AWS

SUMMIT

在AWS上通过Jenkins实现持续集成和持续交付

amazon web services

软件开发流程

持续集成(Continuous Integration)

- 将代码合并到一个中央软件仓库中
- 自动构建(build)
- 测试
- 运行

持续交付和部署
Continuous Delivery & Deployment

- 持续集成概念的扩展
- 当代码产生变化的时候:
  - 触发构建
  - 部署代码到测试环境中
  - 通过测试后部署到生产环境中
- 当流程完成后，程序已经自动的部署到了生产环境中

CI/CD面临的挑战

- 维护一个源代码仓库
- 自动化的构建和测试
- 创建与生产环境接近的测试环境
- 让整个团队看到流程的进展

AWS的CI/CD服务

- CodeCommit
- CodeBuild
- CodeDeploy
- CodePipeline

Jenkins

- 被设计来实现持续集成和持续部署的软件
- 支持多种编程语言,多种操作系统
- 通过插件实现众多功能
- 开源软件

为何要使用Jenkins

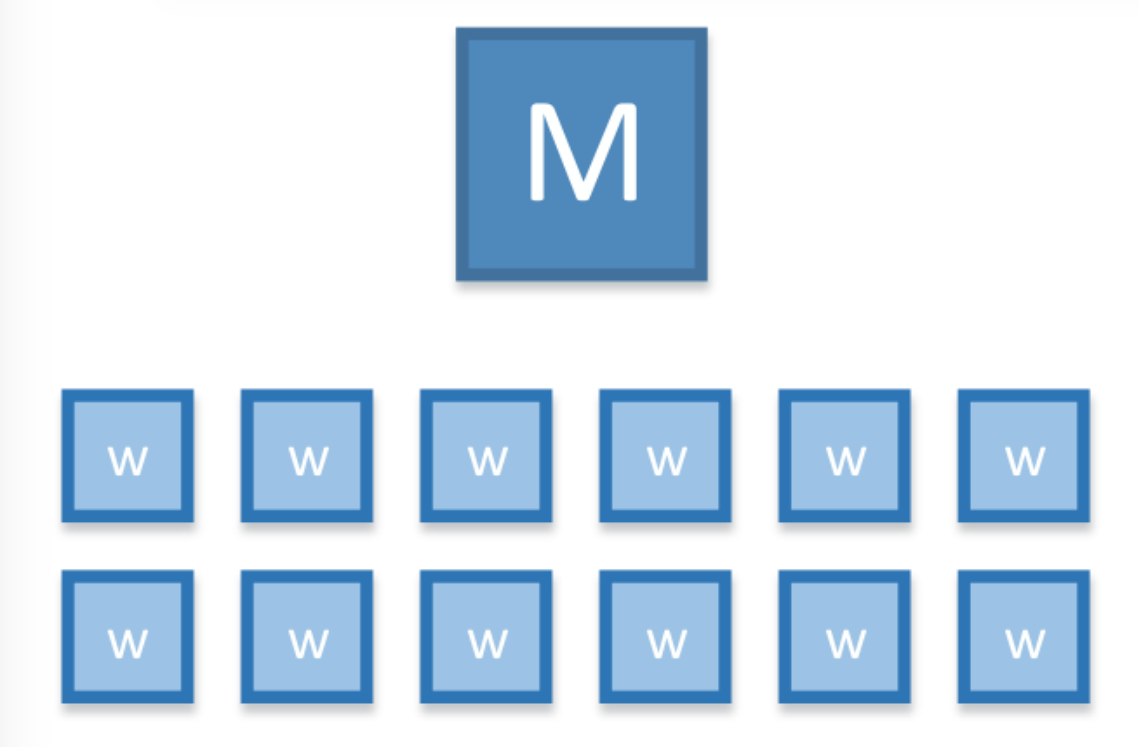
- 有些区域目前还没有AWS Code系列服务或不全
- 现有的项目基于Jenkins
- 希望使用Jenkins特有的功能

