OpenInfra Community Days Korea 2021

쉽고 간편한 네이버 클라우드 플랫폼의 Infrastructure as Code

네이버클라우드
Cloud Advocate 송 창 안

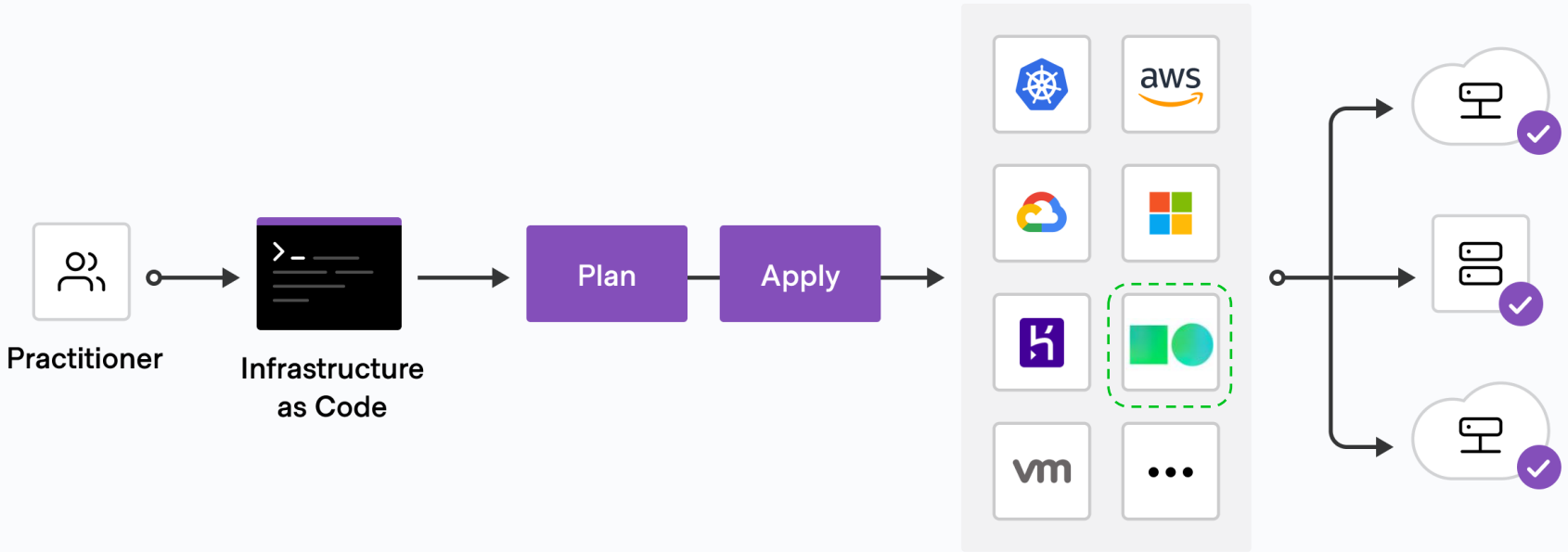
NAVER CLOUD PLATFORM

2021.09.15

01

IaC(Infrastructure as Code) 정의

IaC란?



반복 업무에 대한 자유로운 경험 >>> 애플리케이션 인프라 코드화

코드를 통한 인프라에 대한 효율적인 관리 >>> 클라우드 전반에서 인프라 관리

예측 가능한 관리 >>> 재현 가능한 인프라 구축

Server
Load Balancer
storage
NAS

리소스 명	제공 기능
ncloud_server	서버 인스턴스 리소스
ncloud_block_storage	블록 스토리지 리소스
ncloud_init_script	init 스크립트 리소스
ncloud_launch_configuration	ncloud 시작 구성 리소스
ncloud_lb	로드 밸런서 리소스
ncloud_lb_listener	로드 밸런서 listener 리소스
ncloud_lb_target_group	로드 밸런서 target group 리소스
ncloud_nas_volume	NAS 볼륨 리소스
ncloud_placement_group	배치 그룹 리소스
….	….

