敏捷开发·迭代创新

基于公有云的开发运维DevOps实践

毛郸榕 | Danrong Mao
亚马逊AWS 解决方案架构师

客户分享

musical.ly公司成立于2013年，是一家互联网创业公司，致力于打造全球化的娱乐社交平台，其开发和运营的音乐短视频社交平台musical.ly和视频直播平台live.ly已经成为欧美国家的主流移动社交网络，在全球超过20多个国家和地区的App Store总榜上排名第一，其用户规模仅次于Snapchat、Instagram、Twitter和Facebook等老牌社交应用，成为欧美13~20岁年轻人最喜爱的移动社交应用之一。目前，musical.ly的音乐短视频社交平台musical.ly的日活用户数已经超千万，用户总数超过1.5亿。

从一台Amazon EC2、几百位用户起步，经过短短的三年，musical.ly的日活用户数就超过了千万，同时使用的Amazon EC2数量也超过了2000台。在寻找业务方向、尝试在不同领域开展业务的过程中，AWS云平台给musical.ly提供了充足的灵活性。

客户分享

在运营系统中，musical.ly采用
（1）Amazon EC2作为主要的计算资源以支撑前端应用；
（2）Amazon S3作为统一的存储以存放多媒体资料及系统日志信息；
（3）Amazon CloudFront用来进行全球化的内容分发，降低访问时延以提升用户的使用体验；
（4）Amazon EMR作为灵活开关的大数据处理平台来完成必要的分析任务，
（5）AWS Lambda用于完成对新用户的推送服务。
这些产品的使用极大地降低了musical.ly自己的应用开发工作量，在系统的可靠性、稳定性方面也远远超过自己搭建的系统。

目前，musical.ly所使用的AWS云平台产品包括Amazon EC2、Amazon S3、Elastic Load Balancing、Amazon CloudFront、Amazon Route53、Amazon Kinesis、Amazon ElastiCache、Amazon EMR、Amazon VPC等，打造出一个稳定、可靠、响应迅速的全球化娱乐社交平台

客户分享 - 选择AWS云平台带来的好处主要体现在三个方面

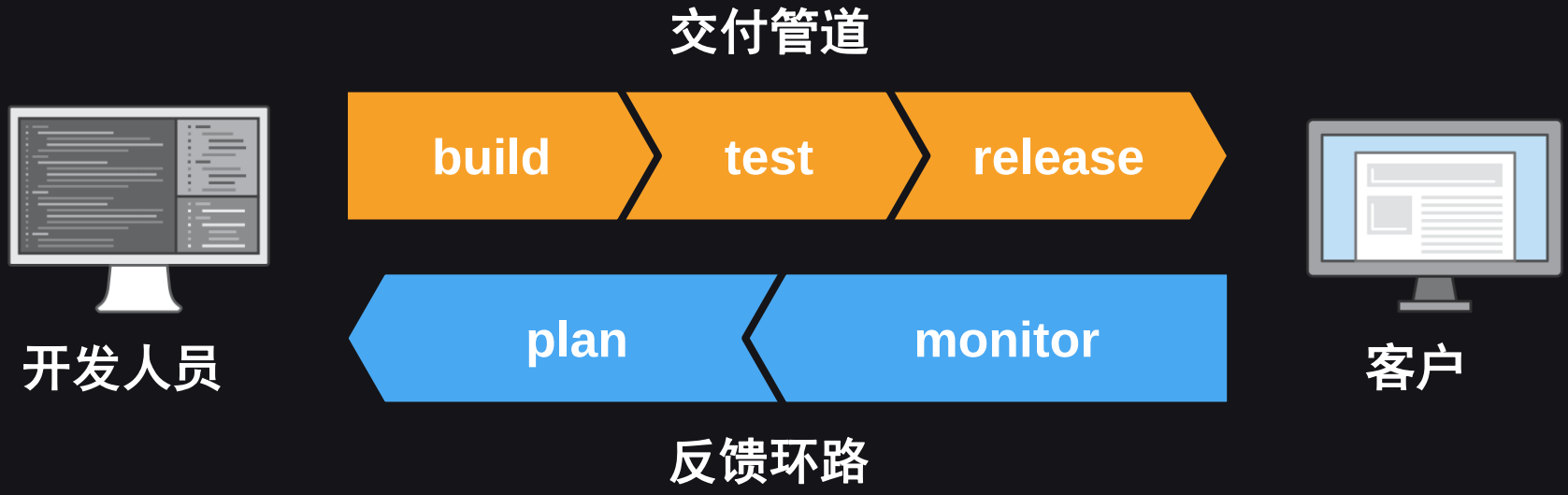
其一：AWS云平台提供的各项服务的稳定性和可靠性非常高，在系统运行的三年多来，从未出现过因云服务不稳定而导致的系统中断。

其二：是AWS云平台提供了非常全面的云服务，这使得musical.ly可以随着业务的增长随时调用新的服务来增添新功能和扩展系统。例如，musical.ly平台最初并不具备大数据分析和处理能力，但随着用户量的激增，分析和处理海量的用户信息，为用户提供更有针对性的信息就成为musical.ly的紧要任务。有了AWS云平台提供的Amazon EMR、Amazon Lambda等服务，musical.ly就能够快速地完成产品的迭代更新和功能扩展，节省了大量的时间成本。

其三：AWS优异的技术支持服务帮助musical.ly节省了大量的人力成本。

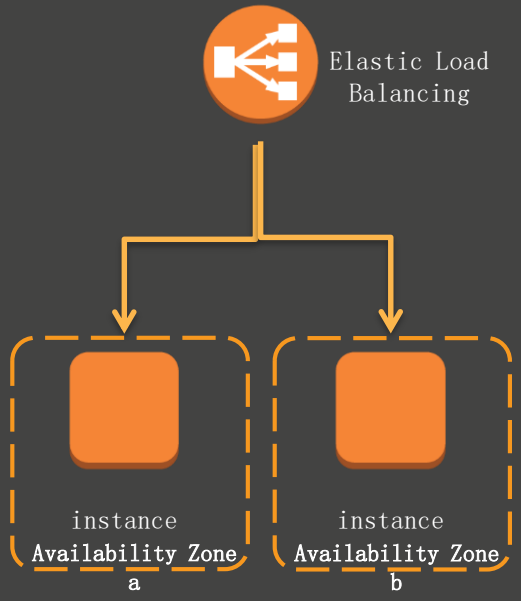
什么是DevOps?

