阿里云

阿里云 DevOps 解决方案

兼具云和 DevOps 的双重优势

目录

01 DevOps 握手

多年来，IT 组织一直将开发和运营团队作为独立的实体进行维护。尽管有着相同的组织目标，但是这些并行运作的团队之间通常会意见不合。组织意识到必须创建新型的合并运作模型，因此 DevOps 方法应运而生。

DevOps 模型并不仅仅是简单地实施用于管理基础设施的敏捷性原则。John Willis 和 Damon Edwards 使用术语 CAMS（文化、自动化、衡量和共享）定义了 DevOps。DevOps 旨在促进开发与运营团队之间的协作。它的目的是通过消除由物理位置、组织功能和业务目标造成的障碍，拉近这两个团队在多个方面的距离。DevOps 可让传统开发人员和运营角色之间的界限变得模糊不清，以最大限度缩小差距并减少孤岛，进而推动各小组之间实现业务连续性、问责制和成果。DevOps 敏捷方法还可帮助企业快速、轻松地应对不断变化的需求。简而言之，DevOps 是开发与运营之间的协作，致力于提高向组织和客户交付产品的效率与创新速度。

本白皮书首先向首席技术官、开发人员和运营经理简单介绍 DevOps 的优势和重要性，然后再深入探究开源容器服务和自动化工具的主要功能，其中包括 Kubernetes、Terraform 和 Packer。接下来讨论业务案例研究，以帮助读者了解 DevOps 解决方案的实施。

02 传统软件开发模型面临的挑战

传统软件开发模型包括 Rational 统一过程、V 模型和瀑布模型。现代流程（尤其是敏捷方法）正在大量取代现有的传统开发模型。这种渐进式改变的主要原因是组织在实施传统软件开发模型时面临的挑战，包括：

人工调停：

传统软件开发模型的一个主要挑战/限制是需要人工干预。此类干预通常会导致不可重复的过程，并会引入人为错误。人工干预还是敏捷性的一个障碍，尤其是在测试和部署方面。手动执行的测试无法实现持续交付和持续集成。此外，手动测试还会增加产生缺陷和从事计划外工作的可能性。手动执行的部署往往存在部署失败的风险，进而导致任务不可靠、计划不周全。

环境不一致：

组织常常在非典型、不统一的环境中运行，这会使开发、测试和生产的配置复杂化。在修复不一致环境生成的错误时，团队通常需花费数天甚至数周时间，既浪费资源又浪费时间。

监控受限：

DevOps 的一个重要优势是能够持续高效地监控，传统开发模型也具备这一优势。DevOps 可以触发自定义警报和多个监控警报，以帮助用户更有效地利用资源。但是，传统模型不具备此类功能。它们完全依赖开发人员执行手动检查操作。此类流程不仅会引入错误，还会延误产品交付或推出时间。

缺乏共享所有权：

传统软件开发模型缺乏共享所有权这一概念，这会导致组织出现沟通问题。我们推出 DevOps 的主要原因之一正是为了填补这种空白。在传统模型中，开发和运营团队是两个不同的团队，它们独立运作，并且只对各自的任务负责。

期望与现实之间存在差距：

传统方法的一个主要诟病点是，模型不考虑现实世界的约束条件。随着时间的推移，主要的开发人员和技术专家都目睹了这一概念的失败，因为他们在开始实施之后常常会遇到障碍。理论上似乎可行的设计在经济上通常不可行或在实践中通常会变得复杂。

测试实践过时：

传统软件开发模型包括过时的测试实践。缺乏持续集成意味着，开发人员需要手动检测错误并手动部署以跟进这些错误。在团队就可行的解决方案达成一致意见之前，开发人员必须不断重复此流程，最后才能将软件推送到生产阶段。基于类似情况的每次发布都意味着会向系统引入更多技术“债务”，进而造成计划外和不可靠的任务。

缺乏灵活性：

传统开发方法因在响应变更管理时不够灵活而饱受诟病。开发人员通常必须在更早阶段执行此流程，以纳入任何需要的修改或编辑内容。传统方法缺乏灵活性这一特点使其不适合敏捷开发方法。

