基础设施事件准备

AWS 操作指南和最佳实践

2017 年 7 月

免责声明

本文档仅用于参考。本文档代表截至其发布之日的 AWS 最新产品服务和实践，如有变更，恕不另行通知。客户负责对本文档中的信息以及 AWS 产品或服务的任何用途进行独立评估，各项产品或服务均按“原样”提供，不作任何类型的明示或默示保证。本文档不形成 AWS、其附属公司、供应商或许可方的任何保证、陈述、合同承诺、条件或担保。AWS 对其客户承担的责任和义务受 AWS 协议制约，本文档不是 AWS 与客户之间任何协议的一部分，也不构成对该协议的修订。

目录

简介 1

基础设施事件准备计划 1

什么是计划基础设施事件？ 1

计划基础设施事件期间会发生什么？ 2

设计原则 2

离散工作负载 3

自动化 6

多样性/弹性 7

成本优化 10

事件管理流程 10

基础设施事件计划 10

计划和准备 11

操作准备 (事件当天) 18

事后活动 19

结论 21

鸣谢 21

延伸阅读 22

附录 22

架构评审详细清单 22

摘要

本白皮书面向在 Amazon Web Services (AWS) 上部署了生产工作负载，希望通过设计和预配置基于云的应用程序，从容动态地处理产品发布或季节性流量高峰等计划扩展事件的客户，介绍了相关操作指南和最佳实践。我们首先介绍了一些常规设计原则，针对基础设施事件计划的多个概念领域提供了具体的最佳实践与操作指南。然后介绍了操作准备方面的考虑因素和实践以及事后的活动。

简介

基础设施事件准备是指针对将影响业务的预期事件和重大事件，进行相应的设计和准备。在事件发生期间，面对所有条件和流量模式变化，公司的 Web 服务均保持可靠、响应迅速和高容错至关重要。这些事件包括向新区域扩张、新产品或新功能发布、季节性事件或重大业务公告或营销事件等。

如不正确计划基础设施事件，可能会对公司的商业声誉、持续经营或财务状况造成负面影响。基础设施故障事件的表现形式可能是意外服务故障、负载相关性能下降、网络延迟、存储容量限制、API 调用率等系统限制、可用 IP 地址数量有限、因监测不充分导致对应用程序堆栈组件的行为了解不足、对第三方服务的意外依赖、组件尚未进行扩展配置或其他意外错误条件。

要减少重要事件期间发生意外故障的风险，公司应投入时间和资源以进行计划和准备，培训员工，以及设计和记录相关流程。针对支持云的某个应用程序或一组应用程序进行基础设施事件计划，要求的投资大小取决于系统的复杂性和全球覆盖范围。不论公司云应用的范围或复杂性如何，本白皮书中提供的设计原则和最佳实践指南都能适用。

使用 Amazon Web Services (AWS)，公司可以通过动态、适应性强、按使用量付费的方式扩展其基础设施，为计划的扩展事件做好准备。Amazon 提供丰富、弹性和可编程的产品和服务，让公司拥有与 Amazon 运行其全球网络所用的同样高度安全、快速、可靠的基础设施，从而灵巧地适应快速变化的自身业务需求。

本白皮书概述了用于指导基础设施事件计划和执行的最佳实践和设计原则，展示了如何通过 AWS 服务来确保应用程序根据业务需要做好纵向和横向扩展的准备。

基础设施事件准备计划

本节介绍了计划基础设施事件的内容以及此类事件发生时通常会出现的活动种类。

什么是计划基础设施事件？

*计划基础设施事件*是一种由业务驱动、预期的、计划的事件窗口，是维护高响应率、高可扩展性、高容错能力 Web 服务的业务关键窗口。需求来源包括营销活动、与公司业务线有关的新闻活动、产品发布、区域扩张或者导致公司 Web 应用程序和底层基础设施的流量增加的类似活动。

计划基础设施事件期间会发生什么？

在大多数计划基础设施事件中，最主要的顾虑是能够提高 Web 基础设施的容量，以满足增加的流量需求。传统的本地环境预配置了物理计算、存储和网络资源，公司 IT 部门必须根据对理论最大峰值的最佳估计预配置额外容量。而风险就在于预配置的容量不足，公司会因 Web 服务器过载、响应缓慢以及其他运行时错误而遭受业务损失。

在 AWS 云中，基础设施是可编程、有弹性的。这意味着它不仅可以快速预配置以响应实时需求，还可以配置为根据系统指标作出自动、智能和动态的响应 – 根据需要增加或缩减资源，例如 Web 服务器群集、预配置吞吐量、存储容量、可用计算内核、流式处理分片的数量等等。

此外，许多 AWS 服务都采用全托管模式，包括存储、数据库、分析、应用程序和部署服务。这意味着，AWS 客户无需担心为高流量事件配置这些服务的复杂性。AWS 全托管服务采用可扩展性和高可用性设计。

通常在准备计划基础设施事件时，AWS 客户会进行系统评审，以评估其应用程序架构和运行准备情况，并考虑可扩展性和容错能力这两个因素。将会考虑流量预估值并与将其正常业务活动的性能进行比较，同时还会确定容量指标并估计额外容量需求，识别并解决任何潜在的瓶颈以及对第三方上下游的依赖关系。如果计划事件包含区域扩张或引入新的受众，则还会考虑地理因素。在计划事件开始之前进行向其他 AWS 区域或可用区的扩展。此外还会检查客户的 AWS 动态系统设置，例如 Auto Scaling、负载均衡、地理路由、高可用性和故障转移措施等，确保正确配置这些设置，以正确处理预期的容量和事务速率增加。考虑并在需要时修改静态设置，例如 AWS 资源限值和内容分发网络 (CDN) 源服务器的地址等。

另外还会在需要时审查和增强监控和通知机制，以保证所发生事件的实时透明性，以及在完成计划事件后的事后分析。

在计划事件期间，如果需要排查问题或实时支持，例如关闭服务器，则 AWS 客户还可以用 AWS 打开支持案例。订阅 AWS 企业级支持计划的客户还享有更大的灵活性，可在需要快速响应时立即与支持工程师交流，并提交关键严重性案例。

事件完成后，AWS 资源将根据流量水平自动缩减，或根据事件的要求继续扩展。

设计原则

计划事件的准备始于开始实施基于云的应用程序堆栈或工作负载时的良好设计。

离散工作负载

不论是正常工作负载还是高流量工作负载，优秀的设计都是有效管理计划事件的必备条件。请务必从一开始就以具体的业务应用程序或产品为中心，采用离散、功能独立的资源分组设计。本节介绍实现这一设计目标的多个维度。

标签

标签用于标识和组织资源，它们是计划基础设施事件中基础设施资源管理的关键组件之一。在 AWS 中，标签属于客户管理的键值标签，用于负载均衡器或 Amazon Elastic Compute Cloud (EC2) 实例等单独托管的资源。通过引用已附加到 AWS 资源的定义明确的标签，可以轻松地确定计划事件工作负载将由整体基础设施中的哪些资源组成，然后可以用此信息进行准备状况分析。标签还可用于成本分配目的。