在AWS上部署Jenkins

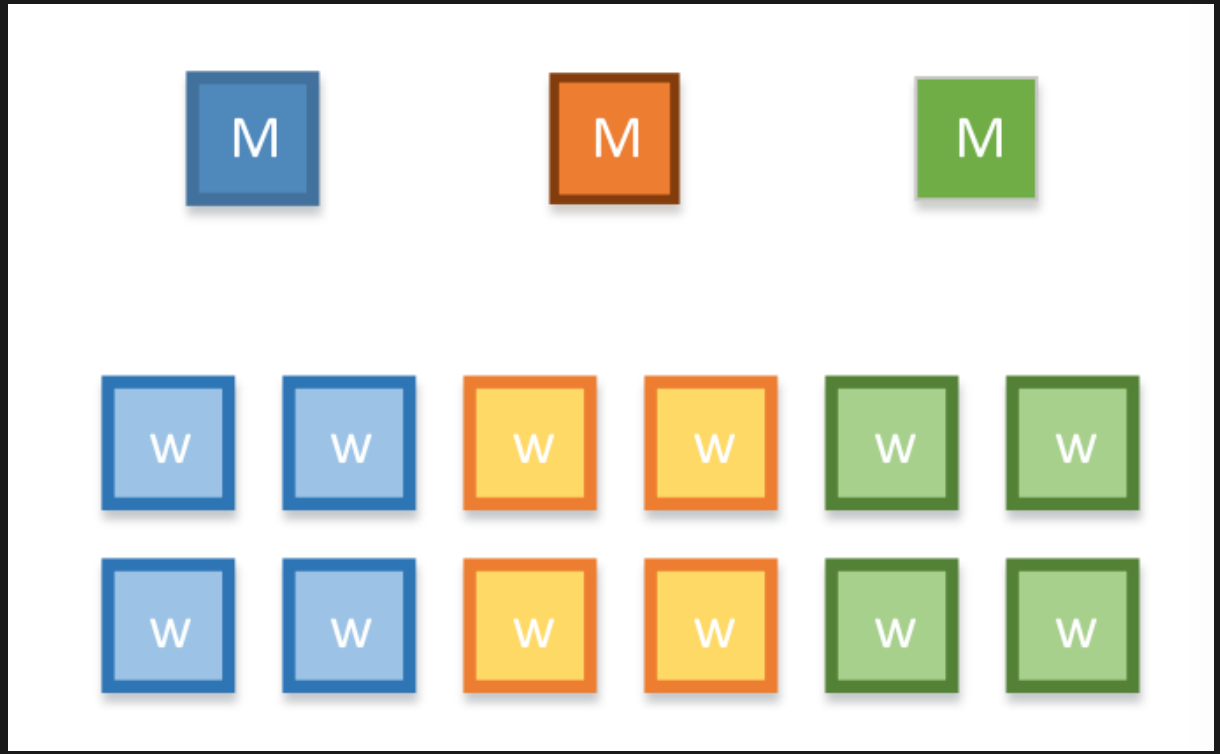
- 单节点部署
- single master mutiple workers
- multiple masters multiple workers
- master节点的高可用

单节点部署

single master multiple workers



multiple masters multiple workers



master节点的高可用

- $JENKINS_HOME
- EFS
- EBS
- auto scaling group

在ECS上部署jenkins

- 每次启动新的worker容器运行
- 自动扩展

创建master

```
# Dockerfile for Jenkins Master

FROM Jenkins

# Add the entry amazon-ecs to plugin.txt to preload the
Amazon ECS plugin

COPY plugins.txt /usr/share/jenkins/plugins.txt

RUN /usr/local/bin/plugins.sh
/usr/share/jenkins/plugins.txt
```

