AWS 初创企业弹性基准

AWS 规范性指导

AWS 规范性指导: AWS 初创企业弹性基准

Table of Contents

AWS 初创企业弹性基准

Amazon Web Services（贡献者）

2026 年 3 月（文件历史记录）

每位初创公司创始人都知道紧张局势：团队正在竞相推出新功能，而客户则期望每个版本都能正常运行。你醒来时会收到一则关于昨晚停电的通知，客户投诉不断涌入，而且你的路线图已经包含了承诺的功能。这是初创企业的日常现实——在创新压力和维护稳定服务的需求之间取得平衡。

优先考虑新功能的本能是很自然的。投资者想要增长，客户要求更多的能力，而竞争对手并没有停滞不前。然而，经验表明，即使是最具创新性的初创公司，韧性问题也可能使他们脱轨。服务中断不仅意味着收入损失；还会削弱你努力与客户建立的信任。实际上，在拥挤的市场中，坚如磐石的可靠性可以成为你无声的差异化因素。

增强初创企业的弹性并不意味着放缓或进行大规模的基础设施投资。这是关于尽早做出明智的选择，防止以后出现问题。可以把它想象成盖房子。一开始就打下坚实的基础比在所有东西都建好后修复结构性问题容易得多。通过将弹性实践融入您的初始架构和运营中，您可以创造竞争优势。您不仅在交付产品；还要提供可靠的体验，从而建立客户忠诚度。

AWS 初创企业弹性基准 (AWS SRB) 是专门为面临这一挑战的团队创建的。它提供了一种实用的途径，可以在保持初创公司与众不同的速度的同时，增强系统的可靠性。没有繁重的流程或企业规模的复杂性，只有随业务增长的实用模式。

本指南介绍如何在启动环境中实现 AWS SRB。它可以帮助您确定哪些因素对弹性至关重要，将有限的资源集中在哪里，以及如何建立可随着公司发展而扩展的实践。

指导范围

在探索建立初创企业弹性的实际步骤时，重要的是要了解该指南在组织更广泛的增长轨迹中的位置。初 AWS 创企业弹性基准与弹性生命周期框架保持一致。它用作初始路线图，旨在帮助您在部署第一个应用程序时以最小的开发开销实施基本的弹性措施 AWS。可以把它当作你的入门套件，用来构建能够从中断中有效恢复的可靠系统。

Well AWS -Architected Framework 和本弹性指南相辅相成，可帮助您构建安全可靠的基础架构。本指南更深入地探讨了弹性实践，而 Well-A AWS rchitected Framework 则提供了更广泛的架构最佳实践。您可以使用AWS Well-Architected Tool中的，根据这些实践来评估您的架构 AWS 账户。

责任共担模式

阐明在云端构建弹性系统的一个重要方面至关重要：与您的初创公司 AWS 之间的合作伙伴关系。云端弹性采用责任共担模式，在确保系统弹性方面，每个团队都扮演着不同的角色。 AWS

AWS 负责为其服务提供支持的底层云基础设施的弹性。可以将其视为构建应用程序的基础。这包括保持数据中心、网络组件和架构中 AWS 服务 使用的核心的弹性。

您的初创公司的责任集中在您在此基础上构建和部署的系统的弹性上。本指南中的建议属于您的责任范围。通过深思熟虑地实施这些做法，您可以创建使用可靠 AWS 基础架构并包含强大恢复功能的应用程序。

这种模式意味着你永远不会孤立地建立弹性。您正在建立在久经考验、可靠的基础上，同时将精力集中在直接影响客户和业务运营的各个方面。本指南探讨了如何充分利用这种责任共担模式，使用可靠 AWS 的基础架构，同时实施切实可行的措施，确保您的应用程序从中断中恢复过来。

第一阶段：设定目标

想象一下，在重大产品发布之前，你的团队正处于最后的冲刺阶段。这些新功能是开创性的，投资者的兴奋情绪也在增强。然后，在例行部署期间，您的核心服务就会停机。当客户投诉充斥你的电子邮件时，有两个问题变得非常明确：你能负担得起离线多长时间？ 您能承受丢失哪些数据的代价？

希望一切都能正常运转并不是一个好策略。你需要一种系统的方法来决定哪些地方的弹性最重要，哪些不重要。这就是业务影响分析 (BIA) 变得至关重要的地方。它可以帮助您做出明智的决定，决定在哪里投资于韧性。BIA 可以帮助您了解系统的哪些部分真正需要坚如磐石的可靠性，哪些部分可以承受一定的灵活性。

