *	80
메모	

- 헬스체크 프로토콜 : HTTP
- 헬스체크 포트 : 80
- URL Path : /
- HTTP Method : HEAD
- 나머지 값은 디폴트 값으로 설정

**Health Check 설정**

Target에 대한 Health Check를 위한 정보를 입력하세요.  
Health Check에 실패한 서버는 로드밸런싱 대상에서 제외됩니다.(• 필수 입력 사항입니다.)

프로토콜 *	HTTP
포트 *	80
URL Path *	/
HTTP Method *	HEAD
Health Check 주기 (초) <small>(?)</small> *	30
정상 임계값 *	2
실패 임계값 *	2

- 포함시킬 서버 : lnxsvr1
- Networking > Load Balancer > +로드 밸런서 생성 > 어플리케이션 로드밸런서 생성 선택
  - 로드밸런서 이름 : nce-cm
  - Network : public IP
  - 부차 처리 성능 : small
  - 대상 vpc : lab1-vpc
  - 서브넷 : (생성된 lb 서브넷 활용)

**로드밸런서 생성**

생성할 로드밸런서 이름을 입력하고 로드밸런서가 할성화될 VPC 및 서브넷을 선택해주세요.  
로드밸런서를 생성하시면 로드밸런서 이용 시간과 트래픽 사용량에 따라 요금이 부과됩니다. (•필수 입력 사항입니다.)

유형	Application
로드밸런서 이름*	ncs-cm
Network*	<input type="radio"/> Private IP <input checked="" type="radio"/> Public IP
부하 처리 성능*	<input checked="" type="radio"/> Small <input type="radio"/> Medium <input type="radio"/> Large
대상 VPC*	lab1-vpc (10.0.0.0/16)
서브넷 선택*	<input checked="" type="checkbox"/> KR-2 lab1-vpc-lb-subnet   10.0.254.0/24 <input type="checkbox"/> KR-1 - 선택하세요 -

각 서비스 Zone에 로드밸런서를 배치할 전용 서브넷을 생성하셔야 합니다. 전용 서브넷에 서버 인스턴스를 위치시키면  
로드밸런싱이 동작하지 않습니다. 각각의 로드밸런서마다 서브넷을 생성할 필요는 없으나 가급적 C클래스(255.255.255.0) 이상을 권고합니다.

- 리스너 설정 : HTTP, 80포트 추가 후 하단의 ‘다음’ 버튼 클릭
- Target Group : nce-cm-lab 선택
- 생성 후 접속정보 확인

<b>접속 정보</b>	ncs-cm-7057057-6b67a2af43b5.kr.lb.naverncp.com 175.106.100.235
<b>부하 처리 성능</b>	Small
<b>액세스 로그 수집</b>	비활성

### DNS에 존 레코드 추가

- Networking > Global DNS > 상단의 도메인 추가
- 이름 : 교육용계정명.ncloudedu.com 선택
- 상단의 레코드 추가
  - 레코드명에 공란
  - 레코드 타입 : A
  - TTL : 15분
  - 레코드값 : 로드밸런서 공인 IP
  - 레코드 추가 선택

- 레코드명에 www
- 레코드 타입 : A
- TTL : 15분
- 레코드값 : 로드밸런서 공인 IP
  
- 레코드명에 server1
- 레코드값 : lnxsvr1 공인 IP
- 레코드 타입 : A
- TTL : 15분
- 추가 및 설정 적용

호스트	도메인	레코드 타입	레코드 값	TTL
@	edu76.ncloudedu.com	SOA	ns1-1.ns-ncloud.com. ns1-2.ns-ncloud.com. 6 21600 1800 1209600 300	300
@	edu76.ncloudedu.com	NS	ns1-1.ns-ncloud.com ns1-2.ns-ncloud.com	86400
server1	edu76.ncloudedu.com	A	175.106.98.12	300
@	edu76.ncloudedu.com	A	175.106.100.235	300
www	edu76.ncloudedu.com	A	175.106.100.235	300

### Free SSL 인증서 발급

- <https://letsencrypt.org/ko/getting-started/> 를 통해 FreeSSL 인증서 발급
- <https://certbot.eff.org/docs/using.html#manual> 참고
- 인증서 발급을 위한 우분투-18.04 임시 서버 생성 (Public Subnet에 생성 후 공인 IP 부여)
- Let's Encrypt 설치

```
[root@krweb001 ~]# apt install certbot
```

- Let's Encrypt 클라이언트 실행

```
[root@krweb001 ~]#
[root@krweb001 letsencrypt]# certbot certonly
```

● 인증서 발급 정보를 받기 위한 정보 입력

```
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator manual, Installer None
Enter email address (used for urgent renewal and security notices)
(Enter 'c' to cancel): 개인 메일 주소
Starting new HTTPS connection (1): acme-v02.api.letsencrypt.org

-----
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.2-November-15-2017.pdf. You must
agree in order to register with the ACME server. Do you agree?

-----
(Y)es/(N)o: y

-----
Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.

-----
(Y)es/(N)o: y
```

● 인증서 발급 도메인 입력 : 계정명.ncloudedu.com 입력

```
How would you like to authenticate with the ACME CA?

-----
1: Spin up a temporary webserver (standalone)

2: Place files in webroot directory (webroot)

-----
Select the appropriate number [1-2] then [enter] (press 'c' to cancel): 1

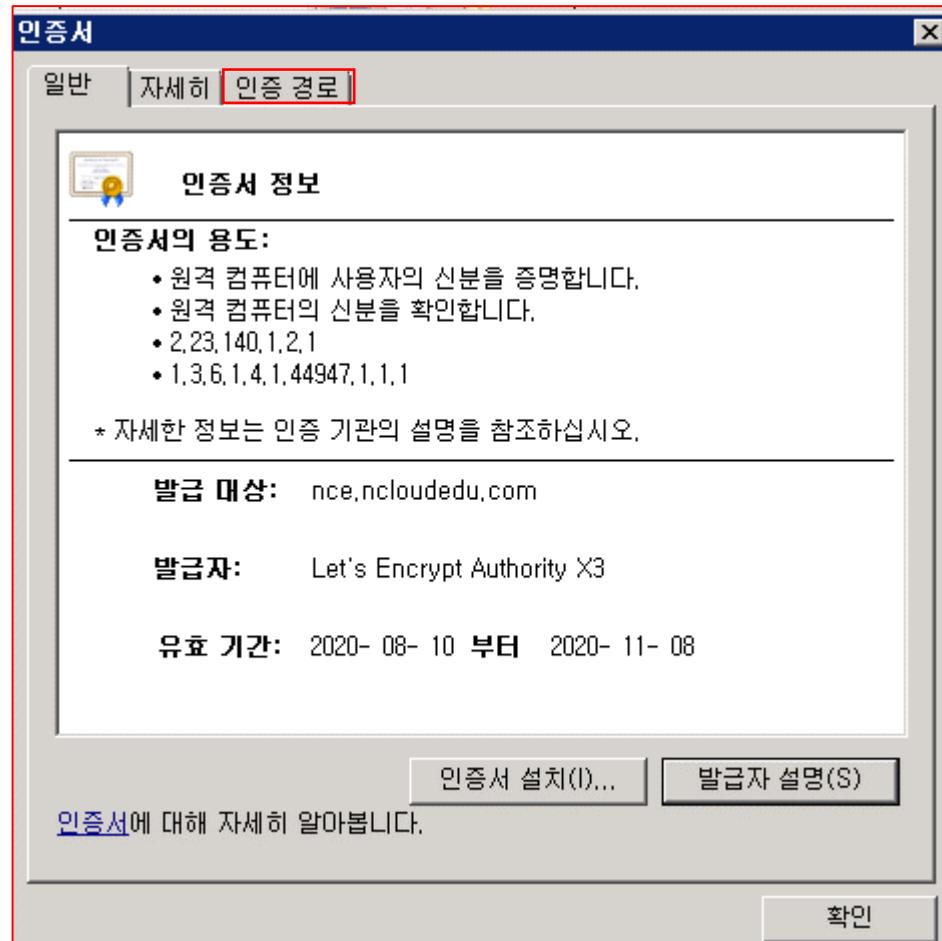
Plugins selected: Authenticator standalone, Installer None

Starting new HTTPS connection (1): acme-v02.api.letsencrypt.org

Please enter in your domain name(s) (comma and/or space separated) (Enter 'c'
to cancel):
```