例如标签可用于组织 EC2 实例、Amazon Machine Image (AMI) 映像、负载均衡器、安全组、Amazon Relational Database Service (RDS) 资源、Amazon Virtual Private Cloud (VPC) 资源、Amazon Route 53 运行状况检查以及 Amazon Simple Storage Service (S3) 存储桶。

有关有效标签策略的更多信息，请参阅 AWS Tagging Strategies。[1]

有关如何创建和管理标签，以及如何将标签放入资源组的示例，请参阅 Resource Groups and Tagging for AWS。[2]

松散耦合

在设计基于云的架构时，应用程序堆栈中每个组件在设计上应尽可能相互独立，从而让基于云的工作负载拥有弹性和可扩展性方面的优势。

若每个组件都采用黑盒设计并配备定义明确的输入输出接口 (例如 RESTful API)，则可以减弱基于云的应用程序堆栈组件之间的相互依赖关系。如果组件不是应用程序，而是共同构成应用程序的服务，这就是所谓的*微服务架构*。对于应用程序组件之间的通信和协调，可以使用事件驱动通知机制来实现组件之间的消息传递，例如 AWS 消息队列，如图 1 所示。

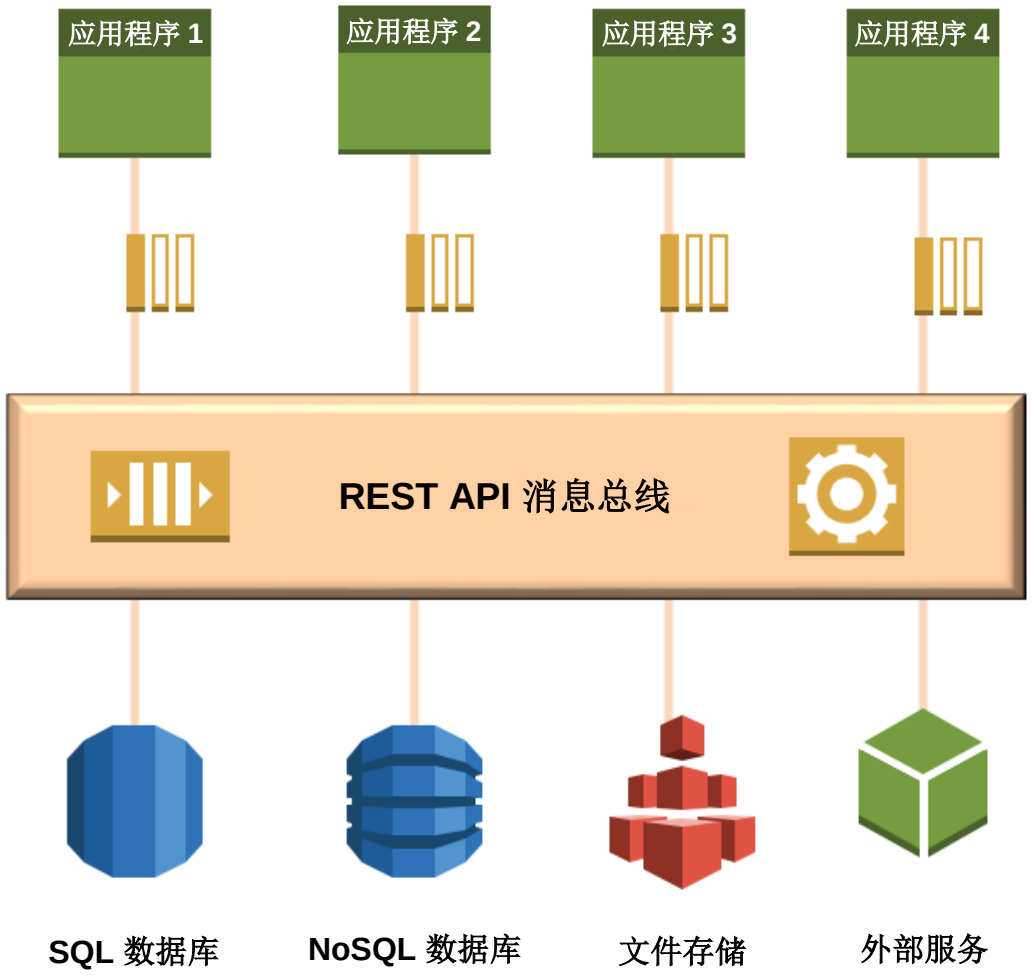

图 1.使用 RESTful 接口和消息队列的松散耦合

使用这种机制，某个组件中的变化或故障传递到其他组件的几率会大大降低。例如，如果多级应用程序堆栈中的某个服务器失去响应，与之松散耦合的应用程序可以设计为绕过未响应的层级，或切换到降级模式的替代事务。

松散耦合的应用程序组件如果采用中间消息队列，将更容易支持异步集成设计。应用程序的组件不使用点对点直接通信，而是使用持久性的中间消息层 (例如 Amazon Simple Queue Service (SQS) 队列或 Amazon Kinesis Streams 等流数据机制)，可以在下游组件处理传入队列的同时，承受一个组件中活动的突然增加。或者在某个组件发生故障时，消息将一直保留在队列或流中，直到发生故障的组件可以恢复时为止。

有关 AWS 所提供消息队列和通知服务的更多信息，请参阅 Amazon Simple Queue Service。[3]

服务，而不是服务器

使用托管服务和服务终端节点，您不再需要担心安全或访问权限、备份或还原、补丁管理或更改控制、监控或报告的设置，也无需关注许多传统的系统管理细节。这些云资源可使用多可用区 (在某些情况下还可采用多区域) 配置进行预配置，保证高可用性和弹性。它们可以向上扩展或向下收缩，通常无需停机，并且还可以通过 AWS 管理控制台或 API/CLI 调用即时进行配置。

托管服务和服务终端节点可以用于为客户应用程序堆栈赋予额外的功能，例如关系数据库和 NoSQL 数据库系统、数据仓库、事件通知、对象和文件存储、实时流媒体、大数据分析、机器学习、搜索、转码以及多种其他工作负载。终端节点是面向 AWS 服务入口点的 URL。例如，https://dynamodb.us-west-2.amazonaws.com 是 Amazon DynamoDB 服务的入口点。

使用托管服务及其服务终端节点，可以将生产就绪资源作为设计解决方案的一部分，用于处理计划基础设施事件期间增加的容量、覆盖面和事务速率，对于执行的功能与托管服务相同的服务器，无需预配置和管理这些服务器。

有关 AWS 服务终端节点的更多信息，请参阅 AWS Regions and Endpoints。[4]有关拥有终端节点的托管服务示例，另请参阅 Amazon EMR、[5]Amazon RDS[6] 和 Amazon ECS[7]。

