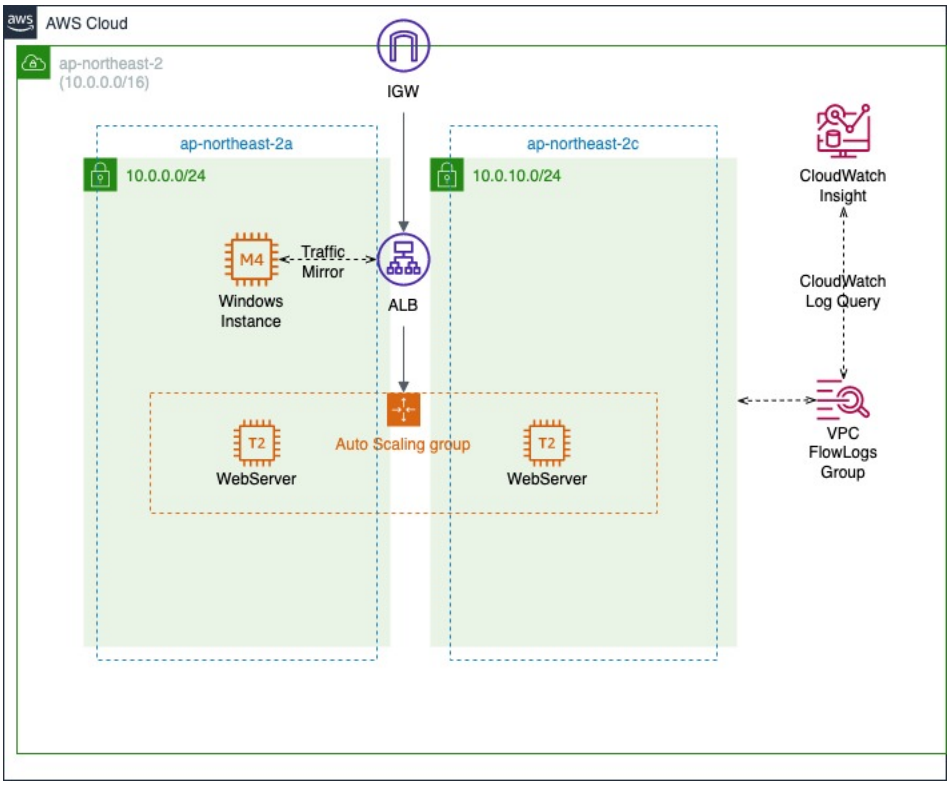
# VPC FlowLog&Traffic Mirror Workshop

## Introduce

본 워크샵은 VPC FlowLog및 Traffic Mirror를 실습하기 위하여 준비되었습니다. VPC FlowLog와 Traffic Mirror를 실제로 보기 위해 가상의 3 Tier 웹 어플리케이션을 구성하고 어플리케이션에서 발생하는 Traffic를 실제로 보고 확인해 보겠습니다.

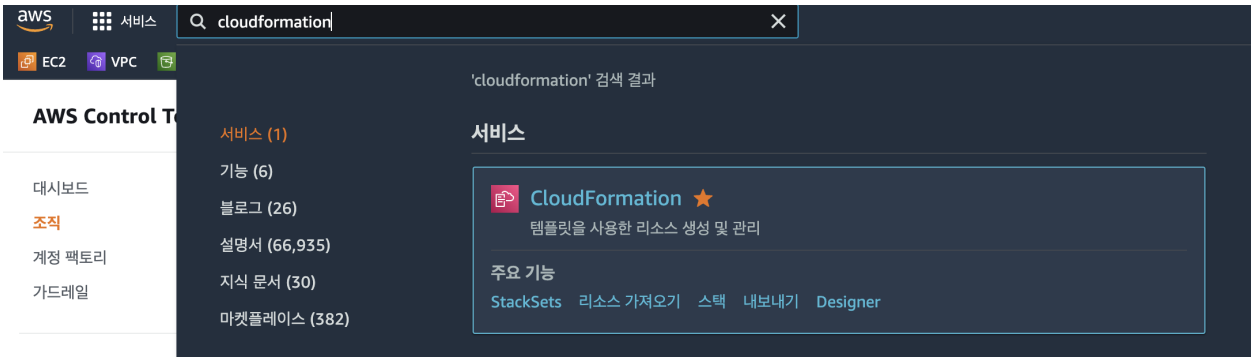



우리는 위와 같은 구성을 생성하여 실습할 것입니다.

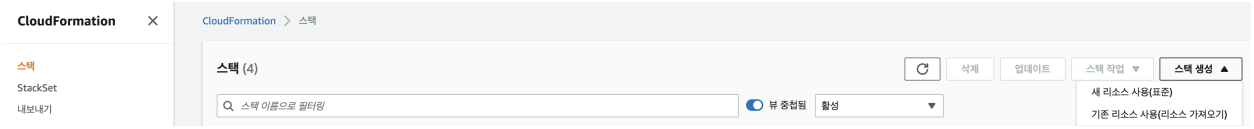
## 1. CloudFormation Template 수행

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/19f88f60-c80e-46c7-abc1-3ea1f7546f1f/VPC_AutoScaling_and_ElasticLoadBalancer.template