03 促使 DevOps 成为首选软件开发模型的因素

随着时间的推移，组织正逐渐将 DevOps 用作他们的首选软件开发模型。RightScale 的年度云状态调查报告显示，DevOps 采用率已从 2013 年的 54% 增长到 2017 年的 78%。Puppet 的 2017 年 DevOps 状态报告披露，到 2020 年，组织很可能会解聘那些无法适应形势并将其组织转型为 DevOps 的首席信息官。随着企业开始了解 DevOps 的优势，DevOps 方法的采用将会呈现大幅增加趋势。下面列出了组织因采用 DevOps 而获得的一些主要优势。

便捷的自动化：

DevOps 模型包括多项功能，例如操作系统修补，设置 CI/CD 以自动执行部署、报告生成和测试等。DevOps 实现的自动化流程涉及创建版本、运行测试用例和生成报告等方面。自动化流程不会出错，并且稳定性、效率和可靠性都得到提升，从而缩短交付时间。

端到端监控：

DevOps 模型的一个重要方面是端到端监控需求。全面的监控有助于用户了解整体应用性能。全面了解应用堆栈可以进一步增强协作和提高效率。此外，这有助于企业快速找出基础设施和软件内存在的问题或复杂情况。

业务敏捷性：

DevOps 可帮助组织对孤岛进行分类，并将重点放在反馈和协作事宜上，以便在多个开发阶段之间迅速切换。Futurum Research 首席分析师兼 Broadsuite Media 首席执行官 Daniel Newman 指出，大型组织可使用 DevOps 提高组织响应速度。而且，DevOps 可帮助各种规模的组织创建更短的迭代流程，让组织加快创新和问题处理速度。

日常工作效率

在典型的 IT 环境中，组织通常会浪费大量资源来反复地等待和解决相同问题。因此，组织必须消除不必要的重复任务，将员工的精力用在为组织创造价值上。DevOps 推动了部署自动化和生产环境标准化。实施 DevOps 模型会创建可预测且可控的部署，以便将团队从琐碎的任务中解放出来并提高工作效率。

通过编排实现自动化：

DevOps 在自动化方面提供全面的控制和协调，涵盖基础设施的整个层次结构。换句话说，编排是一种特殊的自动化方式。用户可以使用 Chef、Ansible 和 Puppet 等工具执行编排。这些工具都有自己明确定义的标准，且与领先的云提供商兼容。

即时交付部署：

DevOps 通过最新工具解决基础设施问题。它通过创建自定义逻辑及其写入功能来实现此目的。此外，DevOps 还使用一键式构建工具帮助实现整个流程的自动化。此类工具与云服务兼容，可提高部署效率。

04 推动业务发展的 DevOps 最佳实践

用户可以通过实施 DevOps 最佳实践，充分利用 DevOps 的众多优势。这些最佳实践旨在确保组织使用敏捷的软件开发方法，从而促进开发与运营团队之间的协调，并提供更好的最终用户体验。

消除 IT 孤岛：

以往，IT 组织被划分为多个学科孤岛。在这种设置中，软件开发遵循流水线式方法，一个部门仅执行一组任务。这种分离会限制主动协作，并会延误合作任务和交付时间。因此，组织的高层管理者必须打破团队之间的职能孤岛。

实施自动测试：

DevOps 最重要的一个方面是实施自动测试。用户应该将测试自动化与测试驱动开发（TDD）和行为驱动开发（BDD）结合起来，以取得最佳成果。要实现自动化，应确定场景和测试用例，选择合适的自动化工具，构建测试环境，运行测试用例，并最终分析结果。敏捷团队会分多个批次执行自动测试任务，这样团队在发现故障时能够立即进行修复。

使用集成式配置管理：

为了最大程度发挥 DevOps 的优势，组织应对软件开发实施集成式配置管理方法。这样，开发团队可以在解决方案级别应用配置管理，同时又不会忘记组织和解决方案中的生产配置问题。在 DevOps 环境中，开发人员必须要有企业意识并着眼于全局。通过集成式配置管理，运营团队可以了解新发布产品的潜在影响，从而使产品发布的规划变得更加简单。