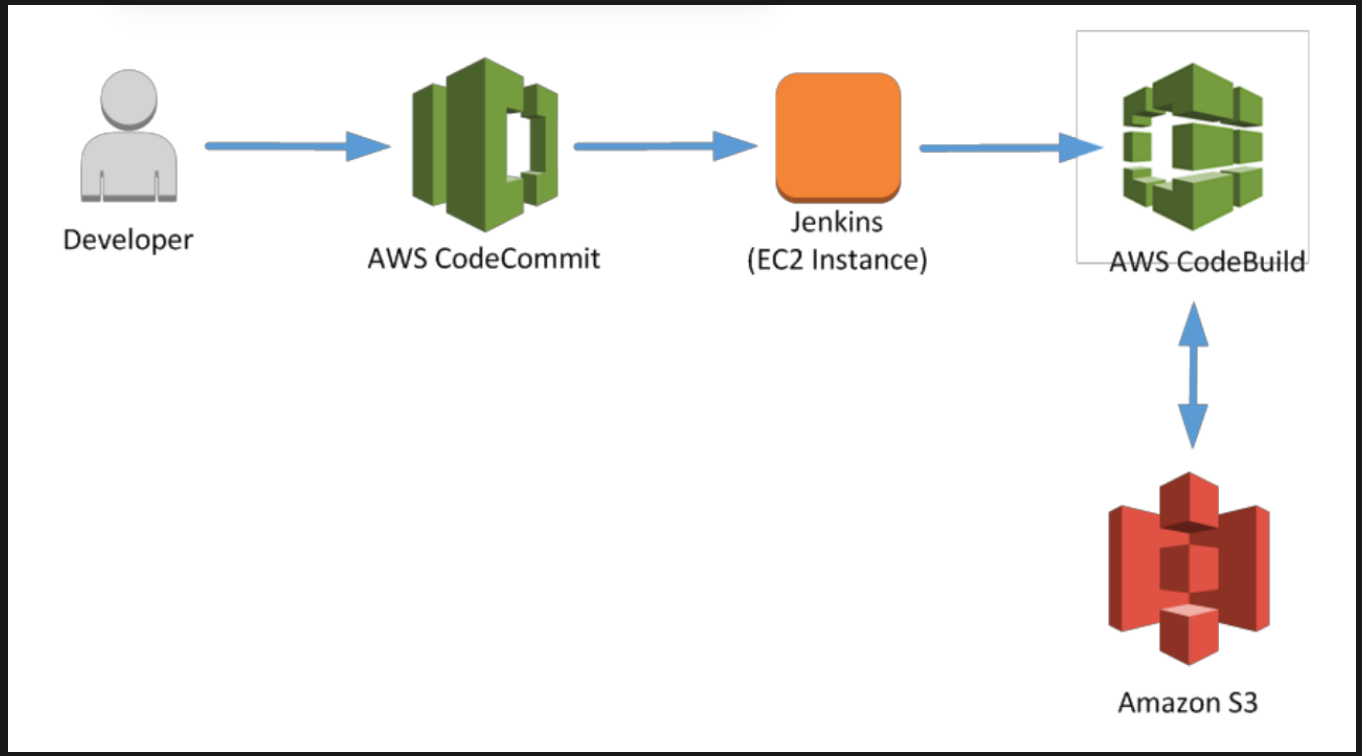
创建data volume

```
# Dockerfile for Jenkins Data Volume Container

FROM Jenkins

VOLUME ["/var/jenkins_home"]

# Creating a Jenkins Data Volume

docker build -t jenkins_dv .

#Tag the jenkins_dv image

 docker tag jenkins_dv:latest <AWS Account
Number>.dkr.ecr.us-east-
1.amazonaws.com/jenkins_dv:latest

# Push the jenkins-dv image to ECR

docker push <AWS Account Number>.dkr.ecr.us-east-
1.amazonaws.com/jenkins_dv:latest
```

配置compose

```
jenkins_master:
  image: jenkins_master
  cpu_shares: 100
  mem_limit: 2000M
  ports:
    - "8080:8080",
    - "50000:50000"
  volumes_from:jenkins_dv
jenkins_dv:
  image: jenkins_dv
  cpu_shares: 100
  mem_limit: 500M
```

