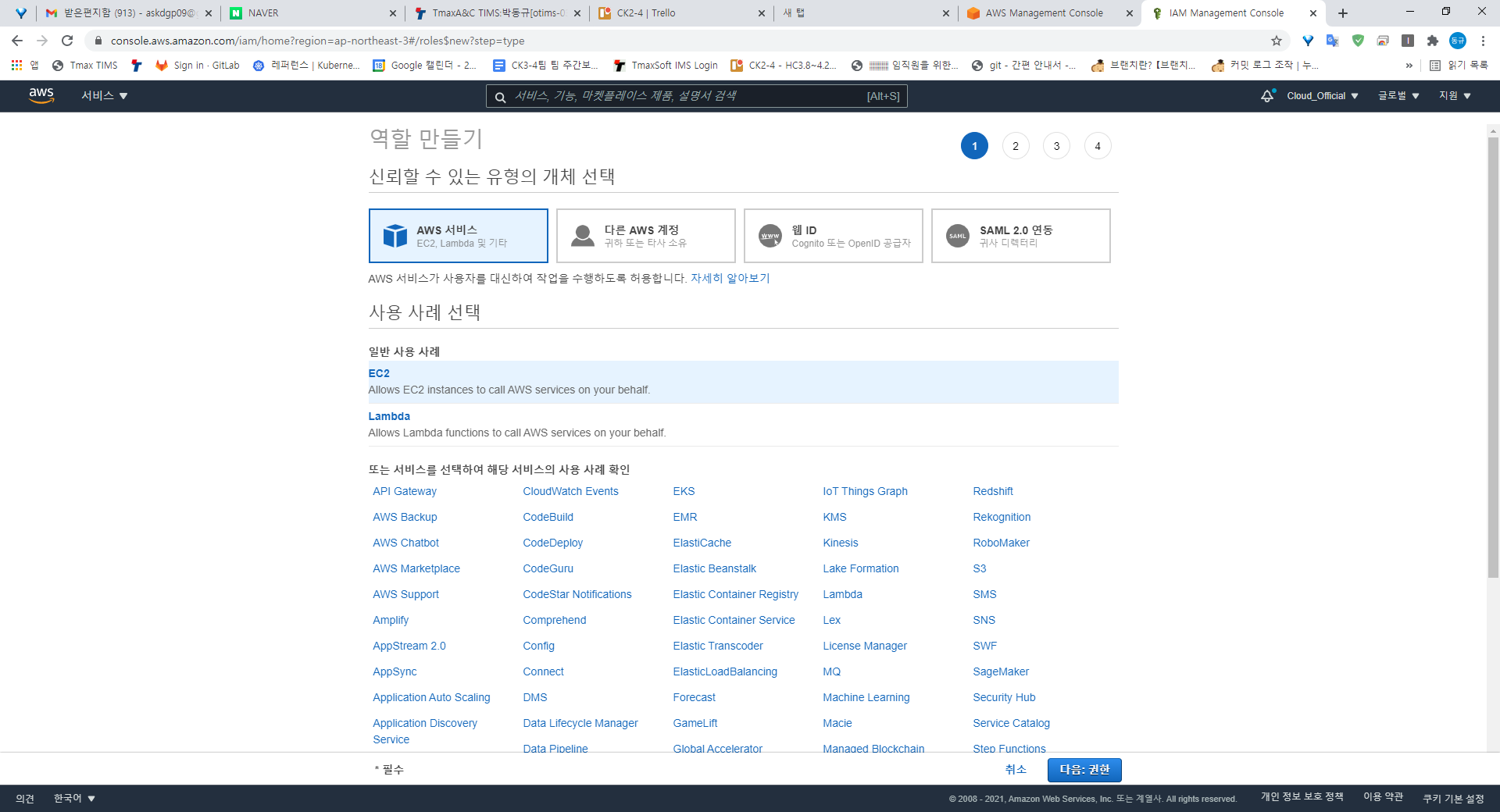
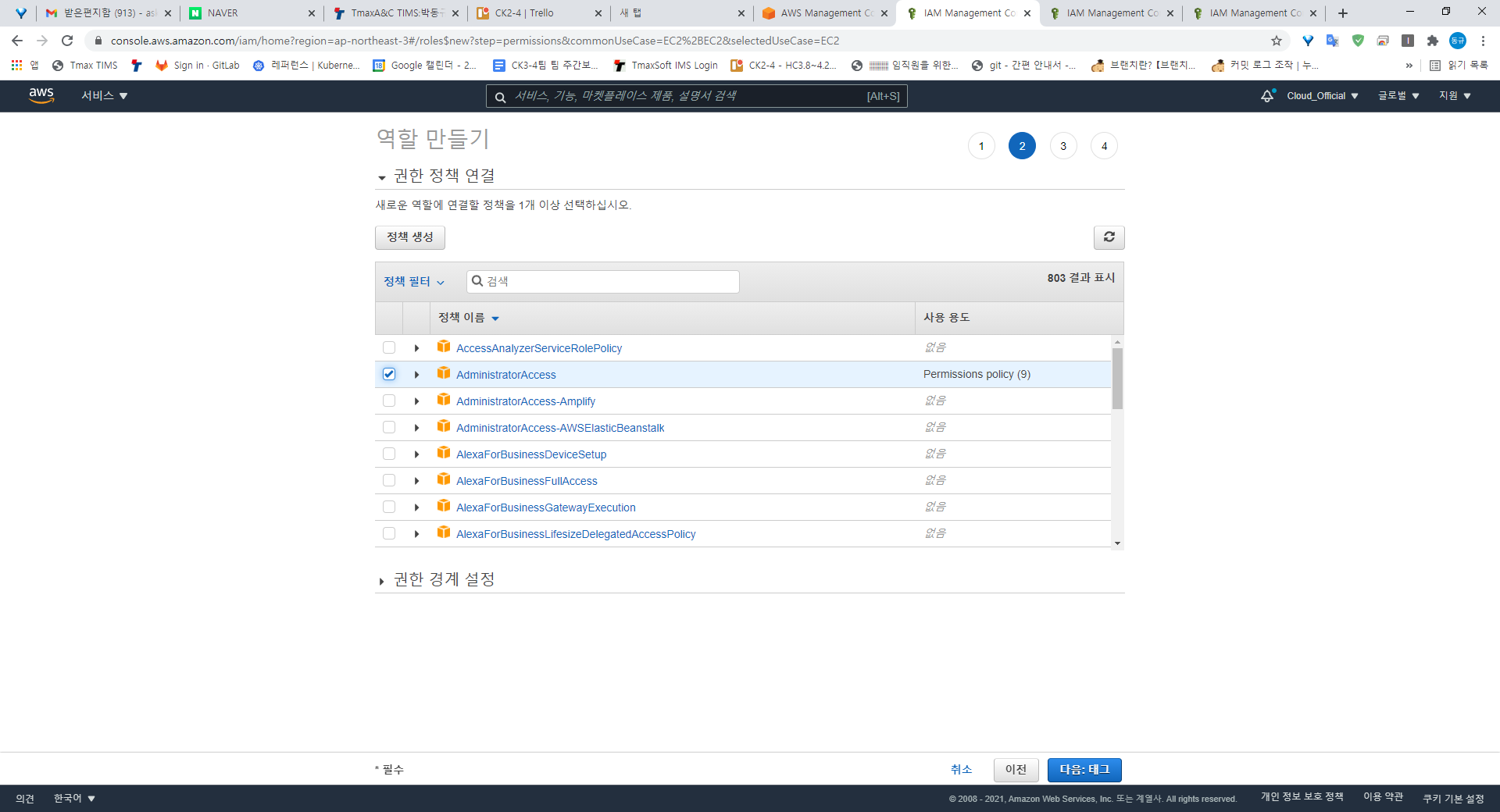
aws ec2에서 ebs, efs csi를 이용한 볼륨 생성 방법