部署持续集成：

DevOps 部署流程合并了多个子流程，其中包括版本管理、代码开发、测试、部署和多项部署后任务。DevOps 的目的是使用 Bamboo 和 Jenkins 等 DevOps 工具自动执行这些任务，以减少人工干预。组织必须持续（而非定期）集成更改，确保能充分利用 DevOps。

确保持续交付：

持续交付是一种软件开发模型，利用该模型，组织以较小批次创建软件，以便能够随时将软件发布到生产环节。ThoughtWorks 的 Martin Fowler 进一步解释说，团队可在下列情况下实现持续交付：

- 软件在整个生命周期内均可部署。
- 开发团队通过开发新功能确保软件可以部署。
- 维持系统的生产就绪性，尽管对其进行了更改。
- 团队可以按需执行任何环境的任何版本的一键式部署。

团队可以通过集成软件、创建可执行文件及运行自动测试来检测任何问题，从而处于持续交付状态。组织可通过极小的部署风险、可跟踪的进度以及接收用户反馈的机会，从持续交付中获益。

专门的监控：

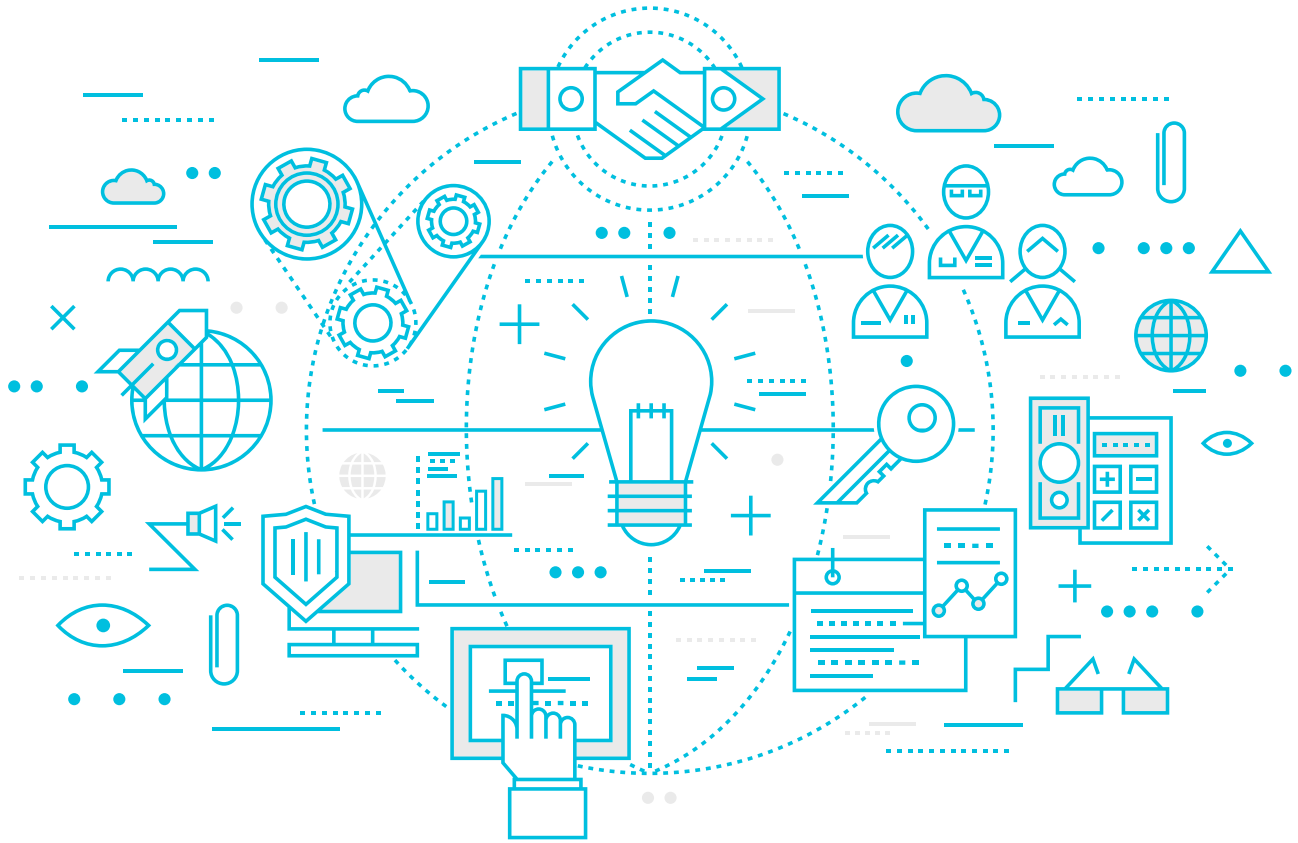
对 DevOps 而言，监控基础设施变得更加重要。事实证明，加载时间跟踪、查询日志和其他关键细节有助于优化应用性能。部署错误对整个应用可能是灾难性的，需要进行应用检查。相关团队可以利用 Geckoboard 等在线工具来确保主动监控。它结合了许多工具中与基础设施、分析、应用、销售和图表相关的多个指标。监控团队必须谨记的一个重要方面是，在监控时设置相关警报，以便及时管理诸如应用无响应等事件。

