

- 배포 자동화 준비 : Travs-ci, S3 생성 및 연동하기
- 1. Travis-ci 계정 생성 및 git 프로젝트 연동
- 2. AWS IAM 사용자 생성 (IAM : spring-travis-deploy)
- 3. S3 버킷 생성
- 4. .travis.yml파일에 S3 설정 추가
- 5. Git push 테스트
  - 파일 및 인스턴스 정리



## 1) Travis-ci,S3 생성 및 연동하기

1. Travis-ci 계정 생성 및 git 프로젝트 연동 (.travis.yml 파일 생성)
2. AWS IAM 사용자 생성 (Travis-ci 접근 권한)
3. S3 버킷 생성
4. .travis.yml 파일에 S3 설정 추가
5. Git push 테스트

Travis CI, S3 생성 및 연동하기

## 배포 자동화 준비 : Travs-ci, S3 생성 및 연동하기

이제 Test & Build를 자동화시키는 작업을 진행할 것이다. Git에 프로젝트 수정사항을 새롭게 push하면 자동으로 Test & Build & Deploy가 진행되도록 개선하는 작업이다.

### CI & CD

코드 버전 관리를 하는 VCS 시스템(Git, SVN 등)에 PUSH되면 자동으로 테스트와 빌드가 수행되어 안정적인 배포 파일을 만드는 과정을 **CI(Continuous Integration - 지속적 통합)**라고 하며, 이 빌드 결과를 자동으로 운영 서버에 무중단 배포까지 진행되는 과정을 **CD(Continuous Deployment - 지속적 배포)**라고 한다.

Travis CI는 Git에서 제공하는 무료 CI서비스이다. Jenkins와 같은 CI도구도 있지만 설치형이기 때문에 이를 위한 EC2 인스턴스가 하나 더 필요하다. 초기 서비스에서 배포를 위한 EC2 인스턴스는 부담스럽기 때문에 오픈소스 웹 서비스인 Travis CI를 사용한다.

\* AWS CodeBuild도 있지만 빌드 시간만큼 요금이 부과되기 때문에 초기 서비스엔 사용하기 부담스럽다.

Travis CI이 해줄 역할은 스프링 프로젝트를 Test, Build, Notify를 해주는 것이다.

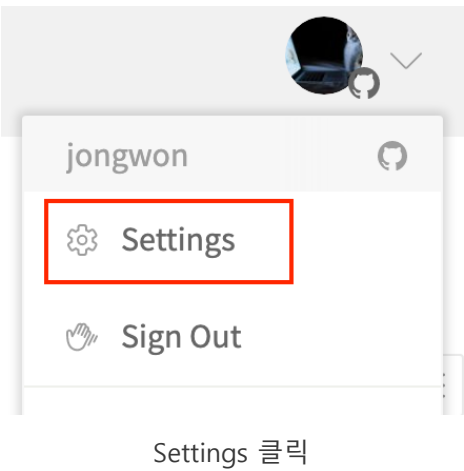
- Test : 스프링 프로젝트를 테스트해준다.
- Build : 스프링 프로젝트를 빌드해준다.
- Notify : Build 완료나 에러 결과를 이메일 혹은 Slack과 같은 메신저로 알림을 준다.

S3란 AWS에서 제공하는 일종의 파일 서버이다. 이미지 파일을 비롯한 정적 파일들을 관리하거나 지금 진행하는 것처럼 배포 파일들을 관리하는 등의 기능을 지원한다. 보통 이미지 업로드를 구현할 때 S3를 많이 사용한다.

실제 배포는 AWS CodeDeploy라는 서비스를 통해서 진행한다. S3연동이 먼저인 이유는 **Jar(실행 파일)**을 전달하기 위해서이다.

# 1. Travis-ci 계정 생성 및 git 프로젝트 연동

1) **Travis-ci 사이트** 접속 > git 계정으로 로그인 > 계정명 > **Settings** 클릭



Settings 클릭

2) 설정 아래 깃허브 저장소 검색창 > 저장소 검색 > **spring\_deploy\_test** 클릭



저장소 찾기

## 3) 프로젝트 설정

Travis CI의 상세한 설정은 프로젝트에 존재하는 .travis.yml파일로 할 수 있다. 해당 파일을 스프링 프로젝트 build.gradle과 같은 위치에 생성한 후 다음 코드를 추가한다.

```
language: java
jdk:
  - openjdk11 # java8이면, openjdk8

branches:
  only:
    - main # git default 브랜치를 등록한다. (ex. master)

# Travis CI 서버의 Home
cache:
  directories:
    - '$HOME/.m2/repository'
    - '$HOME/.gradle'

script: "./gradlew clean build"
```

```
# CI 실행 완료 시 메일로 알림
notifications:
  email:
    recipients:
      - jong9712@naver.com
```

나는 저장소를 만들 때 branch이름을 main으로 설정하였기 때문에 main으로 입력했지만 기본적으로는 master를 입력해줘야 한다. 이렇게 git에 push를 하면 travis-ci를 확인하면 빌드가 성공된 것을 볼 수 있다.