软件开发周期



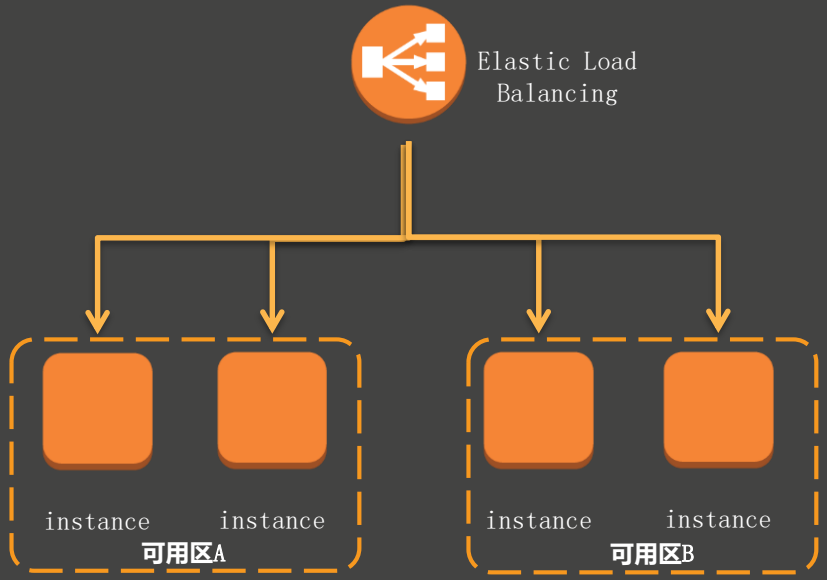
DevOps = 加速软件开发周期，提升效率

公有云的优势 - 弹性



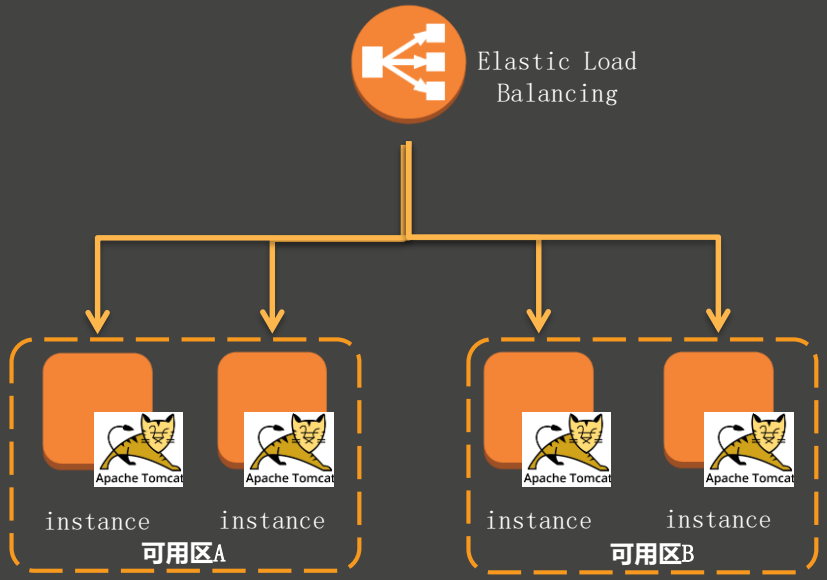
前端负载均衡接收客户端请求，然后发送给后端服务器

公有云的优势 - 弹性



前端负载均衡接收客户端请求，然后发送给后端服务器

公有云的优势 - 弹性 - 部署Tomcat



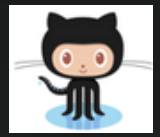
前端负载均衡接收客户端请求，然后发送给后端服务器

一天内代码发布多次，迭代创新

聚焦业务创新，不用把时间浪费在搭建环境上

我们需要基于公有云方式的开发运维工具

常见的开发部署方式



Git Repository

Jenkins

Junit4

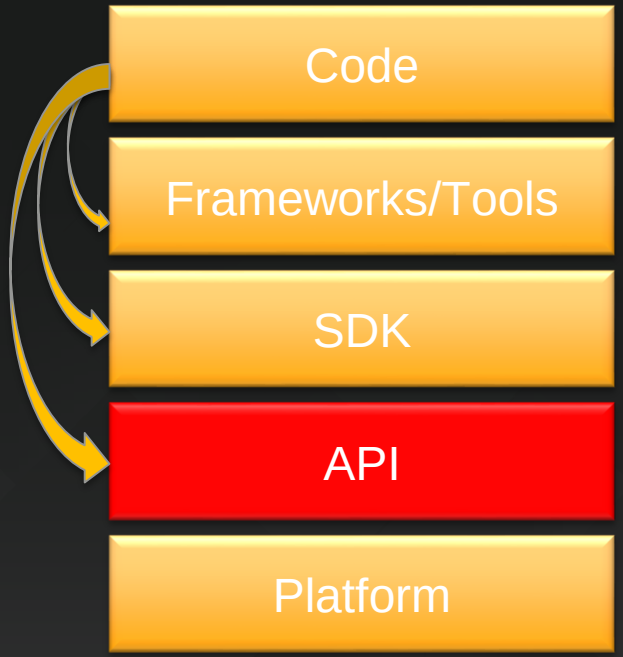
Bucket

EC2

怎么样实现DevOps

amazon web services

从软件开发角度看DevOps



Logic and control

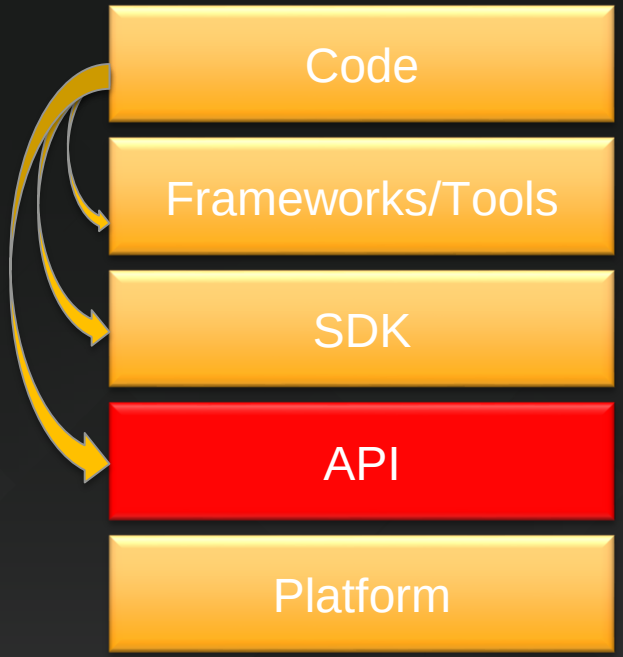
Easy to use

Multiple language

Automation foundation

Infrastructure and OS

从软件开发角度看DevOps



Programs, CLI, Scripts, DML, …

Python, Java, PHP, Node.js, PHP, Ruby, GO, …

REST API

基础服务

亚马逊电商怎么做DevOps?

amazon web services

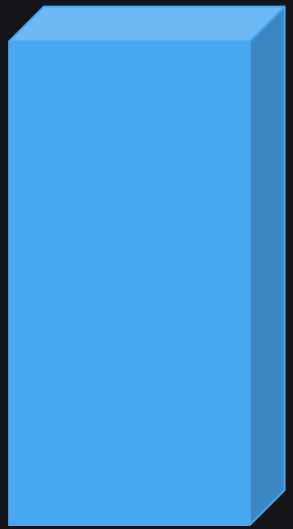
Amazon.com 代码部署

~11.6s	~1,079	~10,000	~30,000
平均代码保护署间隔时间	1小时内的最多代码部署次数	平均每个主机的部署次数	最多相同时刻部署代码的主机数