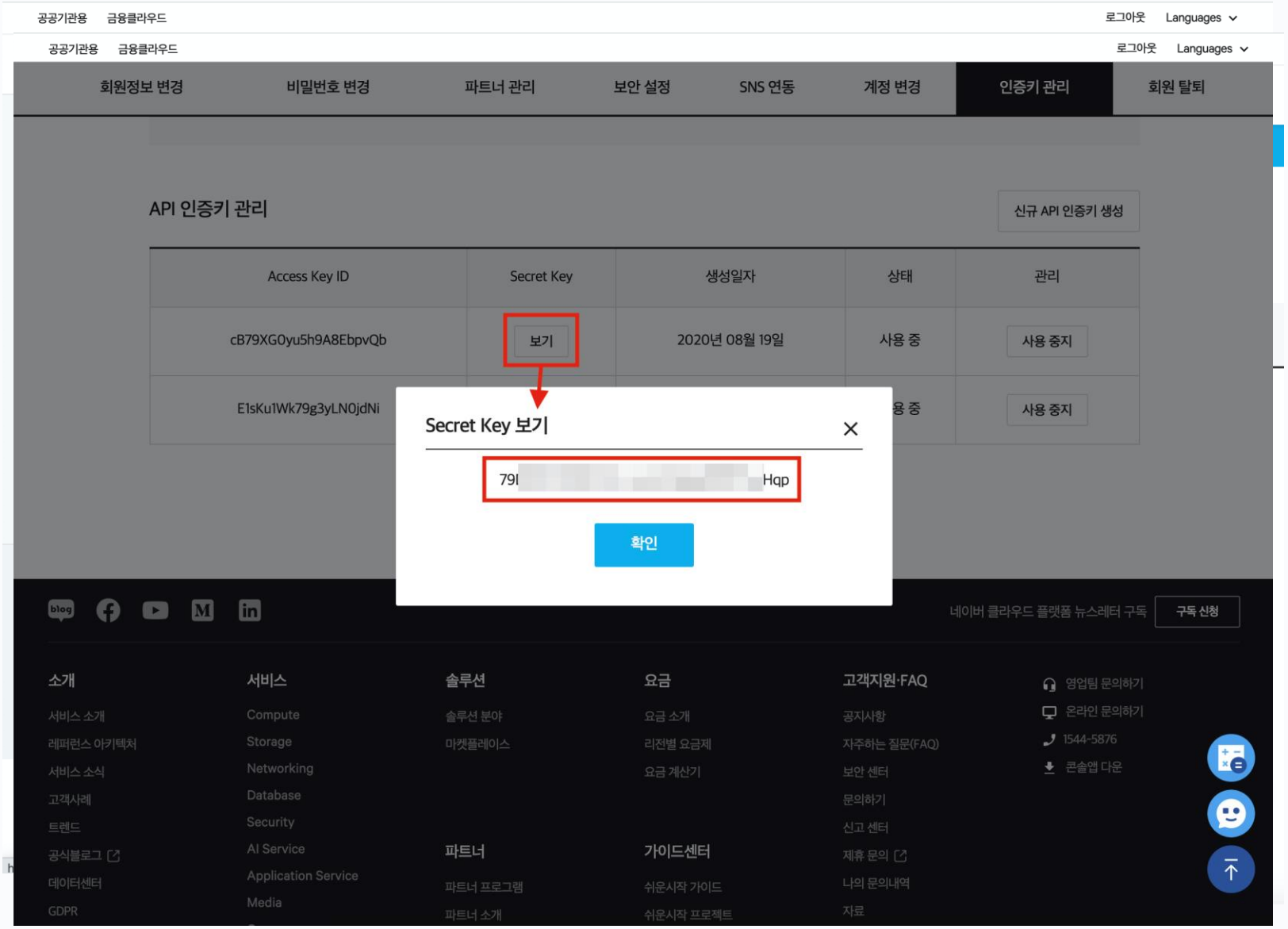
VPC Network

리소스 명	제공 기능
ncloud_vpc	VPC 리소스
ncloud_vpc_peering	VPC peering 리소스
ncloud_subnet	VPC 서브넷 리소스
ncloud_route	Route 리소스
ncloud_route_table	Route 테이블 리소스
ncloud_route_table_association	라우팅 테이블과 서브넷 간의 연결을 생성하기 위한 리소스
ncloud_network_acl	네트워크 ACL policy 명칭 리소스
ncloud_network_acl_rule	네트워크 ACL policy 리소스
ncloud_nat_gateway	NAT 게이트웨이 리소스
….	….