无服务器架构

在计划基础设施事件期间需要响应动态变化的处理负载，使用 AWS Lambda 就是有效解决这一需求的另一个策略。Lambda 是一个事件驱动型的无服务器计算平台，它是一种动态调用服务，运行 Python、Node.js 或 Java 代码通过通知来响应事件，并自动管理该代码所指定的计算资源。Lambda 不需要预配置 Amazon EC2 计算资源。可以配置 Amazon Simple Notification Service (Amazon SNS) 触发 Lambda 函数。有关 Amazon SNS 的详细信息，请参阅 Amazon Push Notification Service。[8]

Lambda 无服务器函数可以根据外部事件或内部系统负载指标，执行访问或调用其他 AWS 服务的代码，例如数据库操作、数据转换、对象或文件检索，甚至扩展操作。AWS Lambda 还可以生成新的通知或自己的事件，甚至还能启动其他 Lambda 函数。

AWS Lambda 可以在计划基础设施事件期间精确控制扩展操作。例如，Lambda 可以用于扩展 Auto Scaling 操作的功能，从而执行多种操作，例如通知也需要扩展的第三方系统，或为预配置的新实例添加额外的网络接口。有关如何使用

Lambda 来自定义扩展操作的示例，请参阅 Using AWS Lambda with Auto Scaling Lifecycle Hooks[9]。

有关 AWS Lambda 的更多信息，请参阅 What is AWS Lambda?[10]

自动化

Auto Scaling

Auto Scaling 是基础设施事件计划的一个关键组件。根据预定义的条件自动扩展应用程序的容量，有助于在计划基础设施事件期间发生流量模式和容量波动时保证应用程序的可用性。

AWS 的 Auto Scaling 功能支持它的多种资源，包括 EC2 实例、数据库容量、容器等。

Auto Scaling 可以用于扩展实例组，例如组成某个基于云的应用程序的服务器组，从而让它们可以根据规定的条件自动扩展。Auto Scaling 甚至还可用于在实例运行状况不佳时保持固定数量的实例。这种自动扩展和保持实例数量的功能，正是 Auto Scaling 服务的核心功能。

Auto Scaling 通过对组中的实例定期进行运行状况检查，保持您指定数量的实例。如果某个实例运行状况不佳，则该组会终止该实例，并启动其他实例代替。

使用 Auto Scaling 策略，可以根据条件的变化自动增加或减少服务器组中运行的 EC2 实例数量。扩展策略生效期间，Auto Scaling 组将会动态调整组的所需容量，或者在知道可预测的潮汐流量时按计划调整容量，并根据需要启动或终止实例。

重启和恢复

对于任何计划基础设施事件，一个重要的设计元素就是建立处理受损实例或服务器以及远程恢复或重新启动它们的过程和自动化功能。

EC2 实例可以设置为在对底层硬件的系统状态检查失败时自动恢复。这时实例将会重启 (必要时使用新硬件)，但将保留其实例 ID、IP 地址、弹性 IP 地址、Amazon Elastic Block Store (EBS) 卷附件和其他详细配置信息。有关 EC2 实例自动恢复的更多信息，请参阅 Auto Recovery of Amazon EC2。[11]

配置管理/编排

对于一个稳健、可靠、响应及时的计划基础设施事件策略而言，用于单个资源状态管理和应用程序堆栈部署的配置管理和编排工具是不可或缺的。

配置管理工具通常用于处理服务器实例、负载均衡器、Auto Scaling、单个应用程序部署和应用程序运行状况监控的预配置和配置。它们还可与数据库、存储卷和缓存层等其他服务集成。

编排工具比配置管理工具高一个抽象层，提供指定不同资源间关系的工具，从而让客户可以将多个资源作为统一的云应用程序基础设施来预配置和管理，无需担心资源之间的依赖关系。

这些工具以代码的形式定义并描述了各个资源及其相互关系，该代码支持版本控制，从而能够根据测试和开发目的退回到以前的版本，或者尝试新的代码分支。此外还可以定义针对基础设施事件优化的编排和配置，然后在事件结束后退回到标准配置。

Amazon Web Services 建议使用下列工具来实现硬件即代码部署和编排：

- **AWS Config with Config Rules** 或 AWS Config Partner，提供详细、可视、可搜索的 AWS 资源、配置历史和资源配置合规功能。
- **AWS CloudFormation** 或第三方 AWS 资源编排工具，用于管理 AWS 资源预配置、更新和终止。
- **AWS OpsWorks、Elastic Beanstalk**、或第三方服务器配置管理工具，用于管理操作系统 (OS) 和应用程序配置修改。

有关硬件即代码管理的方式的更多详细信息，请参阅 Infrastructure Configuration Management。[12]

多样性/弹性

消除单点故障和瓶颈

在计划基础设施事件时，您需对应用程序堆栈进行何单点故障 (SPOF) 或性能瓶颈分析。是否有任何单个服务器、数据卷、数据库、NAT 网关或负载均衡器实例的故障会导致整个应用程序，或其中的重要部分停止工作的情况？

其次，由于基于云的应用程序的流量或事务量会不断扩展，那么随着数据量沿着数据流路径增长，是否有基础设施的任何部分会面临物理限制或制约，例如网络带宽或 CPU 处理周期？

一旦发现这些风险，可以通过多种方式缓释。

面向故障的设计

如前文所述，对于单个资源故障或者流量或事务量波动，配备 RESTful 接口的松散耦合和消息队列是实现弹性的一个好策略。弹性设计的另一个维度是将应用程序组建尽可能配置为无状态。

无状态应用程序无需了解以前的事务，其他应用程序组件之间也只存在松散的依赖关系。它们不存储会话信息。无状态应用程序可以作为池或群集的成员水平扩展，因为任何请求都可由池中的任何实例或群集处理。您可以使用 Auto Scaling 和运行状况检查条件，直接根据需要添加更多资源，以编程方式处理计算、容量和吞吐量需求的波动。使用 Lambda 函数代替 EC2 实例，潜在可以将采用无状态设计的应用程序重构到无服务器架构中。Lambda 函数还拥有内置的动态扩展功能。

如果 Web 服务器等应用程序资源无法避免拥有有关事务的状态数据，则应从应用程序的设计上考虑将有状态的应用程序部分与服务器本身分离。例如，HTTP Cookie 或等效状态数据可能存储在数据库中，例如 DynamoDB 或 S3 存储桶或 EBS 卷中。

对于需要跟踪工作流程中每个步骤的当前状态的复杂多步骤工作流程，可以使用 Amazon Simple Workflow Service (SWF) 集中存储执行历史，从而让这些工作负载变得无状态。

另一个弹性措施是使用分布式处理。如果需要及时处理大量数据，而单个计算资源又无法满足需要，则可以从工作负载设计上将任务和数据分为较小的分段，从而使用计算资源群集并行执行。分布式处理是无状态的，因为正在处理分区数据和任务的独立节点可能发生故障。这时，分布式处理调度引擎将自动在分布式处理群集的另一个节点自动重启失败的任务。

AWS 提供多种分布式数据处理引擎，例如 Amazon EMR、Amazon Athena 和 Amazon Machine Learning，每种引擎均属于托管服务，提供终端节点，让您不必再受安装补丁、维护、扩展、故障转移等复杂任务的困扰。