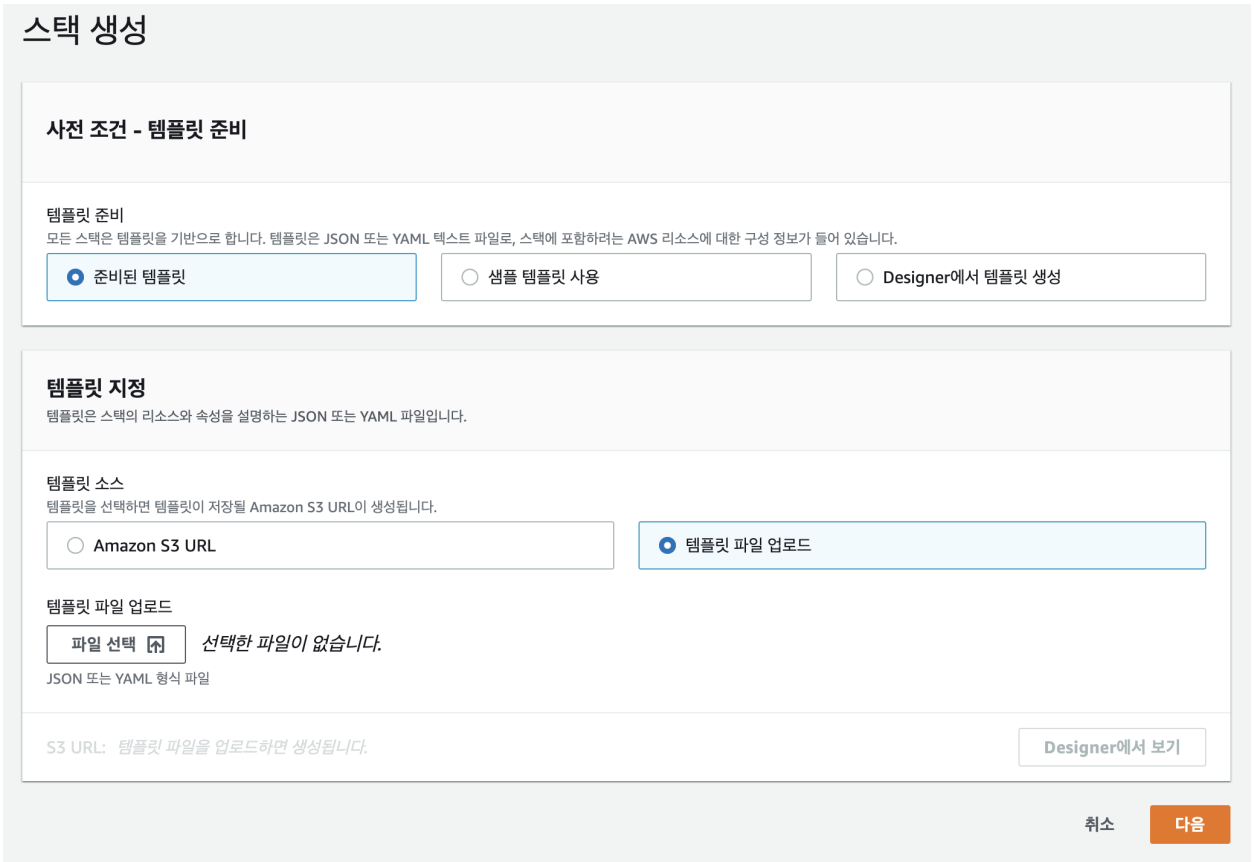
Region : ap-northeast-2



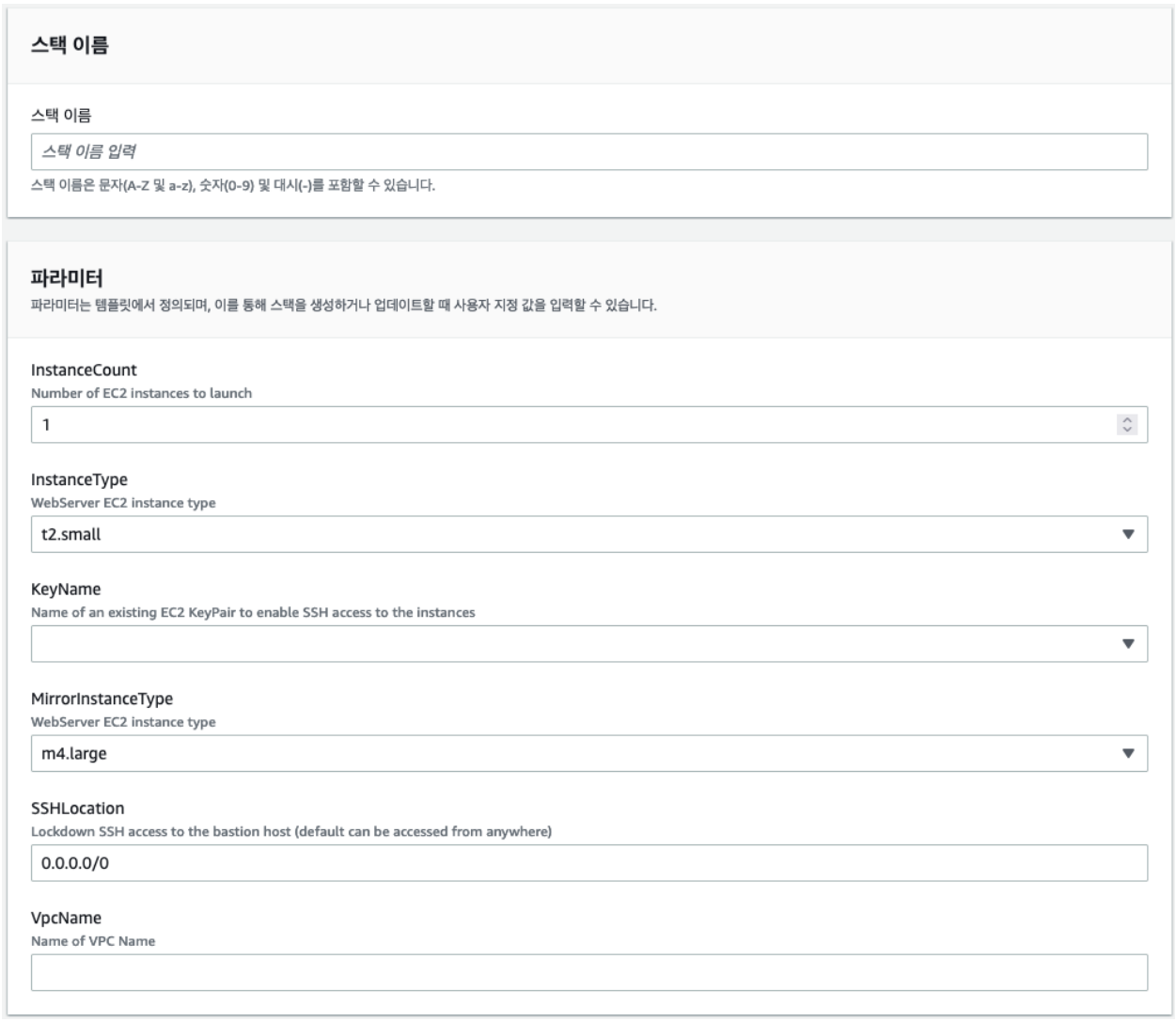

Cloudformation 검색후 해당 콘솔로 이동




스택 생성 클릭 → 새 리소스 사용(표준) 클릭




사전에 다운로드한 템플릿 파일 업로드




InstanceCount - ASG에서 생성한 EC2의 수량입니다. 1로 합니다

InstanceType - 생성될 EC2의 Type입니다. t2.small로 설정합니다.

KeyName - EC2의 Key입니다. 사전에 생성한 Key를 자동으로 찾습니다.

MirrorInstnaceType - Tarffic Mirror Target으로 사용한 EC2의 Type입니다. M시리즈 이상부터 Traffic Mirror 설정이 가능합니다

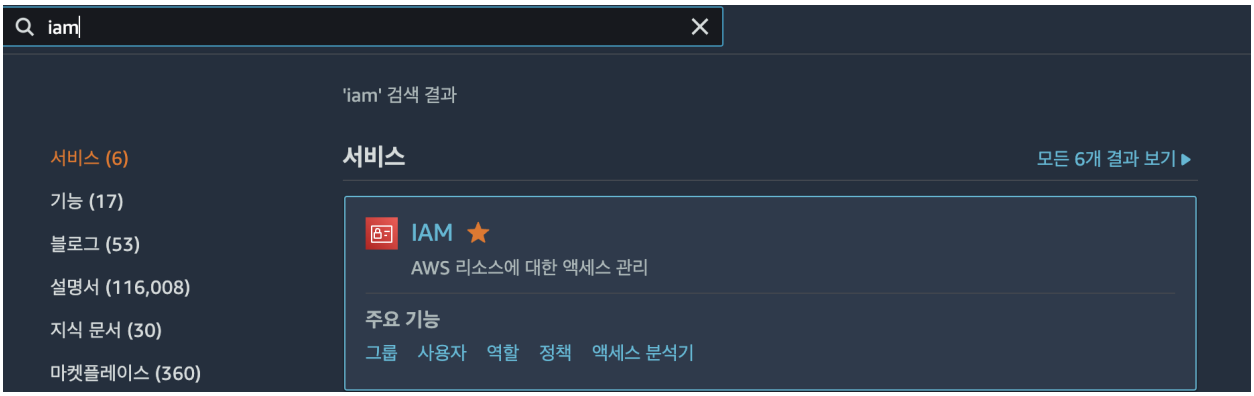
SSHLocation - EC2의 SSH 접근 Source IP입니다. 보안을 위하여 자신의 IP로 변경합니다.

VpcName - 생성될 VPC의 Name입니다.