Jenkins与CodeBuild结合

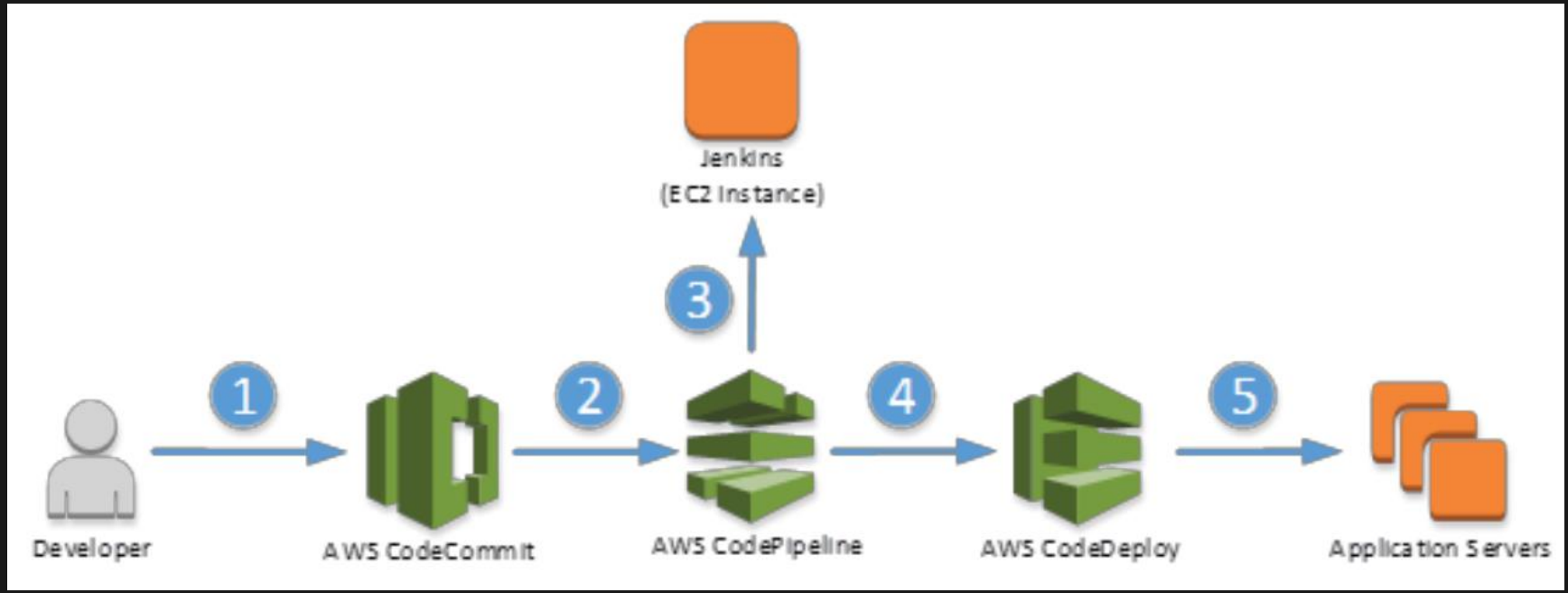


- 安装Jenkins AWS CodeBuild plugin (aws-codebuild.hpi)
- 创建IAM实体,用于调用AWS CodeBuild
- 创建IAM实体,用户AWS CodeBuild执行操作(访问S3,CloudWatch logs)
- 创建AWS CodeBuild项目
- 配置Jenkins AWS CodeBuild plugin