02

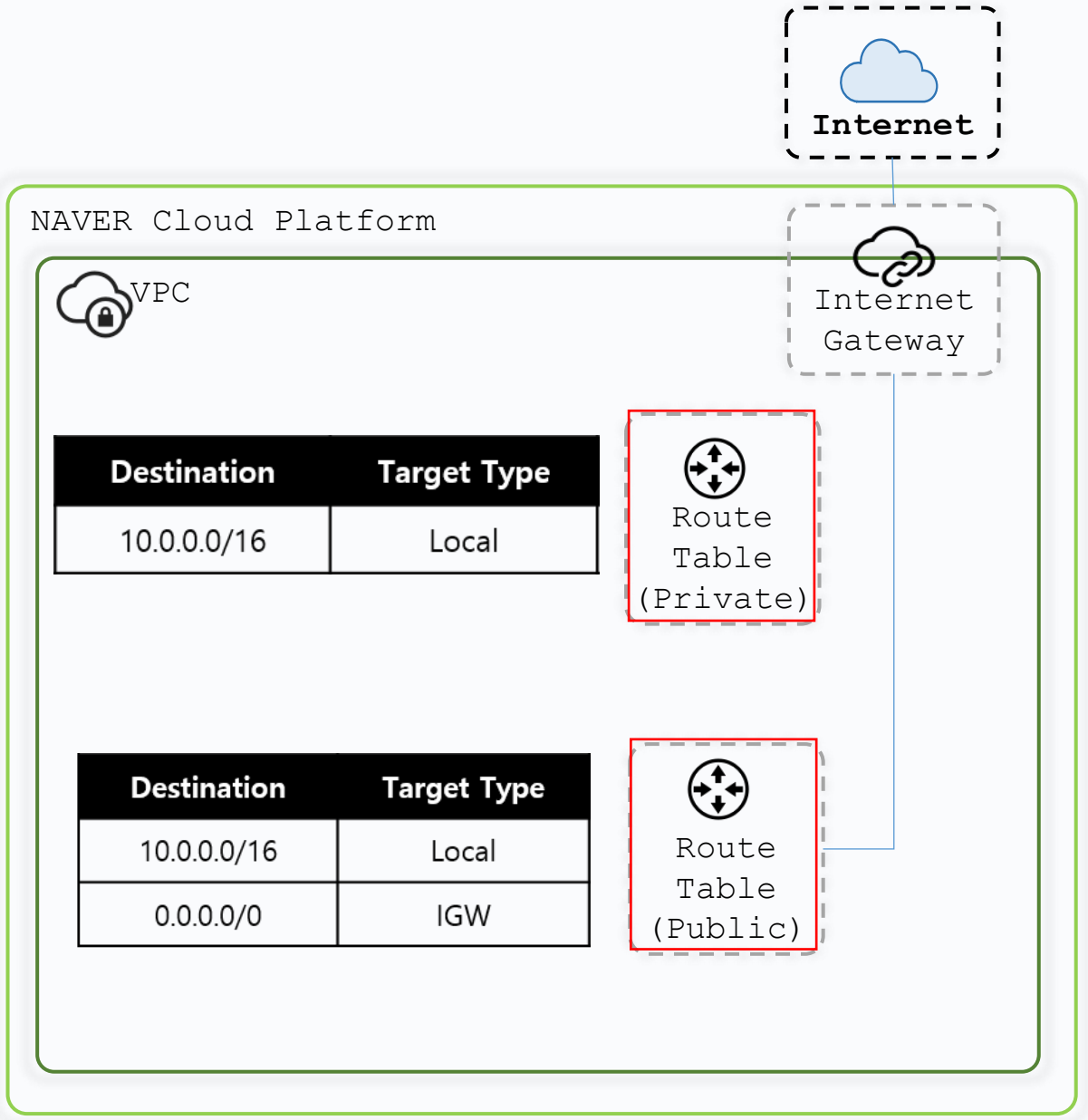
Example 1

VPC 생성 후 서버 생성

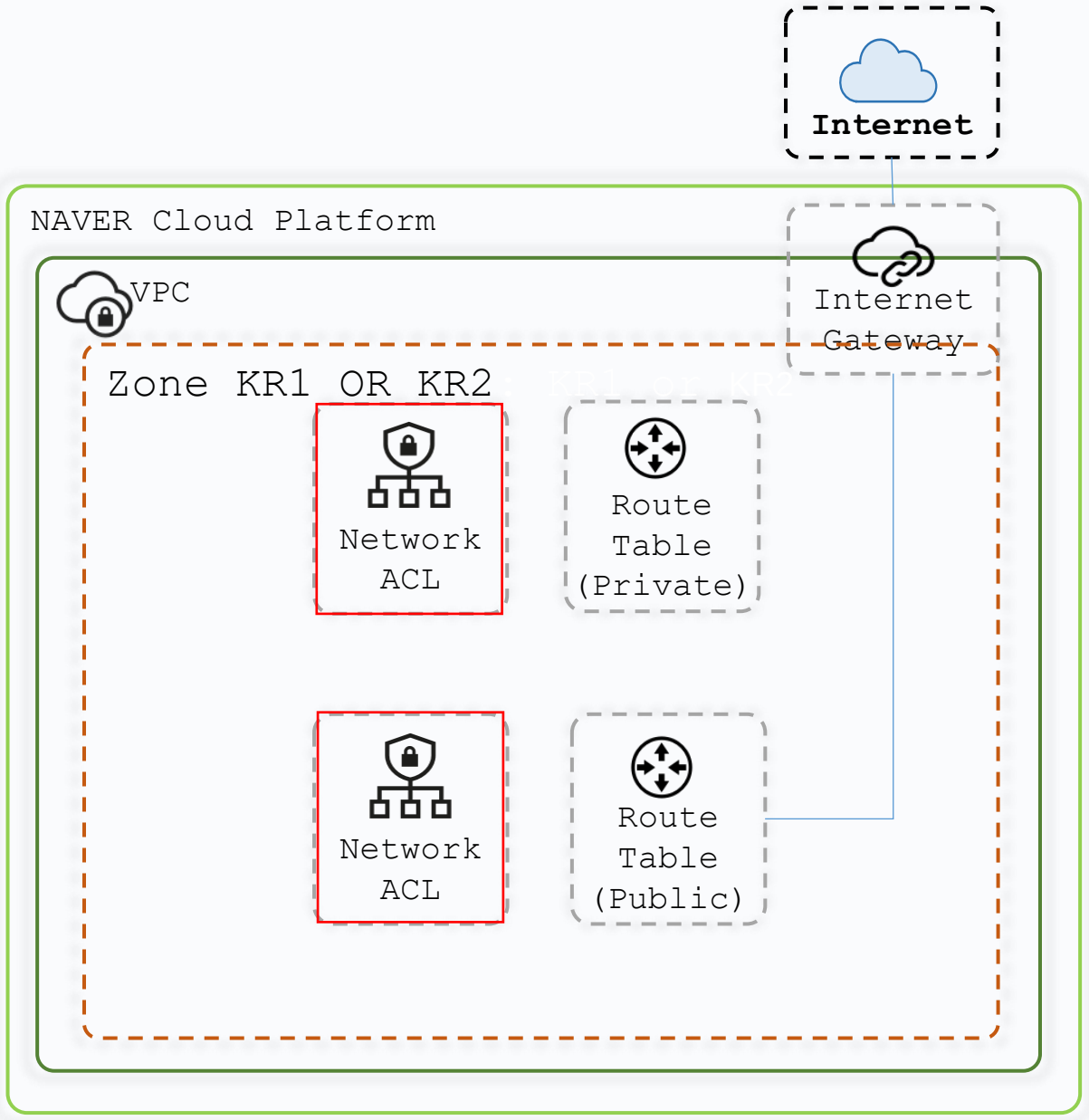

공공기관용 금융클라우드
로그아웃 Languages
회원정보 변경
비밀번호 변경
파트너 관리
보안 설정
SNS 연동
계정 변경
인증키 관리
회원 탈퇴
API 인증키 관리
신규 API 인증키 생성
Access Key ID
Secret Key
생성일자
상태
관리
cB79XG0yu5h9A8EbpvQb
보기
2020년 08월 19일
사용 중
사용 중지
E1sKu1Wk79g3yLN0jdNi
Secret Key 보기
79I Hqp
확인
네이버 클라우드 플랫폼 뉴스레터 구독
구독 신청
소개
서비스 소개
레퍼런스 아키텍처
서비스 소식
고객사례
트렌드
공식블로그
데이터센터
GDPR
서비스
Compute
Storage
Networking
Database
Security
AI Service
Application Service
Media
솔루션
솔루션 분야
마켓플레이스
파트너
파트너 프로그램
파트너 소개
요금
요금 소개
리전별 요금제
요금 계산기
가이드센터
쉬운시작 가이드
쉬운시작 프로젝트
고객지원·FAQ
공지사항
자주하는 질문(FAQ)
보안 센터
문의하기
신고 센터
제휴 문의
나의 문의내역
자료
영업팀 문의하기
온라인 문의하기
1544-5876
콘솔앱 다운