对于流数据的实时处理，的 Amazon Kinesis Streams 可以将数据分为多个分片，供该数据的多个用户使用，例如 Lambda 函数或 EC2 实例。

有关这些类型工作负载的更多信息，请参阅 Big Data Analytics Options on AWS。[13]

多可用区和多区域

AWS 服务在全球多个位置托管，这些位置由区域和可用区组成。区域是指一个独立的地理区域，每个区域都有多个相互隔离的位置，被称为可用区。AWS 允许客户将资源 (例如实例和数据) 放在多个位置。

设计应用程序时，您应当确保它们跨多个可用区和区域分配。除跨可用区和区域分配和复制资源外，您还应使用负载均衡和故障转移机制进行应用程序设计，从而让应用程序堆栈可以在发生故障时自动重定数据流和流量的路由，指向这些替代位置。

负载均衡

使用 Elastic Load Balancing 服务 (ELB)，可以将一组应用程序服务器附加到负载均衡器上，同时仍可以跨多个可用区分配。当特定可用区中的 EC2 实例的运行状况检查，负载均衡器后面的失败一样，负载均衡器会停止将流量发送到这些节点。与 Auto Scaling 配合使用时，运行正常的节点数将与其他可用区自动重新平衡，无需手动干预。

此外，也可以使用 Amazon Route 53 DNS 和基于延迟的路由算法跨区域执行负载均衡。更多信息请参阅 Latency Based Routing。[14]

负载分离策略

在基于云的基础设施中，*负载分离*概念包含将流量重定向或代理至其他位置，从而缓释主系统的压力。在某些情况下，负载分离策略可以采用分流法，选择删除某些流量或减少应用程序的功能以解放处理负载，并能够至少服务一部分传入的请求。

负载分离可以使用多种方法，例如一种方法是基于延迟的 DNS 路由选择，另一种方法是使用缓存。使用 Amazon ElastiCache 等内存缓存层，可以让缓存尽量靠近应用程序。此外也可以使用 Amazon CloudFront 等全球内容分配网络，让缓存层更接近用户侧。

有关 ElastiCache 和 CloudFront 的更多信息，请参阅 Getting Started with ElastiCache[15] 和 Amazon CloudFront CDN。[16]

成本优化

预留实例、竞价型实例与按需实例比较

能否根据系统指标以及其他性能和运行状况检查标准，在云端动态地预配置资源，与能否控制在云端预配置资源的成本紧密相关。使用 Auto Scaling，资源利用率可以与实际处理和存储需求紧密匹配，从而最大程度减少无谓的费用和利用不足的资源。

云端成本控制的另一个维度，是能够选择按需实例、预留实例 (RI) 或竞价型实例。此外还有一个针对 DynamoDB 的预留容量功能。

使用按需实例，您只需为使用的 EC2 实例付费。使用按需实例，您只需要按小时支付计算容量费用，无需长期承诺。

Amazon EC2 预留实例提供了与按需实例定价相比更多的折扣 (最高可达 75%)，并且在特定可用区使用时可以提供容量预留。除可用区预留和价格折扣外，预留实例和按需实例在功能上并无差异。

使用竞价型实例，您可以竞价购买空闲的 Amazon EC2 计算容量。竞价型实例的价格低于按需实例，大幅降低了基于云的应用程序的运行成本。

对于面向云的设计，某些使用案例更适合于使用竞价型实例。例如，由于竞价型实例可以随时在要价超出您的出价时停用，您应考虑仅将竞价型实例用于相对无状态和横向扩展的应用程序堆栈。对于有状态的应用程序或昂贵的处理负载，按需实例、预留实例可能会更好。对于无需考虑容量限制的任务关键型应用程序，预留实例是您的最佳选择。

更多详细信息请参阅 Reserved Instances[17] 和 Spot Instances[18]。

事件管理流程

基础设施事件计划是涉及应用程序开发人员、管理员和商业利益相关者的集体活动。在基础设施事件开始前的几周，您就应举行多轮经常性的会议，由拥有和操作 Web 服务的各项关键基础设施组件的关键技术人员参加。

基础设施事件计划

基础设施事件应在事件日期的数周以前就开始计划。计划事件生命周期的典型时间线如图 2 所示。

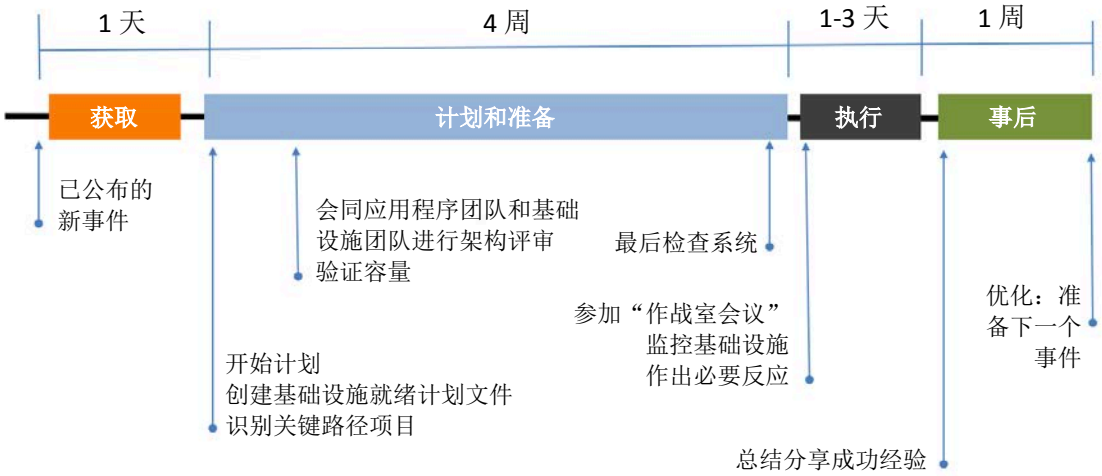

图 2.典型基础设施事件的时间线

计划和准备

计划

在基础设施事件前数周，我们建议执行如下活动计划：

第 1 周：

- 提名成立一个团队来推动基础设施事件的计划和设计。
- 举行利益相关者会议，了解事件的参数 (规模、持续周期、时间、地理覆盖范围、受影响的工作负载) 和成功标准。
- 让任何上下游合作伙伴和供应商参与。

第 2-3 周：

- 进行架构评审，并根据需要进行调整。
- 执行运行评审；根据需要进行调整。
- 遵循本白皮书以及脚注引用中介绍的最佳实践。
- 识别风险并制定缓释计划。
- 制定计划事件操作手册。

第 4 周：

- 根据预期负载检查需要扩展的所有云供应商服务。
- 检查服务限制并在需要时提高限制。
- 按预定义的阈值设置监控控制面板和警报。

架构评审

基础设施事件准备的一个基本内容是对将发生流量暴增的应用程序堆栈进行架构评审。评审的目的是验证和识别应用程序可扩展性和可靠性方面的潜在风险领域，以及识别在事件开始前优化的机会。

AWS 以五个设计支柱为中心，为其企业级支持计划的客户提供了一个检查客户应用程序堆栈的框架。这五个支柱分别是安全性、可靠性、性能效率、成本优化和卓越运行，详见下文所述。