1. IAM role 생성 및 ec2 instance에 연결

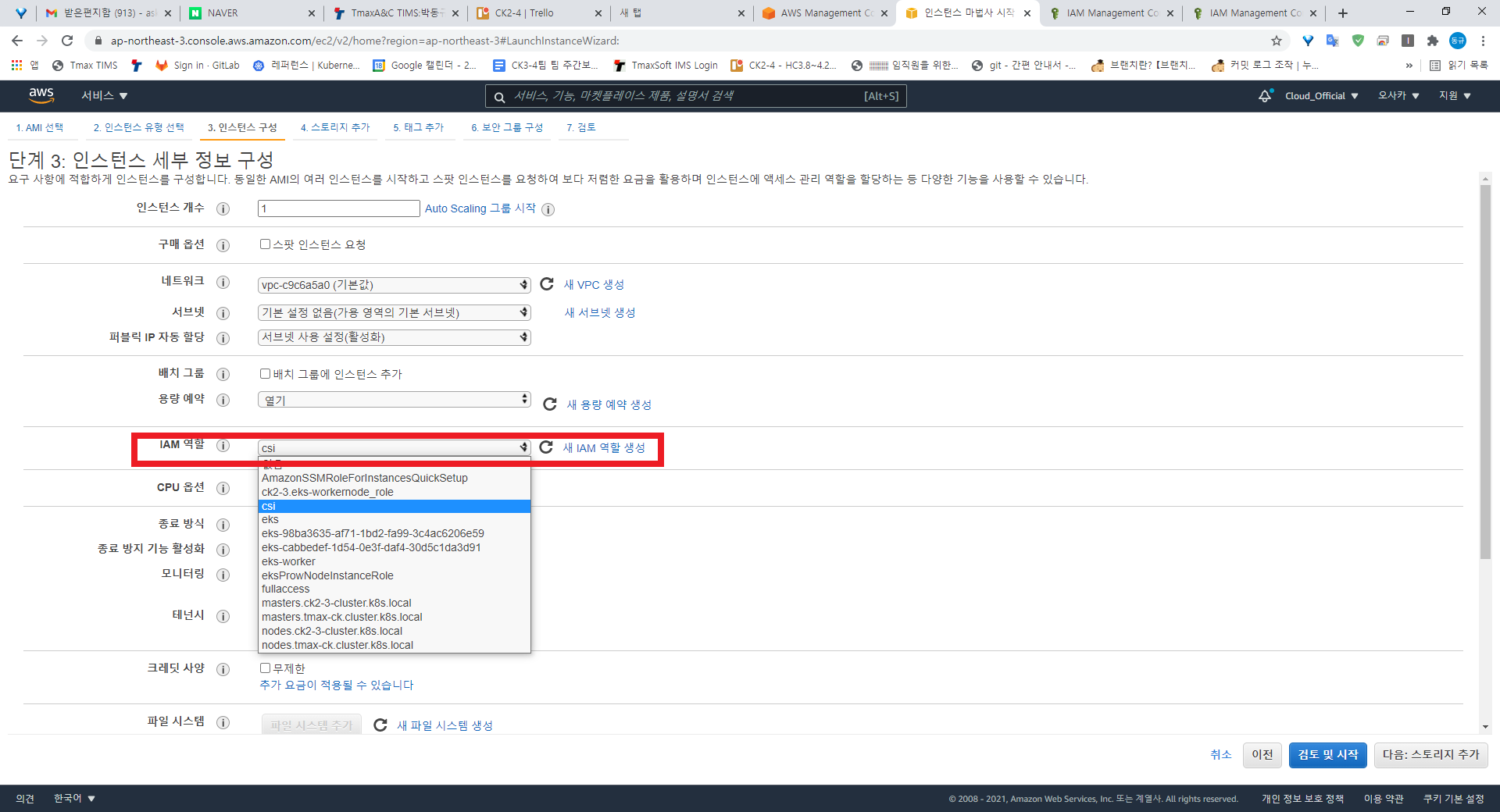
* 원래는 ebs, efs 권한만 있으면 가능하나, 일단은 admin 권한으로 ec2에 적용할 IAM role 생성
* AWS Management Console->IAM->역할(목록)->역할 만들기->EC2->AdministratorAccess-> 생성.