NAVER Cloud Platform

```terraform
terraform {
  required_providers {
    ncloud = {
      source = "navercloudplatform/ncloud"
    }
  }

provider "ncloud" {
  support_vpc = true
  region      = "KR"
  access_key  = var.access_key
  secret_key  = var.secret_key
}
```




```hcl
resource "ncloud_vpc" "vpc" {
  name            = var.vpc_name // "vpc-hashitalks"
  ipv4_cidr_block = "10.0.0.0/16"
}
```




```terraform
resource "ncloud_network_acl" "pub_nacl" {
  vpc_no = ncloud_vpc.vpc.id
  name   = "network-acl-public"
}
resource "ncloud_network_acl" "priv_nacl" {
  vpc_no = ncloud_vpc.vpc.id
  name   = "network-acl-private"
}

```

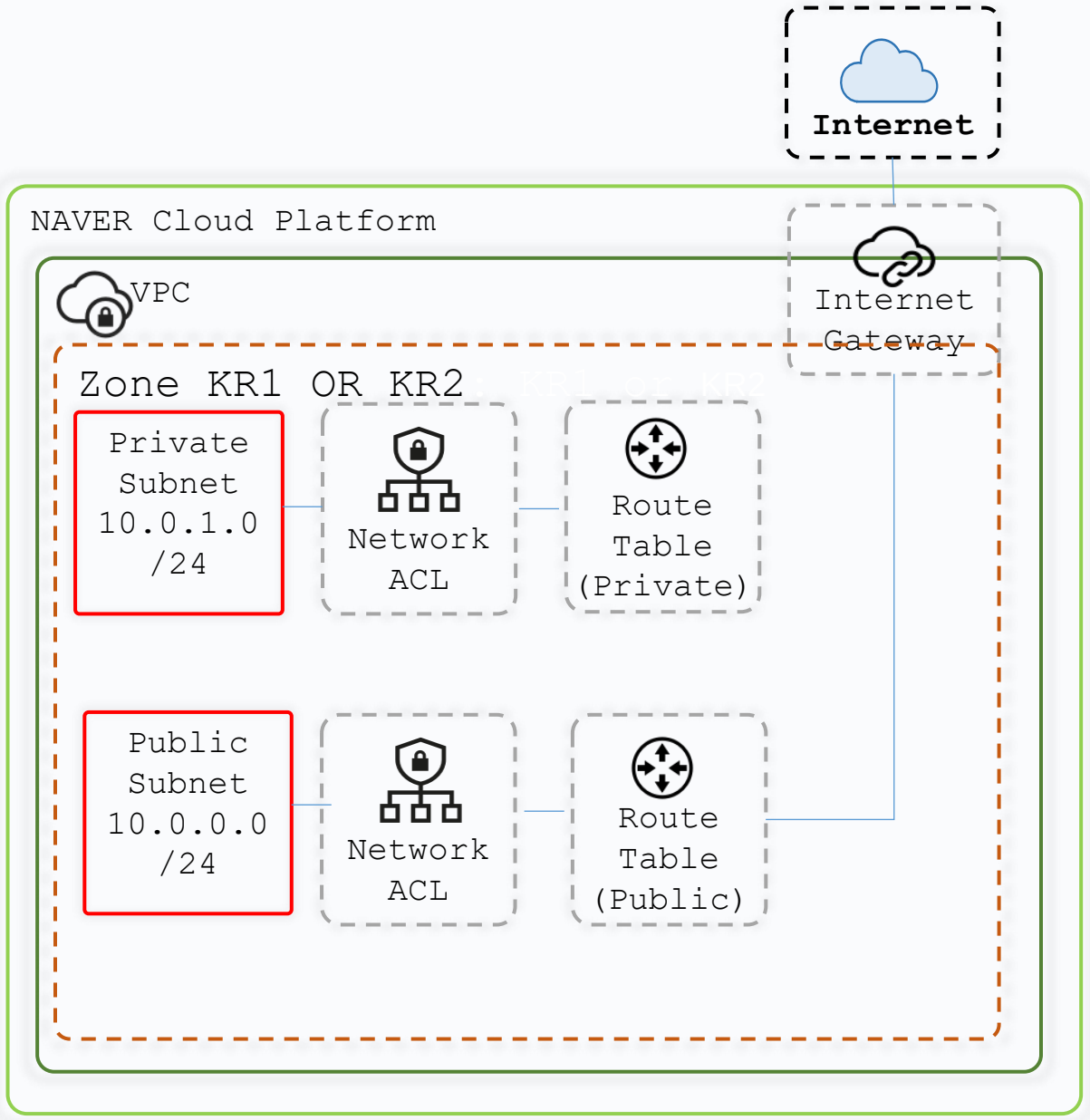
Inbound

프로토콜	접근 소스	포트	허용
TCP	내부 서버	내부 서버 포트	허용
TCP	0.0.0.0/0	32768-65535	허용
TCP	0.0.0.0/0	1-65535	거부
UDP	0.0.0.0/0	1-65535	거부
ICMP	0.0.0.0/0	-	거부

Outbound

프로토콜	접근 소스	포트	허용
TCP	내부 서버	32768-65535	허용
TCP	0.0.0.0/0	32768-65535	허용
TCP	0.0.0.0/0	1-65535	거부
UDP	0.0.0.0/0	1-65535	거부
ICMP	0.0.0.0/0	-	거부