### CM에 인증서 추가

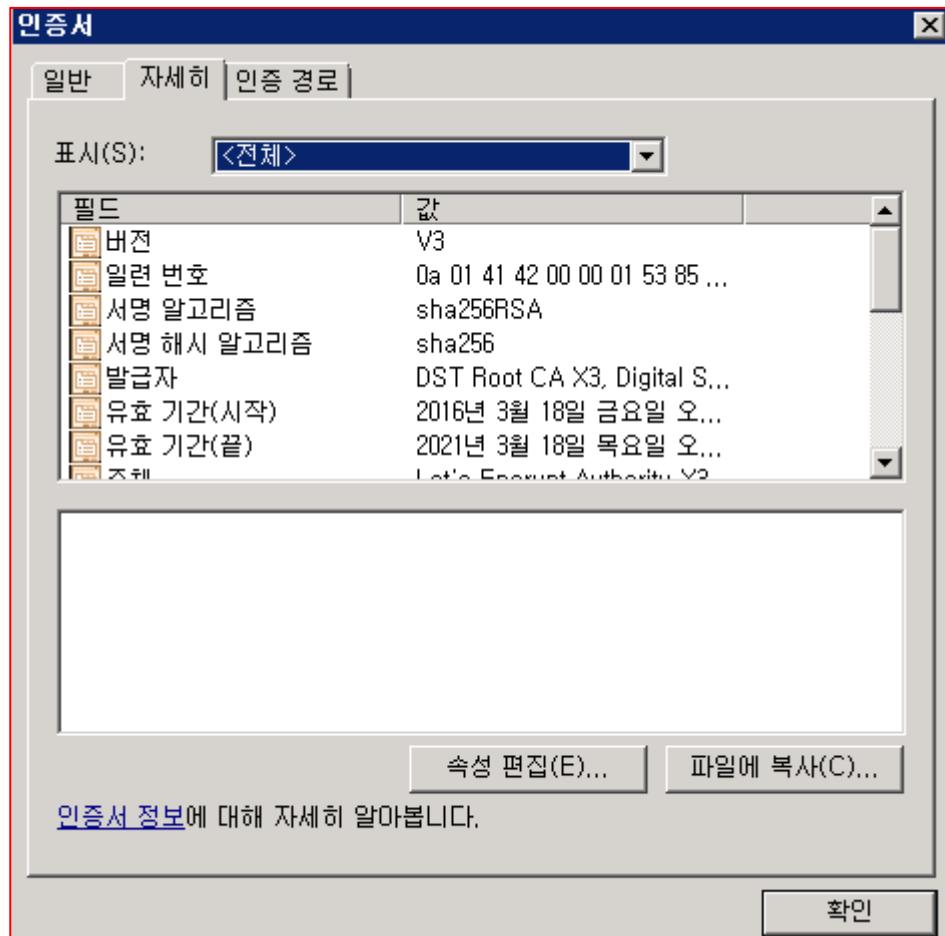
- /etc/letsencrypt/live/계정명.ncloudedu.com 디렉토리에 있는 파일들을 PC로 다운로드
- Cert1.pem 파일을 cert1.crt로 파일명 변경
- 더블클릭 후 인증 경로 확인



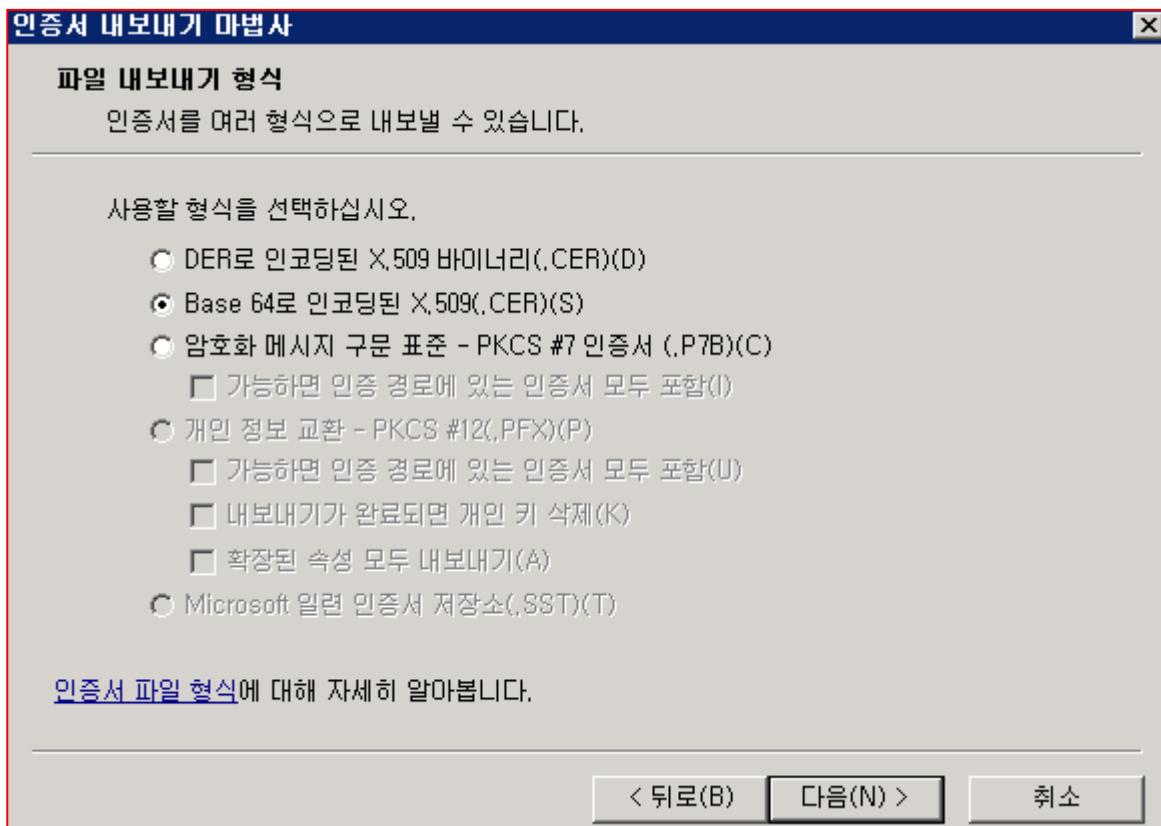
- 상위 인증기관 선택 후 인증서 보기 선택



- 인증서 화면에서 자세히 선택 후 파일에 복사 선택



- Base 64로 인코딩 선택 후 임시 파일에 저장



- DST Root CA X3 에 대해서도 동일 작업 수행
- 만들어진 파일 2개를 lnxsvr1 서버에 업로드
- Security > Certificate Manager > +외부 인증서 등을 선택
- Certificate 이름 : nce
- Private Key 입력
  - /etc/letsencrypt/archive/계정명.ncloudedu.com/privkey1.pem 파일을 열어서 내용 전체 복사

```
[root@krweb001 nce.ncloudedu.com]# cat privkey1.pem
-----BEGIN PRIVATE KEY-----
MIIEvglBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQDZSjY2rz9/JbgR
nuSjTxkM/+uPb4DtCUebo2Wi/yQhn/MWRICsSZtUfTNXXXlbe2vXERS5CqB49kTE
SjUTtct8EiFfuAiXpkUwwfGsbELvp2wnf/z1NsYOOjh83NMHHVcef01VneNIn4xn
0/YqqYuR+0lb0PPxOP1AFwoxDkVFK3d6JVfEnxZrZ2UmZ4PQwQ8+U+ITKXfF+d/D
NgYQQCBGC33p67Q7rqaWjmna+VaDXxU0aTziebx7mvEevflKq7pyplbKDRCr8WAB
Yd4Jm3Y2Nght8YCGJFYZuZaTTb8lgxo0n5khCUoI9a6DnyCMczN3W6c8e8ApdX9e
iUrMa/F1AgMBAECggEBAINDV1tGJcgzwI1+l1GX71gpnHrx+WrUkmMorLiXMXvW
nHrZ9cnDEIJJAvtInf3/X5QF3xSR9pTicAkT9bAtHfAa9W0noE+HC1s+59dK8/Naw
TDWxLff9rk1Jq37pbfc+KL46TkG1m4P7WUloJgnyyHtwBRCqSeCLCKhnhXTU4NII
cN5I88NFyTE6BoA5i/O8xukr9P1hIRLpjRFpzOGRv08aBMT+gAP+QUiZHn9oYQ9X
-----END PRIVATE KEY-----
```

```

yIXM/jRJHfeZVL976Mr3wLvCBq+ysPipdr3LZEICdEB7ecc3its4H4M+CdLXrzUu
/4oNfz38PeEL0tnqQtm9ak5VRiTxBa5hOwf8qXQ4dyECgYEA9cIMqGO65cv8eKoG
WsyC6/HfHQWLk4Ri3XTn5PXN+Vny9HmAA0nlbg+9yejUq6zdg56w7HGVOseSATDV
xhbH25E8DheKZQ+dp/lYpl7tCve08v4Q2iyH10r4nAVWWC91muK0utK9Bs+ueOCc
grPKLFxA/IgtlfE4mOMn+K/7d/0CgYEA4IHpxtOk88PsW9mDNtTWlAR+3YYjqb
k8R+o2Q0AJoK7i0KIDi7YcLXHiFN8e65SobrorgycYhckarznX2VUSnLcQrkGeoQ
zCpU83yxQ9gTho4684s1MEncwvTe+LebraPgt0U/wu5sJhvXmC0vDWDjiAoHmUP
WJeXghYe7NkCgYEAtJNJO6nRxUTh33DjLFB2m4xfJD5i0leB7zwVpxSkWF8qbUza
rQ/HLn1oLXBe1yYwtKOhToWAYuf+r9tGI7vW58zDN4M6DVe0t4/ZZQUQyG8GxUO9
89yljJorHs2ZCz8LA1kt+NgdQmTQxUQYGLqofYDlyeLubbKAp2q0kjQQ560CgYBU
262Nsbpzx3at5AQSnAYTNEtDbRXptw3wiQOmwgKNmzSv+2I1DBSOYI0AaiJsrUMC
g7ZEjNYtpaB36e5wRc8/4HjsiNxgNZhYxSKXLVDWDGW91QSbnr2xvnAtFV2pSkbw
A3tVnHx83aLkxyJlneAGldYe60W/p8rqP+TKHPs5eQKBgEkgI86L3St7gxinLVjC
3XRkJb968W/MN3XQzv0CT/WJC52qEYNi1bWAQloYgf7IDhvRaDazZx4Yqaj/99ZP
VZBCICmi1/F1OjQthW8IZCkvanaXgS+xf4qfN2OelcpC8lw6MS/aYSGeD8x7PINI
nJRlbq6ZOaLuZdZfq7DDyZi0
-----END PRIVATE KEY-----