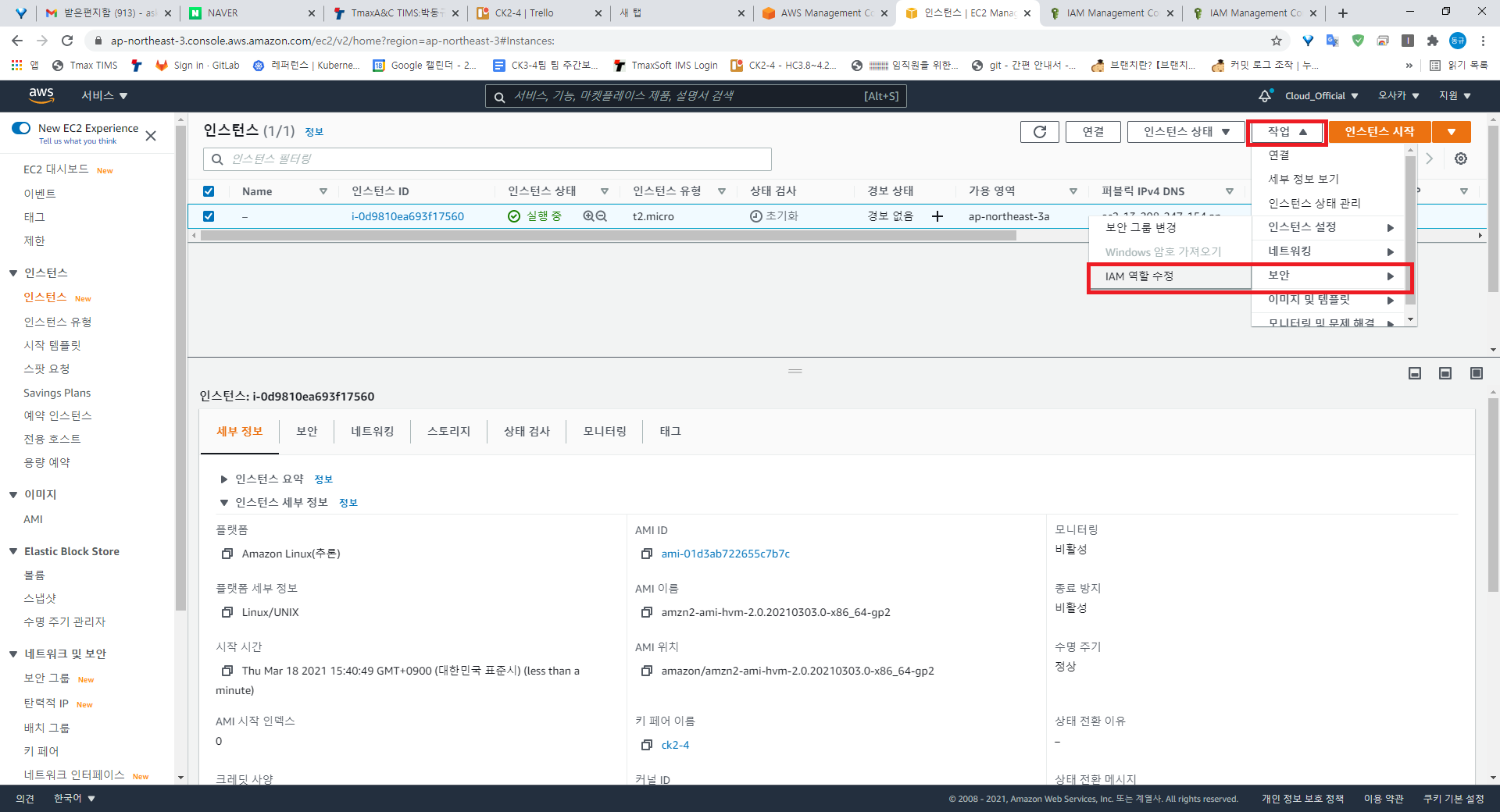


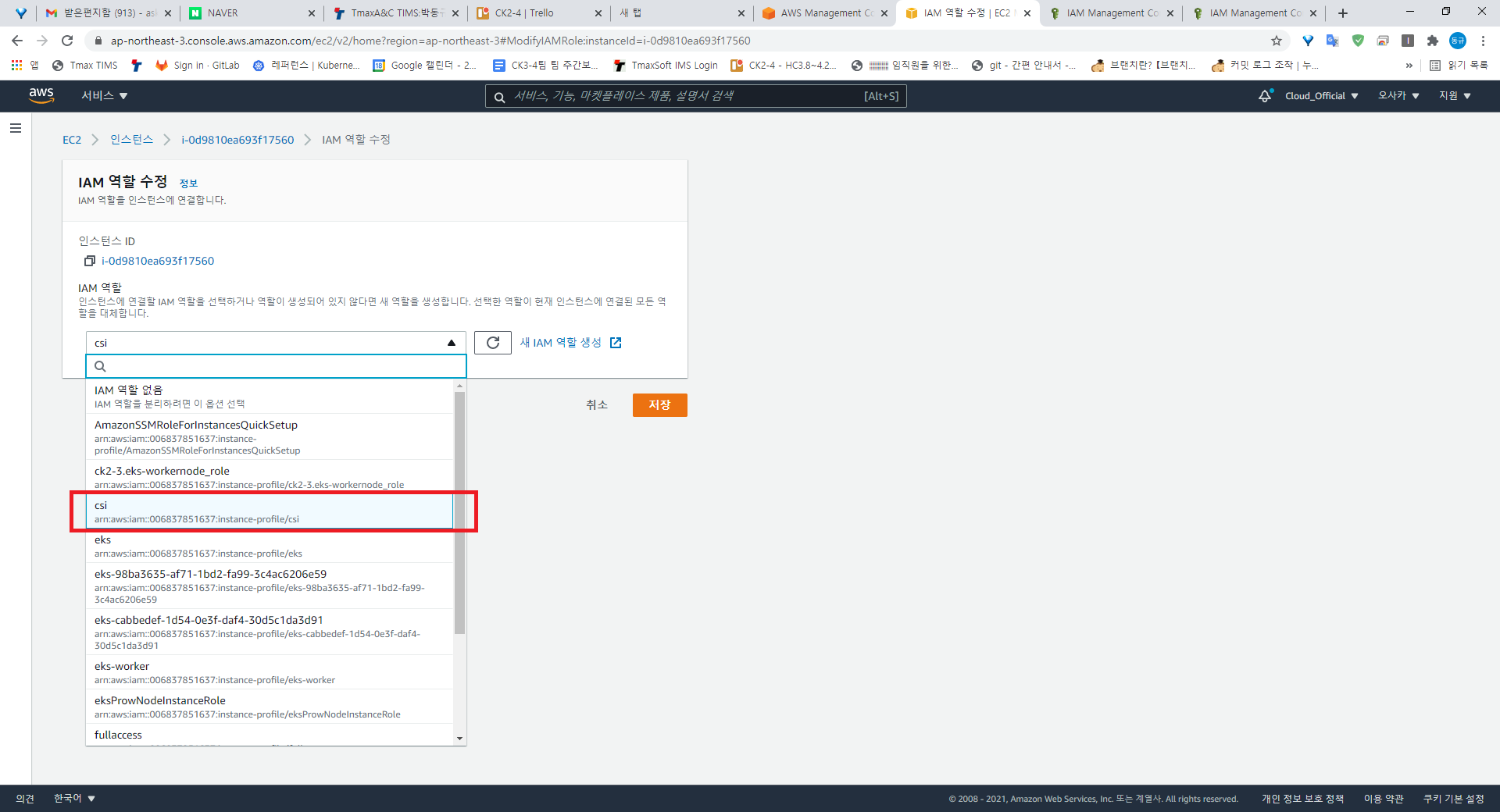
1. EC2 인스턴스에 IAM role 연결

* 1) instatnce 생성시 iam role 연결 (ex - iam role: csi)
  + 인스턴스 생성시 인스턴스 세부 정보 구성에서 IAM role을 생성한 IAM role로 설정