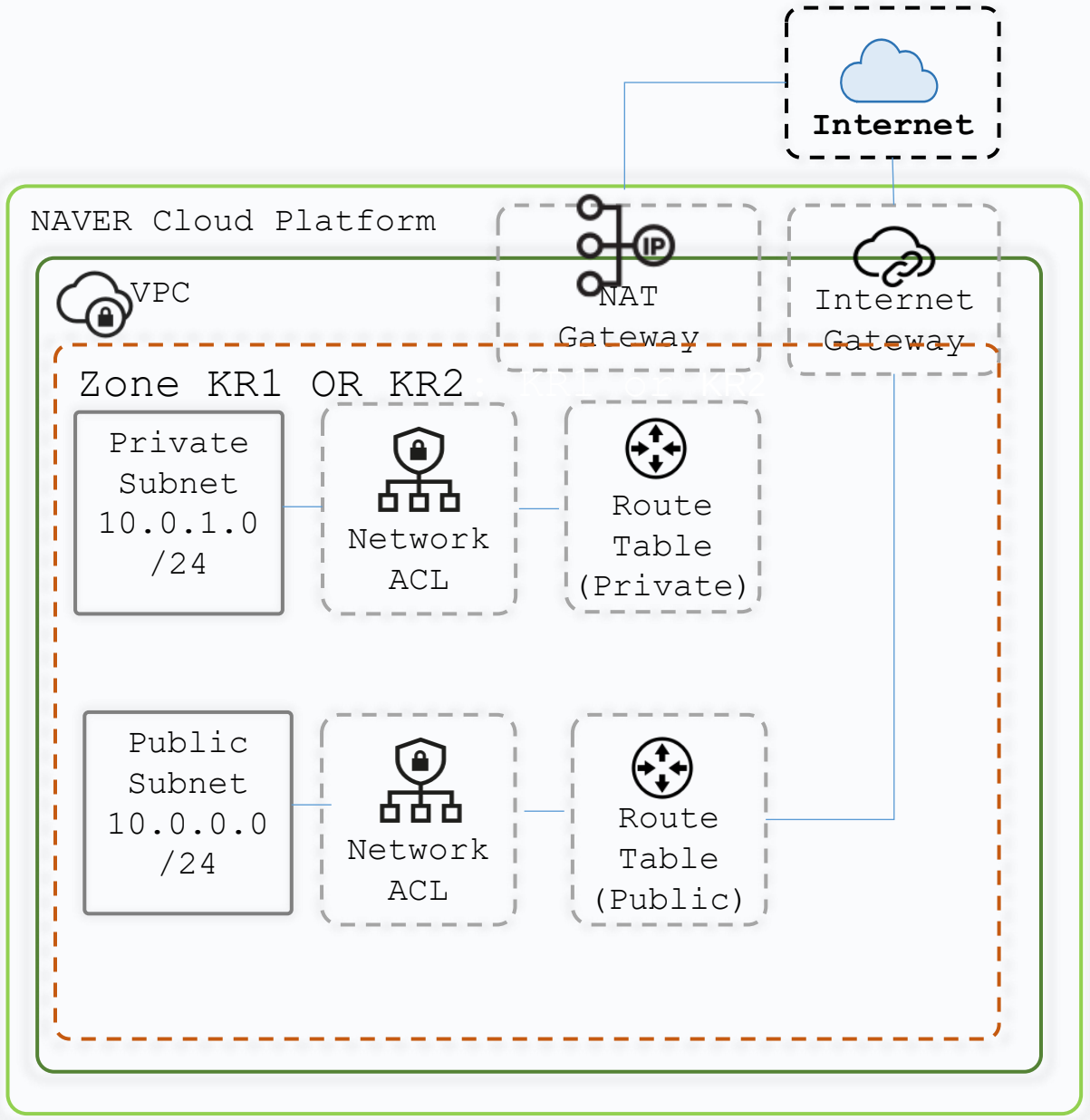
```
locals {
  private_subnet_inbound = [
    [1, "TCP", "${ncloud_server.server_scn_02_public.network_interface[0].private_ip}/32", "8080", "ALLOW"], //
Allow 8080 port from public server
    [2, "TCP", "0.0.0.0/0", "32768-65535", "ALLOW"],
    [197, "TCP", "0.0.0.0/0", "1-65535", "DROP"],
    [198, "UDP", "0.0.0.0/0", "1-65535", "DROP"],
    [199, "ICMP", "0.0.0.0/0", null, "DROP"],
  ]
private_subnet_outbound = [
    [1, "TCP", "${ncloud_server.server_scn_02_public.network_interface[0].private_ip}/32", "32768-65535", "ALLOW"],
// Allow 32768-65535 port to public server
    [197, "TCP", "0.0.0.0/0", "1-65535", "DROP"],
    [198, "UDP", "0.0.0.0/0", "1-65535", "DROP"],
    [199, "ICMP", "0.0.0.0/0", null, "DROP"]
  ]
}
resource "ncloud_network_acl_rule" "network_acl_02_private" {
  network_acl_no = ncloud_network_acl.network_acl_02_private.id
dynamic "inbound" {
    for_each = local.private_subnet_inbound
    content {
      priority    = inbound.value[0]
      protocol    = inbound.value[1]
      ip_block    = inbound.value[2]
      port_range  = inbound.value[3]
      rule_action = inbound.value[4]
    }
  }
dynamic "outbound" {
    for_each = local.private_subnet_outbound
    content {
      priority    = outbound.value[0]
      protocol    = outbound.value[1]
      ip_block    = outbound.value[2]
      port_range  = outbound.value[3]
      rule_action = outbound.value[4]
    }
  }
}
```