```

[root@krweb001 nce.ncloudedu.com]#

- 해당 내용을 복사해서 입력
- Public Key Certificate 입력
  - /etc/letsencrypt/archive/계정명.ncloudedu.com/cert1.pem 파일을 열어서 내용 전체 복사

```

[root@krweb001 nce.ncloudedu.com]# cat cert1.pem
-----BEGIN CERTIFICATE-----
MIIFWjCCBEKgAwIBAgISA71+rhcDtmM2iCICK1DEWfBYMA0GCSqGSIb3DQEBCwUA
MEoxCzAJBgNVBAYTAIVTMRYwFAYDVQQKEw1MZXQncyBFbmNyeXB0MSMwIQYDVQ
QD
ExpMZXQncyBFbmNyeXB0IEF1dGhvcml0eSBYMzAeFw0yMDA4MTAwODQ3NDVaFw0y
MDExMDgwODQ3NDVaMBwxGjAYBgNVBAMTEW5jZS5uY2xvdWRIZHUuY29tMIIBIjAN
BhkqhkkiG9w0BAQEFAAOCAQ8AMIIIBCgKCAQEA2Uo2Nq8/fyW4EZ7ko08ZDP/rj2+A
7QIHm6Nlov8kIZ/zFkZQrEmbVH0zV11yG3tr1xEUuQqgePZExEo1E7XLfBlhX7gl
I6ZFML3xrGxC76dsJ3/89TbGDjo4fNzTBx1XHn9NVZ3jSJ+MZ9P2KqmLkftJW9Dz
8Tj9QBcKMQ5FRSt3eiVXXj8Wa2dIJmeD0MEPPiPiEy3xfnfwzYGEAgRgt96eu0
O66qFo5p2vlWg18VNGk84nm8e5rxHr35Squ6cqSGyg0Qq/FgAWHeCzt2NjYlbfGA
hiRWGbmWk02/CIMaNJ+ZIQIKjfWug58gjHMzd1unPHvAKXV/XoIKzGvxdQIDAQAB
o4ICZjCCAmIwDgYDVR0PAQH/BAQDAgWgMB0GA1UdJQQWMBQGCCsGAQUFBwMB
Bggr
BgEFBQcDAjAMBgNVHRMBAf8EAjAAMB0GA1UdDgQWBTrkf0NgNfuehYkJGPKsrcU
FFzbBDAfBgNVHSMEGDAWgBS0SmpjBH3duubRObemRWXv86jsotBvBgrBgEFBQcB
AQRjMGEwLgYIKwYBBQUHMAGGImh0dHA6Ly9vY3NwLmludC14My5sZXRzZW5jcnlw
dC5vcmcwLwYIKwYBBQUHMAKGi2h0dHA6Ly9jZXJ0LmludC14My5sZXRzZW5jcnlw

```

```
dC5vcmcvMBwGA1UdEQQVMBOCEW5jZS5uY2xdwWRIZHUuY29tMEwGA1UdiARFME
Mw
CAYGZ4EMAQIBMDcGCysGAQQBgt8TAQEBCgwgJgYIKwYBBQUHAgEWGmh0dHA6Ly
9j
cHMubGV0c2VuY3J5cHQu3JnMIIBAYKKwYBAHWeQIEAgSB9QSB8gDwAHYAsh4F
zluizYogTodm+Su5iiUgZ2va+nDnskITLe+LkF4AAAFz18TKIAABAMARzBFAiAI
2oCiYt001WchzwTqHeSwqx6LDkqa4tjBKP6xc9Lc+wIhAJ67RpMgEzBG0SiO5hKi
nAR5xoAiPS9aU6hhCoDxbTiWAHYAb1N2rDhwMRnYmQCKURX/dxUcEdkCwQApBo2y
CJo32RMAAAFz18TK4wAABAMARzBFAiEApz7h7fJy/xOVQPCrP9Jr0rM5t8q5SCAR
zicPbCnSIx8CICyAJ1fv5ajH0w5cAgQhMMRKM69gyJlrBsc7VwLPxQzpMA0GCSqG
Slb3DQEBCwUAA4IBAQAhvno18gkwXpWjxFodmhGo+2Si/JW0EoEiHvln2DeXFmdS
5NEB1hK0sbeDnBBJfT88rFwCrSRN/bAnSPNmKe0dYNqMvyT2Taebo7cHQBIpnA8n
Dweb6zEPqQjOd6evUQp3DTMm1wHjocEJzInRy3HhQaOCzVhyMPDpazuplwd02xG3
IrMCwSe8yKxvze5cLglEoAllCYrB6UDDGjlwjT2aLNHyIzkUWXdzijRLziicbcMK
MXH6ofWYLjK4dJEiRR29CYLsMiMoQ3jclGV+ia0JrsaO126z9B7FVjygfY8ESS+T
IsaJXAqq9HvwkN+x9nV4UBxiVNK14sh7GS0sOaO1
-----END CERTIFICATE-----
[root@krweb001 nce.ncloudedu.com]#
```