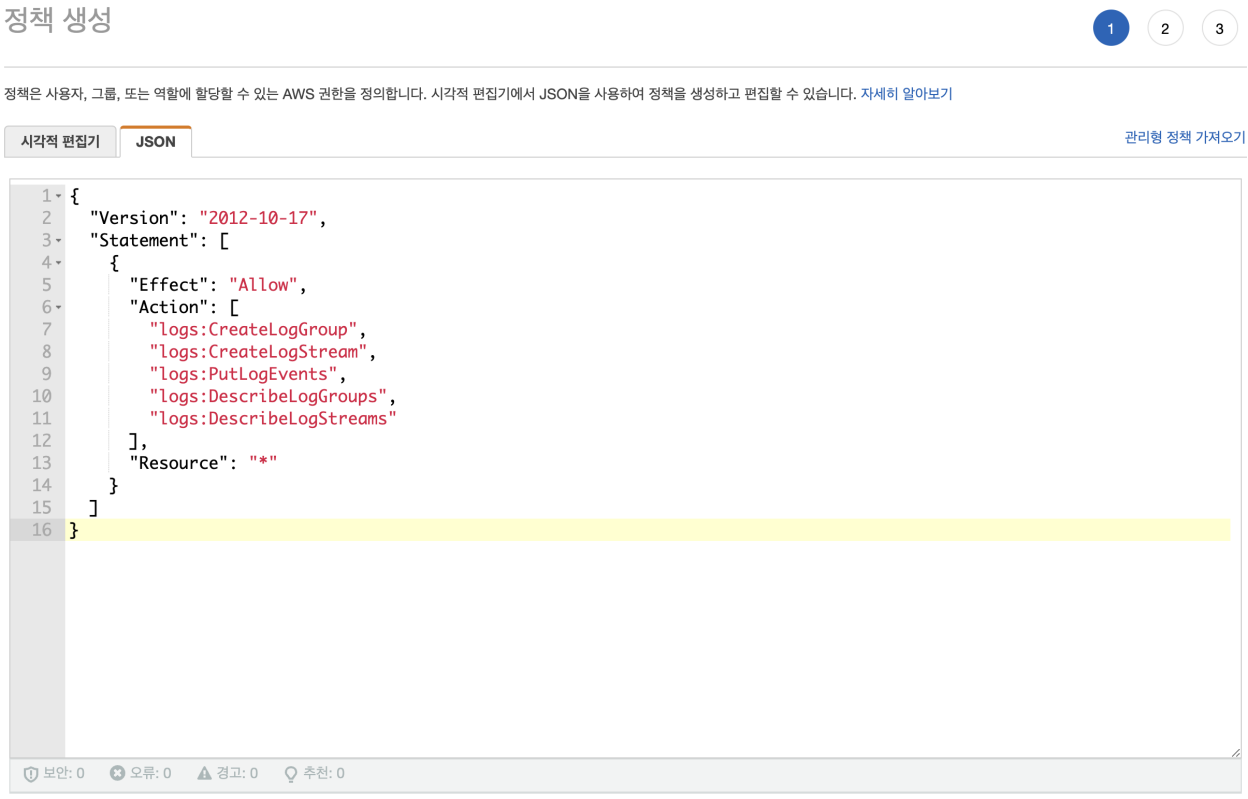
## 2. VPC FlowLog

VPC FlowLog 활성화방안

- CloudWatch Logs를 Target으로 하는 경우 사전에 Log Group과 IAM Role 생성이 필요합니다.