首先绘制您的核心用户旅程。对于每一个人，请问自己以下几点：

- 如果中断会有什么影响？
- 我们必须以多快的速度恢复服务？
- 哪些数据对保护至关重要？

这不仅仅是一项技术练习；它可以帮助您了解可靠性问题对业务的影响。收入损失仅仅是个开始。考虑一下中断会如何削弱客户信任、违反监管要求或给竞争对手带来优势。

通过此分析，您将得出每个用户旅程的两个关键数字：恢复时间目标 (RTO) 和恢复点目标 (RPO)。RTO 定义了您必须以多快的速度恢复该旅程。RPO 定义了您的客户可以容忍多少数据丢失。然后，这些以业务为导向的目标将指导您选择哪些组件以及如何架构它们，而不会对系统的每个部分进行过度设计。

这种方法的美妙之处在于，它可以帮助您将有限的资源集中在最重要的地方。也许您的核心事务处理需要近乎即时的恢复和零数据丢失，但是您的推荐引擎可以容忍更长的停机时间。通过设定明确的目标，您可以创建一个框架，使您可以继续快速开发功能，同时从战略上增强弹性。

清楚地记录这些目标。它们不只是为你的工程团队准备的。当你向企业客户推销或与投资者进行技术尽职调查时，本文档表明你对业务连续性进行了批判性思考。

这些目标会随着创业公司的发展而变化。您的第一千名用户的弹性需求与您的第一个企业客户的弹性需求不同。从你今天可以实际实现的目标开始，但要计划好随着规模的扩大将如何收紧这些目标。

本指南探讨了如何实施满足这些目标的弹性措施。设定这些目标是您至关重要的第一步。它们是您应对创新与稳定之间持续紧张关系的指南针，可帮助您构建一个可靠地为客户提供价值的系统。

第二阶段：设计与实施

本节讨论如何将您的韧性目标变为现实。您已经规划了哪些对您的业务最重要，现在是时候构建它了。如何在不减缓创新的前提下增强韧性？

可以将 AWS 托管服务视为弹性的捷径。与其花费宝贵的工程时间来维护基础架构，不如使用能为您处理冗余的服务。例如，考虑亚马逊简单存储服务 (Amazon S3) S ervice。它会自动将您的数据的多个副本存储在中 AWS 区域 以保持持久性。它不需要额外的密码或深夜的寻呼机任务。

你的核心应用程序组件呢？ 明智的选择可以成倍增加团队的影响力。考虑一下作为服务支柱的数据库。与其构建自己的复制系统，不如考虑使用自动处理故障转移的 Amazon Aurora。这些功能的成本可能会更高，但它们会将团队的注意力从维护基础架构转移到解决业务问题上。这笔费用可以通过更快的功能交付和避免中断期间的收入损失来抵消。

有时，初创公司需要构建自定义解决方案。这就是创新型初创公司的本质。当你这样做时，要保持简单但要聪明。使用 Elastic Load Balancing 和 Amazon EC2 Auto Scaling 群组将您的应用程序分布到多个可用区。设置 Auto Scaling 组的最小容量以处理您的基准流量，即使一个可用区出现故障。这无需复杂的架构模式即可抵御局部故障。随着您的初创公司的发展和客户对更高弹性的需求，您可以发展到更复杂的方法。

我们建议您将生产和开发环境分开 AWS 账户。当你快速移动时，混合它们很诱人，但是这个边界就是你的安全网。它可以防止善意的实验中断你的生产服务。可以把它当作你的 “快速行动，打破现状” 的发展文化的保险——在开发中打破局面，保持生产稳定。

如果您的应用程序依赖于第三方服务，请为其故障做好准备。当您的支付处理器出现问题时，您的系统能否优雅地处理问题？ 构建简单的断路器和备用选项。也许可以将这些交易排队而不是显示错误消息。即使不是完美无瑕，您的客户也会对您保持正常运行表示赞赏。

在构建时记录下来，但要保持实用性。专注于记录关键决策背后的原因，并创建简单的恢复手册。重要的是要在事件发生时做好准备。

你不是在为完美的韧性而建造；而是在为适当的韧性而建造。每花一小时过度设计弹性，就等于没有花在客户要求的功能上的一个小时。使用 AWS 托管服务作为基础，在最重要的位置添加有针对性的弹性，并创建清晰的路径来随着业务的增长扩大弹性。

下一章讨论如何在不消耗工程资源的情况下验证这些设计选择。对于初创公司来说，测试应该是一种合理的提升，也是对应用程序弹性的明智投资。

第三阶段：评估与测试

你已经奠定了坚实的基础，但你怎么知道它真的起作用了呢？ 当你竞相证明产品与市场的契合度时，测试弹性听起来像是一种奢侈品。但是，有一种明智的方法可以在不影响功能开发的情况下做到这一点。本章介绍适合初创企业步伐的精益、实用的测试。