数据驱动的方法：

参与 DevOps 流程的团队必须确保将自己的主要关注点放在性能上。对事实数据的分析是实现这一目标的决定性因素。团队成员还必须认真地与其他团队成员共享应用图表、使用模式和其他任何相关数据，以统一方向和明确目标。此外，测试、可扩展性和部署的纳入有助于进一步简化整个开发流程。开发和运营团队必须齐心协力，使用可重复、内聚和持续改进的方法来创建和提升应用性能。

团队文化

DevOps 从根本上来说是一种不同的运作方式，比简单地实施新工具要复杂得多。Gartner 的预测进一步重申了强大、动态的 DevOps 团队组成的重要性。该报告预测，到 2018 年，在尝试利用 DevOps 但又未明确解决文化基础问题的 I&O 组织中，有 90% 会面临失败。



05 热门 DevOps 工具

DevOps 的基础是两项基本特性及其各自工具的可用性和功能性。DevOps 的成功很大程度上依赖于容器、自动化流程及其工具的使用。本白皮书将讨论企业如何通过容器和自动化充分利用 DevOps。

5.1 容器：

容器是指在具有相同内核类型的多个计算环境中移动时，帮助运行应用或软件的解决方案。这种移动可能是暂存到生产环境，也可能是从物理机移动到云中的虚拟机。此外，通过将软件与周围的环境隔离开，容器还有助于最大程度减少在同一基础设施中处理不同软件的不同团队之间的冲突。每个容器化应用都在一个操作系统上运行，并且与其他容器共享内核，从而使其具有轻便特性。此外，容器还支持微服务，这是 DevOps 通常使用的一种架构。微服务是指被分割的各个小应用部件，以取代大型整体式应用。.

容器服务的优势：

提升业务敏捷性：

利用容器，开发和部署流程对用户来说变得更加简单。这样组织就可以更快地启动新软件和应用，从而提高组织的敏捷性。

加快开发速度：

利用容器，用户可以将一个应用分割成多个独立的小元素，即微服务。这样可最大程度缩减创建、测试、部署和集成之间的周期时间。

优化云利用率：

随着采用云技术的组织越来越多，容器可通过改善云环境以便将多个容器化应用部署到单个云实例来帮助充分利用云服务。

提升面向服务的架构：

容器化可确保每个容器运行单个应用或流程。这样，用户可提高整个开发模型的质量。

轻便且易于使用：

容器极其轻便。此特性有助于轻松管理应用组件。而且，还可以将其部署到多个操作系统和虚拟机上。

各个行业的开发人员使用的最受欢迎的容器服务之一是 Kubernetes。下面将详细介绍 Kubernetes、其功能和用途。

5.1.1 Kubernetes

Kubernetes 是一个开源系统，可帮助开发人员实现容器化应用的自动部署、可扩展性与管理。此外，它帮助用户将构成应用的容器分成逻辑单元，以便于发现和管理。由于 Kubernetes 是一个开源系统，因此组织可自由地利用基础设施，包括私有云、公有云或混合云。久而久之，组织逐渐发现 Kubernetes 是最受青睐的容器工具。由 451 Research 进行的一项最新调查显示，71% 的受访者（包括不同行业的 200 个大型组织）指出他们使用的是 Kubernetes，这进一步证实了该工具的受欢迎程度。