表 1: 架构良好的应用程序的支柱

支柱名	支柱定义	利益相关领域
安全性	通过风险评估和缓释策略，在实现商业价值的同时保护信息、系统和资产的能力。	身份管理、加密、监控、日志记录、密钥管理、专用实例、合规性、监管
可靠性	发生基础设施或服务故障时恢复系统，动态获得计算资源以满足需求，以及减少因错误配置或瞬时网络问题等导致的中断的能力。	服务限制、多可用区和区域、可扩展性、运行状况检查/监控、备份/灾难恢复、联网、自我修复自动化
性能效率	高效地使用计算资源以满足系统要求，以及随着需求变化和技术发展保持这一效率的能力。	相关 AWS 服务、资源利用率、存储架构、缓存、延迟要求
成本优化	避免或消除不必要的成本或欠佳资源的能力。	竞价/预留实例、环境调整、服务选择、容量调整、账户管理、整合账单、资源退役
卓越运行	运行和监控系统以实现商业价值，同时不断改进支持流程和过程的能力。	操作手册、练习手册、CI/CD、游戏日、基础设施即代码、RCA

本白皮书附录中的架构评审项目的详细清单，可用于审核基于 AWS 的应用程序堆栈。

运行评审

架构评审更多侧重于应用程序的设计组件，此外您还需审核您的云运行和管理实践，以评估您在管理云工作负载方面的表现。评审的目的是确定运行差距和问题，并在事件开始前采取措施以减少差距和问题。

AWS 为其企业级支持计划的客户提供了云运行评审工具，对基础设施事件的准备极有价值。评审的重点是评估下列领域：

- 准备 – 您必须拥有正确的组织结构、流程和技术。您应明确定义负责管理应用程序堆栈的员工的职能和职责。流程应事先定义以切合事件的需要。过程应尽可能自动化。
- 监控 – 有效监控衡量应用程序的性能。监控对于发现异常，防止其成为问题至关重要，同时还提供了减轻负面事件影响的机会。
- 操作 – 操作活动需要尽可能利用自动化手段，及时、可靠地执行，同时处理需要升级的意外运行事件。
- 优化 – 使用收集的指标、运行趋势和经验教训进行事后分析，以获取并报告可在未来事件中改进的机会。优化加准备就构成了一个解决运行问题并防止其再次发生的反馈循环。

了解 AWS 服务限制

在计划基础设施事件期间，避免在扩展应用程序或工作负载时超过云服务提供商规定的任何服务限制至关重要。

云服务提供商通常对您可以使用的资源都规定了限制。这些限制一般按账户和区域进行规定，受影响的资源包括实例、卷、流、无服务器调用、快照、VPC 数量、安全规则等。限制的目的是作为一种安全措施，防止企图滥用资源的失控代码或恶意参与者，以及作为一种控制措施，帮助减少计费风险。

随着您在云中的足迹不断扩大，某些限制会逐渐自动上调，但大多数服务都要求您提出支持请求以上调限制。某些服务限制可以通过支持案例上调，但其他服务的限制却无法修改。

AWS 为企业级支持计划和商业支持计划的客户提供了 Trusted Advisor，它提供了一个限制检查控制面板，方便客户主动管理所有的服务限制。

有关各种 AWS 服务限制的更多信息以及如何检查这些限制，请参阅 AWS Service Limits[19] 和 Trusted Advisor。[20]

了解模式

基线

您应在基础设施事件开始前记录关键指标的“正常”值。这有助于您确定在事件完成/结束后，应用程序/服务何时安全地回到正常水平。例如，确定通过负载均

衡器的正常事务速率为每秒 2500 个请求，将有助于在事件后确定何时可以安全地开始缩减过程。

数据流和依赖关系

了解数据如何流过应用程序的各个组件，有利于您识别潜在的瓶颈和依赖关系。数据流中属于数据用户的应用程序层级或组件大小是否正确，并设置为在属于数据生产者的应用程序堆栈层级或组件向上扩展时自动正确扩展？在组件发生故障时，数据是否可以被放入队列，直到该组件恢复时为止？是否有任何下游或上游数据提供者或用户可根据您的事件扩展？

相称性

在评审基础设施事件的准备情况时，另一个考虑因素是应用程序堆栈各组件要求的扩展是否相称。这种相称并不总是一对一的。例如，每秒经过某个负载均衡器事务数量增加了 10 倍，可能需要存储容量、流式处理分片的数量或数据库的数量增加 20 倍才能完成读写操作，因为处理可能会在前端应用程序中进行。

沟通方案

事件开始前，您需要制定沟通方案。收集内部利益相关方和支持群体的名单，识别在各种情形下在事件的各个阶段需要联系哪些人，例如事件开始时、事件进行期间、事件结束时、事后分析、紧急联系人、故障诊断联系人等。

需要联系的人员和群体可能包括以下方面：

- 利益相关方
- 操作经理
- 开发人员
- 支持团队
- 云服务提供商团队
- 网络运营中心 (NOC) 团队

在收集内部联系人名单时，您还应制定一个参与应用程序持续实时交付的外部利益相关者联系名单。这些利益相关者包括支持堆栈关键组件的合作伙伴和供应商，提供外部服务、数据源、身份验证服务等的上下游供应商。

此外部联系人名单中还应包括以下方面：

- 基础设施托管供应商

- 电信供应商
- 实时数据流合作伙伴
- 公关营销联系人
- 广告合作伙伴
- 参与服务设计的技术顾问

要求各提供商提供以下信息：

- 事件时间期间的实时联系点
- 关键支持联系人和升级流程
- 姓名、电话号码和电子邮件地址
- 验证可以联系到实时技术联系人

对订阅企业级支持计划的 AWS 客户还将安排技术客户经理 (TAM)，负责协调并验证安排的专职 AWS 支持人员了解事件并已做好了支持准备。TAM 还会在事件期间召集和参加“作战室”会议，并在必要时推动支持升级。

NOC 准备

事件开始前，您应指示运营和/或开发团队创建实时指标控制面板，以在事件发生时监控生产环境中的各项 Web 服务关键组件。理想情况下，控制面板应自动显示最新指标，每分钟或按照在事件期间适当、有效的其他间隔刷新。

考虑监控以下组件：

- 各服务器的资源利用率 (CPU、磁盘和内存利用率)
- Web 服务响应时间
- Web 流量指标 (用户数、页面访问量、会话数)
- 各访客区域的 Web 流量 (全球客户划分)
- 数据库服务器的利用率
- 营销流转化漏斗，如转化率和流失率
- 应用程序错误日志
- Canary 监控

在 Amazon CloudWatch 中，使用 CloudWatch 自定义控制面板，可以将大部分指标从 AWS 资源收集到一个窗格中。此外，如果 AWS 尚未提供自动提供招标的功能，通过 CloudWatch 还可以将自定义指标导入 CloudWatch。有关 AWS 监控工具和功能的更多详细信息，请参阅本白皮书的“监控”部分。