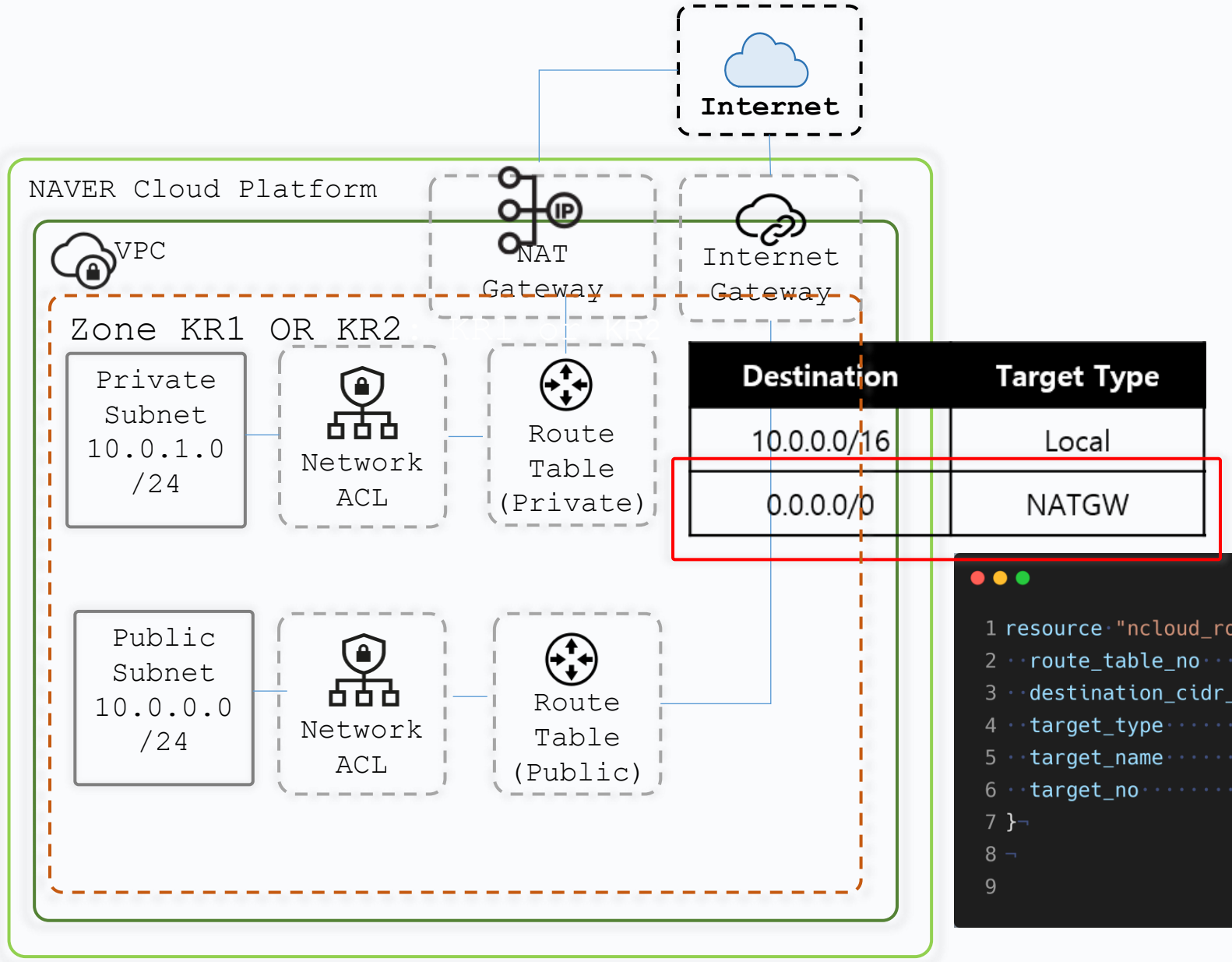

```hcl
resource "ncloud_subnet" "subnet_public" {
  name           = "subnet-public"
  vpc_no         = ncloud_vpc.vpc.vpc_no
  subnet         = cidrsubnet(ncloud_vpc.vpc.ipv4_cidr_block, 8, 0)
  network_acl_no = ncloud_network_acl.pub_nacl.id
  subnet_type    = "PUBLIC" // PUBLIC(Public)
  zone           = var.zone // KR-2
}
resource "ncloud_subnet" "subnet_private" {
  name           = "subnet-private"
  vpc_no         = ncloud_vpc.vpc.vpc_no
  subnet         = cidrsubnet(ncloud_vpc.vpc.ipv4_cidr_block, 8, 1)
  network_acl_no = ncloud_network_acl.priv_nacl.id
  subnet_type    = "PRIVATE" // PRIVATE(Private)
  zone           = var.zone // KR-2
}
```

10.0.0.0/24

10.0.1.0/24

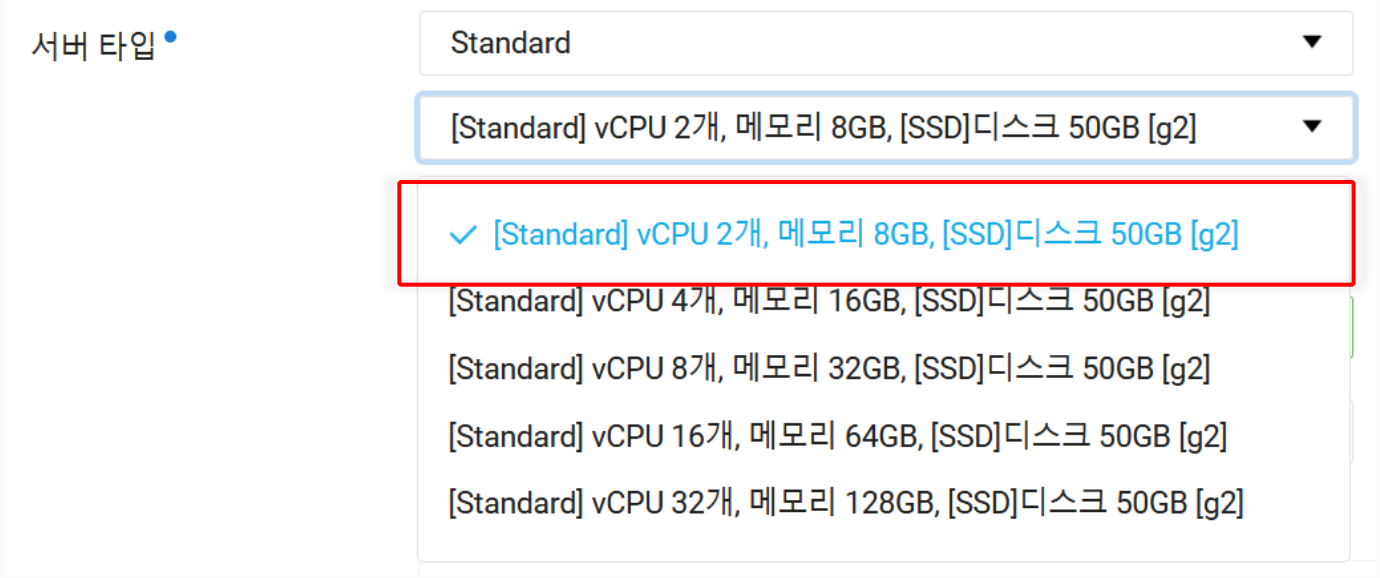



```hcl
resource "ncloud_nat_gateway" "nat_gateway" {
  vpc_no = ncloud_vpc.vpc.id
  zone   = var.zone // KR-2
  name   = "nat-gateway"
}