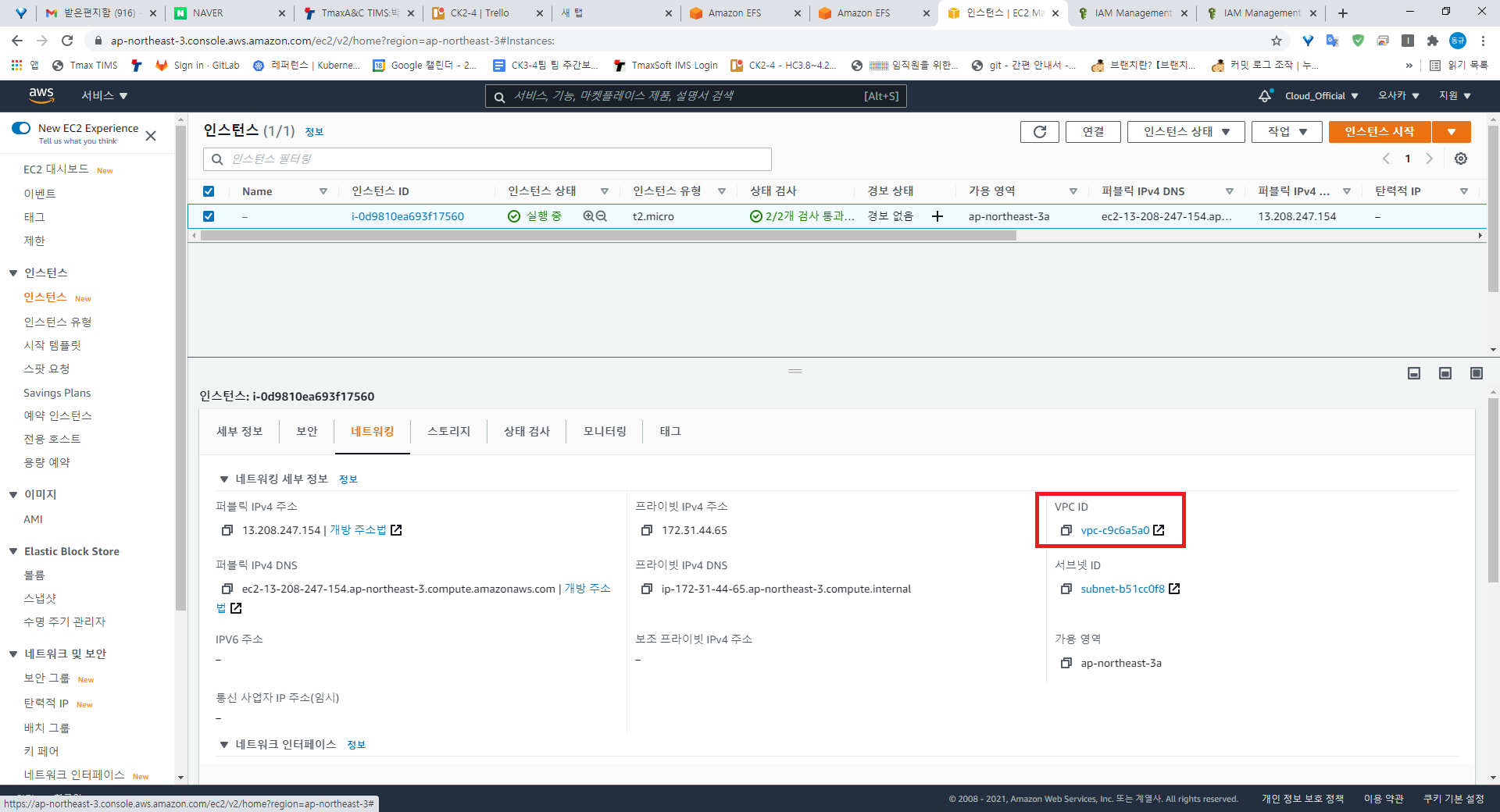
* 2) 이미 instance 생성되어 있을 경우, instance list 창에서 작업-> 보안-> IAM 역할 수정 에서 IAM role 설정

