

# AWS Deploy 과정을 통해 리눅스 환경 탐험하기



신촌 심화 1팀  
서강대학교 멋사

# 0. 사전 준비과정

---

- a. AWS를 가입하고 EC2 인스턴스 를 생성한다.
- b. Pem key를 다운로드하여 하드속에 소중히 보관한다.
- c. 프로젝트를 git repository에 저장해 놓는다.

## Select an existing key pair or create a new key pair



A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. [Learn more about removing existing key pairs from a public AMI.](#)

Choose an existing key pair ▼

### Select a key pair

trippie ▼

☐ I acknowledge that I have access to the selected private key file (trippie.pem), and that without this file, I won't be able to log into my instance.

[Cancel](#)

[Launch Instances](#)

# 1. chmod 400 <이름>.pem

---

AWS가 내가 가진 키가 유효한지 확인하기  
위해서는 권한을 변경해줘야 한다.

- chmod : 리눅스 접근 권한 변경 명령어
  - 400: 절대모드 상 읽기만 가능

# \*파일 권한?

---

## 사용자

파일 소유자 / 파일 소유 그룹 멤버 / 기타 사용자

## 권한

d / r / w / x / -

파일 디렉토리 인지 / 읽기 / 쓰기 / 실행하기 / 불가

drwx-----

-rw-rw----

-r-----

-rw-rw----

\*권한 확인하는 명령어 ls -l

d	r	w	x	r	-	x	r	-	x
	$2^2$	$2^1$	$2^0$	$2^2$	$2^1$	$2^0$	$2^2$	$2^1$	$2^0$
	4	2	1	4	0	1	4	0	1

# 1. chmod 400 <이름>.pem

---

```
-rw-r--r--@ 1 user_name ~~1692 1 13 10:05 test.pem
```

```
$ chmod 400 test.pem  
(권한변경 후)
```

```
-r-----@ 1 user_name ~~ 1692 1 13 10:05 test.pem
```

## 2. \$ ssh -i test.pem ec2-user@52.75.323.321

---

# ssh

= secure shell

다른 컴퓨터에 원격으로 안전하게 접속하는 프로그램



## 2. \$ ssh -i test.pem ec2-user@52.75.323.321

---

# ssh -i [키파일]

-i : identity-file의 약자!  
옵션으로 id 파일을 넘긴다는 뜻.  
그래서 이어서 키파일을 써주는 것!

## 2. `$ ssh -i test.pem ec2-user@52.75.323.321`

---

`ssh -i [키파일] [user_name@host_name]`

우리가 새로 생성한 EC2 인스턴스(컴퓨터)에 접속하려면  
그 컴퓨터의 주소(IP)가 필요하다!

ec2-user라는 이름으로 접속하는건?  
아마존이 그렇게 설정해놔서..

## 2. \$ ssh -i test.pem ec2-user@52.75.323.321

---

정리해 보자

ssh -i test.pem ec2-user@52.75.323.321  
은 무슨뜻이냐면!

ssh 프로그램(방식)을 통해  
인터넷 어딘가 중에서 52.75.323.321주소(IP)에 있는 컴퓨터에  
ec2-user라는 user로 접속할건데  
이 컴퓨터는 잠겨있어서 열쇠가 필요해!  
-i 옵션으로 그 열쇠도 같이 넘길게!  
그 열쇠는 test.pem 이야!

▼ Security Groups

[Edit security groups](#)

**Security group name** launch-wizard-2  
**Description** launch-wizard-2 created 2017-01-14T00:10:32.679+09:00

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ
SSH	TCP	22	0.0.0.0/0
HTTP	TCP	80	0.0.0.0/0

# 3. sudo yum install git-all.noarch

---

- sudo(substitute user do):

관리자(root)가 아닌 계정일 때!