- 해당 내용을 복사해서 입력
- Certification Chain 에 내용 입력
  - 인증서에서 추출한 Chain을 Sub CA + Root CA 순서로 입력

```
[root@krweb001 nce.ncloudedu.com]# cat a.cer
-----BEGIN CERTIFICATE-----
MIIEkjCCA3ggAwIBAgIQCgFBQgAAAOfc2oLheynCDANBgkqhkiG9w0BAQsFADA/
MSQwlgyDVQQKExtEaWdpdGFsIFNpZ25hdHVyZSBUcnVzdCBDby4xFzAVBgNVBAMT
DkRTVCBSb290IENBIFgzMB4XDTE2MDMxNzE2NDA0NloXTDIxMDMxNzE2NDA0Nlow
SjELMAkGA1UEBhMCVVMxFjAUBgNVBAoTDUxIdCdzlEVuY3J5cHQxlzAhBgNVBAMT
GkxIdCdzlEVuY3J5cHQgQXV0aG9yaXR5IFgzMIIBljANBgkqhkiG9w0BAQEFAAO
CQ8AMIIBCgKCAQEAnNMM8FrILke3l03g7NoYzDq1zUmGSXhb418XCSL7e4S0EF
q6meNQhY7LEqxGiHC6PjdeTm86dicbp5gWAf15Gan/PQeGdxyGkOlZHP/uaZ6WA8
SMx+yk13EiSdRxta67nsHjcAHJyse6cF6s5K671B5TaYucv9bTyWaN8jKkKQDIZ0
Z8h/pZq4UmEUez9l6YKHy9v6Dlb2honzhT+Xhq+w3Brvaw2VFn3EK6BlspkENnWA
a6xK8xuQSXgvopZPKiAlKQTGdMDQMcf2PMTiVFrqoM7hD8bEf wzB/onkxEz0tNvjj
/Plzark5McWvxI0NHWQWM6r6hCm21AvA2H3DkwIDAQABo4IBfTCCAXkwEgYDVR0T
AQH/BAgwBgEB/wIBADAObgNVHQ8BAf8EBAMCAYYwf wYIKwYBBQUHAQEEczBxMDIG
CCsGAQUFBzABhiZodHRwOi8vaXNyZy50cnVzdGIkLm9jc3AuaWRlbnRydXN0LmNvb
TA7BgrBqEFBQcwAoYvaHR0cDovL2FwcHMuaWRlbnRydXN0LmNvbS9yb290cy9kc3Ryb
290Y2F4My5wN2MwHwYDVR0jBBgwFoAUxKexpHsscfrb4UuQdf/EFWCFiRAwVAYDVR0gBE0wSzAIbgZngQwBAgEwPwYLKwYBBAGC3xMBAQEwMDAuBgrBqEFBQcC
-----END CERTIFICATE-----
```

```
ARYiaHR0cDovL2Nwcy5yb290LXgxLmxldHNlbmNyeXB0Lm9yZzA8BgNVHR8ENTAz  
MDGgL6AthitodHRwOi8vY3JsLmlkZW50cnVzdC5jb20vRFNUUk9PVENBWDNDUkwu  
Y3JsMB0GA1UdDgQWBBSoSmpjBH3duubRObemRWXv86jsiTANBgkqhkiG9w0BAQsF  
AAOCAQEATPXEfNjWDjdGBX7CVW+dla5cEilaUcne8IkCJLxWh9KEik3JHRRHGJo  
uM2VcGfl96S8TihRzZvoroed6ti6WqEBmtzw3Wodatg+VyOeph4EYpr/1wXKtx8/  
wAplvJSwtmVi4MFU5aMqrSD6ea73Mj2tcMyo5jMd6jmeWUHK8so/joWUoHOugwu  
X4Po1QYz+3dszkDqMp4fkIxBwXRsW10KXzPMTZ+sOPAvexindmjkw8lGy+QsRIG  
PfZ+G6Z6h7mjem0Y+iWlkYcV4PIWL1iwBi8saCbGS5jN2p8M+X+Q7UNKEkROb3N6  
KOqkqm57TH2H3eDJAkSnh6/DNFu0Qg==  
-----END CERTIFICATE-----  
[root@krweb001 nce.ncloudedu.com]# cat b.cer  
-----BEGIN CERTIFICATE-----  
MIIDSjCCAjKgAwIBAgIQRK+wgNaj7qJMDmGLvhAazANBgkqhkiG9w0BAQUFADA/  
MSQwlgyDVQQKExtEaWdpdGFsfNpZ25hdHVyZSBUcnVzdCBDby4xFzAVBgNVBAMT  
DkRTVCBs290IENBIFgzMB4XDIAwMDkzMDIxMTIxOVoxDXTIxMDkzMDE0MDExNVow  
PzEkMCIGA1UEChMbRGlnaXRhbCBTaWduYXR1cmUgVHJ1c3QgQ28uMRcwFQYDVQQ  
D  
Ew5EU1QgUm9vdCBDQSBYMzCCASlwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggE  
B  
AN+v6ZdQCINXtMxiZfaQguzH0yxRMpb7NnDfcAwRgUi+DoM3ZJKuM/IUmTrE4O  
rz5ly2Xu/NMhD2XSktkj4zl93ewEnu1lcCjo6m67XMuegwGMoOifooUMM0RoOEq  
OLi5CjH9UL2AZd+3UWODyOKIYepLYYHsUmu5ouJLGiifSKOeDNoJjj4XLh7dIN9b  
xiqKqy69cK3FCxolkHRyxXtqqzTWMIIn/5WgTe1QLyNau7Fqckh49ZLOMxt+/yUFw  
7BZy1SbsOFU5Q9D8/RhcQPGX69Wam40utolucbY38EVAjqr2m7xPi71XAicPNaD  
aeQQmxkqtIX4+U9m5/wAI0CAwEAaNCMEAwDwYDVR0TAQH/BAUwAwEB/zAOBgN  
V  
HQ8BAf8EBAMCAQYwHQYDVR0OBBYEfMSnsaR7LHH62+FLkHX/xBVghYkQMA0GCSq  
G  
SIb3DQEBBQUAA4IBAQcJGiybFwBcqR7uKGY3Or+Dxz9LwwmgISBd49lZRNi+DT69  
ikugdB/OEIKcdBodfpga3csTS7MgROSR6cz8faXbauX+5v3gTt23ADq1cEmv8uXr  
AvHRAosZy5Q6XkjEGB5YGV8eAlrwDPGxrancWYaLbumR9YbK+rImM6pZW87ipxZz  
R8srzJmwN0jP41ZL9c8PDHlyh8bwRLtTcm1D9SZImJnt1ir/md2cXjbDaJWFBM5  
JDGFoqgCWjBH4d1QB7wCCZAA62RjYjsWvljEubSfZGL+T0yjWW06XyxV3bqxbYo  
Ob8VZRzl9neWagqNdvwYkQsEjgfbKbYK7p2CNTUQ  
-----END CERTIFICATE-----  
[root@krweb001 nce.ncloudedu.com]#
```

### 로드밸런서에 인증서 적용

- Networking > Load Balancer > nce-cm 선택 후 리스너 설정 변경 선택
- 리스너 추가

- 프로토콜 : HTTPS
  - 포트 : 443
  - SSL Certificate 선택 : nce
  - TLS 최소지원 버전 : TLS v1.0
  - Target Group : nce-cm-lab
- 웹 브라우저로 https 로 접근

## Lab 12 Guide

## SENS 발신번호 등록

- AI-Application > Simple & Easy Notification Service > SMS > Calling Number 선택
- 발신번호 등록 선택
- 핸드폰 인증 > 본인인증 선택
- 인증 진행

본인 휴대전화 인증

아래 약관에 모두 동의합니다.

<input checked="" type="checkbox"/> 인증시 개인정보 이용 <a href="#">보기</a>	<input checked="" type="checkbox"/> 인증시 고유식별정보 처리 <a href="#">보기</a>
<input checked="" type="checkbox"/> 통신사 이용약관 <a href="#">보기</a>	<input checked="" type="checkbox"/> 인증사 이용약관 <a href="#">보기</a>
<input checked="" type="checkbox"/> 개인정보 수집 <a href="#">보기</a>	

이름

내국인  남자 여자

생일 년(4자) 월 일

SKT  휴대전화번호

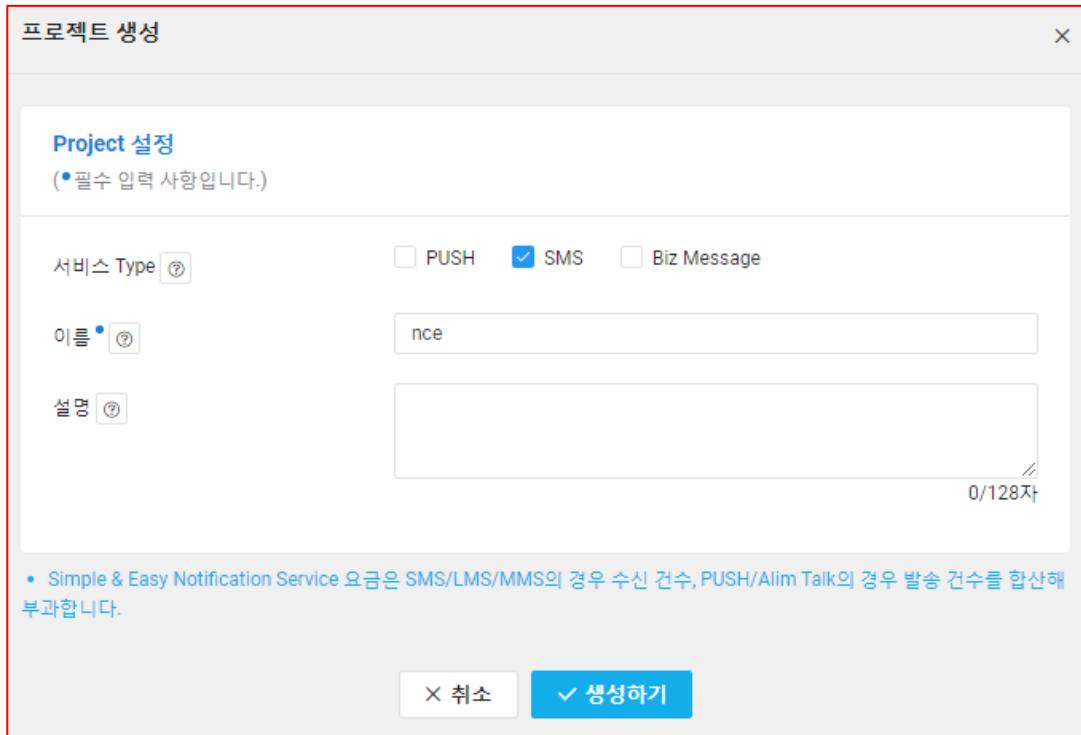
인증번호

인증비용은 네이버 클라우드 플랫폼에서 부담합니다.

확인

## 메시지 전송

- AI-Application > Simple & Easy Notification Service > Project 선택 후 +프로젝트 생성하기 선택
- 서비스 Type : SMS
- 이름 : nce
- 생성하기 선택



- AI-Application > Simple & Easy Notification Service > SMS > Message 선택
- 프로젝트명 : nce
- 프로젝트 명 우측의 'SMS'클릭
- 상단의 '발송하기' 클릭
  - Type : SMS
  - 발송내용 : 일반용
  - 국가코드 : 대한민국
  - 발신번호 : 본인 전화번호 선택
  - 수신번호 : 본인 전화번호 입력 후, 단건추가 클릭
  - 내용에 인사말 입력
  - 발송 설정 : 즉시 발송

### Geolocation 테스트

- Products & Services > Application > GeoLocation 선택 후 이용 신청 선택
- /var/www/html/key2.php 파일 수정

```
<?php  
  
/* SENS, NCP, Geolocation */  
  
$accessKey = "마이페이지 API Key ID";  
$secretKey = "마이페이지 Secret Key";  
  
/* Sens */  
$serviceID = "";  
$phone_no="";  
  
?>
```

- Lnxsvr1 서버 접속

## NAVER Cloud Platform EDU Demo Server

Welcome to NCP

Server name : Inxsvr1  
Client IP : 211.249.70.113



List

[Short URL](#)

[IP 확인 및 비교하기 \(LB를 거칠 때 실제 클라이언트 IP 확인\)](#)

[Geo Location](#)