首先 AWS Resilience Hub，将其视为初始架构评估工具。它为架构的弹性基础提供了有用的基准评估。它通过检查常见的配置模式和潜在的单点故障，帮助您评估基本基础架构设置是否符合您的恢复目标。它可以标记明显的差距，例如缺少多个可用区配置或备份策略不完整。Resilience Hub 补充但不能取代深思熟虑的架构审查和对关键路径的有针对性的测试。

要验证您记录的恢复目标，请在您的开发环境AWS Backup中安排每月的恢复测试。尽管这需要工程时间，但它可能比在实际事件中发现备份不起作用要便宜。将其作为常规开发周期的一部分，例如运行单元测试或代码审查。目标不是完美；而是信心你可以在需要的时候恢复过来。

随着您的初创公司的发展以及客户开始越来越依赖您，请逐步升级您的测试游戏。部署新功能时，请在管道中加入基本的弹性检查。使用尝试简单的混沌实验AWS Fault Injection Service。从您的预制作环境开始，然后从小处着手。在考虑任何生产实验之前，请先测试您的应用程序如何处理开发中的延迟 API 响应。随着您信心的增强，请逐渐扩展这些测试，但请务必先在预制作中进行验证。对于初创公司来说，如果不故意破坏生产中的东西，就足够冒险。

关键是平衡。在测试上花费的每一小时都不是花在开发新功能上的一个小时。但是，一些战略测试可以防止那种失去客户信任的停机。使用提供的自动化工具 AWS 来完成繁重的工作，并专注于对客户最重要的测试。这可以帮助您在不减缓创新的情况下建立对应用程序弹性的信心。

下一章将探讨如何随着创业公司的规模发展而发展这一基础。

第四阶段：运营

您已经构建了一个弹性应用程序并对其进行了测试。现在，日常现实使它继续运转。但是在初创公司中，你无法观看所有操作，也不应该尝试这样做。关键是在不提供太多指标或使团队负担过重的情况下对重要的事情保持警惕。

从客户的角度开始。Amazon S CloudWatch ynthetics 加那利群岛充当自动买家。他们不断测试关键的用户旅程。让他们登录，使用测试账户模拟购买，或者访问关键功能，尤其是在你最繁忙的时候。这可以帮助您了解客户体验，并帮助您在真实用户发现问题之前发现问题。当金丝雀失败时，从客户的角度来看，你会立即知道出了点问题。

在此基础上再接再厉，重点监控支持基础架构。什么信号告诉你有麻烦？ Amazon CloudWatch 可帮助您构建跟踪这些迹象的仪表板。不要只监控技术指标，还要将它们与业务影响联系起来。例如，高 CPU 使用率很重要，但这是因为它可能会降低您使用加那利群岛跟踪的客户体验。

作为一种实用的方法，将您的监控与客户旅程对应起来。如果您运行的是软件即服务 (SaaS) 平台，则可能关心 API 响应时间、身份验证成功率和核心功能可用性。设置提醒，告知您这些指标何时出现偏差。但是，要有选择性。每个警报都应要求采取行动。如果你的团队因为 “可能什么都不是” 而开始忽略警报，那么你设置的指标太多了，或者跟踪的指标有误。

通过您的团队已经使用的工具发送这些警报。如果您的工程师使用的是特定的消息传递应用程序，请在那里发送警报。目标是在不创建新流程的情况下快速感知。当警报触发时，你的团队应该确切地知道它的含义以及该怎么做。

保持操作文档的精简和实用。将包含代码的 runbook 存储在版本控制中，但请记住，它们不是小说。当出现故障时，您的团队需要采取清晰、可操作的步骤。每个警报都应链接到相应的运行手册，并且每个运行手册都应回答三个问题：

- 什么坏了？
- 这为什么非常重要？
- 如何修复此问题？

实施简单的事件管理流程。你不需要复杂的框架，只需要明确定义什么构成事件以及事态升级时该给谁打电话。保留事件日志，因为它们可以帮助您提高应用程序的弹性。

关键是在警惕和开销之间找到最佳位置。使用 AWS 工具尽你所能实现自动化，专注于监控影响客户的指标，并保持流程足够轻松，以便随着增长而发展。

下一章探讨如何在不牺牲使初创企业与众不同的速度和创新的情况下培养韧性心态。归根结底，韧性既关乎人，也关乎技术。

第五阶段：响应与学习

当你经营一家初创公司时，复杂的验尸流程可能会减慢团队的速度。本章探讨了如何从事件中吸取教训而不将其变成官僚主义活动。

将事件学习整合到您现有的节奏中。如果您的团队已经定期开会，请花十分钟时间讨论最近发生的事件。专注于实际问题，例如：

- 运行手册有帮助吗？
- 警报发生的时间是否正确？
- AWS 托管服务能阻止这种情况吗？

专注于行动，而不是责备。在初创公司中，你不是在构建一个完美的系统；而是在构建一个每次出现问题时都会变得更好的系统。

您可以使用票务系统来跟踪事件；无需专门的工具。创建一个简单的模板，其中包括事件时间表、客户影响、采取的恢复步骤和经验教训。如果你积极使用它，这个 cam 就会变成机构记忆。在入职期间查看过去的事件，让新工程师快速上手。设计类似系统时，请在架构评论中参考它们。让他们进入游戏时代，根据实际事件创建逼真的故障场景。该模板记录了发生的事情，经常使用可以将其转化为组织学习。