操作手册准备

您应在准备基础设施事件时制定操作手册。*操作手册*汇集了操作人员在活动期间将执行的过程和操作。事件操作手册可以利用针对例行操作和异常处理的现有操作手册编制。通常，操作手册会包含启动、停止、监督和调试系统的过程，此外还应说明处理意外事件和紧急情况的过程。

操作手册应包括以下部分：

- **事件详细信息**：简要描述事件、成功标准、媒体报道、事件日期以及客户方面和 AWS 方面主要利益相关者的联系详细信息。
- **AWS 服务列表**: 枚举事件期间将使用的所有 AWS 服务，以及这些服务的预期负载、受影响的区域和账户 ID。
- **架构和应用程序评审**：记录负载测试的结果、基础设施和应用程序设计中的任何压力点、针对工作负载的弹性措施、单点故障和潜在的瓶颈。
- **运行评审**：突出监控设置、运行状况标准、通知机制和服务恢复过程。
- **准备清单**：包括服务限制检查，负载均衡器等应用程序堆栈组件预热，流式处理分片、DynamoDB 分区、S3 分区等资源的预配置等考虑因素。更多信息请参阅本白皮书附录中的“架构评审详细清单”。

监控

监控计划

数据库、应用程序和操作系统监控对于确保事件的成功至关重要。应建立全面综合的监控系统，以在基础设施事件期间有效地发现并立即处理严重事故。总体来说，有效的监控策略可以根据对业务的重要性，确保相应水平的应用程序具备相应的监控工具。有效的事故管理策略将会包含 AWS 和客户的监控数据，并具备自己的事件和事故管理工具与流程。实施从 AWS 解决方案的所有区段集中收集监控数据的监控计划，非常有利于调试复杂的故障。

监控计划应解决以下问题：

- 必须为事件设置哪些监控工具和控制面板？

- 监控目标和允许的阈值是什么？哪些事件会触发操作？
- 将会监控哪些资源以及来自这些资源的指标，以及必须按什么频率进行轮询？
- 谁负责执行监控任务？具备哪些监控警报？将会提醒谁？
- 对于常见的和预期的故障，制定了哪些补救计划？如何处理意外事件？
- 发生任何故障时的升级流程是什么？

该策略可以使用以下 AWS 监控工具：

- **Amazon CloudWatch**：开箱即用的解决方案，提供 AWS 控制面板、指标、监控、警报和自动预配置功能。
- **Amazon CloudWatch 自定义指标**：用于操作系统、应用程序和业务指标的收集。Amazon CloudWatch API 允许收集几乎任何类型的自定义指标。
- **Amazon EC2 实例运行状况**：用于查看实例的状态检查以及根据其状态调度事件，例如自动重起或重启实例。
- **Amazon SNS**：用于设置、操作和发送事件驱动型通知。
- **AWS X-Ray**：通过跨系统组件分析数据流，帮助调试和分析分布式应用程序和微服务架构。
- **Amazon Elasticsearch Service**：用于集中收集日志和实时日志分析。适用于快速、启发式发现问题。
- **第三方工具**：用于实时分析，实现全面的堆栈监控和可见性。
- **标准操作系统监控工具**：用于操作系统级监控。

有关 AWS 监控工具的更多详细信息，请参阅 Automated and Manual Monitoring。[21]另请参阅 Using Amazon CloudWatch Dashboards[22] 和 Publishing Custom Metrics。[23]

通知

配置与监控解决方案集成的警报和通知，是基础设施事件设计的一个关键操作要素。这些警报和通知可以配合 AWS Lambda 等服务使用，根据警报触发操作。自动响应运行事件，是以最好的响应速度执行迁移、回退和恢复操作的一个关键要素。

此外还可以使用工具集中监控工作负载，根据可用日志以及与关键运行指标有关的指标创建相关警报和通知。这包括超限异常以及服务或组件故障的警报和通知。理想情况下，每当触及性能阈值下限或发生故障时，系统的机构设计应能根据通知和警报自动自我修复或扩展。

如前文所述，AWS 的服务 (Amazon SQS 和 Amazon SNS) 可以确保针对意外运行事件发出相应的警报和通知，并支持自动响应。

操作准备 (事件当天)

计划的执行

事件当天，参与基础设施事件的核心团队应通过现场电话会议监控实时控制面板。应当已经制定并发放操作手册。确保沟通方案已经明确，并且所有支持人员和利益相关者都了解沟通方案，并且已经建立应急预案。

“作战室”会议

事件期间将提供实时会议网桥，由下列人员参加：

- 主要负责的应用程序和运营团队
- 运营团队领导
- 直接参与技术交付的外部合作伙伴技术人员
- 业务利益相关者

在事件的大部分时间，这一会议网桥上的交谈应当极少。如果出现负面运行事件，可以处理事件的关键人员将已经接入网桥，随时可以行动和提供意见。

向领导层报告

事件期间，每小时向关键相关领导发送电子邮件。邮件应包括以下内容：

- 状态摘要：绿色 (正常)，黄色 (遇到问题)，红色 (重大问题)
- 关键指标更新
- 遇到的问题，补救计划的状态，补救的 ETA
- “作战室”会议网桥的电话号码 (如有任何人希望参加)

事件结束时，应按类似的格式发送最终摘要电子邮件。

应急预案

事件准备过程中的每个步骤都应有相应的回退计划，并已在测试环境中验证。

制定回退计划时请考虑以下问题：

- 事件期间可能发生的最坏情况是什么？
- 哪些类型的事件会导致负面的公共关系影响？
- 哪些第三方组件和服务可能在事件期间发生故障？
- 应当监控哪些将会表明发生不利情况的指标？
- 各种可能情形的回退计划是什么？
- 每个回退过程将需要多长时间？可接受的恢复点目标 (RPO) 和恢复时间目标 (RTO) 是多少？(有关这些概念的更多信息，请参阅 Using AWS for Disaster Recovery[24]。)

请考虑以下类型的回退：

- **蓝/绿色部署：**如果推出新的生产应用程序或环境，请在线保持年以前的生产版本，以便快速退回。
- **热试用：**在另一个区域中启动可以在需要时快速向上扩展的最低环境。如果主区域发生故障，快速扩展备份区域并将流量切换到第二区域。
- **维护模式错误页面：**检查每一层 Web 服务的错误页面设施和触发条件。做好准备，以在需要时为这些错误页面注入更多具体的消息。

针对每种可能的故障情形，对每种回退计划进行测试并相应记录。

事后活动

事后 分析

事后分析往往会被忽视，因为客户通常会急于返回正常运行。但我们建议您要求将事后分析作为任何基础设施事件管理生命周期的一部分。事后分析可让您与参与的各个团队合作，识别可能需要进一步优化的领域，例如操作过程、实施细则、故障转移和恢复过程等。如果应用程序堆栈在事件期间发生中断，这将尤其重要。对事件进行事后分析，也有利于在需要编制根本原因分析 (RCA) 文档时提供记录。