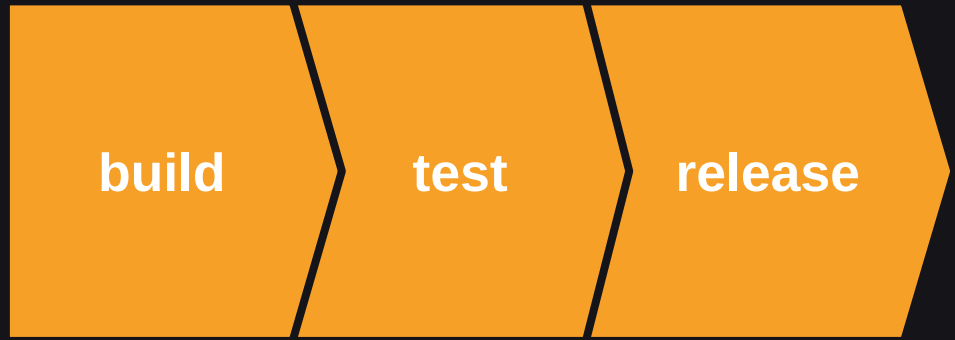

早期，我们也是一个单体架构
A dozen years ago, Amazon.com was a monothic app, very centralized

一体化架构的开发生命周期

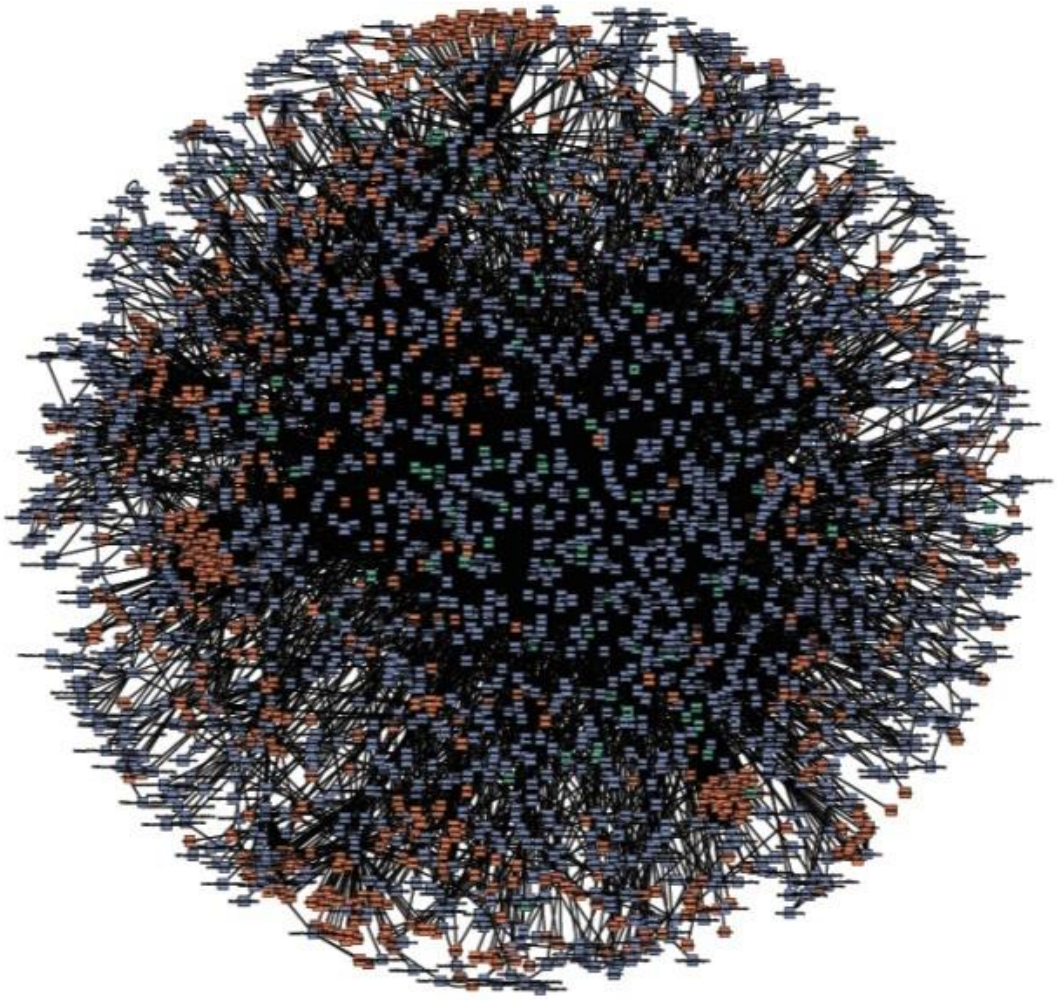
开发人员



应用



delivery pipeline（交付管道）



面向服务的架构

单一目的

通过APIs 连接

高度解耦

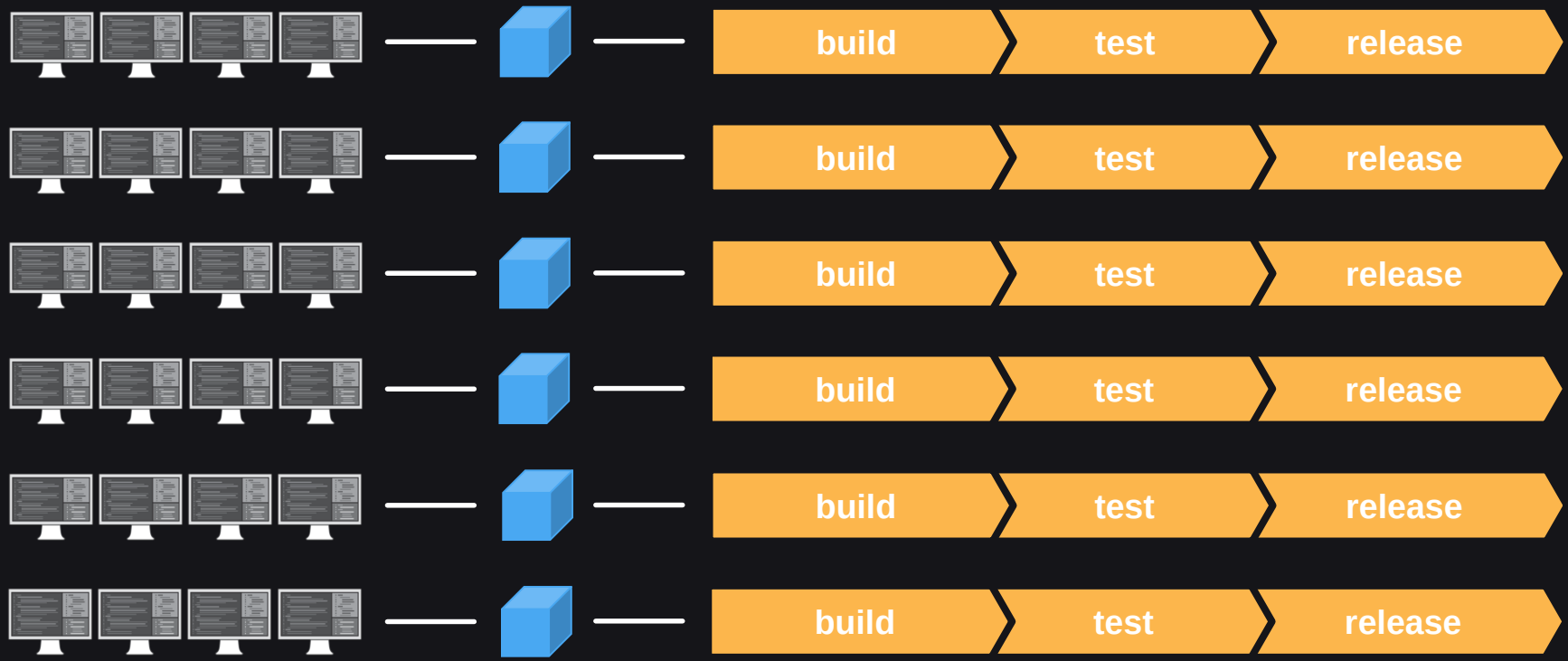
微服务
“Microservices”

工具？

开发人员　　服务

交付管道

Microservice开发生命周期



AWS Code Services

amazon web services

AWS DevOps - 强大的生态系统解决所有软件

开发 | 构建 | 测试 | 部署 | 搭建 | 监控 | 运维

GitHub

Jenkins

CodeStar
CodeCommit

CodeBuild

Solano Labs

CodePipeline

Opsworks

Elastic Beanstalk

Elastic Container Service

Lambda

CodeDeploy

CloudFormation

CloudTrail

CloudWatch

版本管理

自动构建
持续集成

配置管理

持续交付

持续部署

基础设施即代码

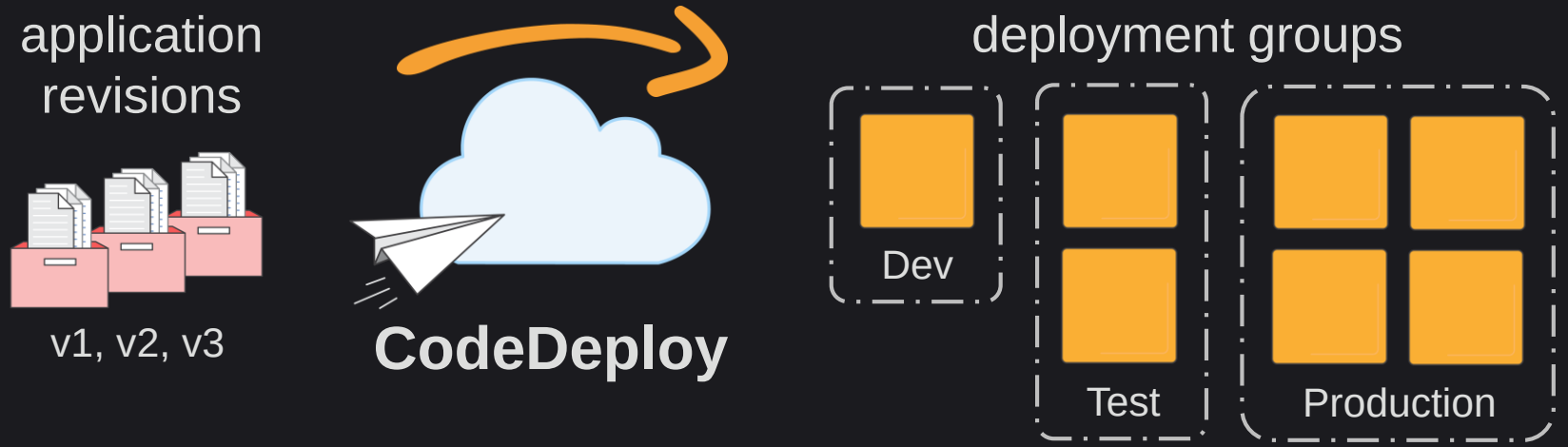
自动化监控

自动化运维

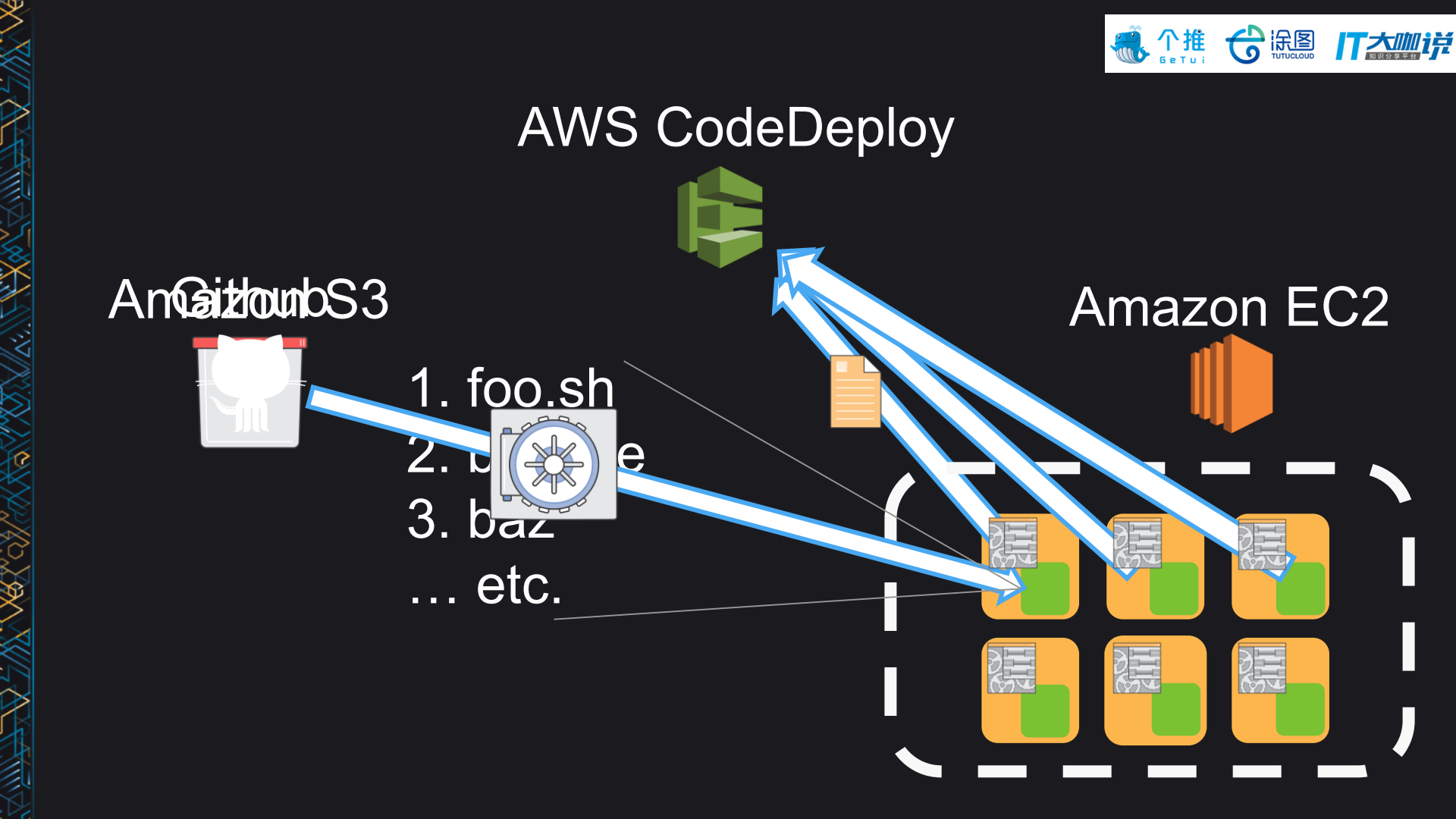
日志管理

AWS CodeDeploy - 代码部署

协调自动化部署的过程



简单可靠的部署
从单个实例扩展到上千个实例
部署到任何服务器上: AWS or on-premises
中央集中控制和监控
代码滚动升级


AWS CodeDeploy
Amazon S3
GitHub
1. foo.sh
2.
3. baz
… etc.
Amazon EC2

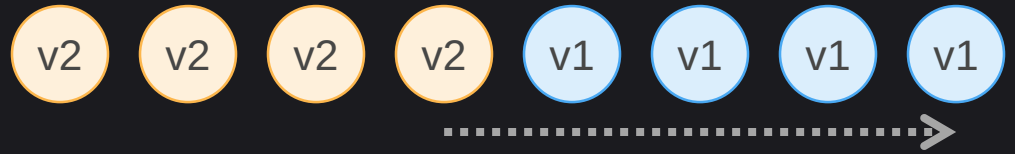
怎么使用CodeDeploy服务进行部署

```
#!/bin/bash
yum -y update
yum install -y ruby
yum install -y aws-cli
cd /home/ec2-user
aws s3 cp s3://aws-codedeploy-cn-north-1/latest/install .
--region cn-north-1
chmod +x ./install
./install auto
```