[Clova Speech to Text](#)

[Clova Text to Speech](#)

[Clova Premium Voice](#)

[Clova Face Recognition](#)

[Papago NMT](#)

[Cloud DB](#)

[Admin Page](#)

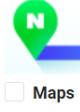
- Geo Location 선택 후 IP 입력하여 확인

### 단축 URL 테스트

- Products & Services > AI-NAVER API > AI NAVER API 선택 후 +Application 등록 선택
- Application 이름 : nce
- Service 선택
- Clova : Clova Voice(Premium)
- Papago translation : All
- Machine Learning : All
- NAVER : All

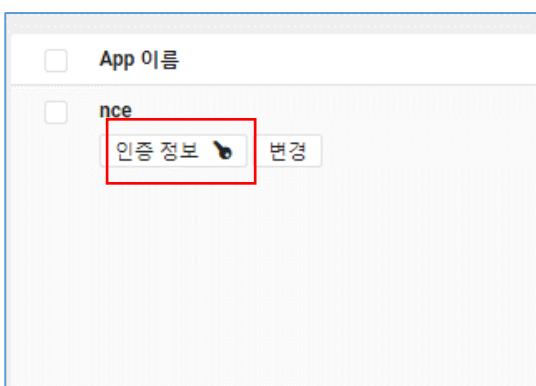
**Service 선택**  
Application에서 이용할 Service 를 선택하세요

---

 <input type="checkbox"/> CLOVA	<input checked="" type="checkbox"/> CLOVA Speech Recognition (CSR) <input checked="" type="checkbox"/> CLOVA Face Recognition (CFR) <input type="checkbox"/> CLOVA Voice - Premium
 <input checked="" type="checkbox"/> Papago Translation	<input checked="" type="checkbox"/> Papago Text Translation <input checked="" type="checkbox"/> Papago Website Translation <input checked="" type="checkbox"/> Papago Doc Translation <input checked="" type="checkbox"/> Papago Language Detection <input checked="" type="checkbox"/> Papago Korean Name Romanizer
 <input checked="" type="checkbox"/> Machine Learning	<input checked="" type="checkbox"/> Pose Estimation <input checked="" type="checkbox"/> Object Detection
 <input type="checkbox"/> Maps	<input type="checkbox"/> Web Dynamic Map <input type="checkbox"/> Mobile Dynamic Map <input type="checkbox"/> Static Map <input type="checkbox"/> Directions 5 <input type="checkbox"/> Directions 15 <input type="checkbox"/> Geocoding <input type="checkbox"/> Reverse Geocoding
 <input checked="" type="checkbox"/> NAVER	<input checked="" type="checkbox"/> CAPTCHA (Image) <input checked="" type="checkbox"/> CAPTCHA (Audio) <input checked="" type="checkbox"/> nShortURL <input checked="" type="checkbox"/> Search Trend

---

- 서비스 환경 등록 > Web 서비스 : server1.도메인명
- 등록 선택
- 인증정보 클릭



- 다음 정보 기록



- /var/www/html/key.php 파일 수정

```
<?php  
  
/* API Key */  
  
$client_id = "클라이언트ID";  
  
$client_secret = "클라이언트Secret";  
  
?>
```

- Lnxsvr1 서버 접속

## NAVER Cloud Platform EDU Demo Server

Welcome to NCP

Server name : Inxsvr1  
Client IP : 211.249.70.113



List

[Short URL](#)

[IP 확인 및 비교하기 \(LB를 거칠 때 실제 클라이언트 IP 확인\)](#)

[Geo Location](#)

[Clova Speech to Text](#)

[Clova Text to Speech](#)

[Clova Premium Voice](#)

[Clova Face Recognition](#)

[Papago NMT](#)

[Cloud DB](#)

[Admin Page](#)

- 
- Short URL 선택 후 www.naver.com 입력하여 확인

### Lab 13 Guide(demo)

#### Curl을 이용한 CPV 구현

- Curl을 이용하여 다음과 같이 명령어 수행
- Lnxsvr1 서버에 접속
- 다음 명령어 수행

```
curl -i -X POST §§
-H "Content-Type:application/x-www-form-urlencoded" §§
-H "X-NCP-APIGW-API-KEY-ID:{애플리케이션 등록 시 발급받은 client id값}" §§
-H "X-NCP-APIGW-API-KEY:{애플리케이션 등록 시 발급받은 client secret값}" §§
-d 'speaker=nara&text=만나서 반갑습니다
&volume=0&speed=0&pitch=0&format=mp3' §§
'https://naveropenapi.apigw.ntruss.com/tts-premium/v1/tts' > a.mp3
```

- a.mp3 파일 확인

#### 웹을 이용한 CPV 구현

- 웹 브라우저로 Lnxsvr1 서버에 접속

NAVER Cloud Platform EDU Demo Server

Welcome to NCP

Server name : Lnxsvr1  
Client IP : 211.249.70.113

NAVER CLOUD PLATFORM

List

[Short URL](#)  
[Geo Location](#)  
[Clova Speech to Text](#)  
[Clova Text to Speech](#)  
[Clova Premium Voice](#)  
[Clova Face Recognition](#)  
[Papago NMT](#)

[Cloud DB](#)

[Admin Page](#)

- Clova Premium Voice 선택
- 음색, 속도, 톤 선택 후 텍스트 입력

## NAVER Cloud Platform Premium Voice DEMO

한국어, 여성 음색 ▾

정상 속도 ▾

기본톤 ▾

변환할 텍스트

안녕하세요.

Submit

- 변환 내용 확인
- Mp3 파일 다운로드

### 웹을 이용한 CSR 구현

- 웹 브라우저로 Lnxsvr1 서버에 접속

## NAVER Cloud Platform EDU Demo Server

Welcome to NCP

Server name : lnxsvr1  
Client IP : 211.249.70.113



List

[Short URL](#)

[IP 확인 및 비교하기 \(LB를 거칠 때 실제 클라이언트 IP 확인\)](#)

[Geo Location](#)

[Clova Speech to Text](#)

[Clova Text to Speech](#)

[Clova Premium Voice](#)

[Clova Face Recognition](#)

[Papago NMT](#)

[Cloud DB](#)

[Admin Page](#)

- Clova Speech To Text 선택
- CPV로 만든 mp3 파일을 업로드
- 결과 확인

파일명 : tts.mp3

Speech to Text 결과 : 안녕하세요 네이버 클라우드 플랫폼에 정 락 수 있니



## Lab 14 Guide(demo)

**Curl을 이용한 Papago Translation 구현**

- Curl을 이용하여 다음과 같이 명령어 수행
- Lnxsvr1 서버에 접속
- 다음 명령어 수행

```
curl -i -X POST @@
-H "Content-Type:application/x-www-form-urlencoded" @@
-H "X-NCP-APIGW-API-KEY-ID:{client ID}" @@
-H "X-NCP-APIGW-API-KEY:{Secret}" @@
-d "source=ko" @@
-d "target=en" @@
-d "text=안녕하세요" @@
'https://naveropenapi.apigw.ntruss.com/nmt/v1/translation'
```

- Windows PC의 경우

```
curl -i -X POST ^
-H "Content-Type:application/x-www-form-urlencoded" ^
-H "X-NCP-APIGW-API-KEY-ID:v7gk4qdsh8" ^
-H "X-NCP-APIGW-API-KEY:33flclsLnf6lVV4Jtly1Oo3c23l8fB82M5ZKKwcP" ^
-d "source=ko" ^
-d "target=en" ^
-d "text=안녕하세요" ^
'https://naveropenapi.apigw.ntruss.com/nmt/v1/translation'
```

- Json 형식으로 요청 가능하며 이 경우 보다 간결한 요청을 보낼 수 있다.

```
curl -i -X POST @@
-H "X-NCP-APIGW-API-KEY-ID:{앱 등록 시 발급받은 Client ID}" @@
-H "X-NCP-APIGW-API-KEY:{앱 등록 시 발급 받은 Client Secret}" @@
-H "Content-Type:application/json" @@
-d @@
'{
  "source": "{원본 언어 코드}",
  "target": "{번역 결과 언어 코드}",
  "text": "{번역할 text}"
}' @@
'https://naveropenapi.apigw.ntruss.com/nmt/v1/translation'
```

- 응답은 어떤 형식이던 Json 형식으로 응답

```
{"message": {"@type": "response", "@service": "naverservice.nmt.proxy", "@version": "1.0.0", "result": {"srcLangType": "ko", "tarLangType": "en", "translatedText": "Hi."}}}
```

### 웹을 이용한 CSR 구현

- 웹 브라우저로 lnxsvr1 서버에 접속

- Papago NMT 선택
- 원번언어와 번역할 언어 선택
- 텍스트 입력
- 결과 확인

변환할 텍스트 : 안녕하세요. 오늘은 날씨가 좋네요.  
번역한 문장 :Hello. The weather is nice today.

-

## Lab 15 Guide

액션을 만들고 트리거를 통해 액션을 실행시키는 방식을 알아봅니다.

액션은 독자적인 특정 액션을 실행시킬 수 있지만, 트리거에 파라미터를 넣어 액션을 호출할 시, 다른 방식으로도 실행이 가능합니다.

