

목차

| | |
|----------------------|---|
| 1. EC2 인스턴스 생성 | 2 |
| □ 목적 | 2 |
| □ 목표 | 2 |
| □ EC2 생성 | 2 |
| □ user data 설정 | 2 |
| - key-pair 설정 | 2 |
| - user data 설정 | 3 |
| □ 태그설정(옵션) | 4 |

1. EC2 인스턴스 생성

□ 목적

EC2 인스턴스를 생성하고 apache2를 설치하는 과정을 테라폼으로 작성한다.

□ 목표

- 테라폼으로 EC2 인스턴스를 생성할 수 있다.
- EC2인스턴스 원격접속을 위해 공개키 인증방식을 적용할 수 있다.
- 테라폼으로 EC2 인스턴스 user-data를 설정할 수 있다.

□ EC2 생성

테라폼에서는 `aws_instance` 리소스를 이용하여 EC2 인스턴스를 생성한다. `ami`와 `instance_type`필드가 필수로 설정해야 한다.

- `ami`: aws ami ID
- `instance_type`: EC2 인스턴스 타입(예: `t2.micro`)

아래 예제는 타입이 `t2.nano`타입이고 AMI가 `ubuntu18.04`인 EC2 인스턴스를 생성하는 예제이다.

```
resource "aws_instance" "web" {  
  ami = "ami-0b50511490117e709" # ubuntu 18.04LTS  
  instance_type = "t2.nano"  
}
```

□ user data 설정

`user data`는 EC2 인스턴스가 생성되고 사용자가 입력한 명령어를 실행하는 기능이다. 테라폼은 `user data` 작업을 수행하기 위해 EC2 인스턴스 원격접속이 필요하다.

- key-pair 설정

EC2 인스턴스의 공개키 인증방식에 사용되는 키 설정을 설명한다.

① 공개키-비밀키 키쌍 생성

EC2 인스턴스에 등록할 공개키와 원격 접속에 사용하는 비밀키를 생성한다. -f인자에 생성할 키 이름을 설정한다. 아래 예는 공개키는 test.pub, 비밀키는 test이름으로 생성된다.

```
ssh-keygen -t rsa -b 4096 -N "" -f test
```

② aws_key_pair 리소스 생성

생성한 키 쌍에서 공개키를 file함수를 사용하여 aws_key_pair 리소스를 등록한다.

```
resource "aws_key_pair" "demo-keypair" {
  key_name     = "deployer-key"
  public_key   = file("./test.pub")
}
```

③ aws_instance 리소스에 키쌍 등록

aws_key_pair 리소스 key_name필드를 활용하여 EC2 인스턴스에 공개키를 등록한다.

```
resource "aws_instance" "web" {
  ami = "ami-0b50511490117e709"
  instance_type = "t2.nano"
  key_name = aws_key_pair.demo-keypair.key_name
  ...
}
```

- user data 설정

EC2 인스턴스 원격접속 후에 실행하는 명령어를 설정한다.

① 의존성 설정

user data작업은 EC2 인스턴스 생성 후에 실행하게 설정한다. EC2 인스턴스 생성 전에 user data를 실행하면, 인스턴스 생성이 끝나지 않았으므로 오류가 발생한다. 실행 순서 설정은 의존성으로 쉽게 제어할 수 있다. 의존성 설정을 위해 빈 리소스인 null_resource를 사용한다.

```
resource "null_resource" "name" {
```

```

depends_on = [aws_instance.web]
...
}

```

② EC2 인스턴스 원격접속

공개키 인증설정은 connection 리소스에 설정한다. 비밀키는 file함수를 사용하여 로드한다.

```

connection {
  type = "ssh"
  user = "ubuntu"
  private_key = file("./test")
  host = aws_instance.web.public_ip
  timeout = "1m"
}

```

③ 실행 명령어 설정

EC2가 생성되고 실행할 명령어를 remote-exec 리소스에 설정한다. 주의사항은 AMI 초기화 작업 전 명령어를 실행하면 오류가 발생할 가능성이 존재한다. 그러므로 초기화 작업이 끝나고 생성되는 파일을 검사한 후 명령어를 실행한다.

실행 권한은 connection 리소스에 설정한 일반 권한을 따른다. 그러므로 root계정이 필요한 명령어는 sudo가 필요하다. 아래 예제는 apache2 리눅스 패키지를 설치하고 index.html파일을 설정한다.

```

provisioner "remote-exec" {
  inline = [
    "until [ -f /var/lib/cloud/instance/boot-finished ]; do sleep 1; done",
    "sudo apt update",
    "sudo apt install apache2 -y && sudo systemctl start apache2",
    "sudo chown -R ubuntu:ubuntu /var/www/html",
    "echo 'hello world' > /var/www/html/index.html"
  ]
}

```

□ 태그설정(옵션)

EC2 인스턴스를 구분하기 태그를 지정한다. 대표적으로 EC2인스턴스 이름인 Name필드를 입력한

다. 아래 예제는 EC2 인스턴스가 생성되면 demo-instance 이름이 설정된다.

```
resource "aws_instance" "web" {
  ami = "ami-0b50511490117e709"
  instance_type = "t2.nano"
  key_name = aws_key_pair.demo-keypair.key_name
  ...
  tags = {
    Name = "demo-instance"
  }
}
```

The screenshot displays the AWS Management Console interface for EC2 instances. At the top, the 'Instances (1/1)' section shows a table with one instance, 'demo-instance', which is in the 'Running' state. Below this, the 'Tags' tab for the instance 'i-07104bc67fcee2dbb' is selected. The 'Tags' section shows a table with one tag: 'Name' with the value 'demo-instance'.

| Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability Zone |
|---------------|---------------------|----------------|---------------|----------------|--------------|-------------------|
| demo-instance | i-07104bc67fcee2dbb | Running | t2.nano | 2/2 checks ... | No alarms + | ap-northeast-2c |

| Key | Value |
|------|---------------|
| Name | demo-instance |

그림 1 EC2 Instance 태그 Name 설정 예