✓ **main** #travs-ci 기본 설정

Commit 00fb8aa

Compare 14620c4...00fb8aa

Branch main

jongwon

JDK: openjdk11 Java

AMD64

#14 passed

Ran for 1 min 42 sec

less than a minute ago

Restart build

빌드 성공

.travis.yml에 입력한 이메일로도 알림이 온 것을 확인할 수 있다.

loosie / spring\_deploy\_test

main

✓ Build #14 passed >

1 min and 42 secs

jongwon

00fb8aa CHANGESET →

#travs-ci 기본 설정

이메일 알림

## 2. AWS IAM 사용자 생성 (IAM : spring-travis-deploy)

S3에는 프로젝트 실행 파일은 jar을 저장하고 전달하기 위한 중간 저장소로 활용한다. 일반적으로 AWS서비스에 외부 서비스가 접근할 수 없다. 그러므로 접근 가능한 권한을 가진 Key를 생성해서 사용해야 한다. AWS에서는 이러한 인증과 관련된 기능을 제공하는 서비스로 IAM이 있다.

1) AWS IAM > 사용자 > 사용자 추가 클릭 > 사용자 이름 입력 > AWS 액세스 유형 선택 > 다음

IAM은 AWS에서 제공하는 서비스의 접근 방식과 권한을 관리한다. 이 IAM을 통해 Travis CI가 AWS의 S3와 CodeDeploy에 접근할 수도 있도록 설정할 것이다.

사용자 세부 정보 설정

동일한 액세스 유형 및 권한을 사용하여 한 번에 여러 사용자를 추가할 수 있습니다. [자세히 알아보기](#)

사용자 이름\*

spring-travis-deploy

+ 다른 사용자 추가

AWS 액세스 유형 선택

해당 사용자가 AWS에 액세스하는 방법을 선택합니다. 마지막 단계에서는 액세스 키와 자동 생성된 비밀번호가 제공됩니다. [자세히 알아보기](#)

액세스 유형\*

☒ 프로그래밍 방식 액세스

AWS API, CLI, SDK 및 기타 개발 도구에 대해 액세스 키 ID 및 비밀 액세스 키 을(를) 생성합니다.


☐ AWS Management Console 액세스


사용자가 AWS Management Console에 로그인할 수 있도록 허용하는 비밀번호 을(를) 생성합니다.


액세스 유형 선택

2) 기존 정책 직접 연결 선택 > 정책 2가지 (s3full..., codedeployf...) 추가 > 다음

▼ 권한 설정

 그룹에 사용자 추가


 기존 사용자에서 권한 복사

 기존 정책 직접 연결

기존 정책 연결

정책 필터 ▼


Q s3full

	정책 이름 ▼
<input checked="" type="checkbox"/>	 AmazonS3FullAccess

정책 추가

정책 필터 ▼

Q codedeployf

	정책 이름 ▼
<input checked="" type="checkbox"/>	 AWSCodeDeployFullAccess

정책 추가

3) 태그(키, 값) 입력 > 검토 후 생성

태그 추가(선택 사항)

IAM 태그는 사용자 사용자에게 추가할 수 있는 키-값 페어입니다. 태그는 이메일 주소와 같은 사용자 정보를 포함하거나 자에 대한 액세스를 구성, 추적 또는 제어할 수 있습니다. 자세히 알아보기

키	값(선택 사항)
<input type="text" value="Name"/>	<input type="text" value="spring-travis-deploy"/>
<input type="button" value="새 키 추가"/>	

검토

선택 항목을 검토합니다. 사용자를 생성한 후 자동으로 생성된 비밀번호와 액세스 키를 보고 다운로드할 수 있습니다.

사용자 세부 정보

사용자 이름	spring-travis-deploy
AWS 액세스 유형	프로그래밍 방식 액세스 - 액세스 키 사용
권한 경계	권한 경계가 설정되지 않았습니다

권한 요약

다음 정책이 위에 표시된 사용자에게 연결됩니다.