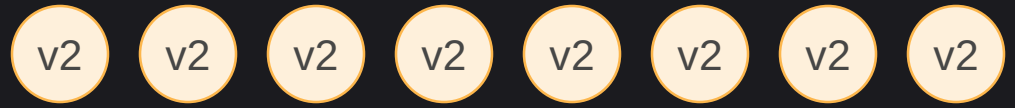
Deployment Config - 在软件开发领域，我们追求极速！快！

一次一台
最少健康主机数目 = 99%

一次一半
最少健康主机数目 = 50%



一次全部
最少健康主机数目 = 0



[其他定制方式]
最少健康主机数目 = 75%

1) 打包 app　2) 指定部署目标　3) 部署!

```
version: 0.0
os: linux
files:
  - source: chef/
    destination: /etc/chef/codedeploy
  - source: target/hello.war
    destination: /var/lib/tomcat6/webapps
hooks:
  ApplicationStop:
    - location: deploy_hooks/stop-tomcat.sh
  BeforeInstall:
    - location: deploy_hooks/install-chef.sh
    - location: deploy_hooks/chef-solo.sh
  ApplicationStart:
    - location: deploy_hooks/start-tomcat.sh
  ValidateService:
    - location: deploy_hooks/verify_service.sh
```

怎么使用CodeDeploy服务进行部署 - 此处用tomcat举例

```
cms.front/
├── appspec.yml
├── css/
├── error.jsp
├── fonts/
├── html/
├── images/
├── index.jsp
├── js/
├── META-INF/
├── scripts/
├── static/
└── WEB-INF/
```

准备代码，此处用tomcat应用举例。
解压代码，在工程目录目录里添加scripts目录和一个appspec.yml，例如下面这样的目录结构，scripts里面放部署时使用的脚本，appspec.yml文件定义代码部署到哪个，用什么用户在什么阶段执行什么脚本

怎么使用CodeDeploy服务进行部署 - 此处用tomcat举例

```yaml
version: 0.0
os: linux
files:
  - source: /
    destination: /home/ec2-user/tomcat/webapps/
hooks:
  BeforeInstall:
    - location: scripts/check_dependencies.sh
      timeout: 300
      runas: root
  AfterInstall:
    - location: scripts/change_permissions.sh
      timeout: 300
      runas: root
  ApplicationStart:
    - location: scripts/start_server.sh
      timeout: 300
      runas: ec2-user
  ApplicationStop:
    - location: scripts/stop_server.sh
      timeout: 300
      runas: ec2-user
```

首次部署，需要安装依赖软件和环境，之后做代码更新时就不需要了

1) 打包 app　2) 指定部署目标　3) 部署

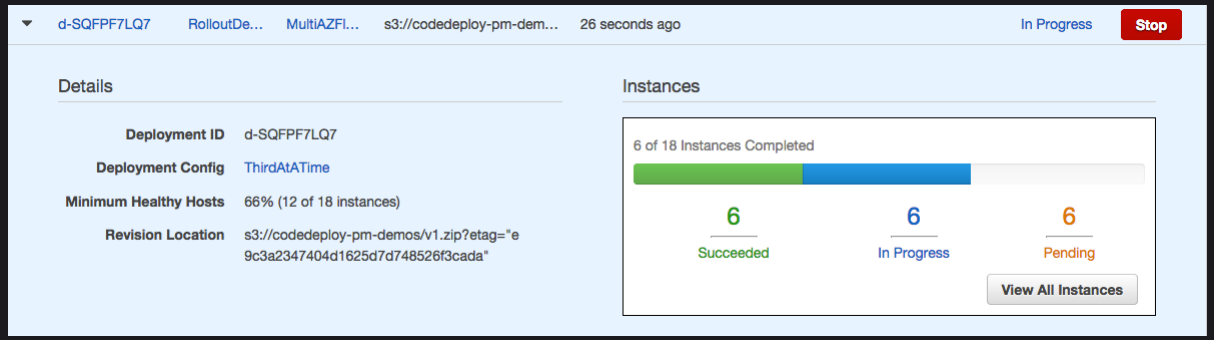
将实例按下述编组:

- 弹性扩展组(Auto-scaling group)
- Amazon EC2标签
- On-premises标签

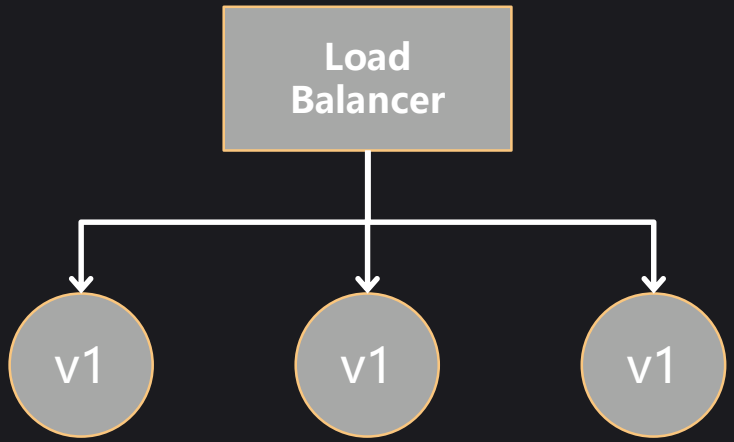
1) 打包 app　2) 指定部署目标　3) 部署

AWS CLI & SDKs
AWS Console
CI / CD Partners
GitHub

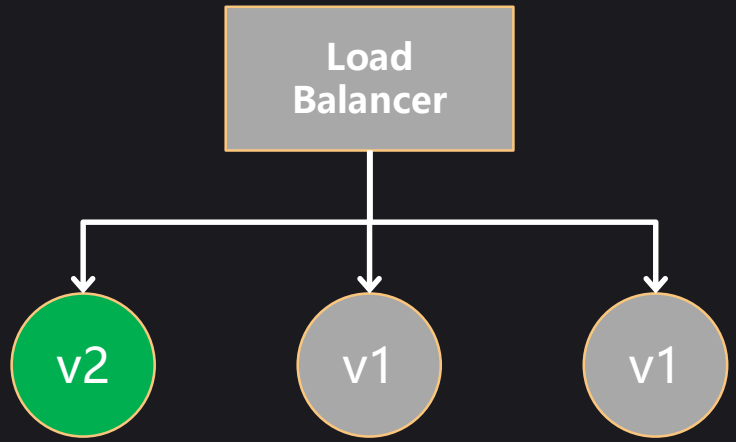
```
aws deploy create-deployment \
--application-name MyApp \
--deployment-group-name TargetGroup \
--s3-location
bucket=MyBucket,key=MyApp.zip
```