应用场景

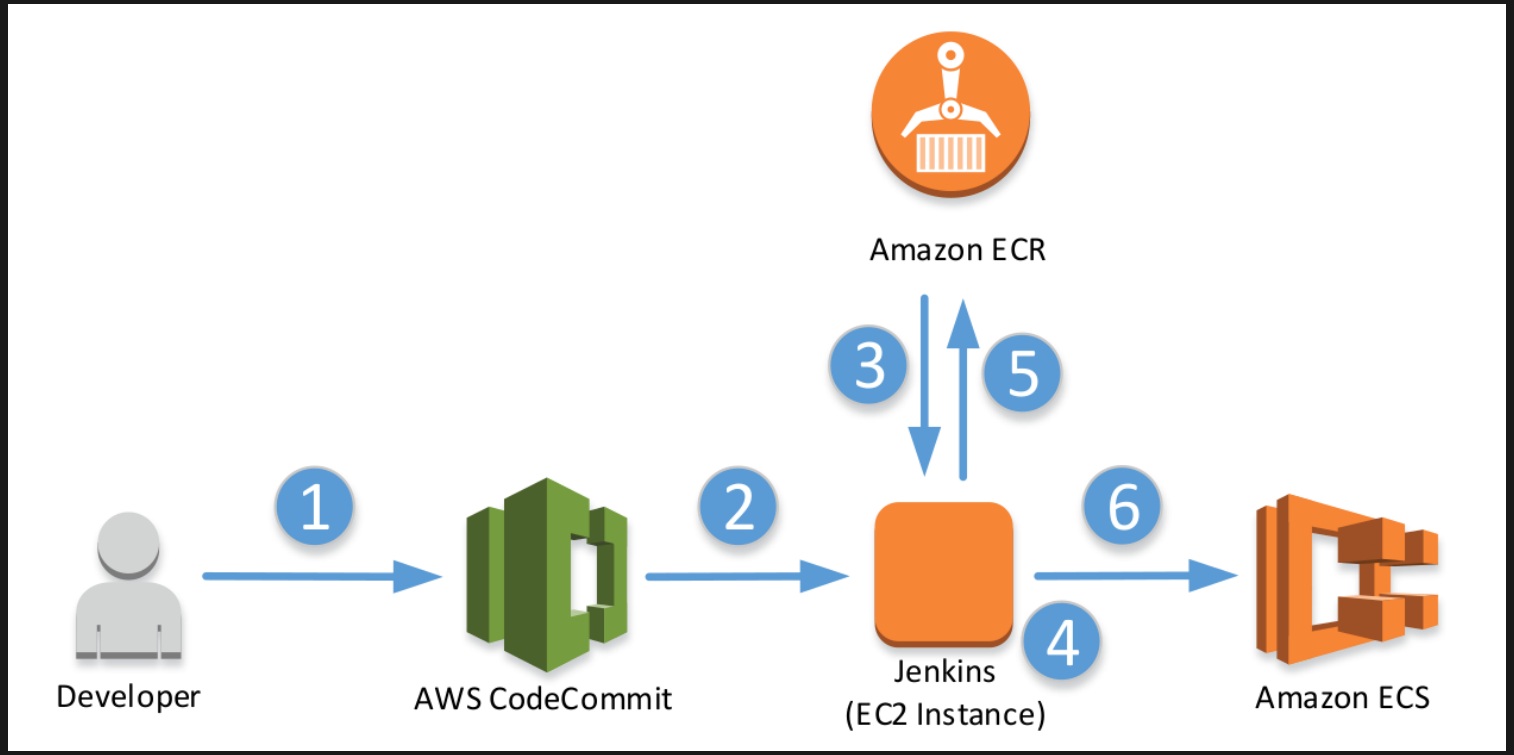
- 标准的CI/CD管道
- 容器管道
- 移动应用
- serverless代码管理
- 自动化的安全框架

标准的CI/CD管道


Jenkins
(EC2 Instance)
3
1
2
4
5
Developer
AWS CodeCommit
AWS CodePipeline
AWS CodeDeploy
Application Servers

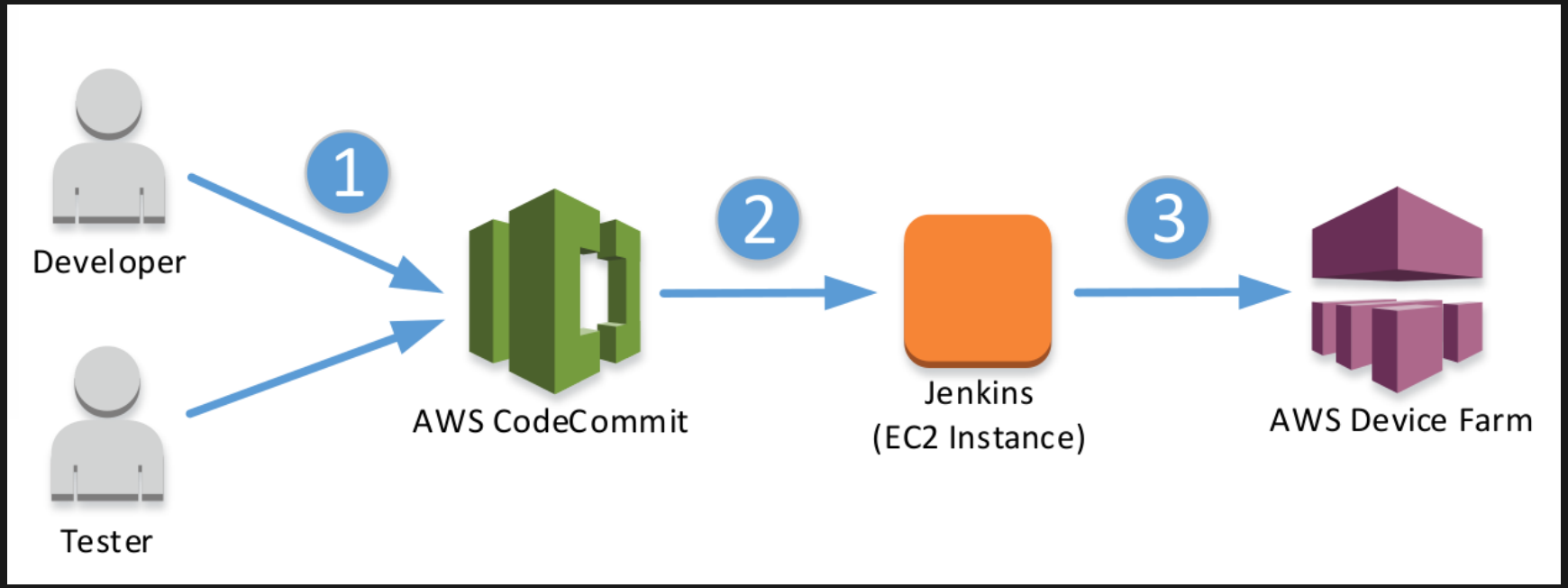
- 开发者将代码提交到AWS CodeCommit
- AWS CodeCommit触发AWS CodePipeline
- AWS CodePipeline调用Jenkins进行构建
- 构建成功后,AWS CodePipeline触发AWS CodeDeploy
- AWS CodeDeploy部署应用到服务器上