收尾流程

基础设施事件结束后，应当立即启动收尾流程。在此期间，建议您继续监控相关应用程序和服务，以确保流量已恢复到正常生产水平。使用在准备阶段创建的任何运行状况控制面板，验证流量和事务速率的正常化。某些事件的收尾期可能是线性、简单的，而对于其他事件，容量的下降可能不均匀，或者更加缓慢的。有些流量模式可能会继续存在。例如，流量暴增后的恢复通常要求简单直接的收尾过程，而向新的地理区域部署或扩展应用程序会有长期性的影响，要求您仔细监控新的流量模式并将额外的监控作为永久应用程序堆栈的一部分。

事件完成后的某个时候，您必须确定何时可以安全地结束事件管理操作。参考以前记录的关键指标的“正常”值，可帮助确定何时宣布事件完成或结束。我们建议将收尾工作分为两个部分，每个部分可以有不同的时间线。首先关注事件操作管理的第一个部分，例如向内外部利益相关者和合作伙伴发送沟通，以及重置服务限制。然后重点处理第二个部分，即收尾工作的技术方面，例如缩减容量的过程、验证环境运行状况和确定是否应当还原或提交架构更改的标准等。

与每个部分关联的时间线取决于事件的性质、关键指标以及客户满意度。下表列举了与各个部分有关的一些常见任务，以帮助您确定结束事件管理的适当时间。

表 2: 操作收尾任务

任务	描述
沟通	通知内外部利益相关者事件已结束。结束沟通的时间应与事件完成的定义保持一致。使用“正常”指标来确定结束沟通的适当时间。另外您也可以分级结束沟通。例如，您可以结束“作战室”网桥，但保留事件升级过程，以防出现事件后的故障。
服务限制/成本控制	虽然您可能禁不住想在事件后保留上调后的服务限制，但请注意服务限制也是一种安全保障。服务限制可防止因账户泄密或错误配置的自动化而超额使用服务，从而保护您和您的成本。
报告和分析	应当制定并向沟通方案中识别的所有内部方面分发数据收集和事件指标的排序规则，附带说明模式、趋势、问题领域、成功过程、临时过程、实践时间线以及是否满足成功标准的分析说明。此外还应制定详细的成本分析，说明支持事件的运行费用。
优化任务	随着时间的推移，企业的运营不断完善，组织也在持续演变。运行优化要求持续收集指标、运行趋势以及事件的经验教训，从而发现改进机会。优化与准备联系起来，就构成了一个解决运行问题并防止其再次发生的反馈循环。

表 3: 技术收尾任务

任务	描述
服务限制/成本控制	虽然您可能禁不住想在事件后保留上调后的服务限制，但请注意服务限制也是一种安全保障。服务限制可防止因源自账户泄密的恶意活动或错误配置的自动化而超额使用服务，从而保护您的运行和运行成本。
收缩过程	释放在准备阶段向上扩展的资源。这些项目是您的架构所独有的，但下列例子较为常见： EC2/RDS 实例大小 Auto Scaling 配置 预留容量 预配置 IOPS
环境运行状况的验证	比较基线指标并检查生产运行状况，以验证在事件后以及缩减过程完成后，受影响的系统行为正常。
架构更改的处置	根据事件性质以及对运行指标的观察，事件准备阶段作出的某些更改可能值得保留。例如，向新地理区域的扩张可能需要永久增加该区域的资源，或者上调某些服务限制或配置参数，例如数据库中的分区数量或者卷中 PIOPS 流的分片数量，可能属于应当保留的性能优化指标。

优化

也许，基础设施事件管理的最重要内容可能是事后分析以及识别观察到的运行和架构挑战及改进机会。基础设施事件很少是一次性事件，它们可能是季节性的或结合新的应用程序版本，也可能是公司向新市场和新区域扩张时，公司增长的一部分。因此，每个基础设施事件都是观察、改进和下一个事件作出更有效准备的机会。

结论

AWS 以模块化的方式提供富有弹性、可编程的产品和服务，从而让客户自由搭建以支持几乎任何规模的工作负载。借助 AWS 基础设施事件操作指南和最佳实践，再加上我们全面的高可用性服务，您的公司可以设计和准备重大的业务事件，确保无缝动态满足持续变化的需求，实现快速响应和全球覆盖。

鸣谢

以下个人和组织对本文档的编撰做出了宝贵贡献：

- Presley Acuna，AWS 企业级支持计划经理
- Kurt Gray，AWS 全球解决方案架构师

- Michael Bozek AWS 高级技术客户经理
- Rovan Omar，AWS 技术客户经理
- Will Badr，AWS 技术客户经理
- Eric Blankenship，AWS 高级技术客户经理
- Greg Bur，AWS 技术客户经理
- Bill Hesse，AWS 高级技术客户经理
- Hasan Khan，AWS 高级技术客户经理
- Varun Bakshi，AWS 高级技术客户经理

延伸阅读

有关运行与架构最佳实践的延伸阅读信息，请参阅 Operational Checklists for AWS。[25]我们建议读者参阅 AWS Well Architected Framework [26]，了解评估他们基于云的应用程序交付堆栈的结构化方法。对于希望 AWS 技术客户经理和支持工程师更多直接参与事件设计、计划和事件当日操作的客户，可以选择 AWS 的基础设施事件管理 (IEM) 技术支持服务。有关 AWS IEM 技术支持服务的更多详细信息，请参阅 Infrastructure Event Management。[27]

附录

架构评审详细清单

是-否-不适用	安全性
□-□-□	根据 AWS 安全最佳实践，至少每 3 个月轮换 AWS Identity and Access Management (IAM) 访问密钥和用户密码以及应用程序所涉及资源的凭证。对每个账户执行密码政策，使用硬件或虚拟 多因子认证 (MFA) 设备。
□-□-□	建立了内部安全流程和控制措施，坚持唯一、基于角色和最低特权的原则，控制使用 IAM 对 AWS API 的访问权限。
□-□-□	删除了任何机密或敏感信息，包括嵌入式公共/私有实例密钥对，并审核了来自任何自定义 Amazon 系统映像 (AMI) 的所有 SSH 授权密钥文件。
□-□-□	为方便起见，对 EC2 实例使用 IAM 角色，而不是将任何凭证嵌入在 AMI 内部。
□-□-□	通过创建 IAM 管理员角色并限制其他功能角色的 IAM 操作，将 IAM 管理员特权与普通用户特权分开。

是-否-不适用	安全性
□-□-□	在 EC2 实例上应用有关 Windows 实例或 Linux 实例的最新安全补丁。我们使用操作系统访问控制，包括 Amazon EC2 安全组规则、VPC 网络访问控制列表、操作系统强化、基于主机的防火墙、入侵检测/防护、监控软件配置和主机清单。
□-□-□	确保往来于组织 AWS 和企业环境的网络连接使用加密协议传输。
□-□-□	执行集中日志和审核管理解决方案，识别和分析对环境的任何异常访问模式或任何恶意攻击。
□-□-□	建立了安全事件和事故管理、关联和报告流程。
□-□-□	确保没有任何安全组拥有对 AWS 资源的无限制访问权限。
□-□-□	我们对前端连接 (客户端到负载均衡器) 使用安全协议 (HTTPS 或 SSL)、最新安全政策和加密协议。客户端与负载均衡器之间的请求会进行加密，这样更加安全。
□-□-□	配置 Amazon Route 53 MX 资源记录集，从而拥有 TXT 资源记录集，包含相应的发件人策略框架 (SPF) 值，为我们的域指定被授权发送电子邮件的服务器。