이어서 입력하는 명령어 줄을 관리자 권한으로 실행 가능

- Yum 이라는 설치 마법사를 이용해 git 을 설치하자.  
(mac은 brew라는 설치 마법사를 사용해요)

# 4. git clone <gitrepository주소>

SGJunghyun / soonhiri Private

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Pulse Graphs Settings

No description or website provided. Edit

67 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

SGJunghyun 형식 수정

app	형식 수정	
bin	my first commit	
config	DB화	
db	형식 수정	a month ago
lib	my first commit	3 months ago
log	my first commit	3 months ago
public	인스타추가	2 months ago
test	DB화	3 months ago
tmp	my first commit	3 months ago
vendor/assets	my first commit	3 months ago
.gitignore	figaro	3 months ago
Gemfile	DB화	3 months ago

Clone with HTTPS Use SSH

Use Git or checkout with SVN using the web URL.

<https://github.com/SGJunghyun/soonhiri.git>

Open in Desktop Download ZIP

## 4. rvm or rbenv를 이용해 루비 설치하기

---

Rvm과 rbenv는 루비를 설치하기 위한  
패키지라고 생각하면 좋다

둘다 많이 쓰이지만 rvm이 더 무겁기 때문에  
요즘은 rbenv로 넘어가는 추세라고한다

우리는 rvm으로 해보겠습니다.

## 4. rvm or rbenv를 이용해 루비 설치하기

---

a. rvm.io 에서 install rvm 하기 : 루비를 쉽게 설치하는 준비작업 (루비버전메니저)

b. 해당 홈페이지에서 시키는대로 차례로 입력해보자

```
$ gpg --keyserver hkp://keys.gnupg.net --recv-keys 409B6B1796C275462A1703113804BB82D39DC0E3
```

```
$ curl -sSL https://get.rvm.io | bash -s stable
```

```
$ source ~/.profile
```



# Ruby Version Manager (RVM)

RVM is a command-line tool which allows you to easily install, manage, and work with multiple ruby environments from interpreters to sets of gems.



## Cut Rubies with ease!

- Install RVM:

```
$ gpg --keyserver hkp://keys.gnupg.net --recv-keys 409B6B1796C275462A1703113804BB82D39DC0E3
```

```
$ #curl -sSL https://get.rvm.io | bash -s stable
```

For all in one installation append `[ --ruby ]` or `[ --python ]` or `[ --ruby --python ]`

# 5. rvm install ruby 2.3.3

---

rvm 설치를 완료했으면 rvm을 이용해  
ruby를 설치한다

2.3.3 버전으로!!

# 7. gem install bundler

---

Bundler gem 설치

\$ bundle install 명령어를 가능하게 해주는 gem

gem 이름, 버전 등  
다른 gem에 대한 모든 관리를 담당하고 있는 gem!

# 8. bundle install

---

\$cd (프로젝트) 를 쳐서 프로젝트 내로 들어간 다음  
bundle install을 통해 모든 gem을 설치!

앗! 그전에 디폴트로 주석처리 되어있는  
`gem 'therubyracer'` 를 활성화시켜주세요!

Rubyracer은  
Javascript 코드를 루비에서 사용할 수 있게 해주는 gem

# 8. gem install passenger

---

Nginx 서버를 사용하기 위해 필요한 gem

Nginx: 웹 어플리케이션을 계속 서버에서 돌게 해주는 웹 서버  
& Rails 앱을 Production 모드로 배포할 수 있게 해줌

Passenger: Rails에서 Nginx를 관리할 수 있게 해주는 모듈  
(\*Python, Node.js에서도 사용 가능)



# 9. sudo passwd

---

Passenger gem을 통해 nginx라는 서버 프로그램을 설치해야 한다

Root 권한이 필요하다!

그것을 위해 sudo passwd 치면, 비번 설정할 수 있음 (처음1회)

# 10. su

---

비번을 설정 했으면

\$su

명령어를 통해 root권한 획득하는 명령어

# 11. # passenger-install-nginx-module

---

10의 Su 권한 획득 후 , passenger-install-nginx-module 입력

위 명령어를 통해 nginx설치한다.