액션을 단독으로 실행시킬 때와 트리거에 파라미터를 넣어 액션을 실행시킬 때, 결과 값이 어떻게 달라지는지에 집중해서 확인해보고, 외부와 통신할 수 있는 URL 주소를 통해서도 트리거를 작동시키는 것 까지 함께 살펴봅니다. 마지막으로는 액션에 코드가 아닌 여러 코드파일로 이루어진 압축파일을 이용하여 네이버 클라우드 플랫폼 내의 다른 상품과 연동하여 사용하는 방법에 대해서도 실습해봅니다.

### 테스트 환경 설정

- Action은 Private Subnet 서버에서 실행할 수 있습니다.
- 테스트 용이므로 lab1-vpc-redis-subnet 을 임시로 이용합니다
- Route Table 설정에 외부 통신은 NAT를 통하여도록 설정합니다.

### 내 패키지 생성

- Cloud Functions 탭 > Action 선택, +패키지 생성 클릭
- 패키지 생성
- 패키지 이름 : hello

### 트리거 생성

- Cloud function > +트리거 생성 클릭
- 트리거 종류 : Basic
- 트리거 이름 : lab-hello
- 외부 연결 주소 생성
- Product : 새로만들기 > lab-hello
- API : 새로만들기 > lab-hello
- Stage : 새로만들기 > lab-hello
- 인증 : none 선택
- 저장하고 액션 연결하기 버튼 클릭 후 저장 버튼 클릭

### 액션 생성

- +액션 생성 클릭
- 트리거 종류: Basic 선택
- 이름: lab-hello
- 다음 버튼 클릭
- 패키지 :hello 선택
- 이름 :helloNCP
- 소스코드 언어 :nodejs:8
- 타입 : 코드
- 코드 :

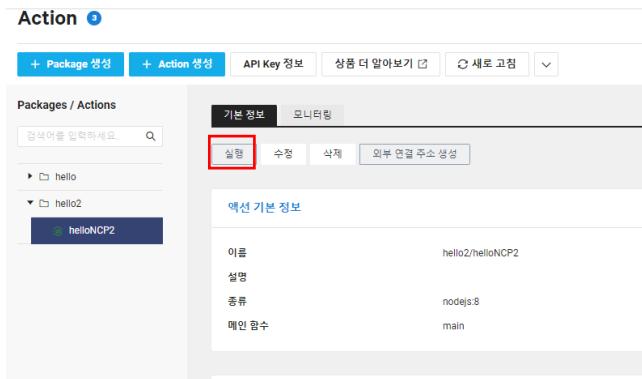
```
function main(params) {  
    return {payload: 'Hello, ' + params.name + ' from ' + params.place + '?'};  
}
```

- VPC :lab1-vpc 선택
- Subnet :lab1-vpc 내 private subnet 선택
- 옵션 설정 :Defualt선택
- 디폴트 파라미터:  

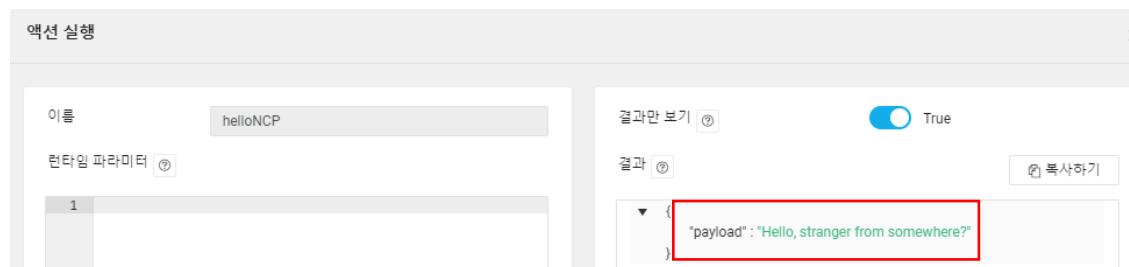
```
{"name":"stranger","place":"somewhere"}
```
- 생성 버튼 클릭

### 액션 단일 동작

- Cloud Functions 탭 > Action > hello 패키지 > helloNCP 액션 선택
- ‘실행’ 버튼 클릭



- 결과만 보기를 True로 변경
- 실행 버튼 클릭
- 아래와 같이 Hello, stranger from somewhere? 가 출력되는지 확인



### 트리거로 액션 동작

- Cloud Functions 탭 > Trigger > Basic > lab-hello 트리거 선택
- 트리거 실행 버튼 클릭
- 런타임 파라미터 :

```
{
    "name" : "NCP",
    "place" : "GangNam"
}
```

- 결과만 보기를 True로 변경
- 실행 버튼 클릭
- Activation ID 확인
- Cloud Functions 탭 > Action > hello 패키지 > helloNCP 액션 선택
- 모니터링 탭 선택

The screenshot shows the 'Monitoring' tab highlighted in red. The interface includes a sidebar with a 'lab' folder containing a 'helloNCP' item, and a main panel titled '실행결과' (Execution Result) displaying monitoring data.

- Activation ID 결과값의 자세히 보기 클릭
- 아래와 같이 Hello, NCP from GangNam? 이 출력되는지 확인

이름	lab/helloNCP	ID	5c50d337c209460190d337c2096 60101
시작시간	2020-07-14 17:20:21 (UTC+09:00)	종료시간	2020-07-14 17:20:21 (UTC+09:00)
실행시간	1ms	메모리	256MB
상태	<span style="color: green;">success</span>		

**결과**  
액션의 실행 결과가 보입니다.  
({"payload":"Hello, NCP from GangNam?"})

### 외부 URL을 통해 호출

- Cloud function > Action > hello 패키지 > helloNCP 클릭 > 외부 연결 주소 생성 버튼 클릭
  - Product : lab-hello
  - API : lab-hello
  - Stage : lab-hello
  - 인증 : None 선택 후, 하단의 완료 버튼 클릭
- 서버에서 외부 연결 호출 URL로 POST 요청을 전송
- 명령어 :

```
curl -X POST <URL주소> -H "Content-Type:application/json" -d
'{"name":"NCP", "place":"Seoul"}'
```

```
[root@cicd-org ~]# curl -X POST https://9flyj7yr2k.apigw.ntruss.com/labtrigger/labtrigger/cI3Uiwc3ga
{"activationId":"7b988bbc668c4720988bbc668c672021"} [root@cicd-org ~]#
```

**액션에 압축파일을 업로드하여 Outbound Mailer를 통해 메일 보내기**

- 아래 파일을 다운로드 받은 후 각 파일의 이메일 주소 및 API 인증키를 수정  
다운로드:  
[https://github.com/NaverCloudPlatform/outbound\\_mailer\\_python\\_sample](https://github.com/NaverCloudPlatform/outbound_mailer_python_sample)
- 아래 부분 중 하이라이트 된 부분을 본인 계정의 API 인증키 및 메일주소로 치환
  1. \_\_main\_\_.py

```
def process_mail():

    current_date = datetime.today().strftime("%Y-%m-%d")

    mail_contents = RssNewsParser().news_rss_parser('NCP+네이버클라우드플랫폼')

    mail_info = {
        "senderAddress": "no_reply@company.com",
        "title": "네이버클라우드 플랫폼 뉴스 - ${current_date}",
        "body": mail_contents,
        "recipients": [
            {
                "address": "ncloudedu@naver.com",
                "name": "홍길동",
                "type": "R",
                "parameters": {
                    "current_date": current_date
                }
            }
        ],
        "individual": True,
```

```

    "advertising": False
}

```

## 2. base\_auth\_info.py

```

access_key = 'AccessKey 값을 입력하세요'
# NCP Access secret
access_secret = ' AccessSecret 정보값을 입력하세요 '

```

## 3. mail\_sender

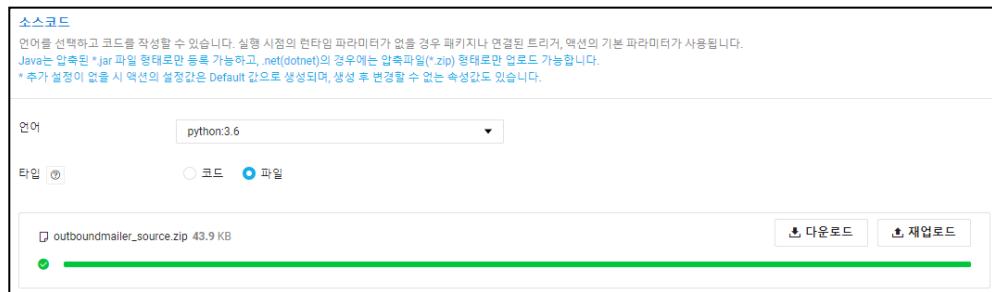
```

"recipients": [
{
    "address": "ncloudedu@naver.com",
    "name": "홍길동",
    "type": "R",
    "parameters": {
        "customer_name": "홍길동",
        "BEFORE_GRADE": "SILVER",
        "AFTER_GRADE": "GOLD
    }
},
],

```

- 수정이 완료되면 5개의 파일을 1개의 압축파일(zip)로 생성
- +액션 생성 버튼 클릭
- 트리거 종류: \*\*트리거 설정 없이 액션 만들기\*\*
- 패키지: hello
- 이름: outboundmailerTest