유형	이름
관리형 정책	AmazonS3FullAccess
관리형 정책	AWSCodeDeployFullAccess

태그

새로운 사용자 에게 다음 태그가 제공됩니다

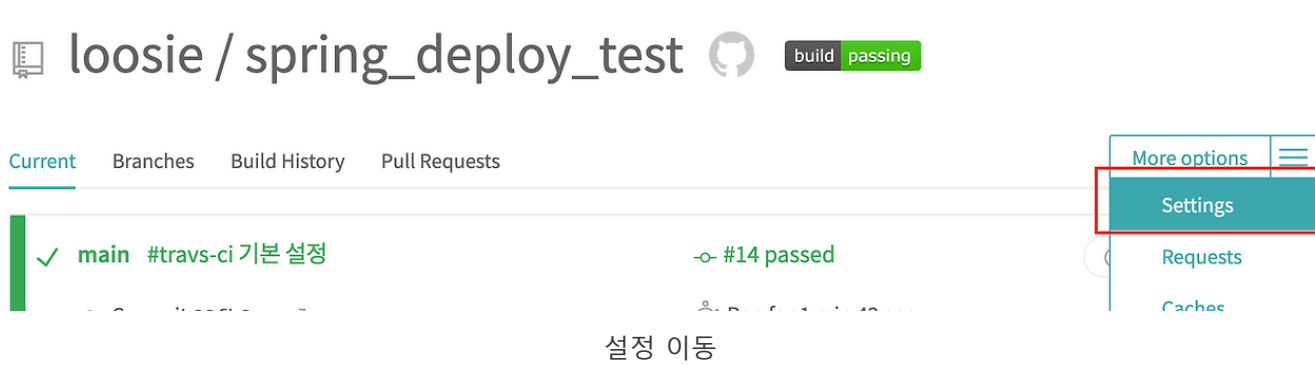
키	값
Name	spring-travis-deploy

최종 검토

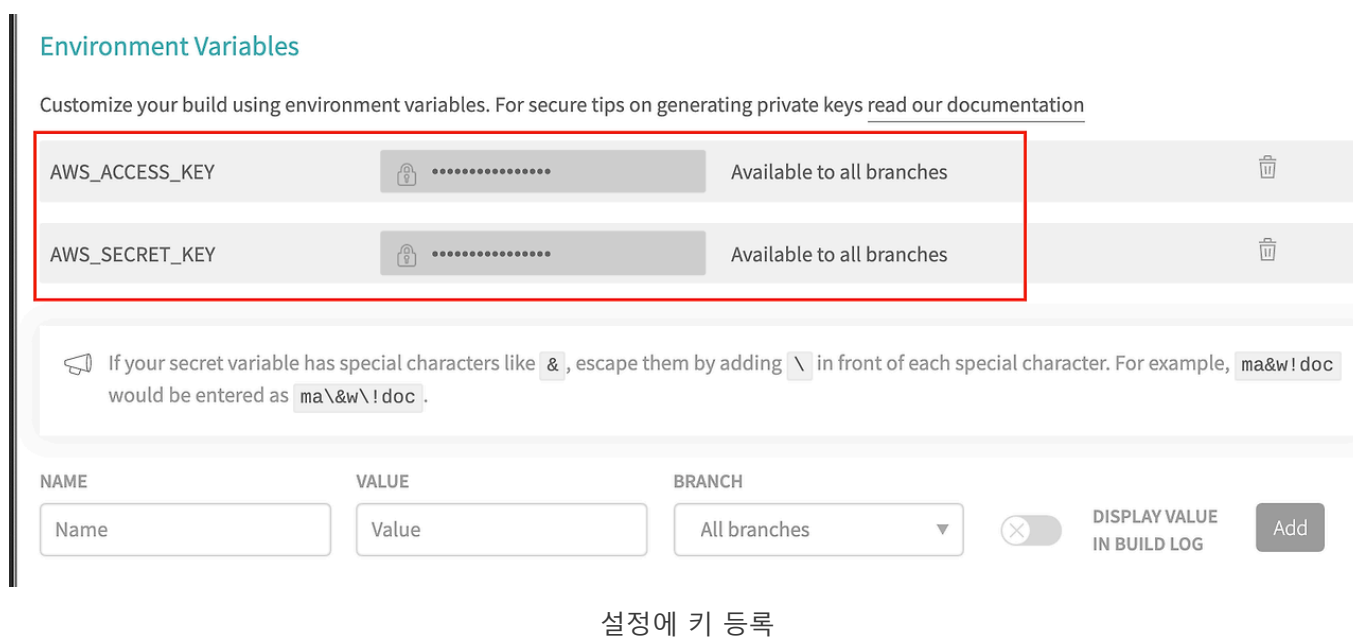
4) 최종 생성이 완료되면 해당 액세스 키와 비밀 액세스 키 복사 > Travs CI 설정 > Environment Variables 항목에 해당 값들 입력