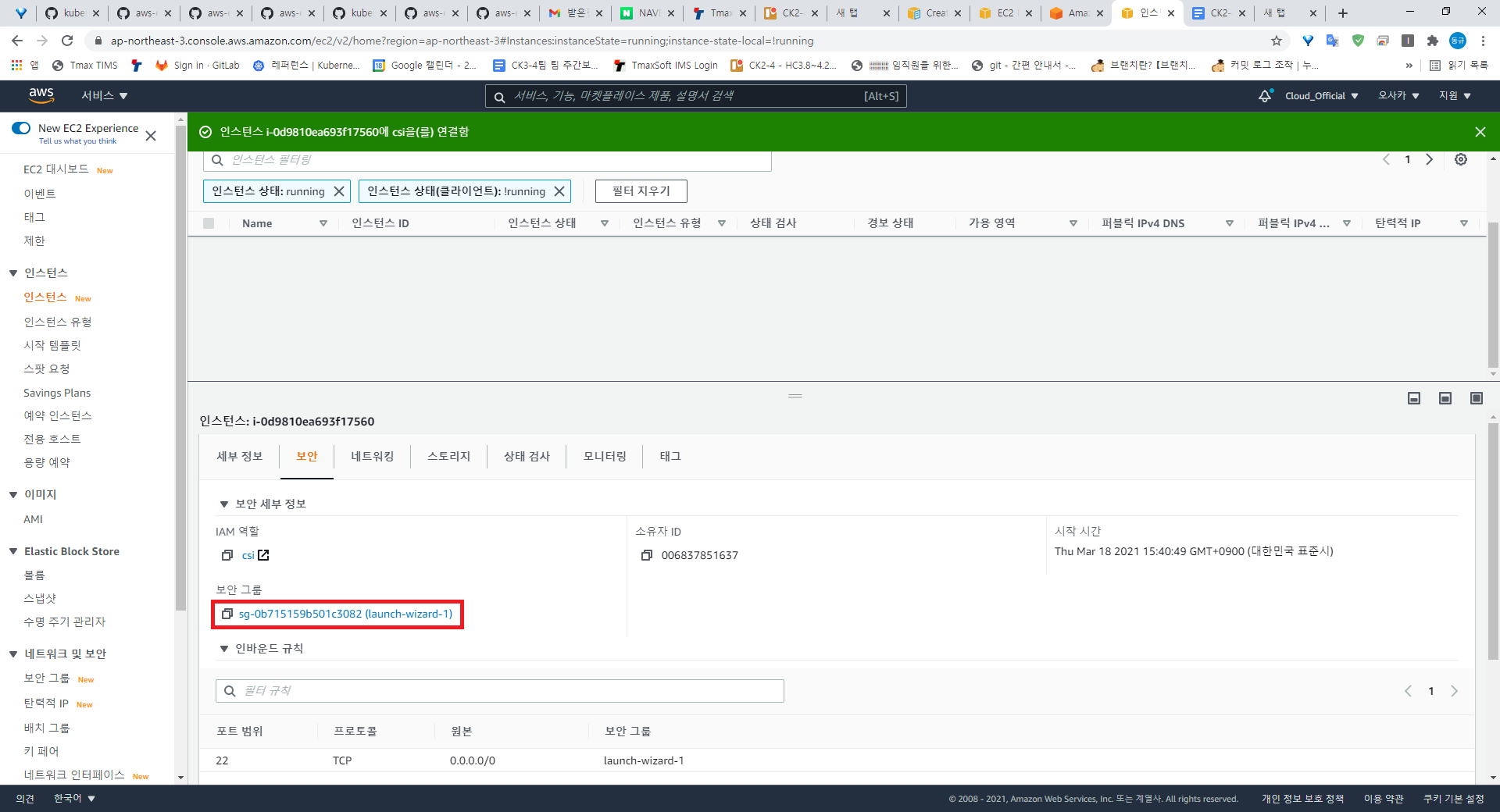




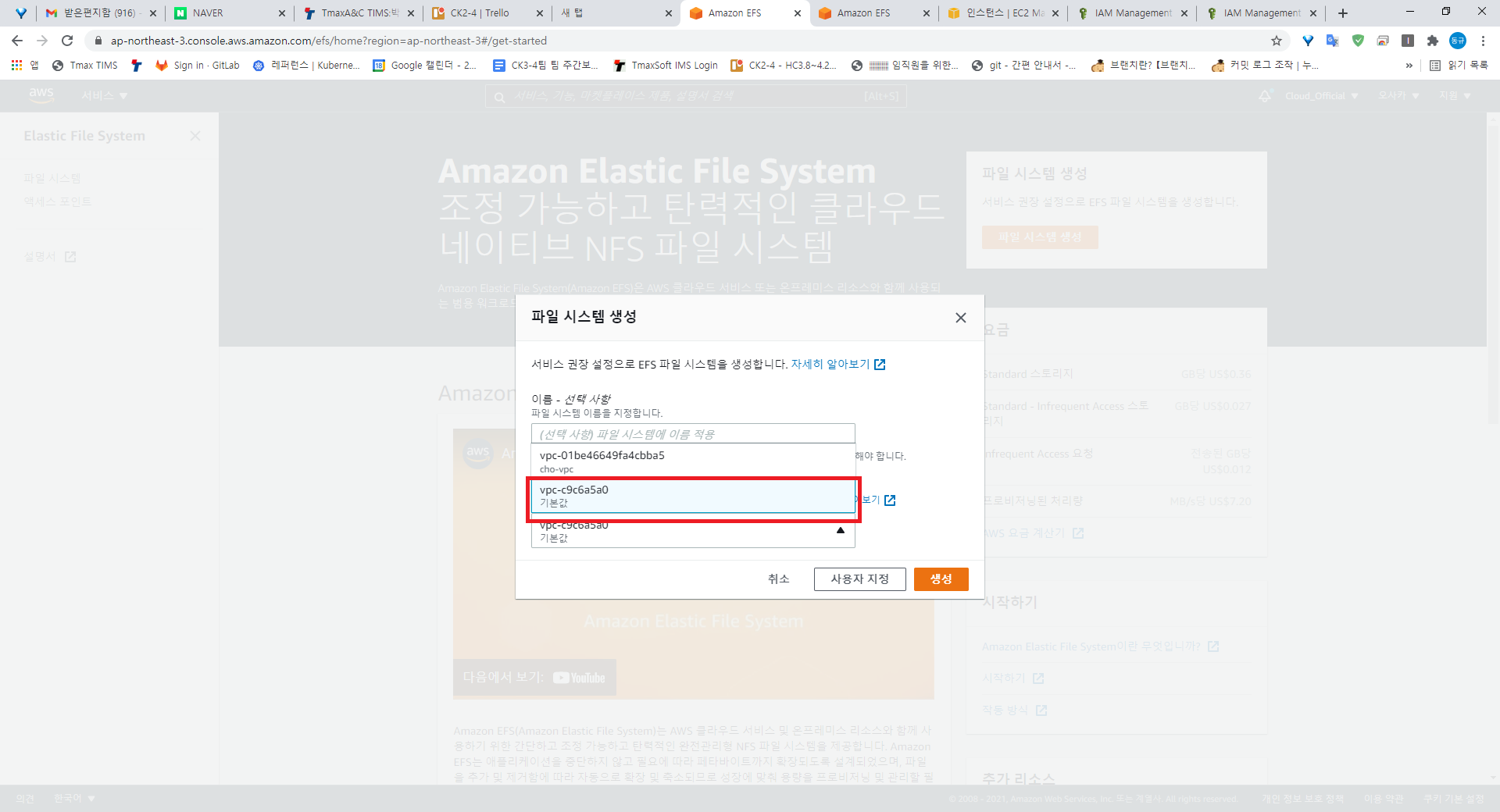
1. efs 생성 및 security group inbound, outbound 규칙 설정

* 참고: efs를 사용하기 위해서는 사용하는 ec2 instance에서 efs 접근 허용 및 efs에서 ec2 instance 접근을 허용하여야 합니다. 이를 위해 efs의 security group과 ec2 instance의 security group에 inbound, outbound 설정이 필요합니다.
* 참고: ec2 instance의 VPC는 instance 창에서 확인 가능
* 참고: ec2 instance의 security group은 instance 창에서 확인 가능