```



Destination	Target Type
10.0.0.0/16	Local
0.0.0.0/0	NATGW

```
resource "ncloud_route" "route_nat" {
  route_table_no         = ncloud_vpc.vpc.default_private_route_table_no
  destination_cidr_block = "0.0.0.0/0"
  target_type            = "NATGW"
  target_name            = ncloud_nat_gateway.nat_gateway.name
  target_no              = ncloud_nat_gateway.nat_gateway.id
}

```




```hcl
data "ncloud_server_image" "id" {
  filter {
    name   = "product_name"
    values = ["ubuntu-18.04?"]
    regex  = true
  }
}

data "ncloud_server_products" "products" {
 server_image_product_code = data.ncloud_server_image.id.id

  filter {
    name   = "product_code"
    values = ["SSD"]
    regex  = true
  }

  filter {
    name   = "cpu_count"
    values = ["2"]
  }

  filter {
    name   = "product_type"
    values = ["STAND"]
  }
}
```

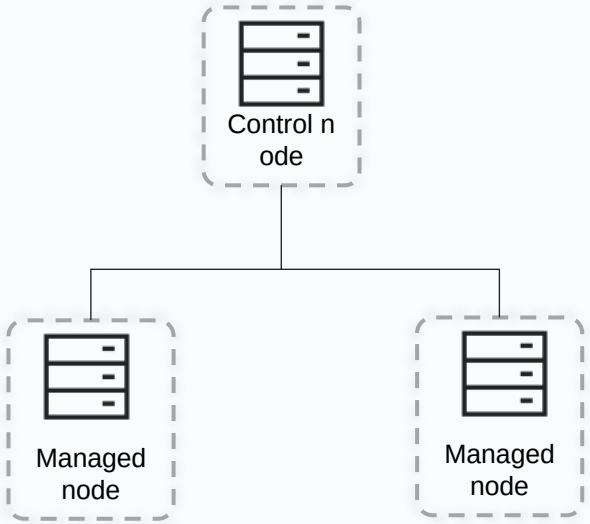
```
resource "ncloud_login_key" "key" {
  key_name = "login-key-name"
}

resource "ncloud_server" "server_public" {
  name                      = "server-public"
  login_key_name            = ncloud_login_key.key.key_name
  subnet_no                 = ncloud_subnet.subnet_public.id
  server_image_product_code =
  data.ncloud_server_products.products.server_products.0.product_code
}

resource "ncloud_server" "server_private" {
  name                      = "server-private"
  login_key_name            = ncloud_login_key.key.key_name
  subnet_no                 = ncloud_subnet.subnet_private.id
  server_image_product_code =
  data.ncloud_server_products.products.server_products.0.product_code

}
```