	사용자	액세스 키 ID	비밀 액세스 키
▶	✔ spring-travis-deploy	AKIA5IGCAKLFFA7D6SXN	***** 표시

액세스 키, 비밀 액세스 키



여기에 **AWS\_ACCESS\_KEY**, **AWS\_SECRET\_KEY**를 변수로 해서 **IAM 사용자에서 발급받은 키 값들**을 등록한다.

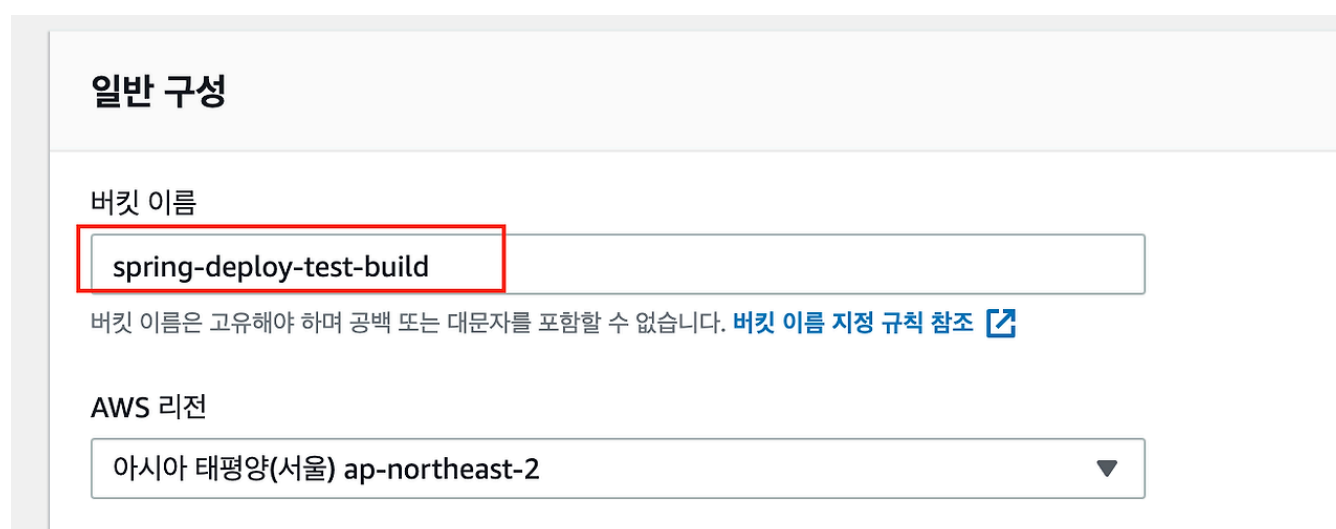


### 3. S3 버킷 생성

Travis CI에서 생성된 Build 파일을 저장하도록 구성하겠다. S3에 저장된 Build 파일은 이후 AWS의 CodeDeploy에서 배포한 파일로 가져가도록 구성할 예정이다. AWS 서비스에서 S3를 검색하여 이동하고 버킷을 생성한다.

#### 1) S3 이동 > 버킷 만들기 > 원하는 버킷명(spring-deploy-test-build) 입력

버킷명은 이 버킷에 배포할 Zip 파일이 모여있는 장소로 의미하도록 짓는 것을 추천한다.



## 2) 퍼블릭 액세스 모든 차단으로 설정 > 버킷 생성

보안으로 관련된 부분으로 모든 차단을 해야 한다. 실제 서비스에서 Jar파일이 퍼블릭일 경우 누구나 내려받을 수 있어 코드나 설정값, 주요 키값들이 다 탈취될 수 있다.