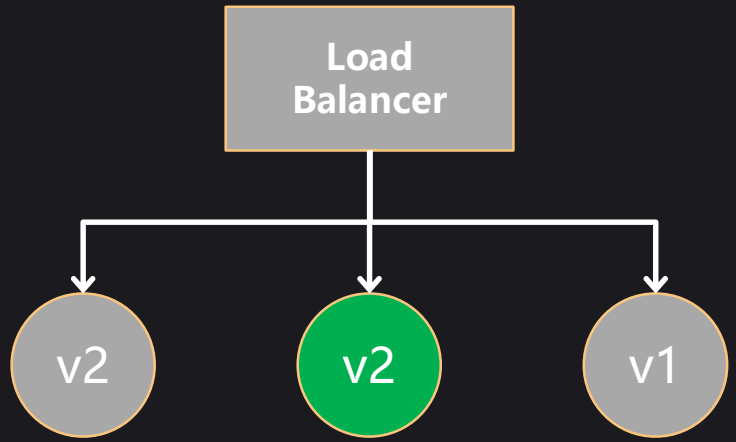
滚动升级—不下线部署



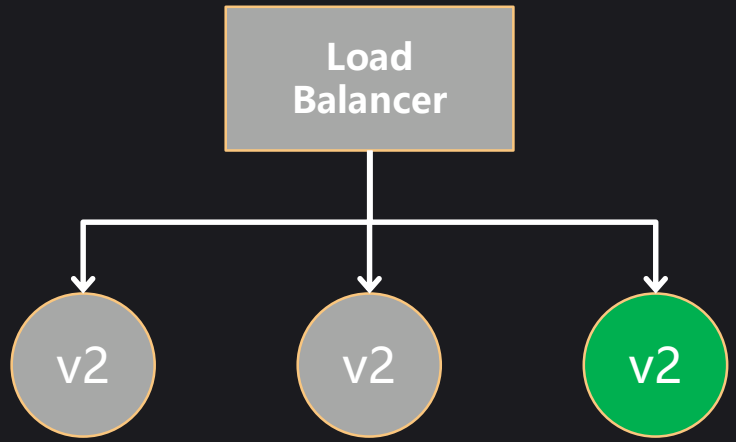
滚动升级—不下线部署



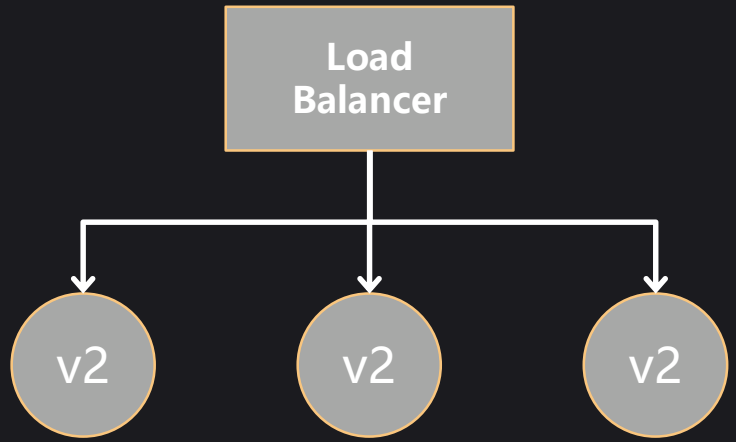
滚动升级—不下线部署



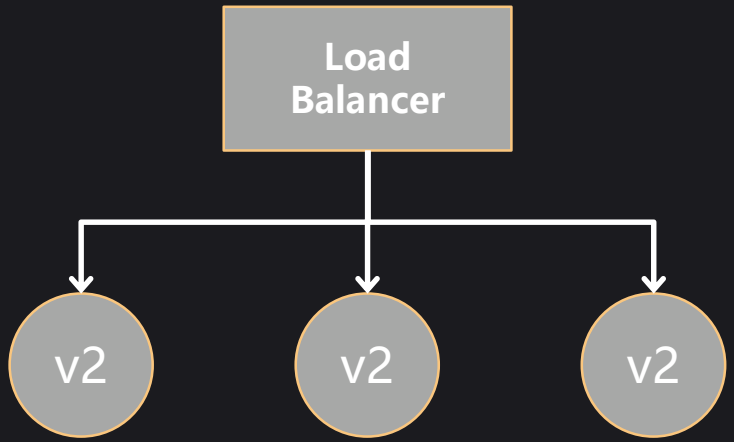
滚动升级—不下线部署



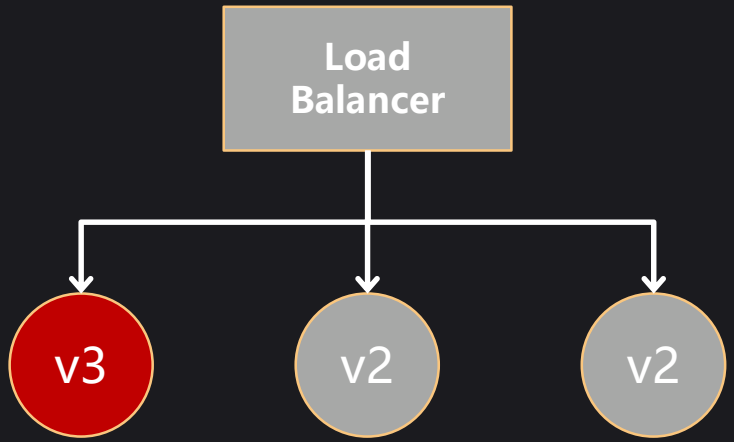
滚动升级—不下线部署



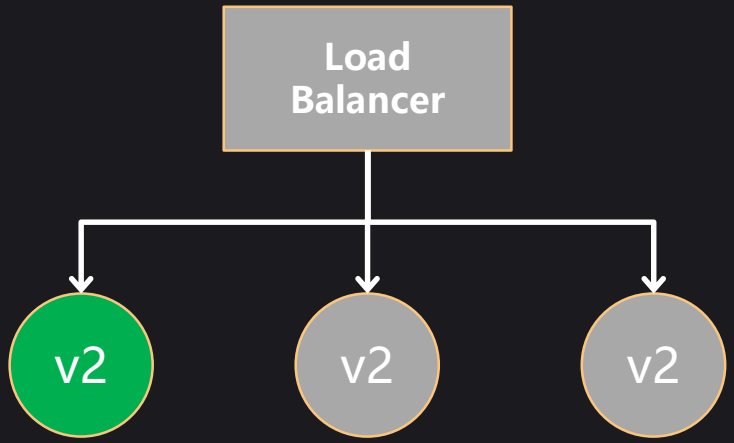
健康检查—捕获任何部署问题



健康检查—捕获任何部署问题

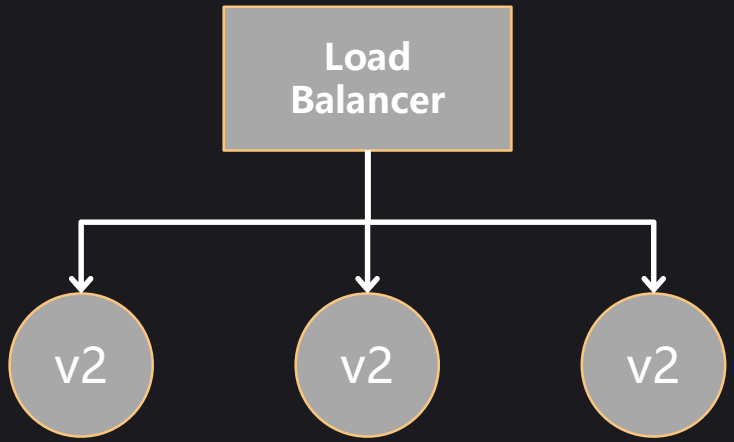


健康检查—捕获任何部署问题



回滚

健康检查—捕获任何部署问题



AWS CodeDeploy - 蓝绿部署

欢迎参加AWS中国区技术培训!

这是用AWS CodeDeploy部署的应用程序!

更多详细步骤请参考 AWS CodeDeploy 文档.

danrongm@amazon.com.

AWS CodeDeploy

部署

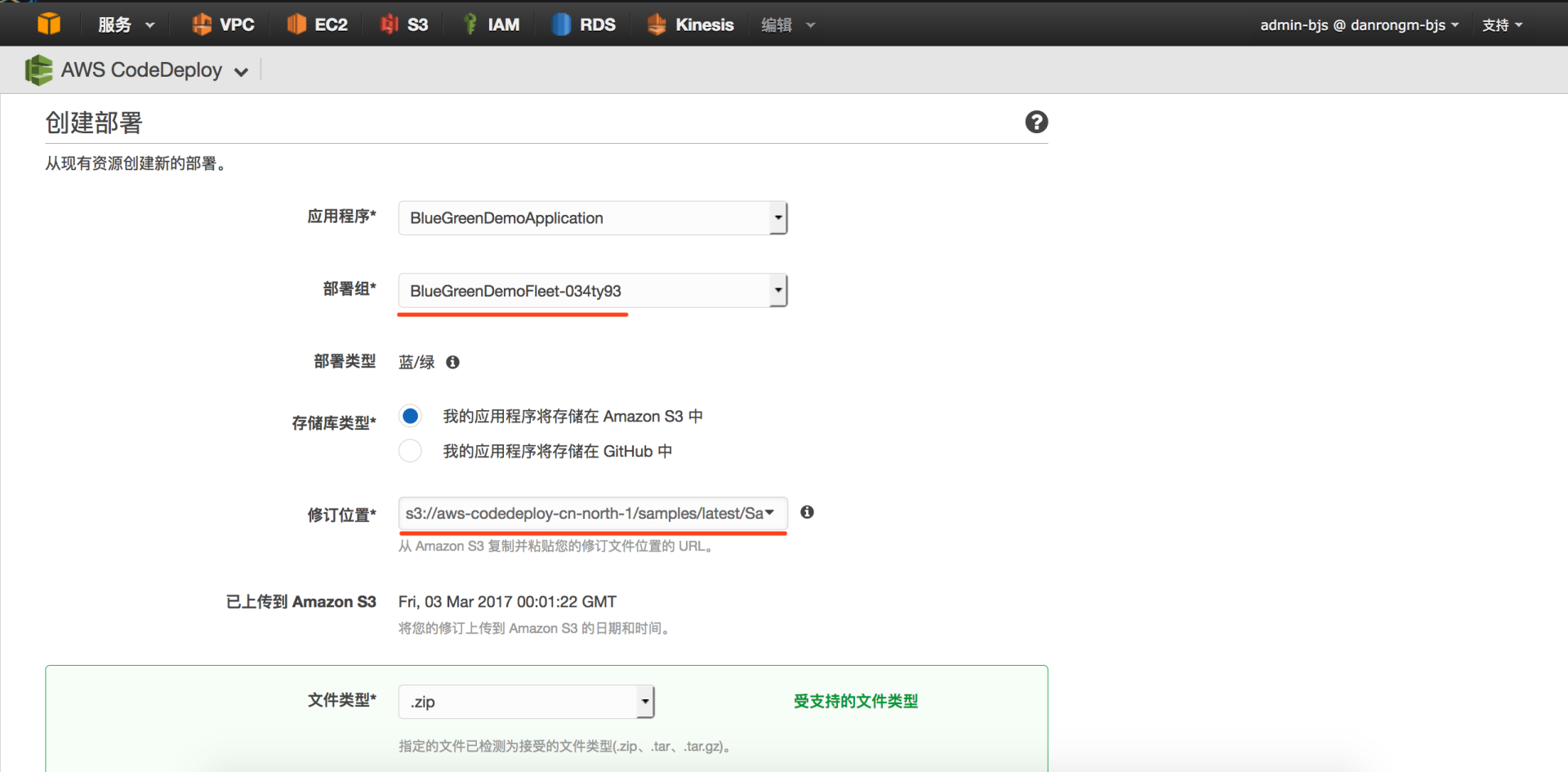
查看、诊断和管理您的部署。

创建部署

过滤器 所有部署 | 按部署 ID 搜索 | 每页部署数 10 | 正在查看第 1 到 6 个部署，共 6 个部署

部署 ID	类型	应用...	部署组	修订...	开...	结...	正在发起事件	状态	操作
d-6NJMMM2NL	蓝/绿	BlueG...	BlueG...	s3://bj...	20...	20...		已成功	
d-XSV3HDWAL	蓝/绿	BlueG...	BlueG...	s3://bj...	20...	20...		已成功	
d-7GSLL7XAL	蓝/绿	BlueG...	BlueG...	s3://bj...	20...	20...		已成功	
d-QLI8AHUAL	蓝/绿	BlueG...	BlueG...	s3://bj...	20...	20...		已成功	
d-DXL4XQNAL	蓝/绿	BlueG...	BlueG...	s3://a...	20...	20...		已成功	
d-FON6N9NAL	就地	BlueG...	BlueG...	s3://a...	20...	20...		已成功	

AWS CodeDeploy



AWS CodeDeploy

ApplicationStop 生命周期事件失败

- [x] 如果实例上的此生命周期事件失败，就不要将此部署到实例上。 了解更多信息

内容选项

选择操作；在部署期间，当目标实例上的文件与相同目标位置的应用程序修订中的文件具有相同名称时，AWS CodeDeploy 将执行该操作。 了解更多信息

- ( ) **使部署失败**
  系统报告出错，并且部署状态更改为“失败”。
- (•) **覆盖内容**
  将应用程序修订中的文件复制到实例上的目标位置，并替换以前的文件。
- ( ) **保留内容**
  不将应用程序修订中的文件复制到实例。现有文件将保留在目标位置中，并被视为新部署的一部分。