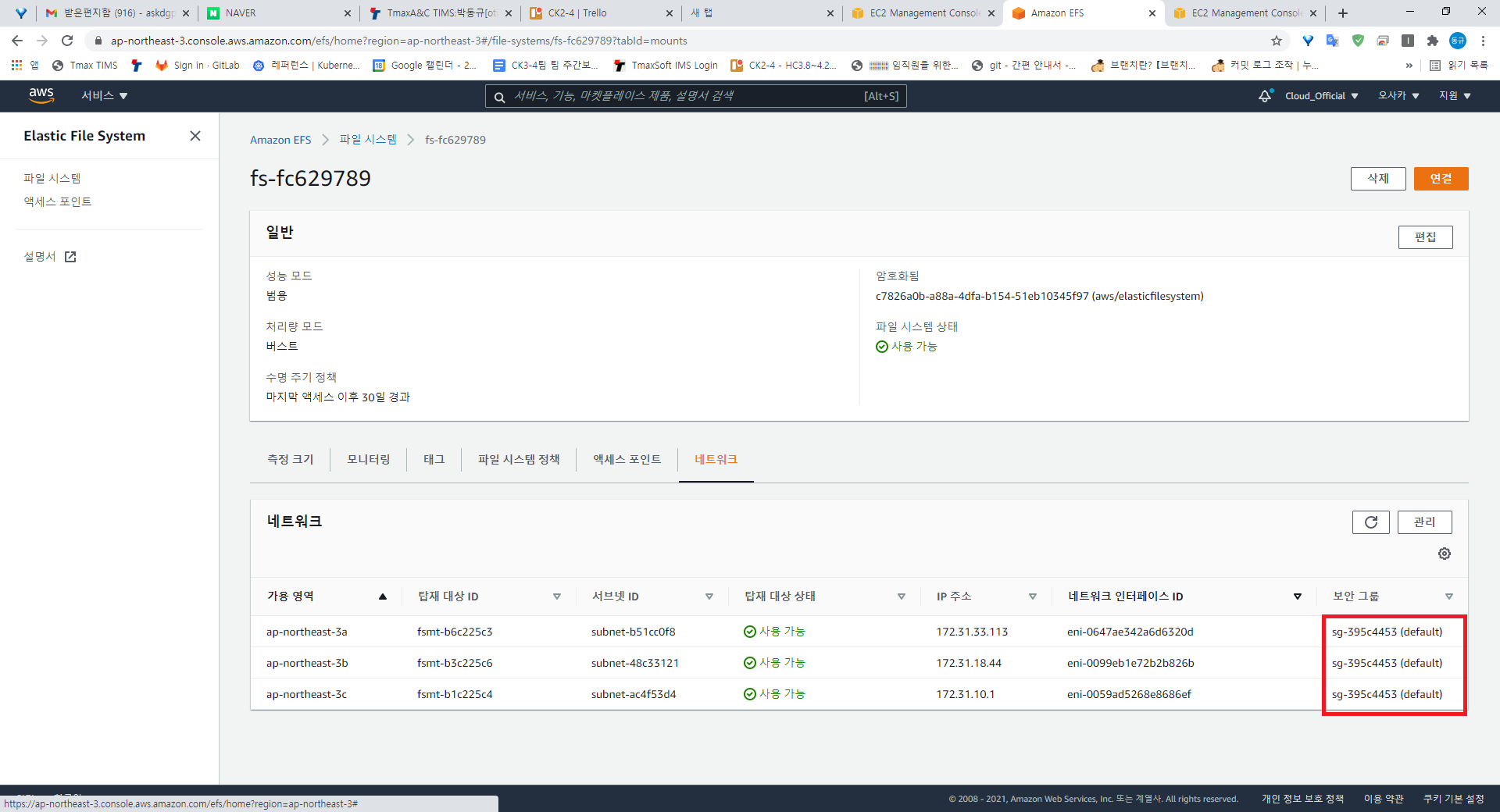




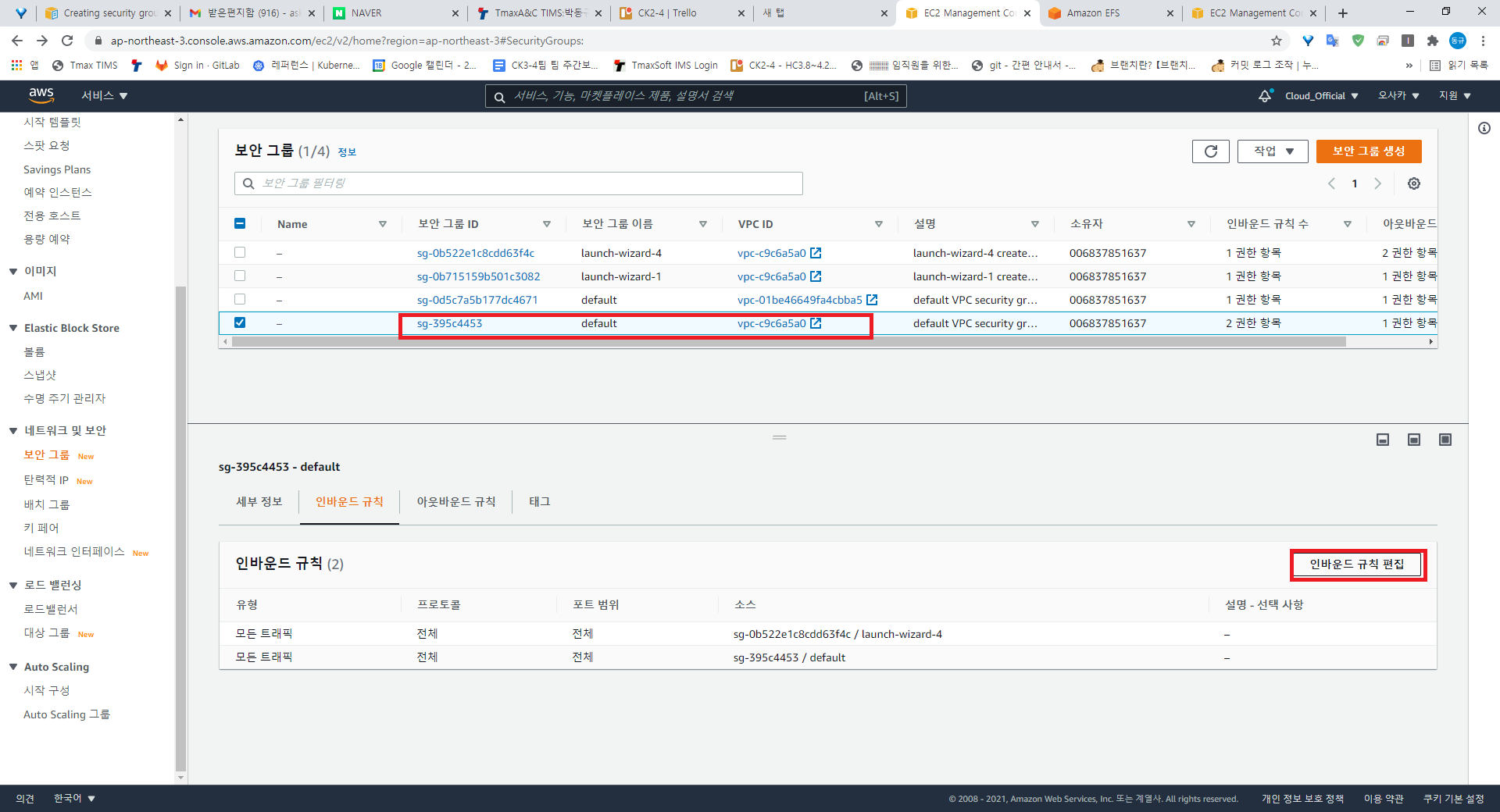
* AWS Management Console-> EFS -> 파일 시스템 생성
  + VPC 선택은 ec2 instance와 동일한 VPC를 선택해야 efs 사용 가능