- 언어: python3.6
- 타입: 파일
- 위에서 만든 압축파일 업로드



- VPC 연결정보에서는 lab1-vpc-redis-subnet 가 생성된 VPC를 선택, Subnet은 lab1-vpc-redis-subnet 선택
- 옵션 설정에서 웹 액션 설정을 True로 변경 후 생성 버튼 클릭
- 생성 완료 후, ‘외부 연결 주소 생성’ 버튼 클릭



- Product, API, Stage를 선택, 만약 선택할 내용이 없는 경우 아래와 같이 새로 생성
  - Product: outboundmailer
  - API: outboundmailer
  - Stage: dev1
  - 인증: None
- 생성된 URL 주소를 복사 후 type 부분을 json으로 수정

-예시:

<https://XXmlgwad.apigw.ntruss.com/outboundmailer/dev1/Vr1la1wcCE/json>

- Curl이나 Postman을 이용하여 POST 형식으로 URL 호출

The screenshot shows the Postman interface with the following details:

- URL: <https://89hhmlgwad.apigw.ntruss.com/outboundmailer/dev1/Vr1la1wcCE/json>
- Method: POST
- Headers: (7)
- Body: JSON
- Response Status: 200 OK (1488 ms, 619 B)
- Body Content (Pretty):
 

```

1 {
2   "result": 201
3 }
```

- 메일이 정상적으로 송신됐는지 확인

The screenshot shows an email inbox with the following details:

- Subject: ☆ 네이버 클라우드 플랫폼 뉴스 - 2021-07-07
- Date: 2021. 7. 7. (수) 19:51
- From: 보낸 사람: VIP <no\_reply@company.com> 주소록에 추가 | 메시지 | 약속조대
- To: 받는 사람: 흥길동<[REDACTED]@navercorp.com>
- Message Preview (Body):
 

네이버 클라우드 플랫폼 뉴스

[아이티데일리] [네이버 클라우드-삼성엔지니어링, 건설기술 혁신 나선다](#)  
 [IT비즈뉴스] [네이버 클라우드, 삼성생명 DT프로젝트 합류...금융 신사업 모델개발 '맞손'](#)  
 [아이티데일리] [네이버 클라우드-삼성생명, 디지털 금융 혁신 '맞손'](#)  
 [매일경제] [ICT 직원들은 열공 중...'클라우드 자격증' 열풍](#)  
 [아이티데일리] [메타넷디플랫폼 컨소시엄, 'NH농협은행 퍼블릭 클라우드 표준 사업자' 선정](#)  
 [아이티데일리] [\[클라우드 MSP 시장②\] 문제는 낮은 수익률...새로운 들파구 모색해야](#)  
 [쿠키뉴스] [NH농협은행, '아마존·네이버·오라클' 클라우드 사업자 선정](#)  
 [팍스경제TV] [NH농협은행, 'NH퍼블릭 클라우드 표준사업자' 선정](#)  
 [아시아투데이] [NH농협은행, NH퍼블릭 클라우드 표준사업자 선정](#)  
 [오늘경제] [NH농협은행, 'NH퍼블릭 클라우드 표준사업자' 선정](#)

## Lab 16 Guide

소스 저장소인 SourceCommit 리파지토리 생성법과 사용 방법에 대해 알아본 후, 서버에서 SourceCommit 리파지토리에 원격으로 연결하여 소스를 저장/업데이트 할 수 있는 방법에 대해 알아봅니다.

SourceCommit 을 이용하는 계정은 Sub Account라는 가정하에, Sub Account를 먼저 생성하고 특정 계정에 특정 권한을 부여하는 방법등에 대해 먼저 실습을 진행합니다.

### Lnxsrv1,Lnxsvr2 서버에 아래 스크립트 반영 필요

```
yum install -y tomcat
systemctl enable tomcat
systemctl start tomcat
yum install -y java-11-openjdk-devel
mkdir -p /var/lib/tomcat/webapps/ROOT/WEB-INF/classes
```

### 서브계정 생성

- Management > Sub Account 클릭
- Dashboard 에서 Sub Account를 위한 접속 페이지 생성
- Sub Accounts 에서 +서브 계정 생성 클릭
- 로그인 아이디 : student
- 사용자 이름 : student
- API 접근 허용
- 로그인 비밀번호 : ncloud<오늘날짜>!      Ex) 7월 20일인 경우, ncloud0720!
- 생성이 완료된 student 계정을 클릭

로그인 아이디	사용자 이름	Console 접근	API 접근	상태
student	student	✓	✓	● 사용 중
dev01	dev01	✓		● 사용 중

- 하단 정책 탭에서 추가 버튼 클릭
- SOURCE\_COMMIT, BUID, DEPLOY, PIPELINE 권한 선택 후 추가 버튼 클릭

### SourceCommit 리파지토리 생성

- Dev Tools > SourceCommit 클릭
- + 리파지토리 생성 버튼 클릭

- 리파지토리 이름 : lab-repo
- 하단의 다음 버튼 클릭
- File safer 연동 안함
- 하단의 생성 버튼 클릭

#### Sub Account 접속 및 HTTPS 접근용 Git Client 설정

- 다른 브라우저를 하나 더 띄워, Sub Account 접속 페이지로 들어간 후, student 계정으로 로그인
- lab-repo 리파지토리 선택 후, GIT 계정/GIT SSH 설정 버튼 클릭
- Git Client 패스워드를 'ncp!@#123' 으로 설정 후 적용 버튼 클릭

#### Git Client SSH 접근용 자격증명 발급

- Lnxsvr1 서버에 접속
- gitlab 이란 이름의 디렉토리 생성  

```
$ mkdir ~/gitlab
```

```
$ cd ~/gitlab
```
- ssh-keygen 명령어 입력

```
$ ssh-keygen -t rsa -C '이메일'
```

Enter file in which to save the key (/root/.ssh/id\_rsa):id\_rsa\_lab

Enter passphrase (empty for no passphrase):ncp!@#123

Enter same passphrase again:ncp!@#123

- 생성된 public rsa key 결과값을 복사

```
$ cat /root/gitlab/id_rsa_lab.pub
```

<결과값/ ssh-rsa ~>

- SourceCommit > Git 계정/Git SSH 설정 클릭 > GIT SSH 설정에 해당 값 입력 후 등록 클릭

## SSH 퍼블릭 키 등록

```
ssh-rsa
AAAAB3NzaC1yc2EAAAQABAAQCO8o27ez6EJ0mVkcmy62+INdybGopiVWLUAWh/xqGLTPmBDpYqmej
9Lg668pJi+ruvz2AYtkhUcqKLIOZkiDlZEoc4fTXotsjOTZCGMionC18E4dyUh2ZnLMenlo2F2L+4b0IJH/1zS/+IR6BsUx
lsubWiPVbCbxWiWtiMailb4V74/nBvm1B+TEQiRri93AgPmuIR82BOJBYoZOpI3QWeUYEZtixckB0tQ6Zxon6xz+MVAf+
Z+a7xmAScIBj8qw13pYLWqTaH31qe3WMulqD8fUIT7mcg3E4kvznrKGH07Nd5WcViygMfhQ0aSgwhOuFo9FCz72M
W4UCIWiY5 mignon.kim@navercorp.com
```

등록

- lnxsvr1 서버에서 ssh key 를 추가 후 저장

```
$ eval $(ssh-agent)
```

```
$ ssh-add ~/gitlab/id_rsa_lab
```

```
Enter passphrase for /root/gitlab/id_rsa_lab:ncp!@#123
```

```
$ ssh-add -l
```

- ssh config 파일을 생성하여, private key 파일 경로 저장

```
$ mkdir ~/.ssh
```

```
$ vi ~/.ssh/config
```

```
Host devtools.ncloud.com
```

User <SSH 키>

```
IdentityFile ~/gitlab/id_rsa_lab
```

## 등록된 SSH 퍼블릭 키

SSH 키	업로드 날짜	상태	
입력할 SSH 키	2020-12-15 20:24 (UTC+09:00)	활성화 비활성화	삭제

## 서버에 로컬 리파지토리 생성 후 원격 리파지토리 업데이트

- lnxsvr1 서버에 접속
  - gitlab 아래 sourcecommit 디렉터리 생성
- ```
$ mkdir ~/gitlab/sourcecommit
```
- ```
$ cd ~/gitlab/sourcecommit
```
- 로컬 리파지토리 생성 및 사용자 정보 설정

```
$ git init
```

```
$ git config --global user.name "사용자 이름"
```

```
$ git config --global user.email "사용자 이메일"
```

```
$ touch readme.txt
```

- 로컬 리파지토리에 readme.txt 파일 추가 후 커밋

```
$ git add readme.txt
```

```
$ git commit -m "First Commit"
```

```
$ git status
```

- 원격 저장소(Sourcecommit) 등록 후 확인

```
$ git remote add origin <리파지토리 URL>
```

```
$ git remote -v
```

- 원격 저장소의 master 브랜치를 가져온 후, 원격 저장소에 업데이트

- \$ git pull --rebase origin master

- \$ git push origin master

- SourceCommit에 들어가 readme.txt 파일이 추가되었는지 확인

## Lab 17 Guide

앞서 로컬 리포지토리에서 SourceCommit 리파지토리와 어떻게 연동하여 소스 업데이트를 할 수 있는지 살펴보았습니다. 이번 Lab에서는 업데이트한 소스를 네이버 클라우드 상에서 어떻게 빌드할 수 있는지 알아봅니다.