随着初创公司的发展，模式也随之出现。也许某些组件更频繁地出现故障，或者某些类型的更改可能会导致问题。使用这些模式来指导弹性投资。如果数据库故障转移导致问题，请考虑改进您的多可用区设置。如果第三方服务中断是常见的主题，可以考虑改进断路器。

目标不是防止所有可能的故障。这是不可能的，而且会减慢你的速度。目标是在快速成长的同时，快速学习、快速适应并保持应用程序足够可靠。利用每一次事件的机会，让你的系统更具弹性，让你的团队知识更丰富，让你的客户对你的服务更有信心。对于初创公司来说，速度和学习效果都非常完美。创建轻量级流程，帮助您在不减缓创新的情况下从事件中吸取教训。最佳弹性实践是您的团队实际使用的实践。

初创企业的下一步行动

还记得那个让你心跳加速的深夜部署吗？ 还是主要客户询问您的弹性措施的那一刻？ 初创公司的韧性之旅不是要消除这些时刻，而是要充满信心地面对它们。

本指南描述了实现弹性的实用途径，包括：设定切合实际的恢复目标、使用 AWS 托管服务、实施精益测试、监控重要内容以及从事件中学习。这种方法对初创公司来说非常强大，因为它可以与您一起成长。同样的框架可以帮助您的初创公司从当今的问题中恢复过来，为将来的企业客户提供服务奠定了基础。

可以将弹性想象成产品路线图；你不能同时构建所有内容。从保护您的核心服务开始。添加对关键客户旅程的监控。实施能够发现明显问题的基本测试。记录你学到的东西。每个步骤都易于管理且实用，最重要的是，它们不会阻止您使用配送功能。

随着初创公司的发展，韧性需求也在不断变化。企业客户可能会问更棘手的问题。高峰流量可能会创下新高。国际扩张带来了新的挑战。因为你已经在基础中建立了韧性，所以你已经做好了适应的准备。

初创公司的目标不是完美的正常运行时间或零事故。目标是建立一个足够可靠的系统，以赢得客户的信任，同时保持使初创公司与众不同的速度。通过遵循这种方法，你可以创造出比创建完美系统更有价值的东西；你可以从每一个挑战中吸取教训，同时跟上创业公司的雄心壮志，从而创建一个每天都变得更好的系统。

韧性之旅并没有结束。它会随着你发布的每一个功能、你赢得的每一个客户以及你处理的每一个事件而演变。本指南可帮助您构建一个框架，帮助您自信而切实地向前迈出一步。归根结底，创业公司的成功并不是要在弹性和速度之间做出选择。这是为了找到明智的方法来实现这两者。

资源

AWS Well-Architected 框架

- 可靠性支柱
  - REL05-BP01 实现优雅降级，将适用的硬依赖关系转换为软依赖关系
  - REL06-BP01 监控所有组件的工作负载
  - REL06-BP03 发送通知（实时处理和警报）
  - REL08-BP03 将弹性测试作为部署的一部分进行集成
- 弹性责任共担模型

AWS 规范性指导

- 上的 Backup 和恢复方法 AWS
- 弹性生命周期框架：持续改进弹性的方法

其他 AWS 资源

- AWS 故障隔离边界（AWS 白皮书）
- 为云应用程序设定 RPO 和 RTO 目标（AWS 博客文章）
- AWS（AWS 白皮书）上的部署选项概述

贡献者

以下个人参与了本指南的编撰。

编写

- Dylan McCarroll，助理解决方案架构师， AWS
- 初创企业解决方案架构师 Igor Fil AWS
- Parth Shah，高级解决方案架构师， AWS
- Vrajesh Prajapati，解决方案架构师， AWS

审阅

- 首席技术专家 Clark Richey AWS
- 布鲁诺·埃默尔，首席技术专家 AWS

技术写作

- Lilly AbouHarb，高级技术撰稿人， AWS

文档历史记录

下表介绍了本指南的一些重要更改。如果您希望收到有关未来更新的通知，可以订阅 RSS 源。

变更	说明	日期
重大更新	更新了建议和自始至 AWS 服务 终的用法。	2026年3月6日
初次发布	—	2024 年 1 月 24 日

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。