IAM 콘솔로 이동

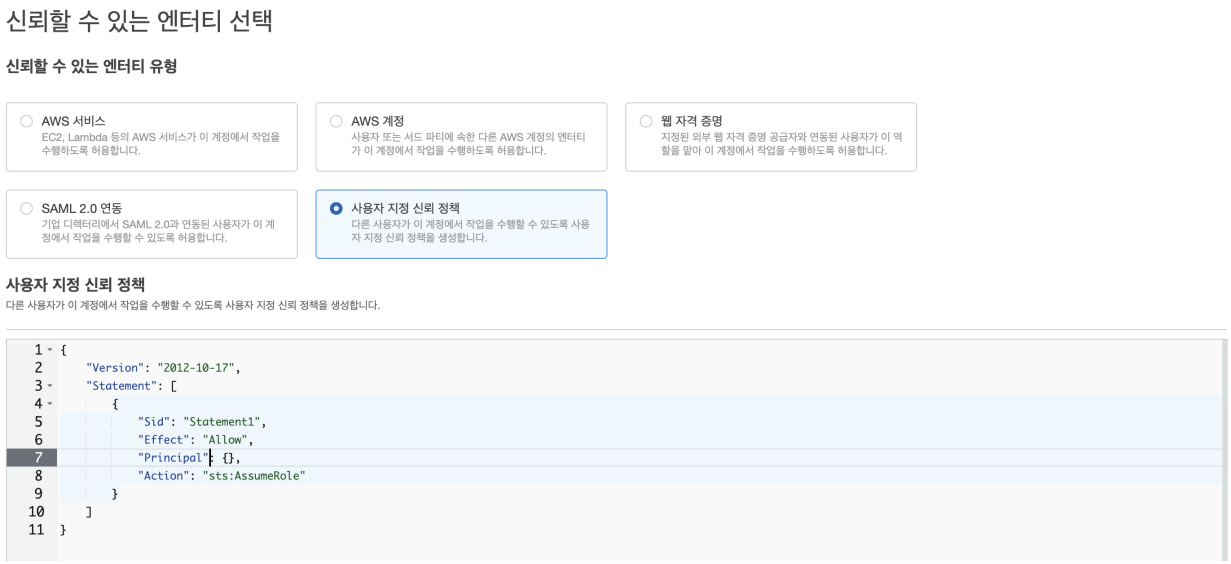



정책 생성

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Resource": "*"
```

```
    }
  ]
}
```

정책 이름은 `VPCFlowLogToCWLogsPolicy`로 합니다




다음으로 사이드에 역할을 클릭하여 역할을 생성합니다.

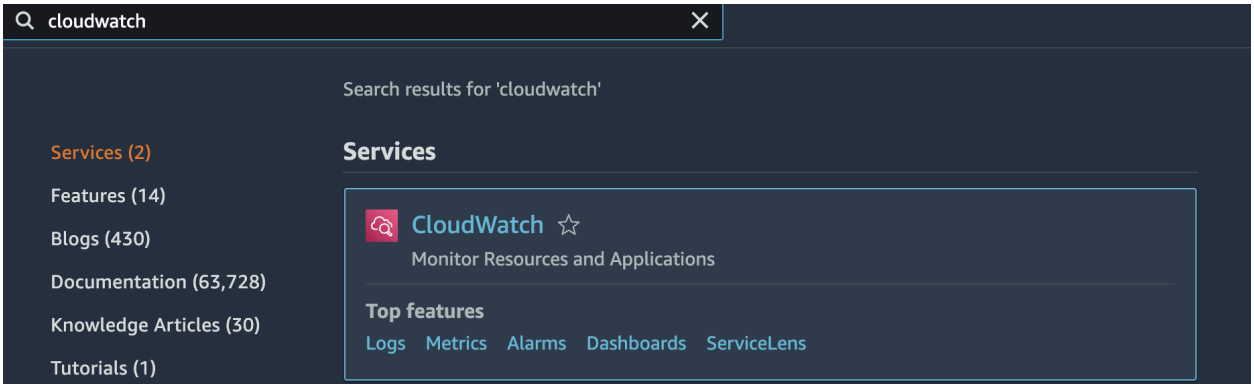
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "vpc-flow-logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

## 권한 추가

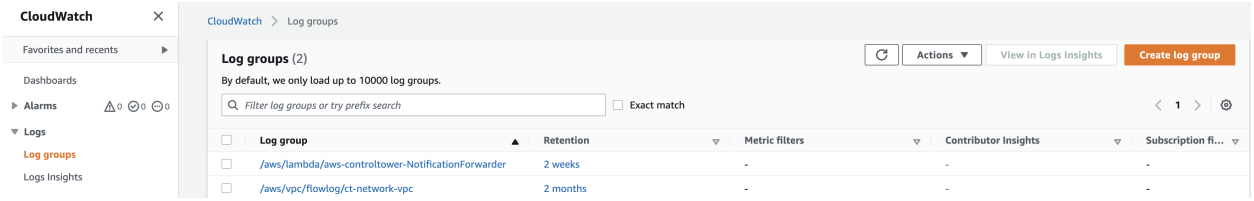

이전에 생성한 Policy를 선택합니다. 역할의 이름은 `VPCFlowLogToCWLogsRole`로 합니다.

> S3를 Target으로 한 경우 별도의 Role, Policy 생성 없이 자동으로 Target S3에 Policy가 생성됩니다.

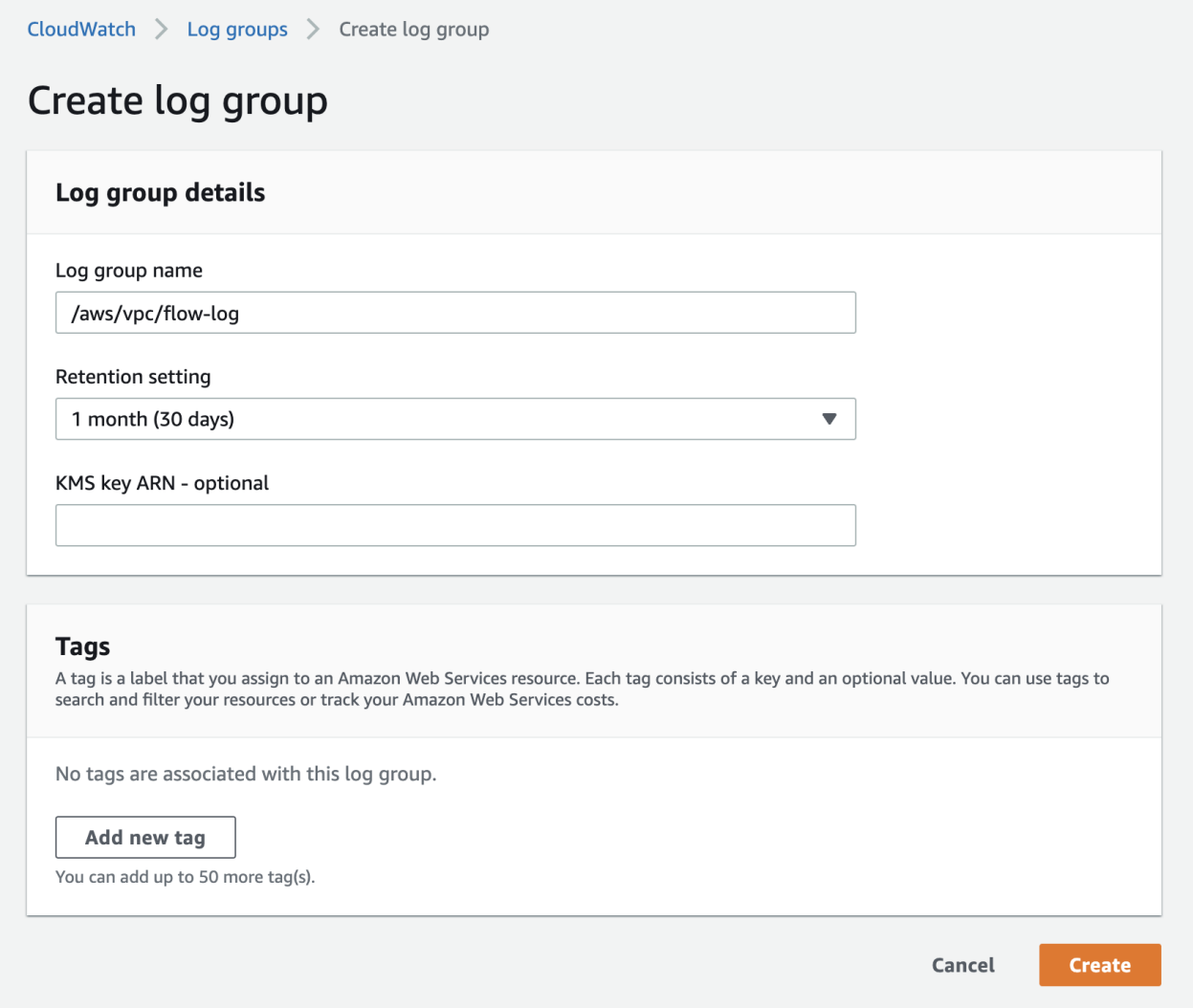