回滚配置覆盖

您可以配置仅适用于此部署的回滚选项。将会使用您在此处选择的回滚选项，替代为配置组配置的选项。

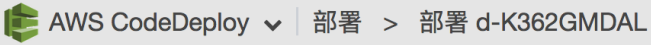
- [x] 部署失败时回滚
- [ ] 达到警报阈值时回滚

- [ ] 禁用回滚

*必填项

取消 部署

AWS CodeDeploy



部署: d-K362GMDAL

部署 已成功

部署进度

步骤 1
预置替换实例
已预配 3 个替换实例，共 3 个

步骤 2
在替换实例上安装应用程序
已更新 3 个实例，共 3 个

步骤 3
将流量重新路由到替换实例

步骤 4
正在终止原始实例
已终止 3 个替换实例，共 3 个

接收流量的实例

监控在您的负载均衡器之后将流量由一组实例重新路由到另一组实例的进度。

原始	替换
0	3
0 个原始实例，共 3 个原始实例	3 个替换实例，共 3 个替换实例
0 个实例未能取消注册	0 个实例注册失败

AWS CodeDeploy

部署详细信息

实例活动

过滤器 状态 环境 流量 每页实例数 10 正在查看第 1 到 6 个实例，共 6 个实例

实例 ID	环境	流量	开始时间	结束时间	持续时间	状态	最新事件	事件
i-00fec37a61b11ded5	原始	否	2017 年 5 月 18 日 4:04:25 PM UTC	2017 年 5 月 18 日 4:04:59 PM UTC	34 秒	已成功	AfterBlockTraffic	查看事件
i-024adbf24d3c2b2c2	替换	是	2017 年 5 月 18 日 4:01:41 PM UTC	2017 年 5 月 18 日 4:03:03 PM UTC	1 分钟, 22 秒	已成功	AfterAllowTraffic	查看事件
i-036d4084be805b763	替换	是	2017 年 5 月 18 日 4:02:07 PM UTC	2017 年 5 月 18 日 4:04:17 PM UTC	2 分钟, 11 秒	已成功	AfterAllowTraffic	查看事件
i-05efa156045ec857f	替换	是	2017 年 5 月 18 日 4:01:54 PM UTC	2017 年 5 月 18 日 4:03:41 PM UTC	1 分钟, 47 秒	已成功	AfterAllowTraffic	查看事件
i-0af55fc19f9f7012a	原始	否	2017 年 5 月 18 日 4:04:25 PM UTC	2017 年 5 月 18 日 4:04:59 PM UTC	34 秒	已成功	AfterBlockTraffic	查看事件
i-0f28169e3320a24e2	原始	否	2017 年 5 月 18 日 4:04:26 PM UTC	2017 年 5 月 18 日 4:05:00 PM UTC	34 秒	已成功	AfterBlockTraffic	查看事件

AWS CodeDeploy

AWS CodeDeploy > 部署 > 部署 d-K362GMDAL > 实例 i-00fec37a61b11ded5 事件

事件：实例 i-00fec37a61b11ded5

部署组详细信息

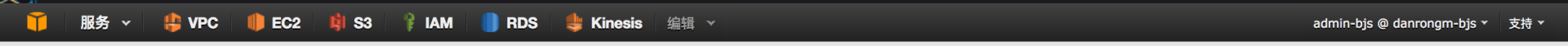
部署组	BlueGreenDemoFleet-voxxmpl
部署 ID	d-K362GMDAL
部署配置	CodeDeployDefault.OneAtATime
正常运行的最少主机数	2 个实例，共 3 个实例

修订

修订位置	s3://aws-codedeploy-cn-north-1/samples/latest/SampleApp_Linux.zip?etag="5f6607299a7cc834efffbcc5134e1424"
已创建修订	2017 年 5 月 18 日 3:30:11 PM UTC
描述	Application revision registered by Deployment ID: d-8CTNSFEAL

事件	开始时间	结束时间	持续时间	状态	日志	操作
BeforeBlockTraffic	2017 年 5 月 18 日 4:04:25 PM UTC	2017 年 5 月 18 日 4:04:25 PM UTC	不到一秒钟	已成功		
BlockTraffic	2017 年 5 月 18 日 4:04:27 PM UTC	2017 年 5 月 18 日 4:04:58 PM UTC	31 秒	已成功		
AfterBlockTraffic	2017 年 5 月 18 日 4:04:59 PM UTC	2017 年 5 月 18 日 4:04:59 PM UTC	不到一秒钟	已成功		

AWS CodeDeploy



事件：实例 i-024adbf24d3c2b2c2

部署组详细信息

部署组	BlueGreenDemoFleet-voxxmpl
部署 ID	d-K362GMDAL
部署配置	CodeDeployDefault.OneAtATime
正常运行的最少主机数	2 个实例，共 3 个实例

修订

修订位置	s3://aws-codedeploy-cn-north-1/samples/latest/SampleApp_Linux.zip?etag="5f6607299a7cc834efffbcc5134e1424"
已创建修订	2017 年 5 月 18 日 3:30:11 PM UTC
描述	Application revision registered by Deployment ID: d-8CTNSFEAL

事件	开始时间	结束时间	持续时间	状态	日志	操作
ApplicationStop	2017 年 5 月 18 日 4:01:41 PM UTC	2017 年 5 月 18 日 4:01:41 PM UTC	不到一秒钟	已成功		
DownloadBundle	2017 年 5 月 18 日 4:01:42 PM UTC	2017 年 5 月 18 日 4:01:42 PM UTC	不到一秒钟	已成功		
BeforeInstall	2017 年 5 月 18 日 4:01:44 PM UTC	2017 年 5 月 18 日 4:01:45 PM UTC	1 秒	已成功		
Install	2017 年 5 月 18 日 4:01:46 PM UTC	2017 年 5 月 18 日 4:01:46 PM UTC	不到一秒钟	已成功		
AfterInstall	2017 年 5 月 18 日 4:01:47 PM UTC	2017 年 5 月 18 日 4:01:48 PM UTC	不到一秒钟	已成功		
ApplicationStart	2017 年 5 月 18 日 4:01:49 PM UTC	2017 年 5 月 18 日 4:01:49 PM UTC	不到一秒钟	已成功		
ValidateService	2017 年 5 月 18 日 4:01:51 PM UTC	2017 年 5 月 18 日 4:01:51 PM UTC	不到一秒钟	已成功		
BeforeAllowTraffic	2017 年 5 月 18 日 4:02:30 PM UTC	2017 年 5 月 18 日 4:02:30 PM UTC	不到一秒钟	已成功		
AllowTraffic	2017 年 5 月 18 日 4:02:31 PM UTC	2017 年 5 月 18 日 4:03:02 PM UTC	31 秒	已成功		
AfterAllowTraffic	2017 年 5 月 18 日 4:03:03 PM UTC	2017 年 5 月 18 日 4:03:03 PM UTC	不到一秒钟	已成功		

AWS CodeDeploy - 蓝绿部署

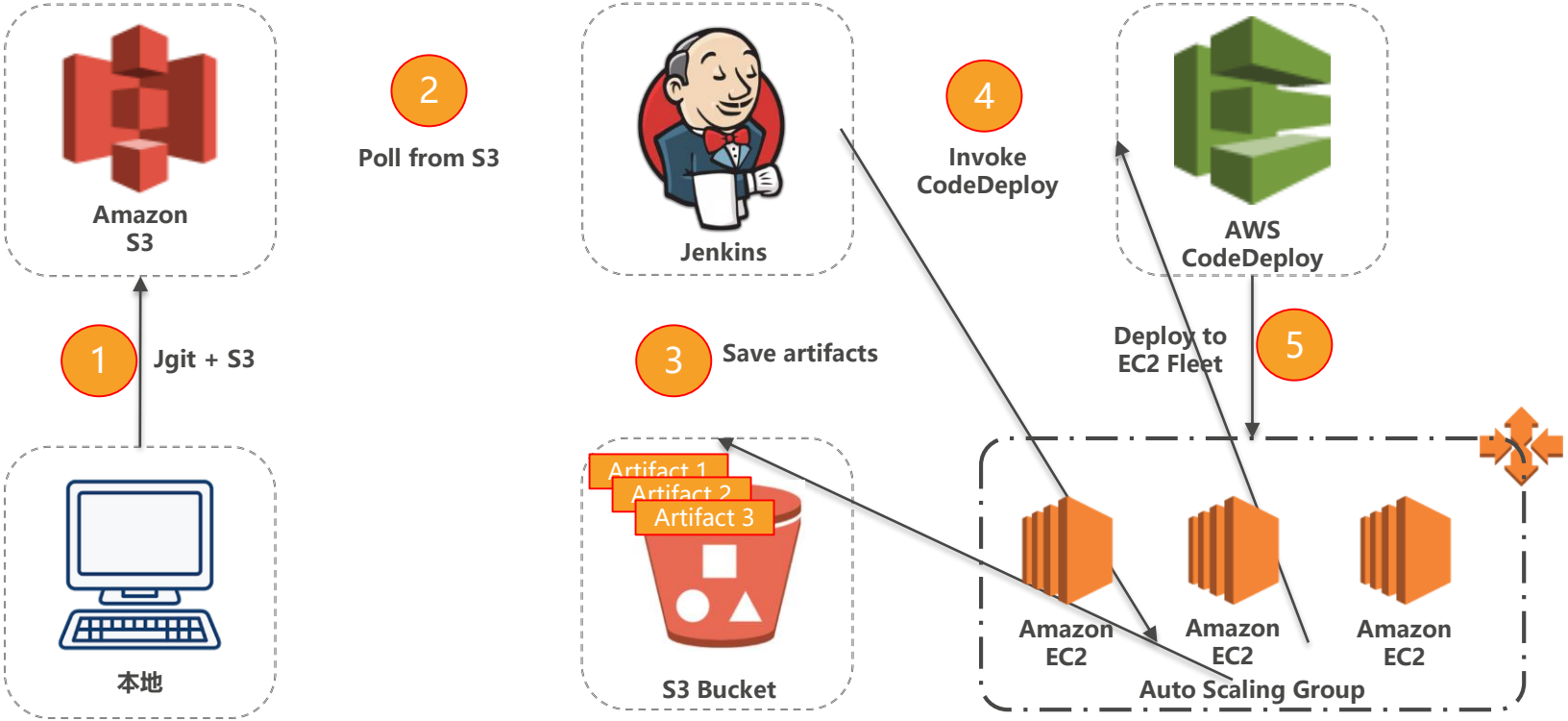
欢迎参加AWS中国区技术培训!