03

Example 2

서버 생성 후 ansible 로 프로비저닝
로드밸런서 와 NAS 연동

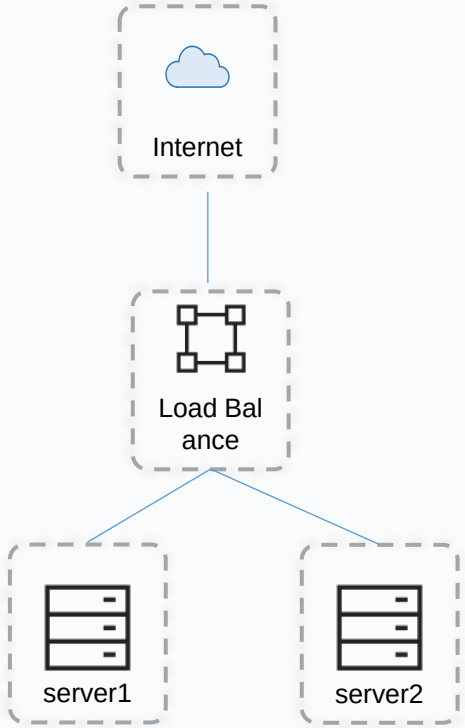
ansible –i inventory playbook.yaml



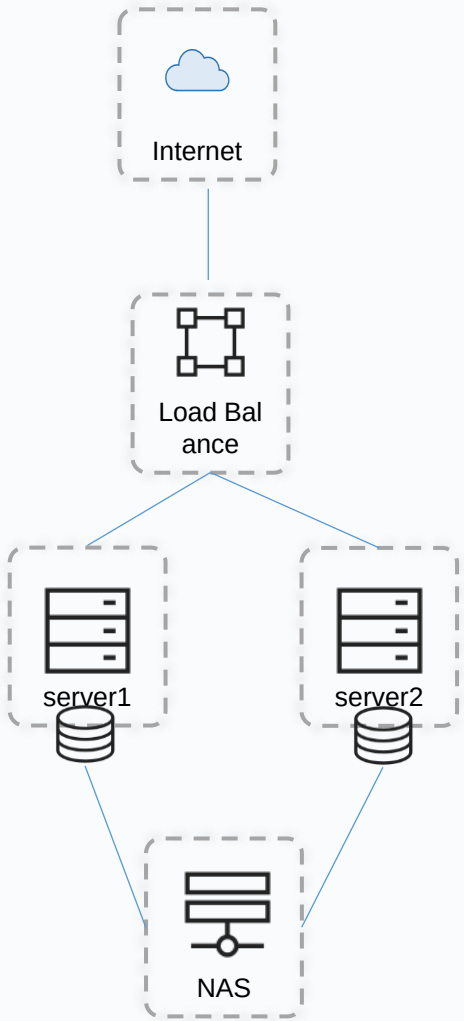

```
resource "null_resource" "ssh" {
  provisioner "local-exec" {
    command = <<EOF
      echo "[ncloud]" > inventory
      echo "${ncloud_server.server.name}
  ansible_host='${ncloud_port_forwarding_rule.forwarding.port_forwarding_public_ip}'
  ansible_port='${ncloud_port_forwarding_rule.forwarding.port_forwarding_external_port}'
  ansible_ssh_user=root
  ansible_ssh_pass='${data.ncloud_root_password.rootpwd.root_password}'" >> inventory

EOF

  }

  provisioner "local-exec" {
    command = <<EOF
      ANSIBLE_HOST_KEY_CHECKING=False \
      ansible-playbook -i inventory playbook.yml

EOF

  }
}
```

```yaml
---
- hosts: ncloud
  become: true
  tasks:
    - name: ping
      ping:

    - name: enable ngnix repo
      copy:
        src: nginx.repo
        dest: /etc/yum.repos.d/nginx.repo
        owner: root
        group: root
        mode: '0644'
        backup: yes
```

```yaml
    - name: install packages
      yum:
        name: "{{ packages }}"
        state: latest
        update_cache: yes
      vars:
        packages:
        - git
        - nginx

    - name: Make sure a service is running
      systemd:
        state: started
        name: nginx
        enabled: yes
```




```hcl
resource "ncloud_load_balancer" "lb" {
  name                  = "ncloud-terraform-test-lb"
  algorithm_type        = "RR"
  description           = "ncloud-terraform-test-lb is best!!"

  rule_list {
    protocol_type       = "HTTP"
    load_balancer_port  = 80
    server_port         = 80
    l7_health_check_path = "/"
  }

  server_instance_no_list = [ncloud_server.server[0].id, ncloud_server.server[1].id]
  internet_line_type      = "PUBLC"
  network_usage_type      = "PBLIP"
  region                  = "KR"
}
```




```
resource "ncloud_nas_volume" "nas" {
  volume_name_postfix           = var.nas_volume_name_prefix
  volume_size                   = "500"
  volume_allotment_protocol_type = "NFS"
  server_instance_no_list       = [ncloud_server.server[0].id,
ncloud_server.server[1].id]
}

resource "null_resource" "nas" {
  provisioner "local-exec" {
    when = create
    command = <<EOF
      echo "mount -t nfs ${var.nasserver}:/${ncloud_nas_volume.nas.volume_name}
/usr/share/nginx/html" > mount.sh
EOF

  }
}
```

04

Example 3

베어메탈 서비스 구성

NAVER
CLOUD PLATFORM

```
resource "ncloud_server" "bm" {
  name                      = var.server_name
  server_image_product_code = data.ncloud_server_image.image.id
  server_product_code       = data.ncloud_server_product.prod.id
  login_key_name            = ncloud_login_key.key.key_name
  raid_type_name            = "5"
  zone                      = "KR-2"
}

resource "ncloud_public_ip" "public_ip" {
  server_instance_no = ncloud_server.bm.id
}
```

```
data "ncloud_server_image" "image" {
    infra_resource_detail_type_code = "BM"
    filter {
        name = "product_name"
        values = ["centos-7.8-64"]
    }
}

data "ncloud_server_product" "prod" {
  server_image_product_code = data.ncloud_server_image.image.id
    filter {
        name = "product_description"
        values = ["^(.*)2\\.2 GHz(.*)20 cores(.*)"]
        regex = true
    }
}
```

참고 리소스

- NaverCloudPlatform / terraform-provider-ncloud
  https://github.com/NaverCloudPlatform/terraform-provider-ncloud
- Naver Cloud Platform 테라폼 공식 페이지
  https://registry.terraform.io/providers/NaverCloudPlatform/ncloud/latest/docs
- [이렇게 사용하세요!] 네이버 클라우드 플랫폼에서 terraform 활용[1] : ansible 연동
  https://blog.naver.com/n_cloudplatform/222078603490
- [이렇게 사용하세요!] 네이버 클라우드 플랫폼에서 terraform 활용[2]: Loadbalancer 서비스와 ansible 연동
  https://blog.naver.com/n_cloudplatform/222094469557
- [이렇게 사용하세요!] 네이버 클라우드 플랫폼에서 terraform 활용[3] : Loadbalancer, Ansible과 NAS 서비스 연동하기
  https://blog.naver.com/n_cloudplatform/222108965637
- [이렇게 사용하세요!] 네이버 클라우드 플랫폼에서 terraform 활용[4] : 간단하게 베어메탈 서버 구성하기
  https://blog.naver.com/n_cloudplatform/222272175921

참고 리소스

- NCP(Naver Cloud Platform) 테라폼 workshop
  https://docmoa.github.io/03-Public%20Cloud/NCP/
- [이렇게 사용하세요!] Terraform을 활용한 네이버 클라우드 플랫폼 VPC 인프라 구성하기
  https://blog.naver.com/n_cloudplatform/222189643849
- NAVER Cloud Platform x HashiCorp 웨비나 다시 보기
  https://youtu.be/Dqwk7fYHhVQ
- Naver Cloud Platform x HashiCorp Webinar Season 2 예제
  https://github.com/ncp-hc/season2

The End of Document

Thank You

NAVER
CLOUD PLATFORM

ncloud.com