> KDF를 Target으로 할때 Source Account/Different Account로 나뉘어 설정이 가능합니다
>
> - Same Account인 경우 KDF에 Log를 Delivery할수 있는 권한만 부여되면 정상작동합니다.
> - Different Account는 Role Federation설정이 되어야만 정상적으로 작동합니다.



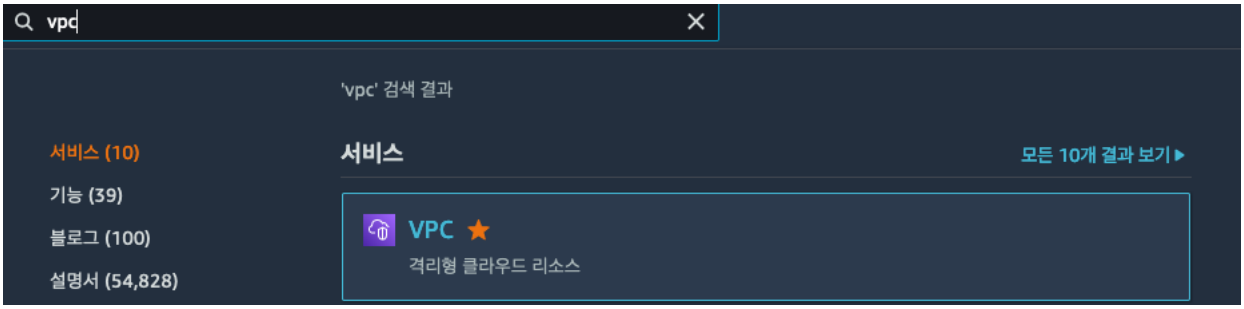

생성이 완료된후 CloudWatch Logs Group을 생성해 줘야 합니다. CloudWatch Console로 이동합니다.



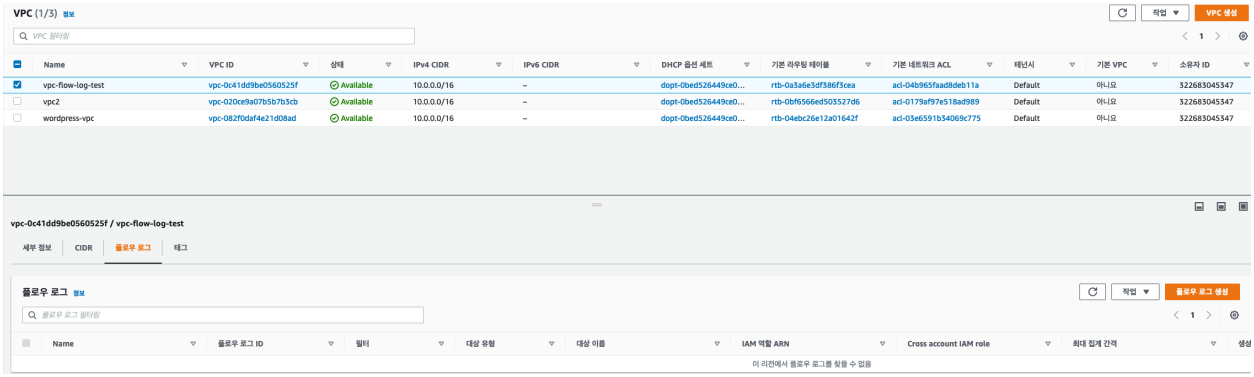

Logs Group으로 이동하여 Create log group를 클릭합니다.



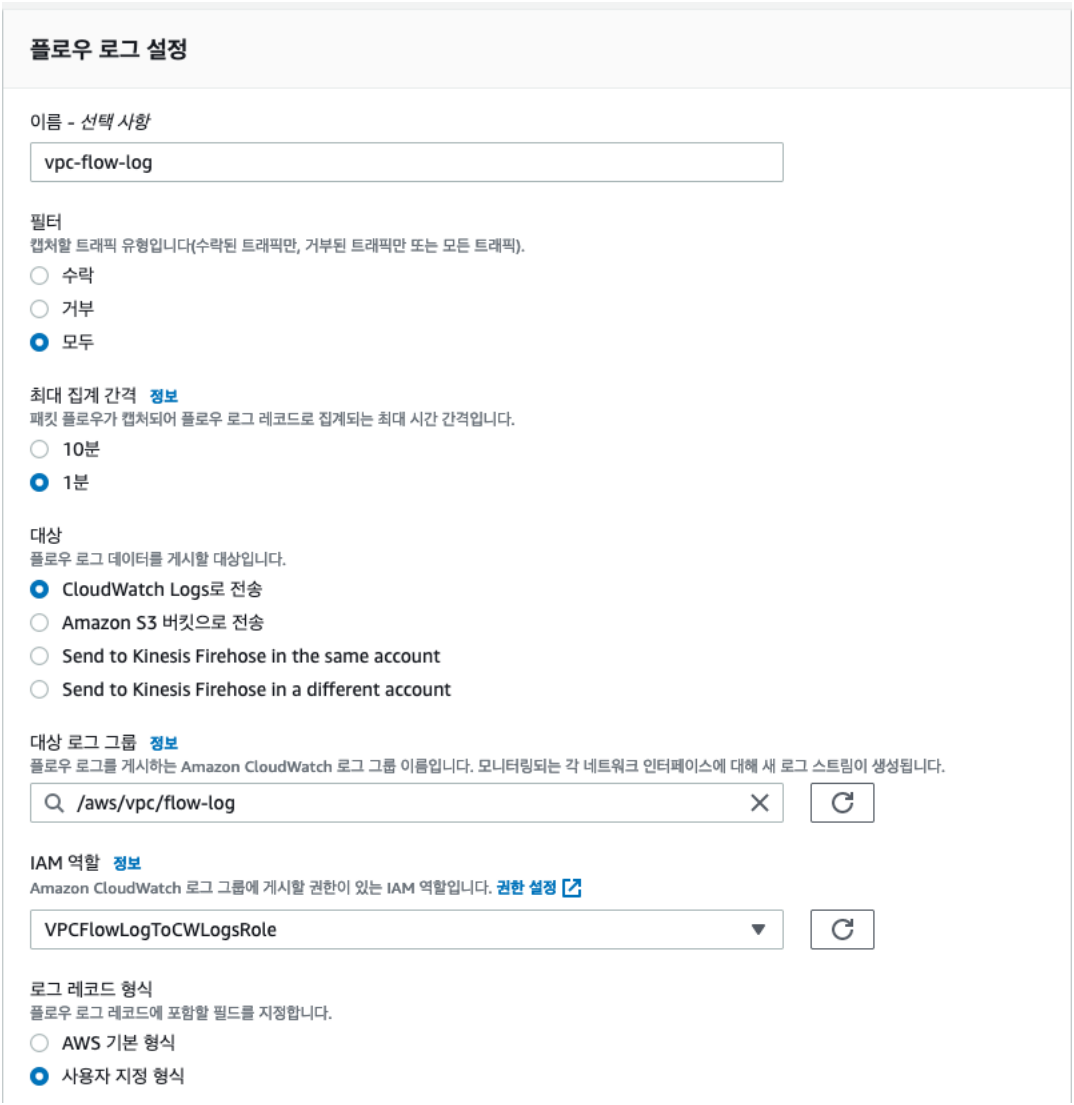

로그 그룹 이름을 입력하고 만료기간은 1개월로 설정하여 생성합니다.




VPC Console로 이동합니다

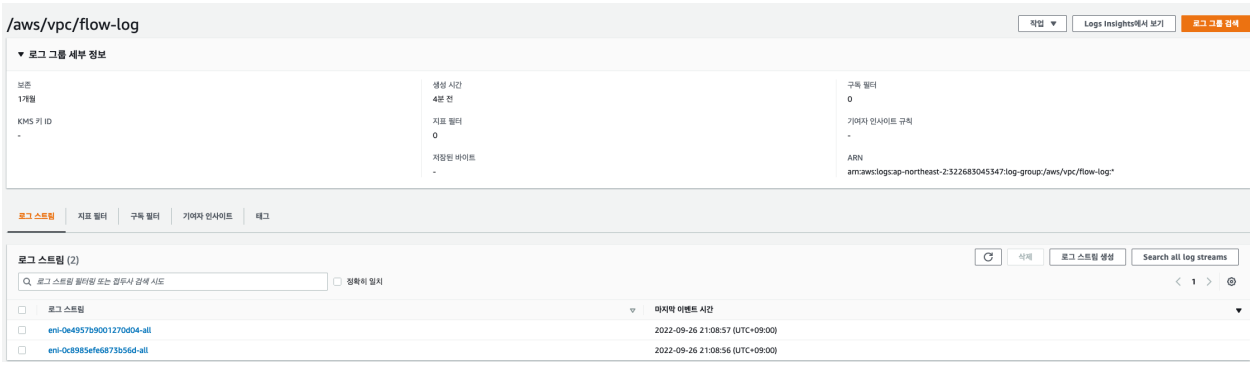



사전에 CloudFormation Stack으로 생성한 VPC를 선택하고 FlowLog를 생성합니다.



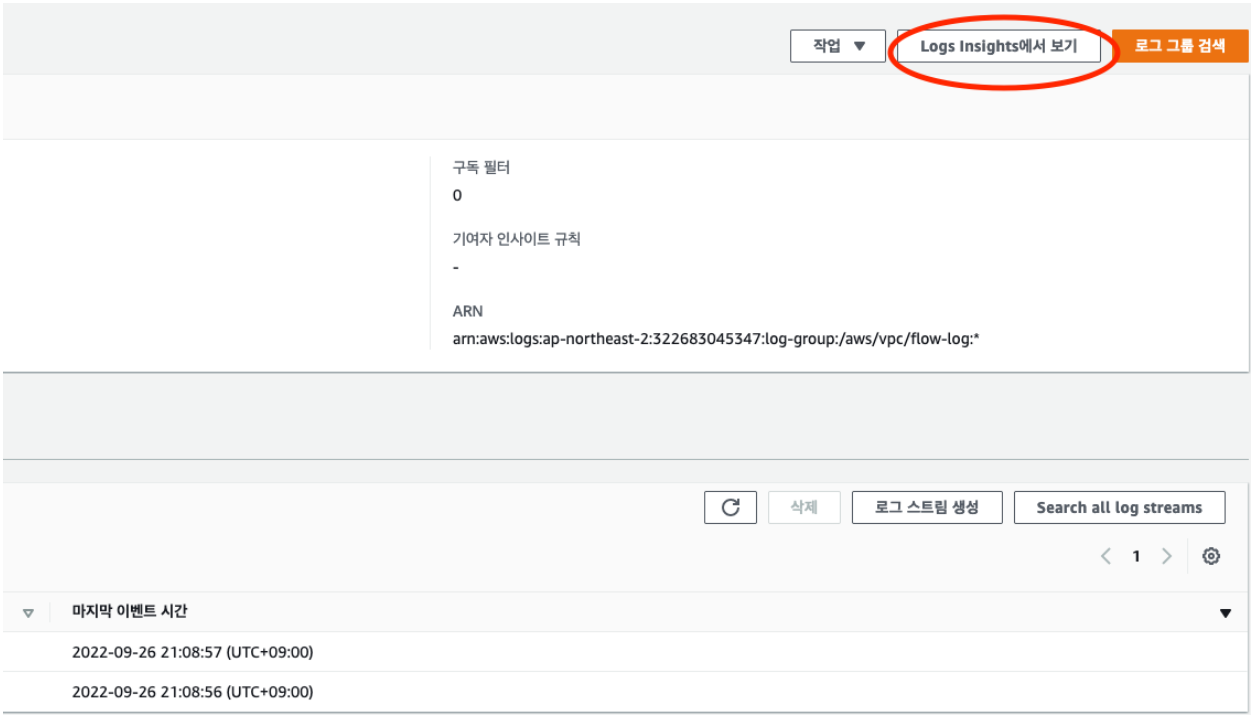

사전에 생성한 CloudWatch Log Group과 IAM 역할을 선택합니다.

