# Deep Learning 환경 구성

## 1. 인스턴스 생성

AWS Console에 접속하여 새로운 EC2 인스턴스를 생성합니다. 다음의 정보를 참고하여 인스턴스를 생성하며, 기타 설정은 모두 기본 설정대로 구성합니다.

Region : us-west-2 (Oregon)

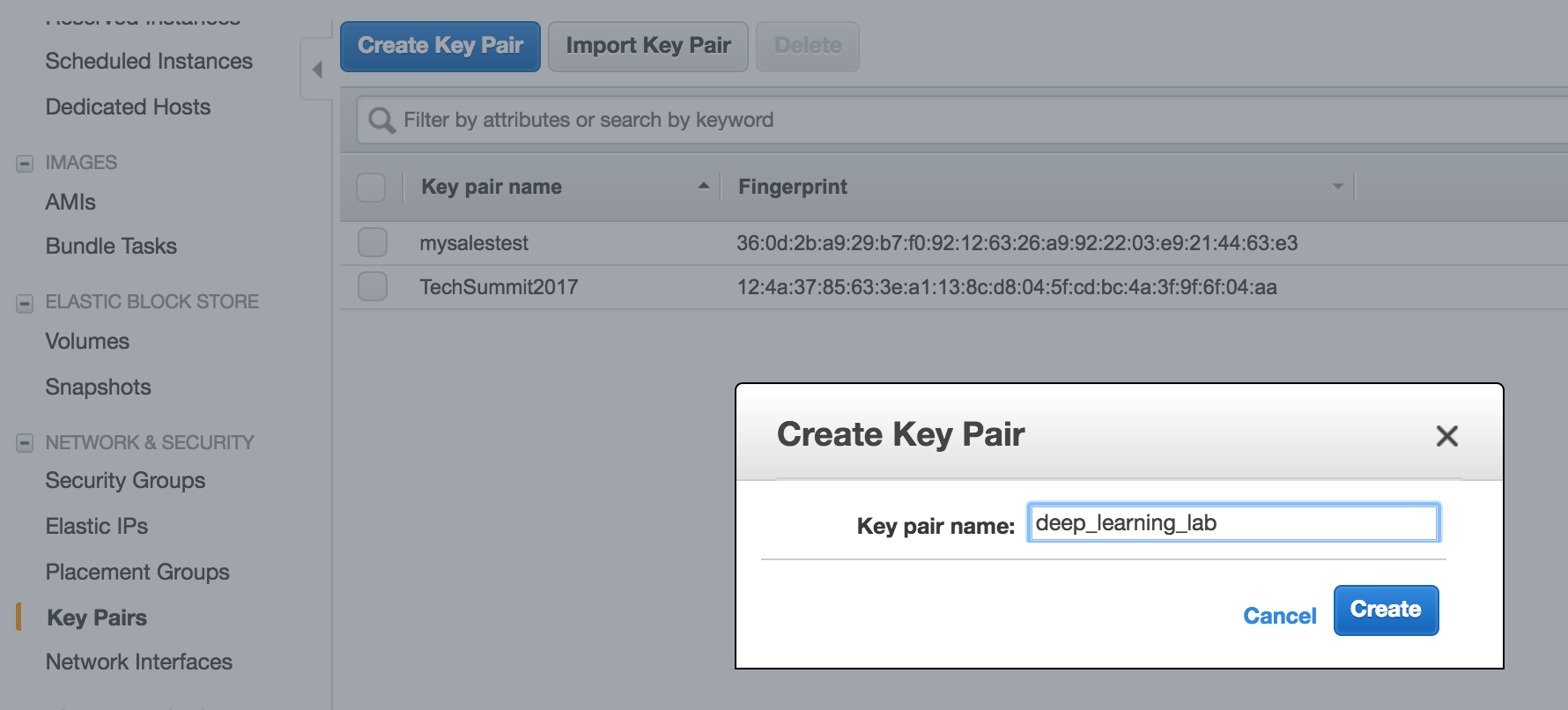
AMI 이름: Deep Learning AMI

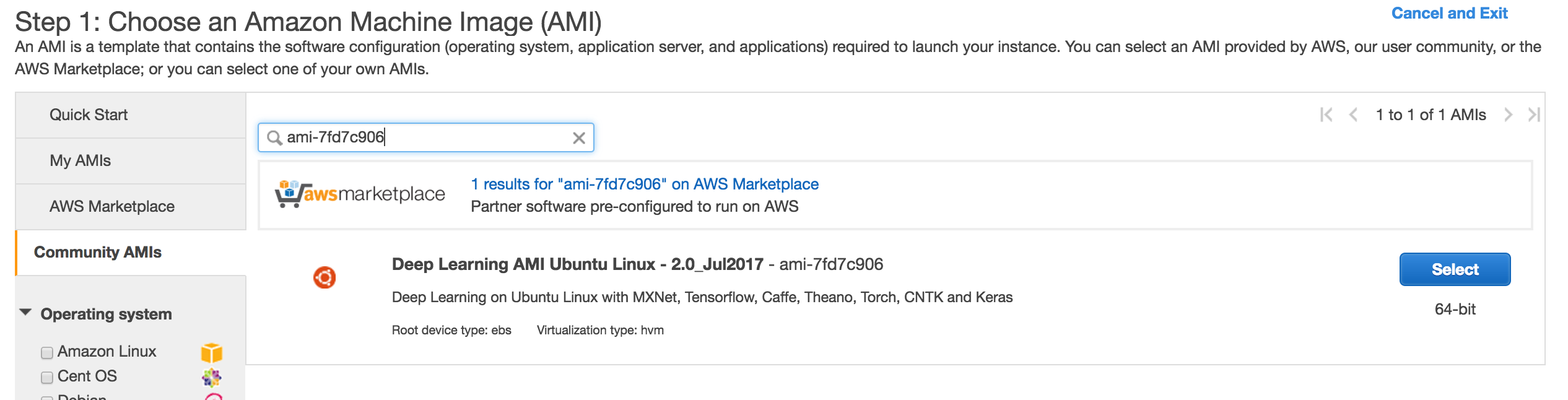
Instance type : p2.xlarge, g2.xlarge, 또는 c4.8xlarge

VPC : Default VPC (Auto-assign Public IP : Enable)

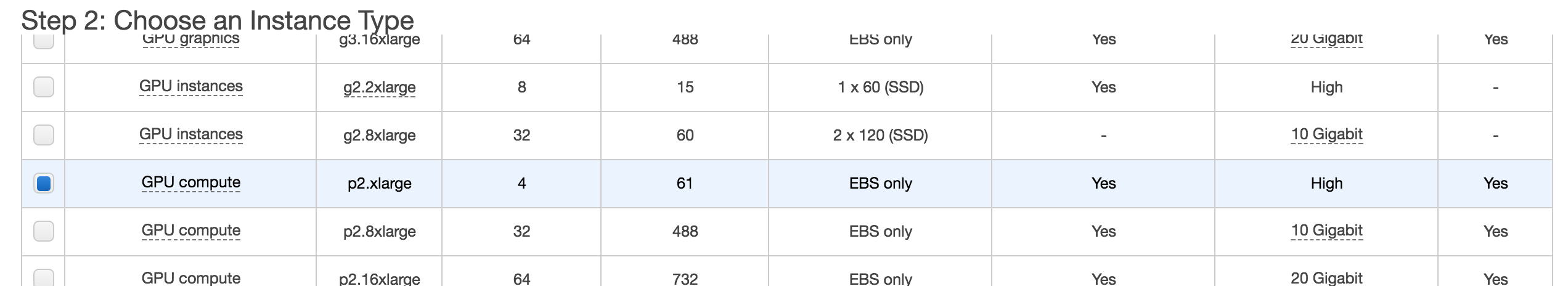
Tag : 자유롭게 Name Tag를 사용하시기 바랍니다.

Key Pair : us-west-2 (Oregon)에 미리 Key pair를 생성하지 않았다면, EC2 콘솔 화면에서 “Key Pairs”메뉴를 선택하여 Key Pair를 생성하고, \*.pem 파일을 다운로드합니다.





\* AWS Market Place에서 Deep Learning AMI로 검색합니다.



\* p2.xlarge의 기본 limit는 1개입니다. (limit에 주의합니다.)

## 2 인스턴스 접속

인스턴스 생성이 완료되면, Putty (Windows) 혹은 Terminal (MAC OS X)를 사용하여 호스트에 접속합니다. 이 때 user id는 ubuntu를 사용합니다.

### 2.1 Windows (Putty) 를 사용하여 접속

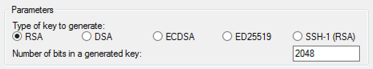
### 2.1.1 SSH 키 변환

## PuTTYgen을 사용하여 프라이빗 키 변환 (http://docs.aws.amazon.com/ko\_kr/AWSEC2/latest/UserGuide/putty.html)

PuTTY에서는 Amazon EC2에서 생성된 프라이빗 키 형식(.pem)을 기본적으로 지원하지 않습니다. PuTTY에는 PuTTYgen이라는 도구가 있는데, 이 도구는 키를 필요한 PuTTY 형식(.ppk)으로 변환할 수 있습니다. PuTTY를 사용하여 인스턴스에 연결하기 전에 프라이빗 키를 이 형식(.ppk)으로 변환해야 합니다.