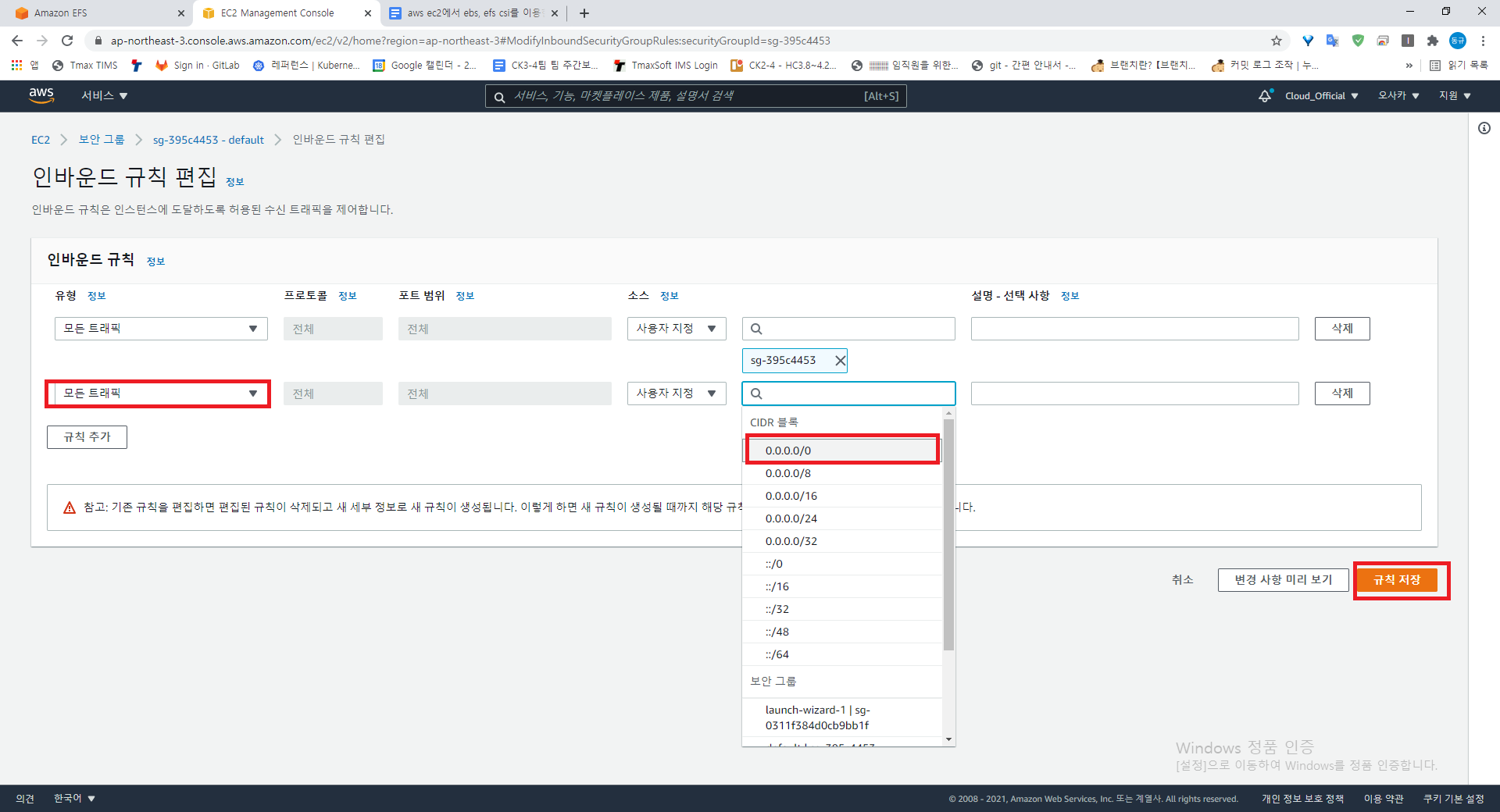
* 참고로, 사용자 지정을 통해 security group을 따로 설정하지 않으면 efs는 해당 vpc의 default security group으로 생성됩니다.



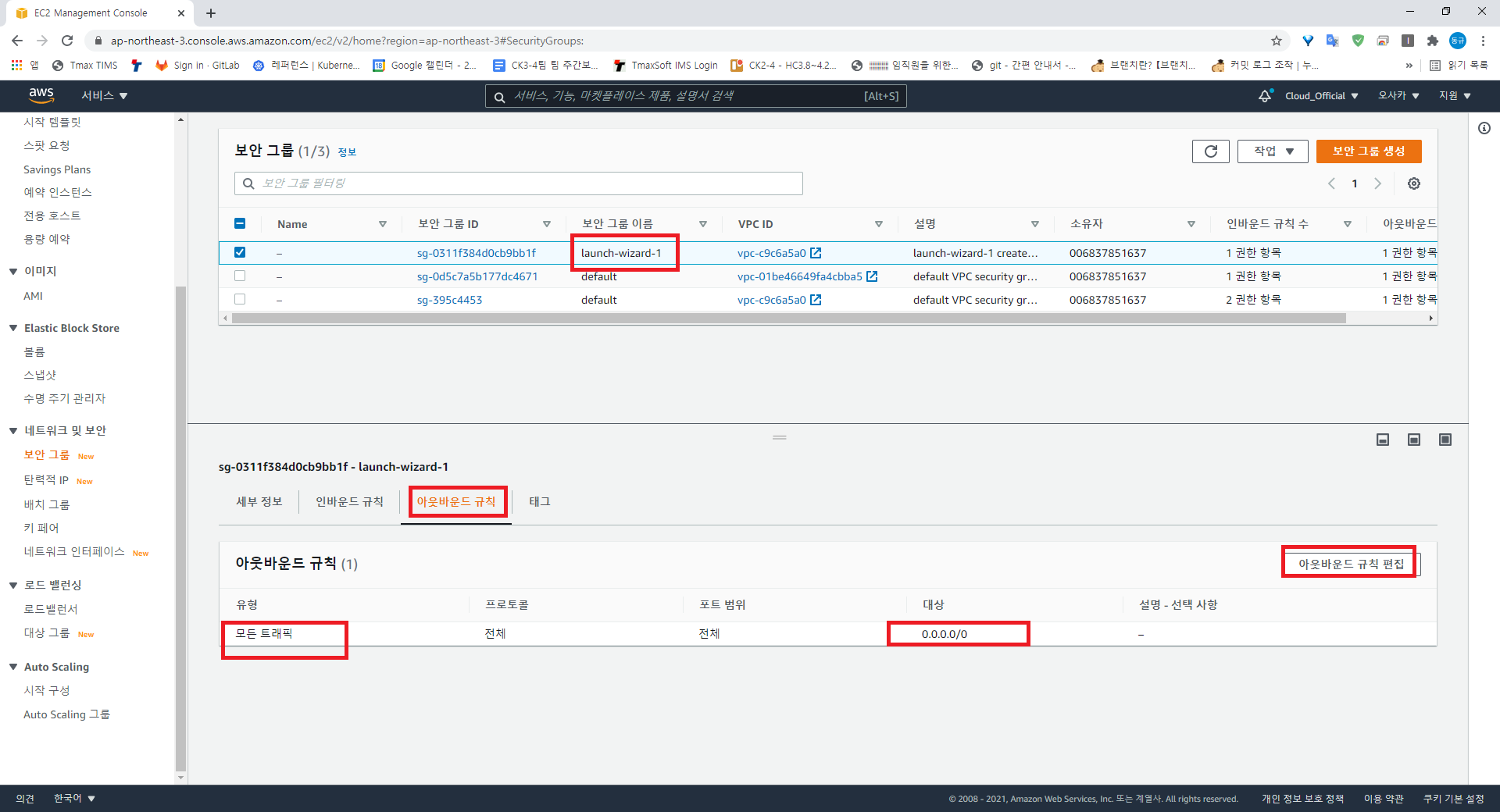
* security group의 inbound, outbound 규칙 설정
  + AWS Management Console -> ec2 -> 보안 그룹(목록)
    - efs security group(vpc의 default security group)의 inbound 규칙 편집