- 로그 레코드 형식은 `사용자 지정 형식`으로 선택하고 `모두 선택`을 클릭 합니다.




일정 시간이 흐르면 위와 같이 CloudWatch Log group에 스트림이 생성됩니다.

### 3. CloudWatch Logs Insights 활용




Logs Insights를 클릭합니다.

- VPC Flow logs 상세 정보

- CloudWatch Insight 쿼리 구문

> CloudWatch Logs Insights 쿼리 구문
> CloudWatch Logs Insights에서는 로그 그룹을 쿼리하는 데 사용할 수 있는 쿼리 언어를 지원합니다. 쿼리 구문은 일반 함수, 산술 및 비교 연산, 정규 표현식을 포함하되 이에 국한되지 않는 다양한 함수 및 연산을 지원합니다. 여러 쿼리 명령이 포함된 쿼리를 생성할 수 있습니다. UNIX 스타일 파이프 문자(|)를 사용하여 쿼리에서 쿼리 명령을 구분합니다.
> https://docs.aws.amazon.com/ko_kr/AmazonCloudWatch/latest/logs/CWL_QuerySyntax.html

`@message`를 그대로 사용하면 검색이 어려울 수 있습니다. `parse`문을 이용하여 `@message`의 컬럼을 분리하여 정규화 할 수 있습니다.