这是用AWS CodeDeploy部署的应用程序!

更多详细步骤请参考 AWS CodeDeploy 文档.

danrongm@amazon.com.

Building CI & CD in AWS



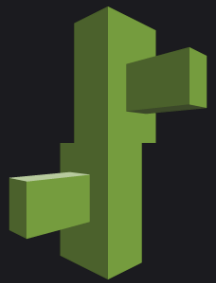
AWS基础设施DevOps Services

amazon web services

基于AWS的基础设施层面DevOps实践要素

高级服务

自己动手



Elastic Beanstalk

OpsWorks

CloudFormation

代码/命令行

便利

控制

IAM

CloudWatch

Python 代码 --启动2个EC2 实例

runec2.py

```
1  import boto.ec2
2
3  conn = boto.ec2.connect_to_region("cn-north-1")
4  conn.run_instances(
5      'ami-981d8fa1',
6      min_count=2,
7      max_count=2,
8      key_name= ‘test-cn’
9      instance_type='t2.micro',
10     security_groups=['wslinux'],
11     )
12
```

CloudFormation

https://aws.amazon.com/cn/blogs/china/cloudformation-migrating/

CloudFormation 一键部署所有云端资源

模板

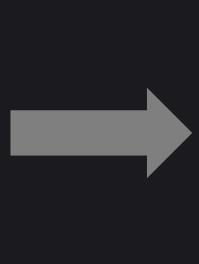
JSON 格式的文件

参数定义

需要的资源

具体配置

CloudFormation



框架

创建堆栈

更新堆栈

错误检查和回滚

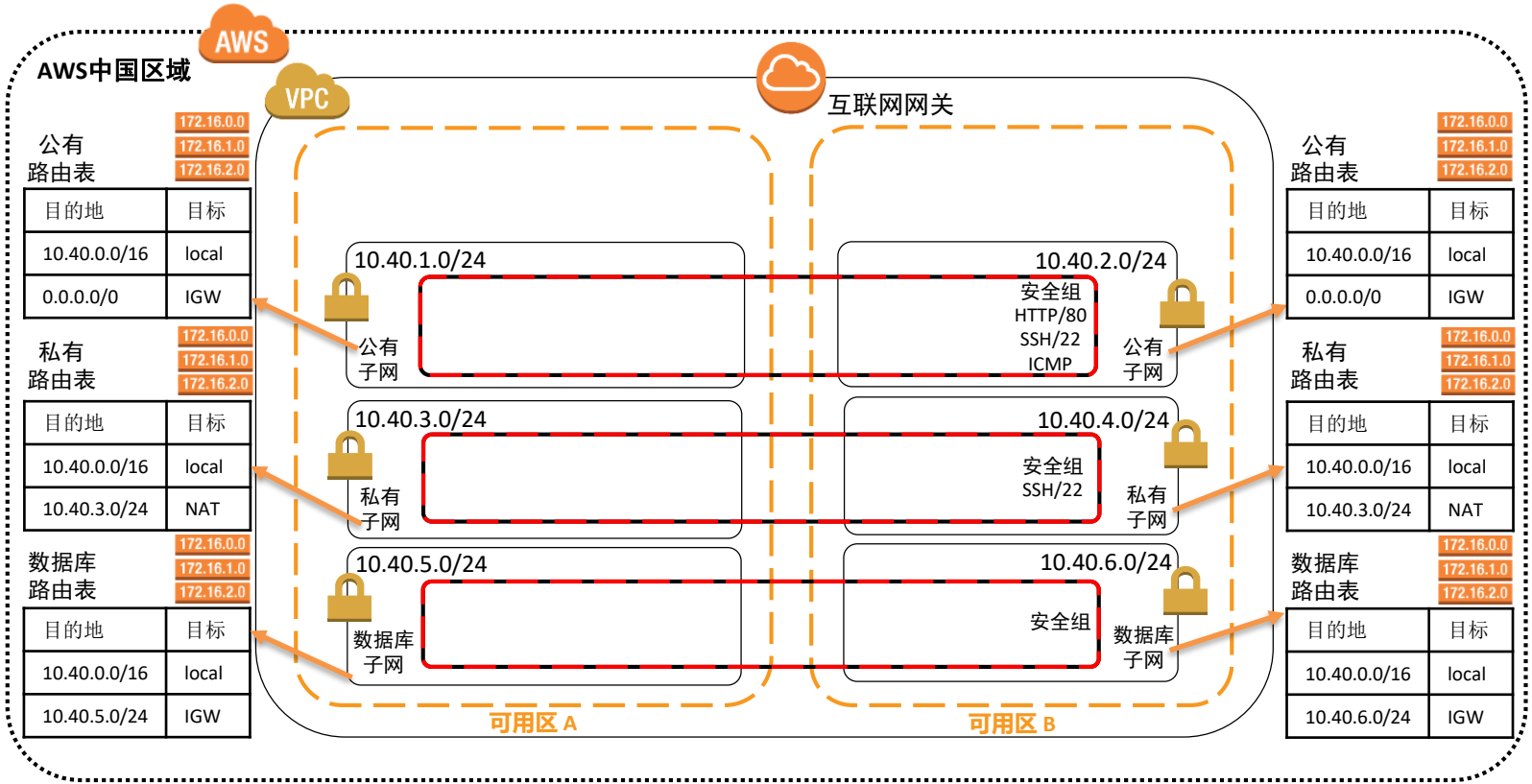
堆栈

配置好的AWS资源

完整的SAWS服务支持

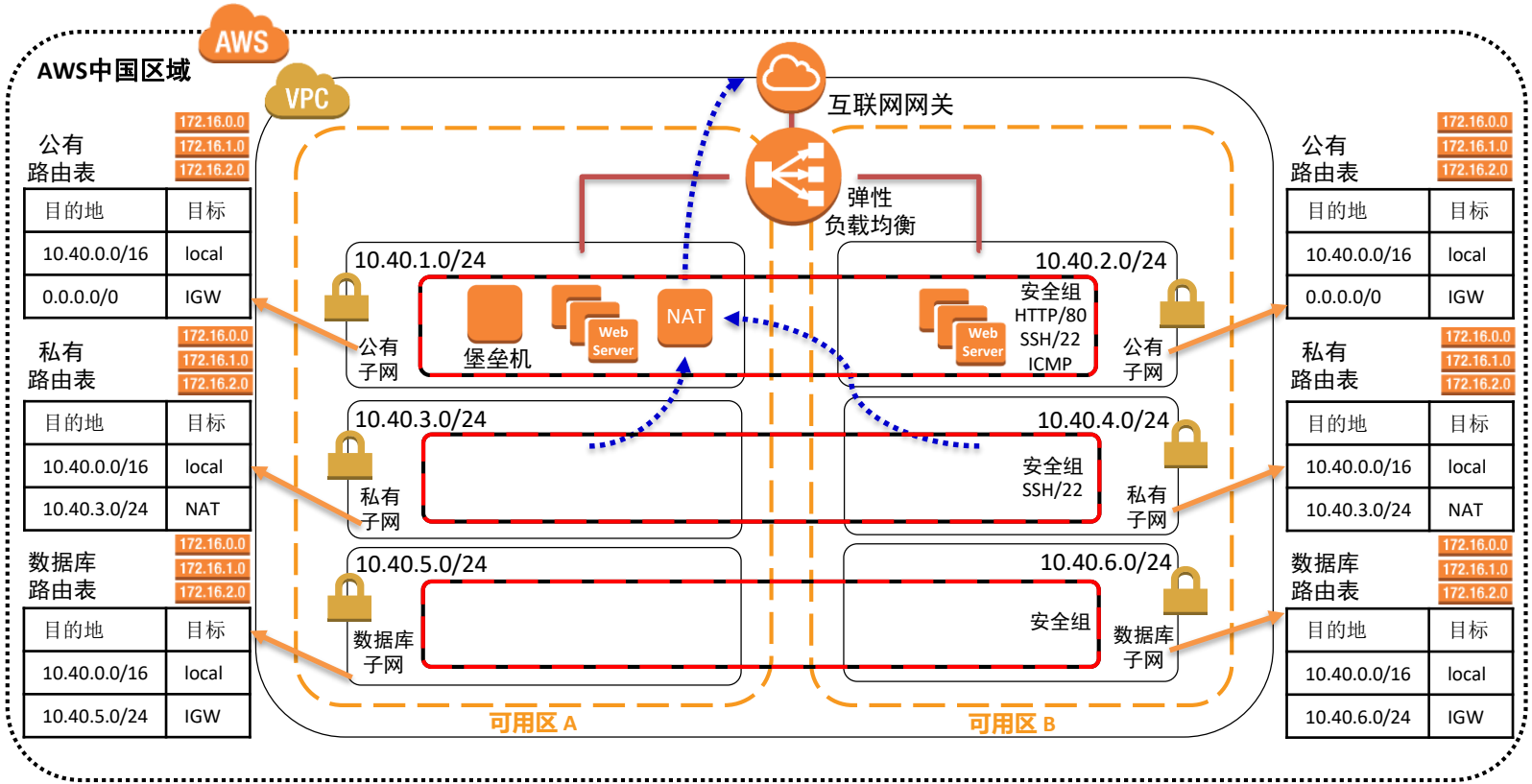
可定制化

上云之前，需要在云端搭建网络环境



https://s3.cn-north-1.amazonaws.com.cn/danrong-share/external/aws-template-bjs-addELB-V3.txt

尽可能缩短上云初始化设置，一键搞定云端网络环境



https://s3.cn-north-1.amazonaws.com.cn/danrong-share/external/aws-template-bjs-addELB-V3.txt

Web-Application-1	i-b77a500f	t2.medium	cn-north-1b	running	2/2 的检查...	无

实例: i-b77a500f (Web-Application-1) 弹性 IP: 54.223.121.26

描述 | 状态检查 | 监控 | 标签

实例 ID	i-b77a500f	公有 DNS	ec2-54-223-121-26.cn-north-1.compute.amazonaws.com.cn
实例状态	running	公有 IP	54.223.121.26
实例类型	t2.medium	弹性 IP	54.223.121.26*
私有 DNS	ip-10-40-2-82.cn-north-1.compute.internal	可用区	cn-north-1b

Welcome to nginx on the Amazon Linux AMI!