빌드하기 전, 빌드 결과물을 저장할 수 있는 Object Storage 생성부터 시작합니다.

### Object Storage 생성

- 다시 마스터 계정으로 돌아와 Storage > Object Storage 에서 + 버킷 생성 클릭
- 버킷 이름 : gitlab<생년월일>
- 나머지는 default선택으로 두고 버킷 생성
- 버킷을 선택 후, 새폴더 버튼 클릭
- 폴더명 : sourcebuild

### 빌드용 파일 업로드

- HelloServlet.java 파일을 만들어 원격저장소에 push

```
$ vi ~/gitlab/sourcecommit/HelloServlet.java
```

```
import java.io.*;  
  
import javax.servlet.*;  
  
import javax.servlet.http.*;  
  
  
public class HelloServlet extends HttpServlet {  
  
    public void doGet(HttpServletRequest req, HttpServletResponse res)  
throws ServletException, IOException {  
  
        PrintWriter out = res.getWriter();  
  
        out.println("Hello, NCP!");  
  
    }  
  
}
```

```
$ git add HelloServlet.java  
$ git commit -m "HelloServlet.java added"  
$ git push origin master
```

### 빌드 프로젝트 생성

- Dev Tools > SourceBuild > + 빌드 프로젝트 생성 클릭
- 빌드 프로젝트 이름 : gitlab
- 빌드 대상 : Sourcecommit
- 빌드 대상 리파지토리 : lab-repo
- 브랜치 : master
- 빌드 환경 이미지 : SourceBuild에서 관리되는 이미지
- 운영 체제 : ubuntu 16.04
- 빌드 런타임 : java
- 빌드 런타임 버전 : java8 버전
- 타임 아웃 : 60초
- 컴퓨팅 유형 : 2vCpu 4GB 메모리
- 하단의 ‘다음’버튼 클릭
- 빌드 명령어 :
  1. 빌드 전 명령어 :

```
apt-get -y update  
apt-get install -y tomcat7
```

2. 빌드 명령어

```
javac -classpath /usr/share/tomcat7/lib/servlet-api.jar HelloServlet.java
```

- 하단의 다음 버튼 클릭
- 빌드 결과물 : 결과물 저장

- 빌드 결과물 경로 : ./HelloServlet.class
- 업로드 할 Object Storage : 앞에서 생성한 버킷 선택
- Object Storage 폴더 경로 : sourcebuild
- 저장될 파일 이름 : HelloServlet
- 나머지는 Default 설정 그대로 남겨두고 확인
- 생성 완료 후, gitlab 빌드를 클릭 후, 상단의 ‘빌드로 이동’클릭
- 우측 상단의 ‘빌드 시작하기’ 버튼 클릭



작업 결과가 Success 인지 확인

## Lab 18 Guide

소스 코드 빌드가 완료되면, 실제로 서버에 배포해야 합니다. 이번 Lab에서는 SourceDeploy를 통해 쉽고 편하게 빌드 이미지를 배포하는 방법에 대해 실습해봅니다. 마지막으로 SourcePipeline을 통해 빌드와 배포 자동화를 어떻게 할 수 있는지에 대해 알아봅니다.

### Agent 설치

- ncloud.com 메인 > 마이페이지 > 계정 관리 > 인증키 관리 > 신규 API 인증키 생성 (없을 경우)

Access key ID와 Secret Key ID를 메모장에 저장 (Access Key의 상태가 사용 중 이어야 함)

- accesskey와 secretkey 을 개인 API 인증키로 치환하여, 아래 명령어 수행

```
$ echo $'NCP_ACCESS_KEY=accesskey\nNCP_SECRET_KEY=secretkey' >
/opt/NCP_AUTH_KEY
```

```
$ wget https://sourcedeploy-
agent.apigw.ntruss.com/agent/vpc/download/install
```

```
$ chmod 755 install
```

```
$ ./install
```

```
$ rm -rf install
```

```
$ service sourcedeploy start
```

```
$ service sourcedeploy status
```

### 배포 프로젝트 생성

- Dev Tools > SourceDeploy > + 배포 프로젝트 생성 클릭
- 프로젝트 이름 : gitlab
- dev stage : 설정
- 적용 서버 : lnxsvr1 선택
- 배포 프로젝트 생성 버튼 클릭
- gitlab 프로젝트를 선택 후, 배포 시나리오 부분에서 ‘생성’ 버튼 클릭



- 배포 시나리오 이름 : test
- 배포 전략 : 기본
- 배포 과정 : 순차배포
- 배포 파일 위치 : Source Build
- 빌드 프로젝트 선택 : gitlab
- 소스 파일 배포 경로 : /.
- 배포 경로 : /var/lib/tomcat/webapps/ROOT/WEB-INF/classes 입력 후 추가 버튼 클릭

배포 전 실행		실행 계정		실행 명령		명령 추가
		명령어를 수행할 서버 계정입니다.		서버에서 실행할 명령어입니다.		+ 추가
		데이터가 없습니다.				

파일 배포		소스 파일 경로		배포 경로		파일 배포 추가
		/.		/var/lib/tomcat/webapps/ROOT/WEB-INF/classes		+ 추가

- 하단의 다음 클릭 후, 배포 시나리오 생성 버튼 클릭

### 배포 시나리오 실행

- gitlab 프로젝트 선택 후, 상단의 배포로 이동 버튼 클릭



- test 시나리오를 클릭 후, 배포 시작하기 클릭
- 배포가 끝나고 시나리오의 상태가 배포 완료 인지 확인
- Lnxsvr1 서버의 /var/lib/tomcat/webapps/ROOT/WEB-INF/classes 디렉터리에 HelloServlet 클래스 파일이 배포된 것을 확인

### 웹페이지 접속

- 서버에서 web.xml 파일을 열고, </description> 밑에 url 정보 추가

```
$ cd /var/lib/tomcat/webapps/ROOT/WEB-INF
```

```
$ wget https://kr.object.ncloudstorage.com/nce/web.xml
```

\$ vi /var/lib/tomcat/webapps/ROOT/WEB-INF/web.xml를 통해 아래 정보가 /description 하단에 있는지 확인

```
<servlet>
    <servlet-name>HelloNCP</servlet-name>
    <servlet-class>HelloServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>HelloNCP</servlet-name>
    <url-pattern>/hello</url-pattern>
</servlet-mapping>
```

- tomcat 서비스 재시작

```
$ systemctl restart tomcat
```

- 웹브라우저를 열고 <lnxsvr1 서버IP>:8080/hello 페이지에 접속하여 Hello, NCP! 문구가 보이는지 확인

## 파이프라인 생성

- lnxsvr2 서버에 Hello, 이름! 를 출력하기 위한 사전 작업을 진행
- lnxsvr2 서버에서 아래 명령어 실행 (deploy를 위한 사전 작업 및 web.xml 파일 다운로드)

```
$ wget https://me2.do/xiXnMsHP -O /root/init.sh && chmod 755 init.sh && sed -i -e 's/\#\!/ init.sh && bash init.sh'
```

- lnxsvr1 서버에서 HelloServlet.java 출력 문구 수정 후 리파지토리에 변경내용 업데이트

```
$ vi ~/gitlab/sourcecommit/HelloServlet.java
```

```
($ cd /gitlab/sourcecommit)
```

문구 수정 >> Hello, 수강생 성함!

```
$ git add HelloServlet.java
```

```
$ git commit -m "HelloServlet.java Revised"
```

```
$ git push origin master
```

- SourceCommit > lab-repo 에 들어가 Code 및 Commit 내역 업데이트 확인
- SourceDeploy > gitlab 선택 > 배포환경 > 설정 변경 클릭 > 적용 서버를 lnxsvr2 로 변경
- Dev Tools > SourcePipeline > + 파이프라인 생성
- 파이프라인 이름 : gitlab
- 작업추가 클릭
- 작업 이름 : lab-build
- 타입 : SourceBuild
- 프로젝트 : gitlab
- 작업 추가 클릭
- 이름 : lab-deploy

- 타입 : SourceDeploy
- 프로젝트 : gitlab
- 스태이지 : dev
- 시나리오 : test
- lab-build 작업의 + 버튼을 클릭, 선행작업 없음 선택 후 확인 클릭
- lab-deploy 작업의 + 버튼을 클릭, 선행작업 lab-build 선택 후 확인 클릭
- 하단의 ‘다음’ 클릭 후 파이프라인 생성 클릭
- 파이프라인 ‘gitlab’ 선택 후 상단의 ‘파이프라인으로 이동’클릭
- 상단의 파이프라인 실행하기 클릭
- lnxsvr2 서버에서 톰캣 재실행
- 웹 브라우저에서 <lnxsvr2 서버 IP>:8080/hello 로 접속하여 Hello, 이름! 출력 여부 확인