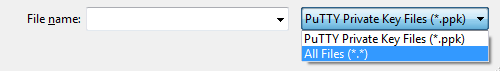
**개인 키를 변환하려면**

1. PuTTYgen을 시작합니다(예: [**Start**] 메뉴에서 [**All Programs > PuTTY > PuTTYgen**] 선택).
2. **Type of key to generate**에서 **RSA**를 선택합니다.



이전 버전의 PuTTYgen을 사용하는 경우 [**SSH-2 RSA**]를 선택합니다.

1. **Load**를 선택합니다. 기본적으로 PuTTYgen에는 확장명이 .ppk인 파일만 표시됩니다. .pem 파일을 찾으려면 모든 유형의 파일을 표시하는 옵션을 선택합니다.



1. 인스턴스를 시직할 때 지정한 키 페어에 대한 .pem 파일을 선택한 다음 [**Open**]을 선택합니다. [**OK**]를 선택하여 확인 대화 상자를 닫습니다.
2. [**Save private key**]를 선택하여 PuTTY에서 사용할 수 있는 형식으로 키를 저장합니다. PuTTYgen에서 암호 없이 키 저장에 대한 경고가 표시됩니다. **Yes**를 선택합니다.

참고

프라이빗 키의 암호는 추가 보호 계층이므로 프라이빗 키가 공개되었더라도 이 암호가 없으면 사용할 수 없습니다. 암호문 사용의 단점은 인스턴스에 로그온하거나 인스턴스에 파일을 복사하기 위해 사용자가 개입해야 하기 때문에 자동화를 어렵게 만든다는 것입니다.

1. 키 페어에 사용된 키에 대해 동일한 이름을 지정합니다(예: my-key-pair). PuTTY가 자동으로 .ppk 파일 확장자를 추가합니다.

이제 개인 키가 PuTTY에 사용하기에 올바른 형식으로 되어 있으므로 PuTTY의 SSH 클라이언트를 사용하여 인스턴스에 연결할 수 있습니다.

### 2.1.2 SSH 터널링 설정

Putty를 열고, 세션정보를 이전과 같이 입력합니다. (port = 22, 호스트(IP) 입력)

Connection -> SSH -> Tunnels 메뉴로 이동하여 “destination port” box항목에 127.0.0.1:8888을 입력하고, Local and auto 옵션을 선택합니다. “source port” 항목에는 8888을 입력한 다음 “add” 버튼을 클릭합니다.

### 2.1.3 SSH 접속

<http://docs.aws.amazon.com/ko_kr/AWSEC2/latest/UserGuide/putty.html> 를 참고하여 접속합니다.

### 2.2 MAC OS X를 사용하여 접속

Terminal (혹인 기타 사용하는 툴)을 사용하여 local shell을 열고, 다음의 ssh command를 사용하여 접속합니다.

$ ssh –i /<pem file이 있는 경로>/\*.pem –L 8888:localhost:8888 ubuntu@<ec2 instance의 public ip address>

## 3.EC2 인스턴스에 로그인한 후, MXNet 업그레이드

$ sudo pip install --upgrade mxnet-cu80

$ sudo pip3 install --upgrade mxnet-cu80

## 4. Github 동기화

역시, EC2 인스턴스에서 명령을 수행합니다. Lab에서 사용할 Gibhub를 동기화합니다.

새로운 SSH Terminal을 열어 (single / distributed 환경 모두) 다음의 command를 통하여 lab 자료들을 sync합니다. (home directory에서 바로 실행)

$ git clone https://github.com/muhyun/deeplearning-on-aws-workshop.git

## 4. Jupyter Notebook 실행

Jupyter notebook은 실행 코드, 계산식, 시각화 및 설명을 담을 수 있는 오픈 소스 웹 어플리케이션입니다. 데이터 클리닝, 변환, 수학적 시뮬레이션, 통계 모델링, 머신 러닝 등에 폭넓게 사용되고 있습니다.

Deep Learning AMI를 통하여 인스턴스를 구동하면, 이미 Jupyter Notebook이 설치되어 있습니다. 다음의 순서로 구성하고, 접속 테스트를 수행합니다. 이 단계가 진행되기 전 단일 환경 인스턴스 및 분산 환경의 마스터 노드에 각각 SSH 세션을 가지고 있어야 합니다.

$ cd deeplearning-on-aws-workshop

$ jupyter notebook

notebook이 실행되면, foreground상태로 멈춰있게 됩니다. 이 상태로 해당 창을 최소화하고 유지하시는 편이 안전합니다.

노트북의 웹 브라우저를 열고 jupyter notebook 실행 결과 나온 URL을 복사해서, jupyter notebook을 접근하시면 됩니다.

감사합니다. 준비를 모두 마치셨습니다.