This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

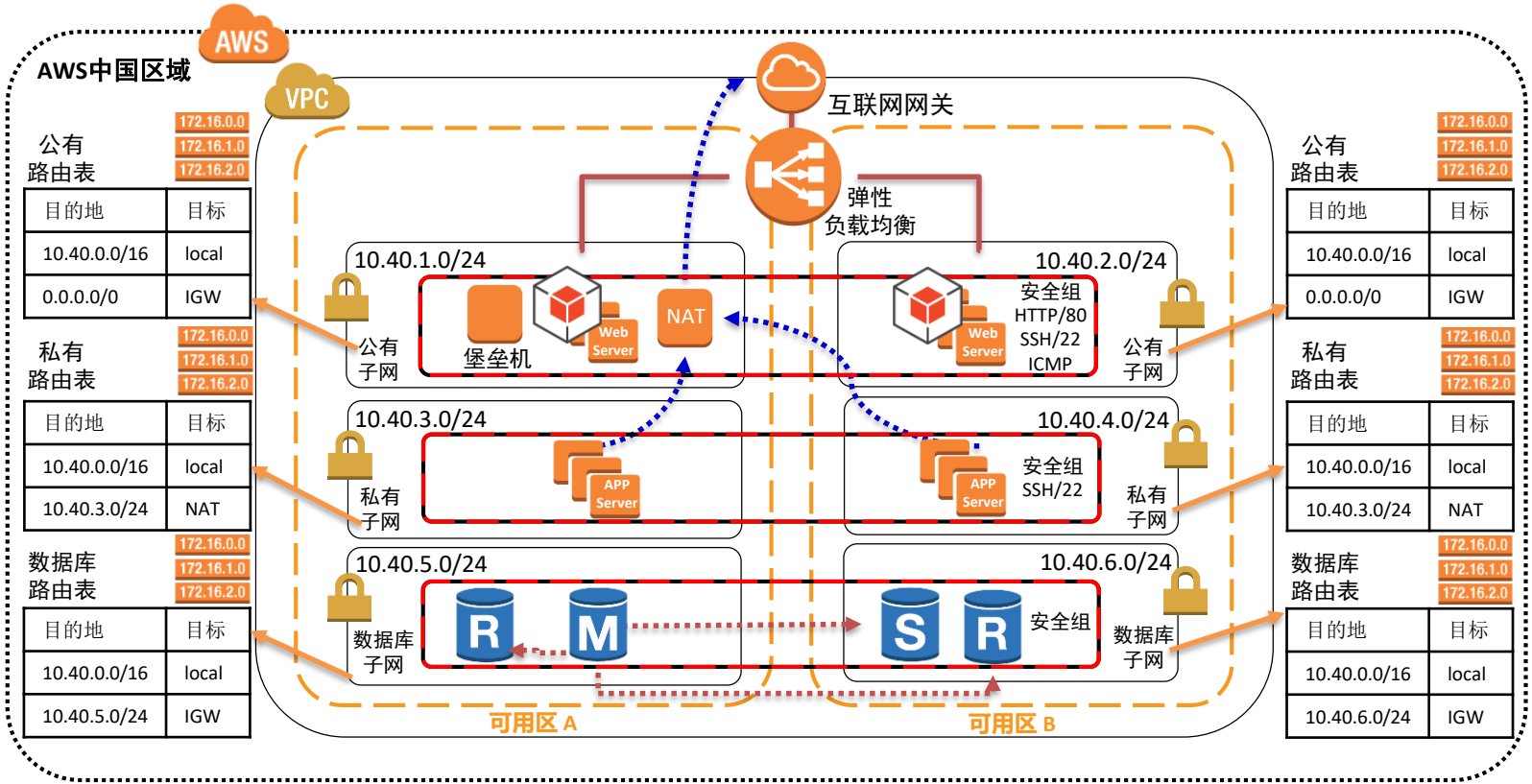
Website Administrator

This is the default `index.html` page that is distributed with **nginx** on the Amazon Linux AMI. It is located in `/usr/share/nginx/html`.

You should now put your content in a location of your choice and edit the `root` configuration directive in the **nginx** configuration file `/etc/nginx/nginx.conf`.

NGINX POWERED BY ELECTRICITY

一键搞定云端网络环境，您需要做什么？ - 仅仅需要在服务器上部署



https://s3.cn-north-1.amazonaws.com.cn/danrong-share/external/aws-template-bjs-addELB-V3.txt

Elastic Beanstalk - 部署应用需要提供的信息

ElasticBeanstalk

你的代码

01 Region

02 堆栈（容器）类型

03 单实例e 或 负载均衡+弹性扩展组

04 数据库(RDS) 可选

支持的平台

Java

node.js

php

docker

Microsoft .net

python

AWS DevOps - 强大的生态系统解决所有软件

开发 | 构建 | 测试 | 部署 | 搭建 | 监控 | 运维

GitHub

Jenkins

CodeStar

CodeCommit

CodeBuild

Solano Labs

CodePipeline

Opsworks

Elastic Beanstalk

Elastic Container Service

Lambda

CodeDeploy

CloudFormation

CloudTrail

CloudWatch

版本管理

自动构建
持续集成

配置管理

持续交付

持续部署

基础设施即代码

自动化监控

自动化运维

日志管理

初创企业支持计划

AWS为初创企业提供支持上云的云创计划到后续为云创成员专设的GTM全方位支持

GTM (go-to-market) 支持计划

亚马逊发明家 amazon launchpad

- 帮助硬件类初创企业在全球市场快速部署渠道
- 独享推广页面，支持影音视频内容播放
- 享受亚马逊物流、客服及支付等全方位支持

FbStart from facebook

- 云创计划成员可获得2,500美元Facebook广告credits

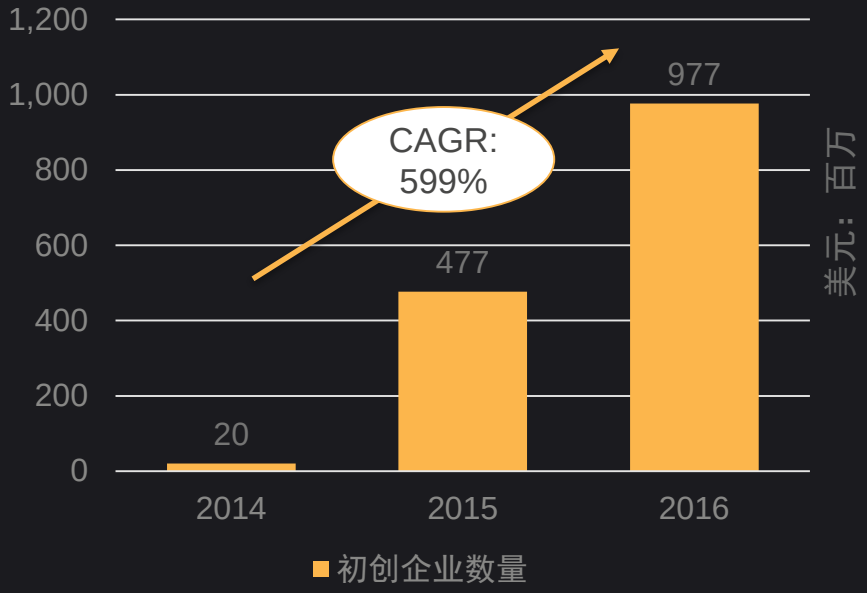
The Alexa Fund
$100 million in investment to fu innovation

- 云创计划成员可获得Alexa Fund投资的优先推荐资格
- Alexa Fund是亚马逊旗下专注于声音科技的早期投资基金（基金规模1亿美元）

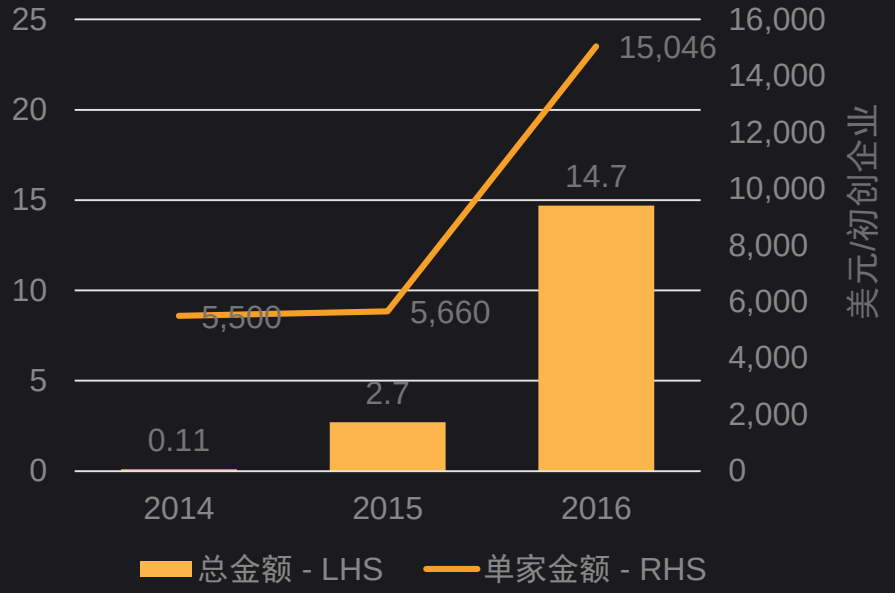
云创计划

至2016年，云创计划为近1,500家中国初创企业提供了近1,700万美元的额度支持

‘14-16 云创计划支持的中国初创企业数量



‘14-16 云创支持中国初创企业的免费额度



Thank You!

软件定义世界: 一切皆软件，一切皆API