(위 명령어 과정 계속 진행하면서 오류 계속 잡아내기)

모든 설치과정이 끝나면 exit 입력해서 su 권한 끝내기



# 12. \$ sudo vi /opt/nginx/conf/nginx.conf

---

차근차근 해석해보면 ,

Sudo : 관리자 권한으로

Vi : vim이라는 text편집기로 열기

/opt/nginx/conf/nginx.conf :  
/opt/nginx/conf 경로에 있는 nginx.conf 파일을 열겠다.

## 12. \$ sudo vi /opt/nginx/conf/nginx.conf

---

server 위쪽에 새로 server{} 를 입력해서 그 사이에

```
server {  
    listen 80;  
    server_name likelion.com;  
    passenger_enabled on;  
    root /home/ec2-user/Myproject/public;  
}
```

# 13. 'Figaro' gem 추가

---

Production mode 환경 변수를 숨겨주는 Figaro gem

Gem "figaro"  
추가 후 bundle install  
bundle exec figaro install

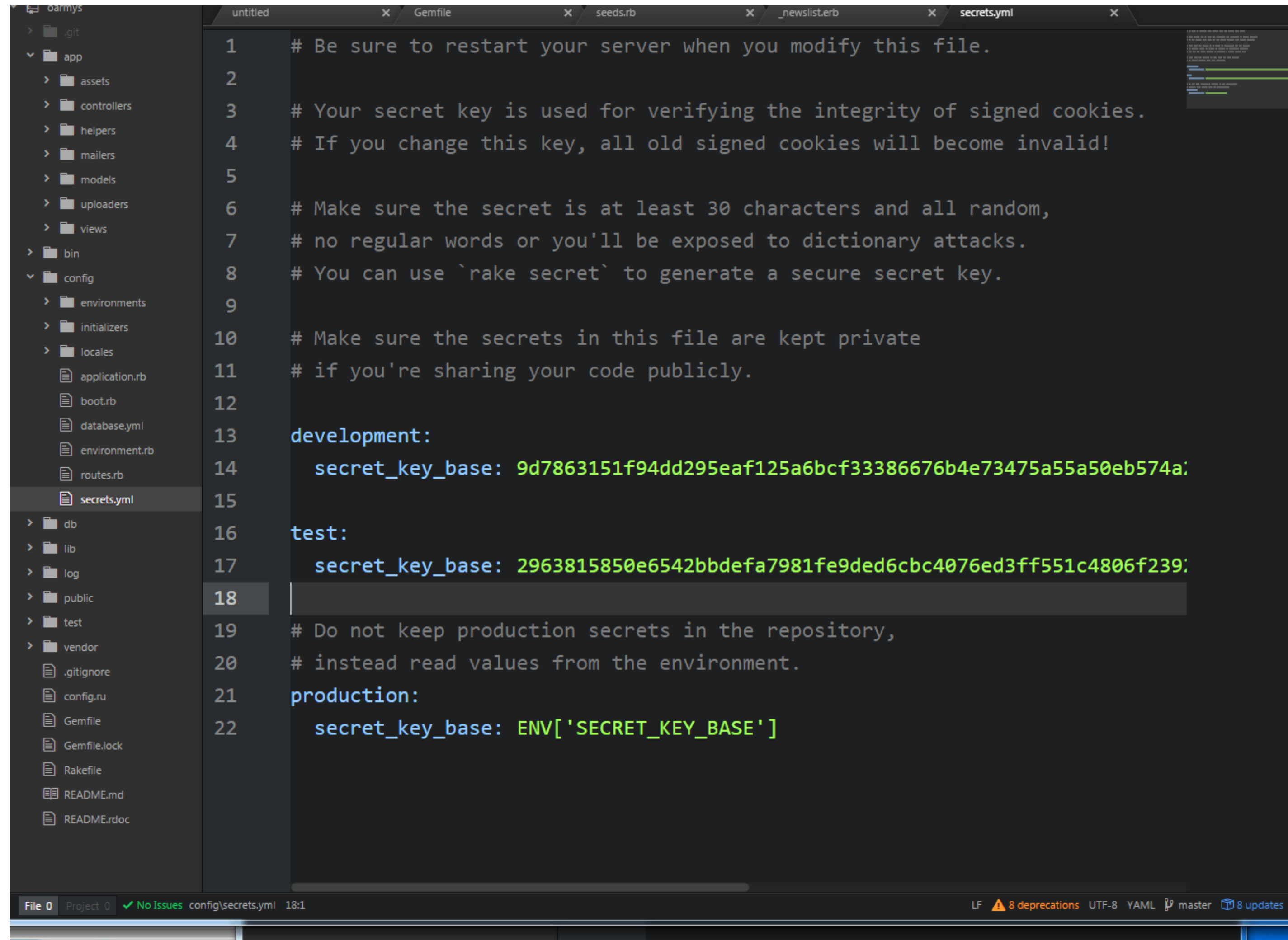
Config 파일에 application.yml 이라는 파일이 생성됨

# 13. 'Figaro' gem 추가

---

application.yml 파일에  
SECRET\_KEY\_BASE : '암호화된 메시지'  
를 입력

# 13. 'Figaro' gem 추가



```
1 # Be sure to restart your server when you modify this file.
2
3 # Your secret key is used for verifying the integrity of signed cookies.
4 # If you change this key, all old signed cookies will become invalid!
5
6 # Make sure the secret is at least 30 characters and all random,
7 # no regular words or you'll be exposed to dictionary attacks.
8 # You can use `rake secret` to generate a secure secret key.
9
10 # Make sure the secrets in this file are kept private
11 # if you're sharing your code publicly.
12