### 이 버킷의 퍼블릭 액세스 차단 설정

퍼블릭 액세스는 ACL(액세스 제어 목록), 버킷 정책, 액세스 지점 정책 또는 모두를 통해 버킷 및 객체에 부여됩니다. 이 버킷 및 해당 객체에 대한 퍼블릭 액세스가 차단되었는지 확인하려면 모든 퍼블릭 액세스 차단을 활성화합니다. 이 설정은 이 버킷 및 해당 액세스 지점에만 적용됩니다. AWS에서는 모든 퍼블릭 액세스 차단을 활성화하도록 권장하지만, 이 설정을 적용하기 전에 퍼블릭 액세스가 없어도 애플리케이션이 올바르게 작동하는지 확인합니다. 이 버킷 또는 내부 객체에 대한 어느 정도 수준의 퍼블릭 액세스가 필요한 경우 특정 스토리지 사용 사례에 맞게 아래 개별 설정을 사용자 지정할 수 있습니다. [자세히 알아보기](#)

#### ☒ 모든 퍼블릭 액세스 차단

이 설정을 활성화하면 아래 4개의 설정을 모두 활성화한 것과 같습니다. 다음 설정 각각은 서로 독립적입니다.

#### ☒ 새 ACL(액세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단

S3은 새로 추가된 버킷 또는 객체에 적용되는 퍼블릭 액세스 권한을 차단하며, 기존 버킷 및 객체에 대한 새 퍼블릭 액세스 ACL 생성을 금지합니다. 이 설정은 ACL을 사용하여 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 권한을 변경하지 않습니다.

#### ☒ 임의의 ACL(액세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단

S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 모든 ACL을 무시합니다.

퍼블릭 액세스 차단 설정

버킷 생성이 완료되었다면 이젠 스프링 프로젝트로 다시 돌아가 설정을 변경해주면 된다.

### 버킷 (7) Info

버킷은 S3에 저장되는 데이터의 컨테이너입니다. [자세히 알아보기](#)

Q spring-de

이름



AWS 리전



spring-deploy-test-build

아시아 태평양(서울) ap-northeast-2

버킷 생성완료

## 4. .travis.yml파일에 S3 설정 추가

Travis CI에서 빌드하여 만든 Jar 파일을 S3에 올릴 수 있도록 다음 코드를 #1에서 만든 .travis.yml에 추가한다.

```
language: java
jdk:
  - openjdk11
```

```
branches:
  only:
    - main
```

```
# Travis CI 서버의 Home
cache:
  directories:
    - '$HOME/.m2/repository'
    - '$HOME/.gradle'

script: "./gradlew clean build"

# CI 실행 완료 시 메일로 알림
notifications:
  email:
    recipients:
      - jong9712@naver.com

##### 새롭게 추가한 코드 #####

before_deploy:
  - zip -r spring_deploy_test *
  - mkdir -p deploy
  - mv spring_deploy_test.zip deploy/spring_deploy_test.zip

deploy:
  # S3로 파일업로드
  - provider: s3
    access_key_id: $AWS_ACCESS_KEY # Travis repo settings에 설정된 값
    secret_access_key: $AWS_SECRET_KEY # Travis repos settings에 설정된 값
    bucket: spring-deploy-test-build # s3 버킷
    region: ap-northeast-2
    skip_cleanup: true
    acl: private # zip 파일 접근을 private으로
    local_dir: deploy # before_deploy에서 생성한 디렉토리
    wait-until-deployed: true
    on:
      all_branches: true # master 말고 다른 모든 브랜치 허용

#####
```

**before\_deploy**

- deploy 명령어가 실행되기 전에 수행된다.
- CodeDeploy는 **Jar 파일은 인식하지 못하므로** Jar+기타 설정 파일들을 모아 압축한다.

**zip -r spring\_deploy\_test \***

- 현재 모든 파일을 spring\_deploy\_test 이름으로 압축한다.

**mkdir -p deploy**

- deploy라는 디렉토리를 Travis CI가 실행중인 위치에 생성한다.

**mv spring\_deploy\_test.zip deploy/spring\_deploy\_test.zip**

- 압축한 파일을 deploy/spring\_deploy\_test.zip로 이동시킨다.

**deploy**



- S3로 파일 업로드 혹은 CodeDeploy로 배포 등 외부 서비스와 연동될 행위들을 선언한다.

**local\_dir:deploy**

- 해당 위치의 파일들만 S3로 전송한다.

**on.all\_branches:true**

- 만약 git branch이름 master가 아니라면 해당 설정을 true로 설정해줘야 한다.