容器管道


Amazon ECR
3
5
1
2
6
4
Developer
AWS CodeCommit
Jenkins
(EC2 Instance)
Amazon ECS

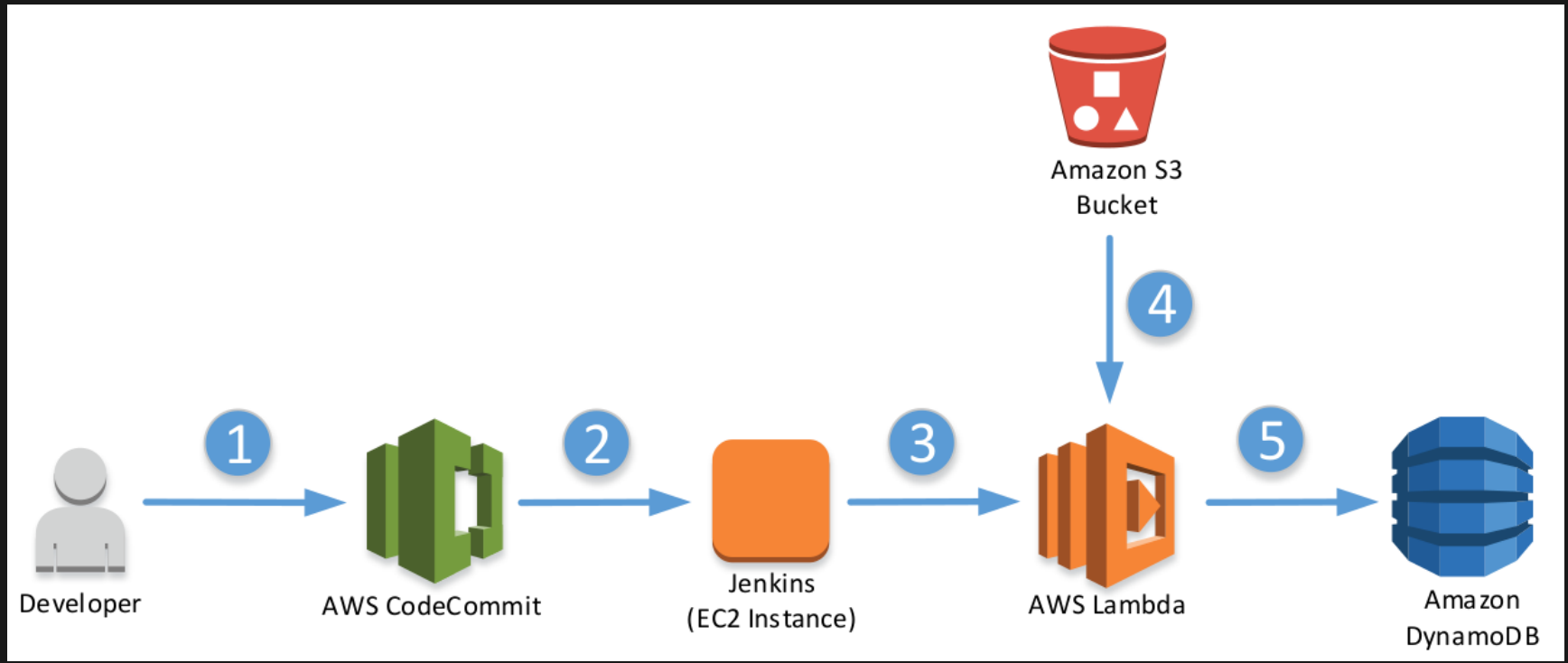
- 开发者提交代码到AWS CodeCommit
- Jenkins检测到AWS CodeCommit的事件
- Jenkins从ECR拉取Docker image
- Jenkins重新构建Docker image
- Jenkins推送新构建的Docker image到Amazon ECR
- Jenkins使用新的image启动服务