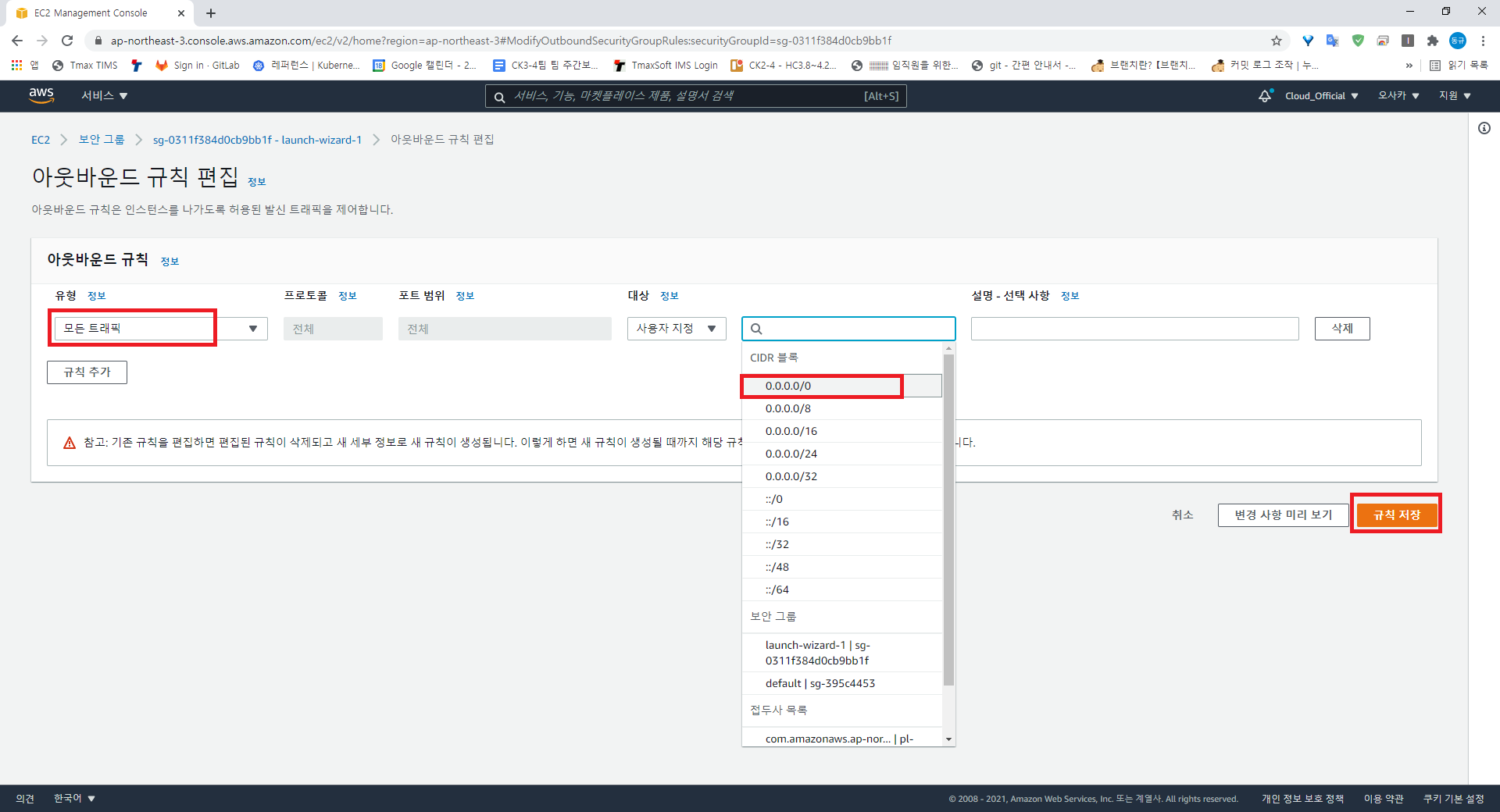


* + - efs security group의 inbound 규칙에 접근 허용할 ec2 instance들의 모든 security group을 nfs로 추가하는 것이 정식 방법이지만, 해당 방법에서는 간단한 사용을 위해 모든 트래픽 허용 규칙을 추가합니다.



* 접근 허용할 모든 ec2 instance의 security group의 outbound 규칙에 efs security group을 NFS로 추가하는 것이 정식방법이지만, 간단한 사용을 위해 모든 트래픽 허용 규칙을 추가합니다. 참고로, ec2 instance의 security group의 outbound 규칙에 이미 모든 트래픽 허용으로 설정되어 있을 수도 있습니다.
  + ex) ec2 instance의 security group: launch-wizrd-1





1. ebs csi 배포 및 volume 생성

* ec2 instance에 kubernetes는 이미 구축되어 있다고 가정함
* 1) ebs csi 데몬 배포 (git 설치했다고 가정)
  + kubectl apply -k "github.com/kubernetes-sigs/aws-ebs-csi-driver/deploy/kubernetes/overlays/stable/?ref=release-0.9"
* 2) dynamic provisoning을 통한 volume 생성 및 pod 연결
  + <https://github.com/kubernetes-sigs/aws-ebs-csi-driver/tree/master/examples/kubernetes/dynamic-provisioning/specs>
    - 해당 link에 예시 파일 존재 (storageclass.yaml, calim.yaml, pod.yaml)
    - 해당 예시에는 storageclass에 volumeBindingMode: WaitForFirstConsumer 가 존재하여 pvc만 생성시에는 pending 상태이고, 사용할 pod까지 생성해야 실제로 ebs로부터 volume 생성 및 pod에 mount 진행함

1. efs csi 배포 및 volume 생성

* 1) efs csi 데몬 배포
  + kubectl apply -k "github.com/kubernetes-sigs/aws-efs-csi-driver/deploy/kubernetes/overlays/stable/?ref=release-1.2"
* 2) dynamic provisioning
  + 위에서 설명했던 efs 생성 과정을 통해 efs 는 이미 생성되어 있어야함
  + 해당 link에 예시 파일 존재 (storageclass.yaml, pod.yaml)
    - https://github.com/kubernetes-sigs/aws-efs-csi-driver/tree/master/examples/kubernetes/dynamic\_provisioning/specs
  + 예시에서 storageclass.yaml의 fileSystemId 수정(생성한 efs ID 입력 - efs ID는 aws의 efs 창에서 확인 가능)