## 5. Git push 테스트

해당 로그가 나온다면 Travis CI 빌드가 성공한 것이다.

```
476
▶ 477 Installing deploy dependencies
493 Logging in with Access Key: *****6SXN
494 Beginning upload of 1 files with 5 threads.
▶ 495 Preparing deploy
▶ 497 Deploying application
500 Done. Your build exited with 0.
```

빌드 성공

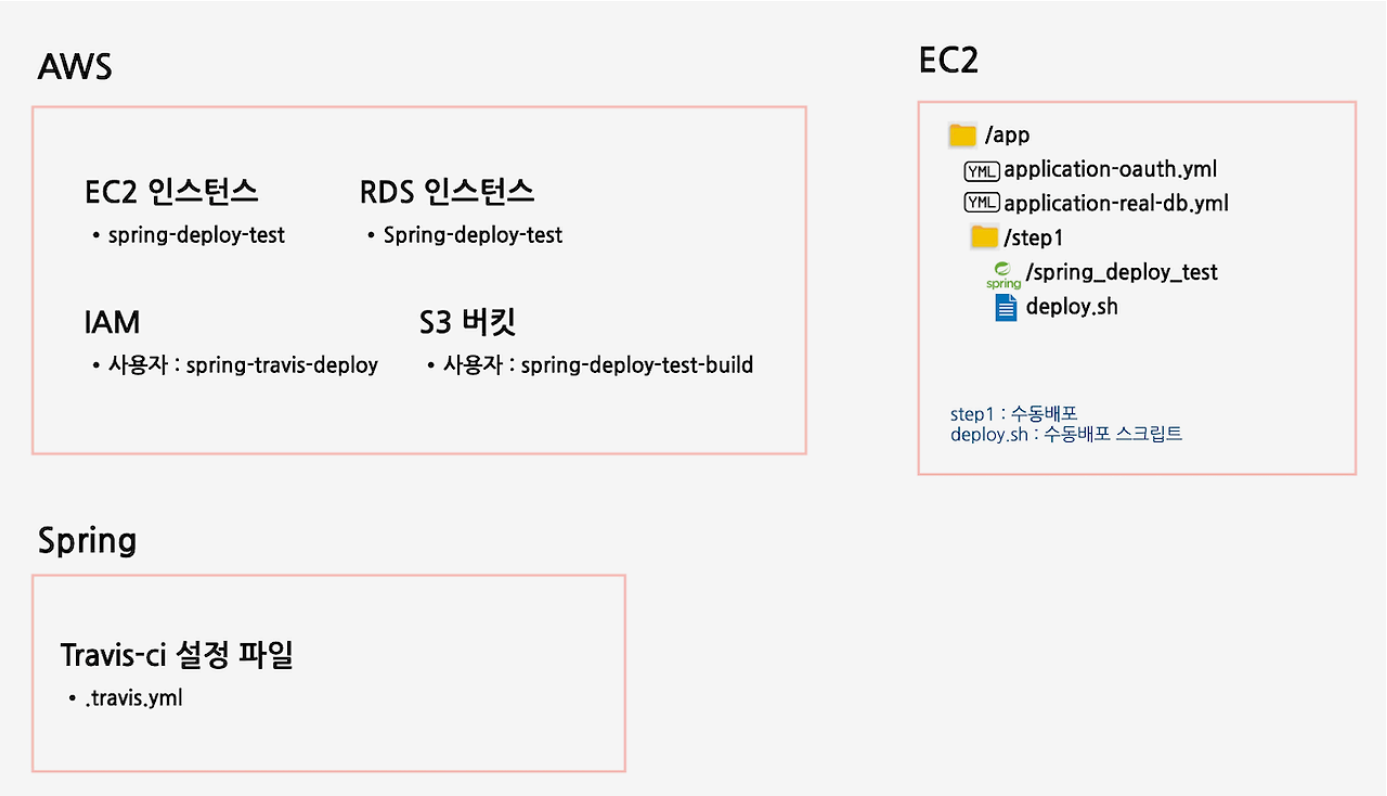
S3 버킷도 확인해보면 업로드가 성공한 것을 확인할 수 있다.



s3 버킷 파일 전달 완료

## 파일 및 인스턴스 정리

현재까지 AWS 환경과 EC2(Linux), Spring 환경에 생성된 인스턴스 및 파일 구조는 다음과 같다.



파일 및 인스턴스

이제 Travis CI와 S3연동이 완료되었으니 마지막으로 CodeDeploy로 배포까지 해주면 된다.