移动应用


Developer
Tester
1
AWS CodeCommit
2
Jenkins
(EC2 Instance)
3
AWS Device Farm

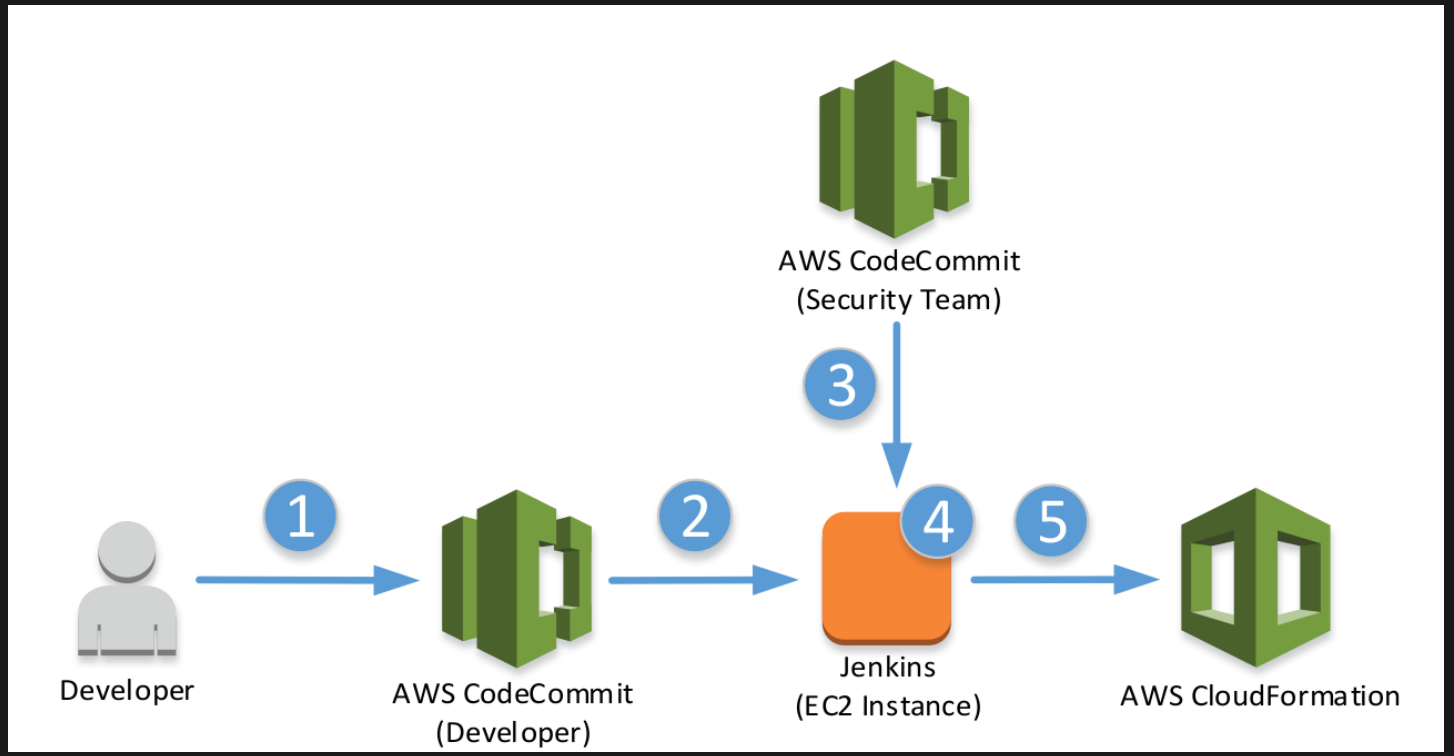
- 开发工程师和测试工程师使用Eclipse产生一个Maven build,然后推送到AWS CodeCommit
- Jenkins根据code生成最新的app
- Jenkins推送app到AWS Device Farm进行测试