```
fields @timestamp
| parse @message "* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *" as account_id, action, az_id, bytes, dstaddr, dstport, end, fl
| sort @timestamp desc
| limit 20
```

Source와 Destination IP주소 쌍 네트워크의 트레픽을 요약할 수 있습니다.

```
fields @timestamp
| parse @message "* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *" as account_id, action, az_id, bytes, dstaddr, dstport, end, fl
| stats sum(bytes) as Data_Transferred by srcaddr, dstaddr, flow_direction
| sort by Data_Transferred desc
| limit 2
```

Instance ID별로 분리하여 데이터 통계를 얻을수 있습니다

```
fields @timestamp
| parse @message "* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *" as account_id, action, az_id, bytes, dstaddr, dstport, end, fl
| stats sum(bytes) as Data_Transferred by instance_id
| sort by Data_Transferred desc
| limit 5
```

거부된 SSH접근 요청 내역을 요약하여 확인할 수 있습니다.

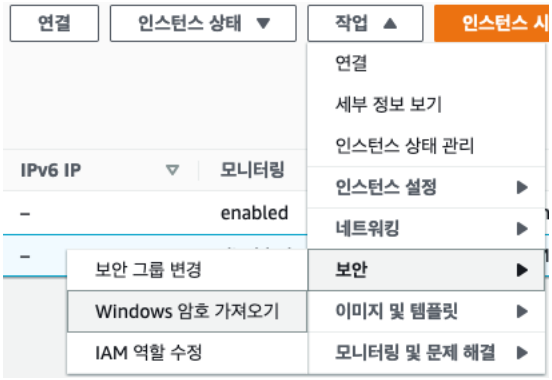
```
fields @timestamp
| parse @message "* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *" as account_id, action, az_id, bytes, dstaddr, dstport, end, fl
| filter action = "REJECT" and protocol = 6 and dstport = 22
| stats sum(bytes) as SSH_Traffic_Volume by srcaddr
| sort by SSH_Traffic_Volume desc
| limit 2
```

또는 모든 트래픽중 요청이 거부된 내역이 있는 IP를 찾을 수 있습니다.

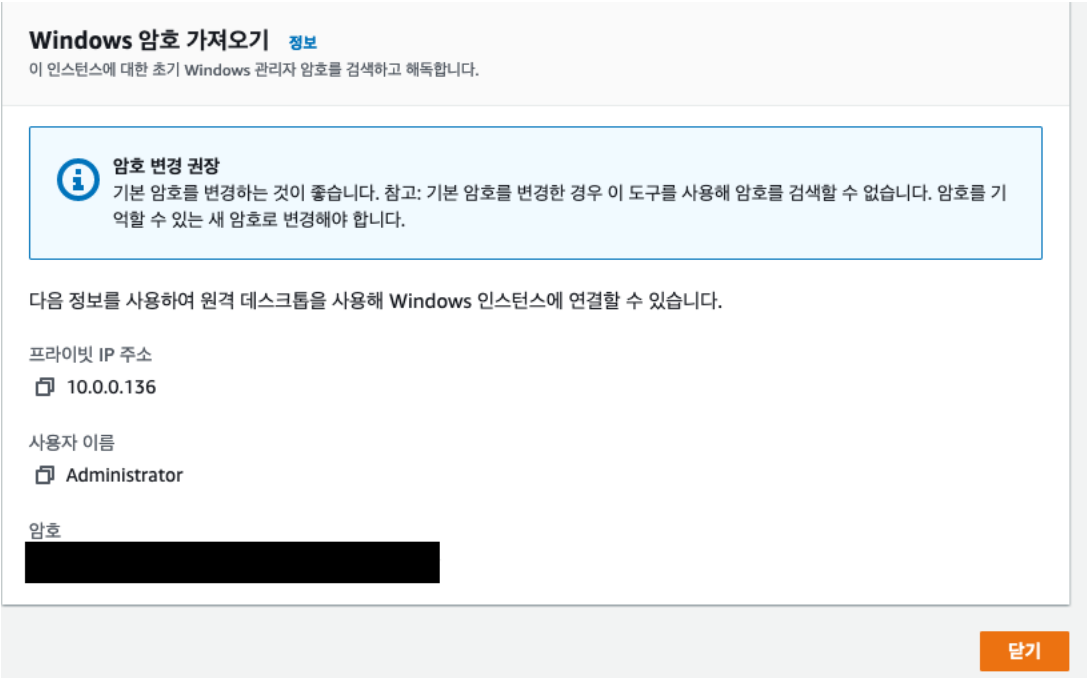
```
fields @timestamp
| parse @message "* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *" as account_id, action, az_id, bytes, dstaddr, dstport, end, fl