13 development:
14   secret_key_base: 9d7863151f94dd295eaf125a6bcf33386676b4e73475a55a50eb574a:
15
16 test:
17   secret_key_base: 2963815850e6542bbdefa7981fe9ded6cbc4076ed3ff551c4806f239:
18
19 # Do not keep production secrets in the repository,
20 # instead read values from the environment.
21 production:
22   secret_key_base: ENV['SECRET_KEY_BASE']
```

## 14. rake db:migrate RAILS\_ENV=production

---

rake db:migrate => migration 을 실행

\*원래는 데이터베이스를 변경/관리하기 위해서는  
쿼리문을 써서 SQL을 관리해야하지만  
Rails에서는 migration이라는게  
데이터베이스 관련된 것을 손쉽게 변경/관리 할 수 있게 해줌

RAILS\_ENV=production => production 모드에서

15. \$ sudo /opt/nginx/sbin/nginx

---

Nginx 서버를 띄우는 것!

## 16. RAILS\_ENV=production rake assets:precompile

---

Asset pipe line을 사용하는 명령어

Asset = Javascript, Stylesheet(css), images 등 을  
모두 합쳐버려서

사용자들이 웹페이지에 보다 빠르게 접속할 수 있게!



## 16. \$ touch tmp/restart.txt

---

수정 사항이 있을 때마다 서버 재부팅!

\$touch

: 파일의 '수정 시간'을 갱신하거나  
해당 파일이 없으면 새로운 파일을 생성함

tmp = timestamp의 약자

Passenger는 tmp 폴더에 있는 restart.txt라는 파일의  
수정 시간이 업데이트 될때마다 서버를 스스로 재시작 한다.

대장님이 안 알려줌!!!!!!!!!! 중요!!!

<%= image\_path '파일명' %>

---

asset pipe line을 쓰면  
asset들을 서버에 압축해서 올려주기 때문에  
기존의  는  
먹히지 않는다...  
(\*파일들 위치와 이름이 마음대로 변경된다...)

대신 파일 이름만 동일하면 그 파일이 있는 위치를 알아서  
찾아주는 메서드를 사용하면 된다!

예) 

신촌 심화 1팀입니다.  
감사합니다.



**likeLion**<sup>®</sup>  
SOGANG UNIVERSITY