serverless

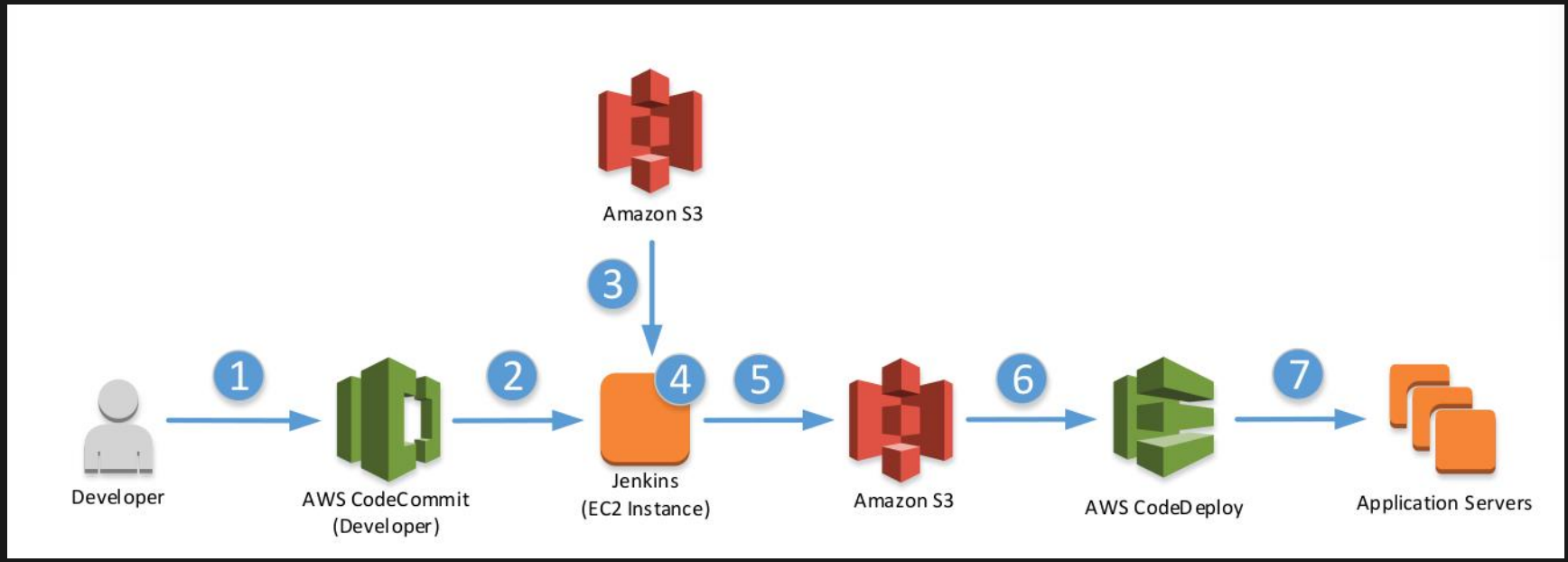

Amazon S3
Bucket
4
1
2
3
5
Developer
AWS CodeCommit
Jenkins
(EC2 Instance)
AWS Lambda
Amazon
DynamoDB

- 开发者提交代码到AWS CodeCommit
- Git hooks通知Jenkins
- Jenkins执行构建和单元测试,并通过post-build step调用AWS CLI把最新代码更新给AWS
- AWS Lambda函数获取Amazon S3存储桶中的对象
- AWS Lambda处理S3对象,并将结果写入到Amazon DynamoDB

整合security review(1)



整合security review(2)



总结

AWS

SUMMIT

amazon web services