| filter action="REJECT"
| stats count(action) as redjects by srcaddr
| sort redjects desc
```

## 4. Traffic Mirror Setting

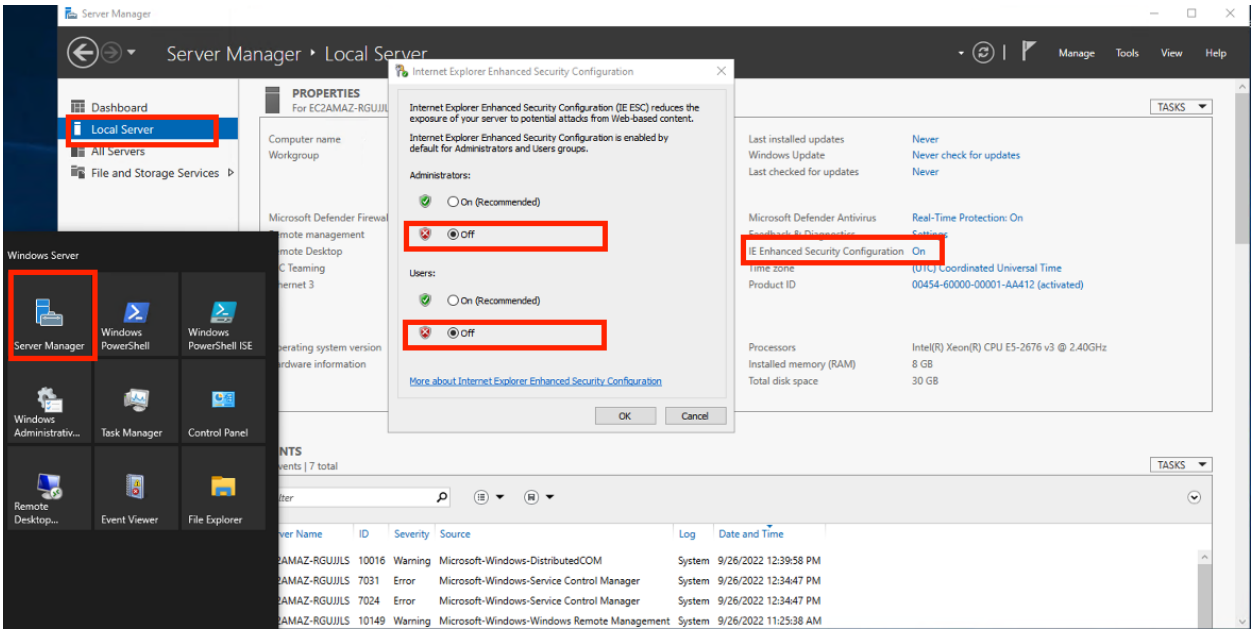
사전에 생성한 EC2의 암호를 먼저 찾아야 합니다. EC2 Dashboard로 돌아가서 Windows Instance를 선택합니다.




작업 → 보안 → windows암호 가져오기를 클릭합니다. 클릭후에 초기 EC2 생성전 만들어둔 Key를 업로드 합니다.




위와같이 접속 정보를 알 수 있습니다. Mac이라면 Microsoft Remote Desktop을 사용하여 접속합니다.

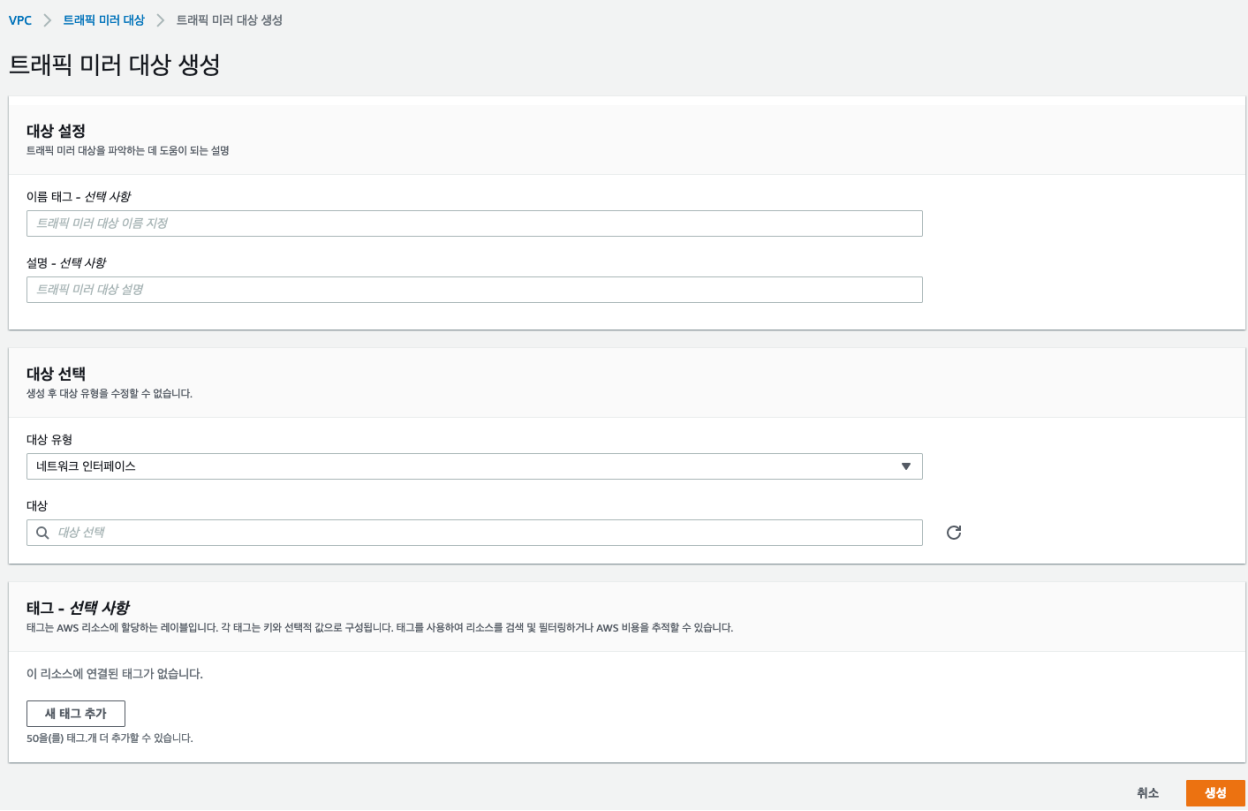



작업의 편의를 위하여 Internet Explorer Enhanced Security Configuration을 전부 Off시켜줍니다

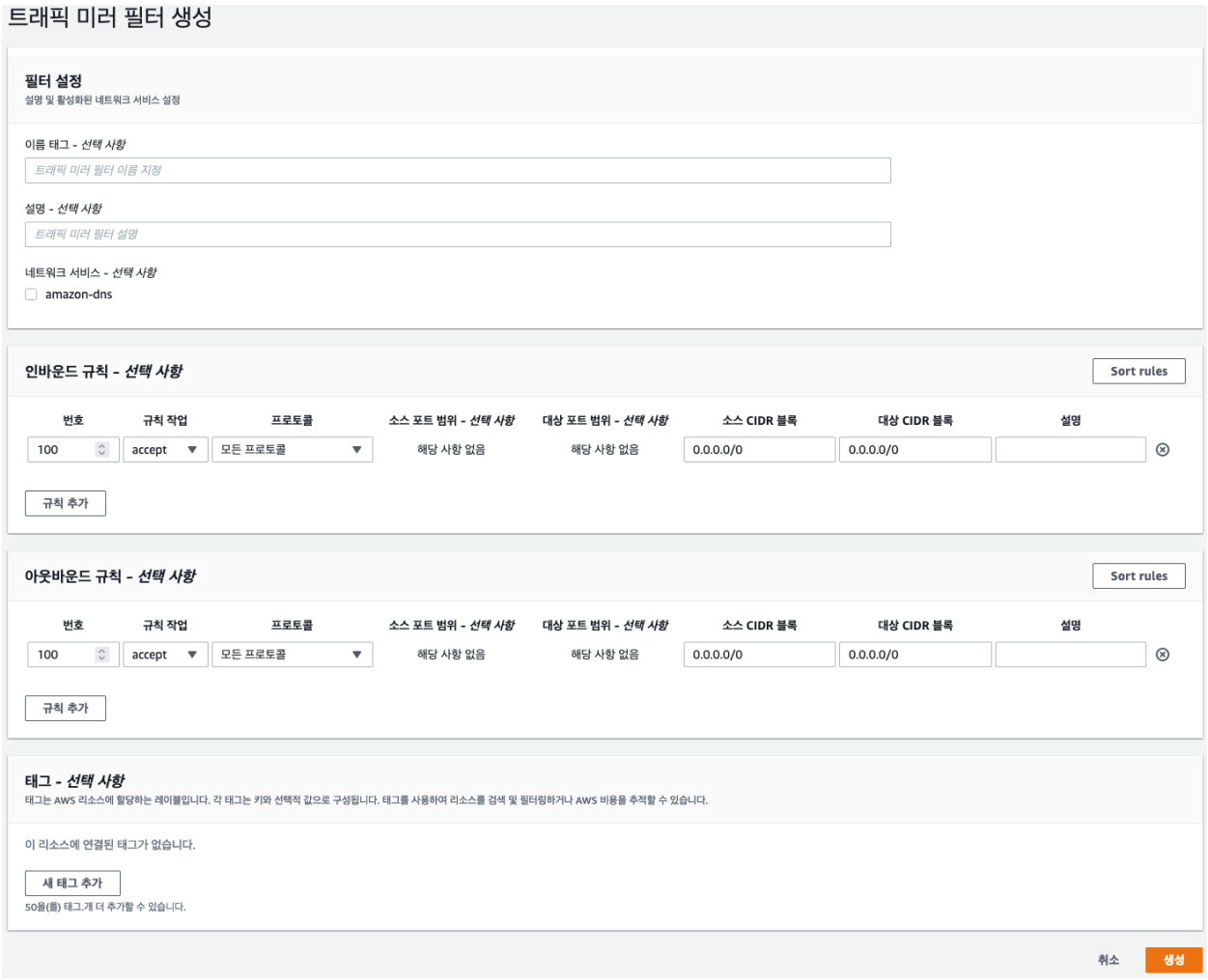
다음은 Wireshark를 Download및 설치합다.

1. https://www.wireshark.org/#download
2. Windows Installer 64-bit를 전부 Default로 설치합니다.

정상적으로 설치가 되었으면 Wireshark를 실행했을때 Ethernet이 보입니다.

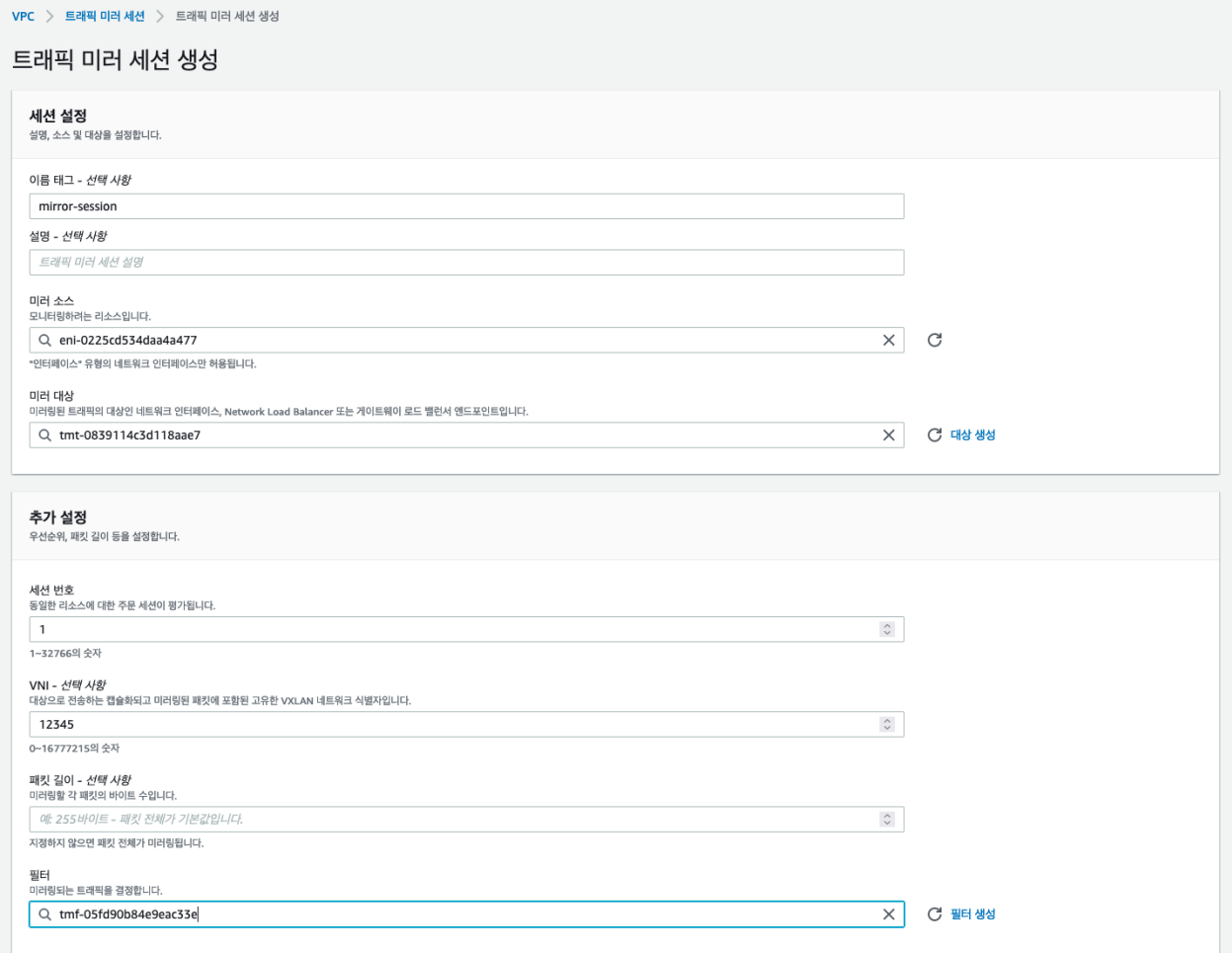



VPC Console로 돌아가 Mirror Target을 생성합니다. Target은 Network Interface로 선택후 Windows Instance의 eni를 선택합니다.



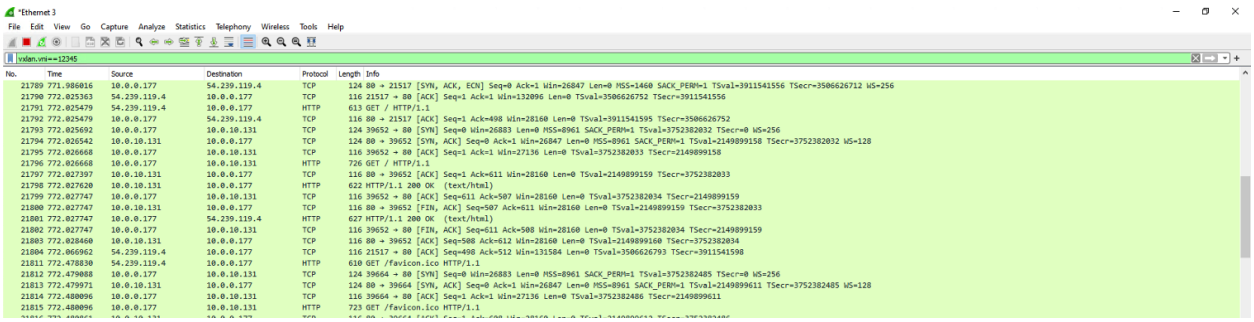

다음은 Mirror Filter를 생성합니다.

1. Inbound Rule
    a. 100/ accpet / all protocols / 0.0.0.0/0 / 0.0.0.0/0
2. outbound Rule
    a. 100/ accpet / all protocols / 0.0.0.0/0 / 0.0.0.0/0




다음은 Mirror Session을 생성합니다.

1. Mirror Source : 패킷 수집이 필요한 eni입니다.
2. Mirror Target : 패킷을 수집하여 트래킹할 Target입니다. 사전에 설정한 Target을 지정합니다
3. Session Number: 1 로 기입합니다.
4. VNI : 12345로 기입합니다. 옵셔널한 값이지만 구분의 용이를 위하여 설정합니다
5. Filter : 기존에 설정한 Filter를 지정합니다.




설정이 완료되면 Wireshark의 서버에서 Mirror Session에서 설정한 vni로 filter하여 트래픽을 확인할 수 있습니다(`vxlan.vni==12345`)