是-否-不适用	可靠性
□-□-□	在 EC2 实例群上部署我们的应用程序，然后部署到 Auto Scaling 组中，确保根据预定义的扩展计划自动执行水平扩展。了解更多。
□-□-□	在 Auto Scaling 组配置中使用 Elastic Load Balancing 运行状况检查，确保 Auto Scaling 组根据底层 EC2 实例的运行状况作出反应。(仅在 Auto Scaling 组中使用负载均衡器时适用。)
□-□-□	跨多个可用区部署应用程序的关键组件，在可用区之间恰当复制数据。使用 Elastic Load Balancing、Amazon Route 53 或任何相关第三方工具，测试这些组件中的故障会如何影响应用程序的可用性。
□-□-□	在数据库层，在多个可用区部署 Amazon RDS 实例，通过同步复制到位于不同可用区的备用实例，从而增强数据库可用性。
□-□-□	制定了发生停机或性能下降时自动或手动故障转移的流程。
□-□-□	使用 CNAME 记录将 DNS 名称映射到我们的服务。不使用 A 记录。
□-□-□	为 Amazon Route 53 记录集配置了更低的生存时间 (TTL) 值，这样可避免在重新确定流量路由时，DNS 解析程序请求更新后的 DNS 记录发生延迟。(例如，DNS 故障转移检测并响应某个终端节点的故障时可能出现这种情况。)
□-□-□	至少配置两个 VPN 隧道以提供冗余，以防在 AWS 终端节点发生中断或进行计划维护。
□-□-□	使用 AWS Direct Connect，并始终配置两个 Direct Connect 连接，从而提供冗余，以防设备不可用。在不同的 Direct Connect 位置预配置连接，从而提供冗余，以防位置不可用。 此外还配置了到我们虚拟专用网关的连接，从而跨多个 Direct Connect 连接和位置配置了多个虚拟接口。
□-□-□	使用 Windows 实例，并且确保使用最新的半虚拟化驱动程序。半虚拟化驱动程序有助于优化驱动程序性能，减少运行时问题和安全风险。此外，我们还确保在 Windows 实例上

是-否-不适用	可靠性
	运行最新版本的 EC2Config 代理。
□-□-□	拍摄 Amazon Elastic Block Store (EBS) 快照卷，确保在发生任何故障时可实现时间点恢复。
□-□-□	为操作系统和应用程序/数据库数据 (视情况) 使用单独的 Amazon EBS 卷。
□-□-□	在任何 Linux 实例应用上最新的内核、软件和驱动程序补丁。

是-否-不适用	性能效率
□-□-□	在上线前对 AWS 托管的应用程序组件进行全面测试，包括性能测试。此外还进行负载测试，以确保 EC2 实例大小、IOPS 数量，RDS 数据库实例大小等的正确。
□-□-□	根据服务限制运行使用情况检查报告，确保当前对各 AWS 服务的使用量等于或低于服务限制的 80%。了解更多
□-□-□	为我们的应用程序 (Amazon CloudFront) 使用内容分发网络 (CDN) 以利用缓存，以及作为优化内容交付并自动将内容分配到距离用户最近的边缘站点的方式。
□-□-□	我们知道，Amazon CloudFront 收到的某些动态 HTTP 请求报头 (用户代理、日期等) 会降低缓存命中率并增加源上的负载，从而影响性能。了解更多
□-□-□	确保一个 EC2 实例的最大吞吐量大于所连接 EBS 卷的合计最大吞吐量。还使用有 PIOPS EBS 卷的 EBS 优化实例，以获得预期的卷性能。
□-□-□	确保解决方案的设计没有基础设施瓶颈，数据库或应用程序设计没有压力点。
□-□-□	使用 Amazon CloudWatch 或第三方合作伙伴工具，部署应用程序资源监控，配置基于任何性能超限的警报。
□-□-□	我们的设计避免在附加到我们应用程序实例的任何安全组使用大量规则。安全组中如有大量规则，可能会降低性能。

是-否-不适用	成本优化
□-□-□	我们知道，基础设施事件可能涉及某些过度预配置的容量，需要在事件结束后清理，以避免不必要的成本。
□-□-□	确保所有基础设施组件的大小合理，包括 EC2 实例大小、RDS 数据库实例大小、缓存群集节点大小和数量、Redshift 群集节点大小和数量以及 EBS 卷大小。
□-□-□	方便时我们使用竞价型实例。竞价型实例非常适合启动和结束时间灵活的工作负载。竞价型实例的典型使用案例：批量处理、报告生成和高性能计算工作负载。
□-□-□	建立最低可预测应用程序容量要求，并利用预留实例的优势。 . 预留实例可让您预留 Amazon EC2 计算容量，小时费率的折扣明显优于按需实例的价格。

Notes

[1] https://aws.amazon.com/answers/account-management/aws-tagging-strategies/

[2] https://aws.amazon.com/blogs/aws/resource-groups-and-tagging/

[3] https://aws.amazon.com/sqs/

[4] http://docs.aws.amazon.com/general/latest/gr/rande.html

[5] https://aws.amazon.com/emr/

[6] https://aws.amazon.com/rds/

[7] https://aws.amazon.com/ecs/

[8] https://aws.amazon.com/sns/

[9] https://aws.amazon.com/blogs/compute/using-aws-lambda-with-auto-scaling-lifecycle-hooks/

[10] http://docs.aws.amazon.com/lambda/latest/dg/welcome.html

[11] https://aws.amazon.com/blogs/aws/new-auto-recovery-for-amazon-ec2/

[12] https://aws.amazon.com/answers/configuration-management/aws-infrastructure-configuration-management/

[13] https://d0.awsstatic.com/whitepapers/Big_Data_Analytics_Options_on_AWS%20.pdf

[14] http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html#routing-policy-latency

[15] https://aws.amazon.com/elasticache/

[16] https://aws.amazon.com/cloudfront/

[17] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts-on-demand-reserved-instances.html

[18] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html

[19] https://docs.aws.amazon.com/general/latest/gr/aws_service_limits.html

[20] https://aws.amazon.com/about-aws/whats-new/2014/07/31/aws-trusted-advisor-security-and-service-limits-checks-now-free/

[21] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring_automated_manual.html

[22] http://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch_Dashboards.html

[23] http://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/publishingMetrics.html

[24] https://aws.amazon.com/blogs/aws/new-whitepaper-use-aws-for-disaster-recovery/

[25] http://media.amazonwebservices.com/AWS_Operational_Checklists.pdf

[26] http://d0.awsstatic.com/whitepapers/architecture/AWS_Well-Architected_Framework.pdf

[27] https://aws.amazon.com/premiumsupport/iem/