另外，Kubernetes 可在公有云和私有云中运行，这也提高了它的受欢迎程度。451 Research 进行的同一项调查显示，57% 的 IT 决策者表示，他们更喜欢在“容器即服务”(CaaS) 和内部 CaaS 上托管容器。

5.1.1.2 Kubernetes 的功能

再次重申一下，Kubernetes 如此受欢迎的一个主要原因是它对各种功能的强大支持，例如：

自动装箱：

通过 Kubernetes，用户可以根据资源需求和限制自动放置容器。它在不影响资源可用性的情况下实现这一点。而且，用户可以将“尽力而为”的工作负载与关键负载相混合，以提升利用率并充分利用资源。

自动回滚和推出：

让 Kubernetes 成为最受青睐的容器服务之一的一项功能是，它可以定期推出对应用及其配置的更改。而且它还能同时监控应用运行状况，确保不会一次性消除所有实例。如果发生错误，Kubernetes 可撤消更改。

自我修复：

如果容器运行失败，则 Kubernetes 允许用户重启容器，并在节点无响应时重新计划并替换它。对于未通过用户定义的运行状况检查的容器，Kubernetes 会立即将其销毁，以免出现任何错误或问题。

负载均衡和服务发现：

Kubernetes 会为容器分配它们自己的 IP 地址，并为一组容器分配一个 DNS 名称，从而确保在各个容器间顺利实现负载均衡。此外，用户也不必费心修改应用以使用不寻常的服务发现机制。

5.1.1.3 阿里云和 Kubernetes

为了方便使用 Kubernetes 来管理阿里云中的容器应用，阿里云容器服务提供了对 Kubernetes 集群的支持。利用阿里云资源编排服务 ROS 的应用部署功能，用户只需使用 ROS 模板单击一下，即可创建高度可用且安全的 Kubernetes 集群。

Kubernetes 集群整合了阿里云的存储、网络、虚拟化和安全功能，以提供可简化集群创建和扩展的高性能应用管理。

在阿里云上部署的 Kubernetes 可为容器化应用的部署、扩展和管理提供便利。它进一步关注容器化管理和应用开发，并随附以下功能：


弹性扩展和自行修复


服务发现和服务器负载均衡


服务发布与回滚


机密和配置管理

5.1.1.4 业务场景

要求：

一家位于阿联酋的休闲娱乐公司希望创建一个能够在 OTA 模式下运行的以目的地为中心的 Web 和移动应用。此外，它还希望改进旅游平台的生态系统，以纳入产品和销售管理、收入管理、库存管理、内容管理以及用于 B2C 和 B2B 的其他功能。

挑战：

要求实施完全托管的支持系统：

与所有组织一样，该组织必须确保支持系统在所有客户交互中都可以访问。

避免基础设施束缚：

该组织主要担心他们的工作最终会将他们束缚在现有环境中。甚至是使用标准工具和框架构建的基础设施也存在这种可能性。

需要类似 PaaS 的解决方案：

该组织的主要要求是用于运行应用的可靠模型，而无需负责硬件和软件基础设施维护。采用 PaaS 解决方案可确保简单性、可扩展性和可靠性。

支持多地区备份：

该组织希望通过跨地区的数据复制消除单点故障风险。

滚动升级：

该组织希望各种应用全天候可用，这要求开发人员每天多次部署应用的新版本。

运行容器化应用：

运行容器化应用是一个极其不稳定而且复杂的过程。为解决此问题，组织需要一个工具，在容器化应用从开发切换至生产时顺利运行容器化工作流。

Kubernetes 为何成为首选工具

Kubernetes 可满足组织构建用于运行复杂应用的大型集群的所有要求。使用 Kubernetes 还可以通过对可扩展性和回滚升级的支持来避免基础设施束缚。此外，它还为用户提供一个框架，用于建立应用之间的交互规则。

Kubernetes 还提供一个用于应用部署的统一平台，可减轻组织手动执行这些任务的负担。而且，呈现给用户的工作单元处于“服务”级别。Kubernetes 让用户能够收集一段时间内的详细数据，并使用该数据发现某些趋势，以帮助指示系统是否将要出现故障。

此外，它还帮助用户定义何时移入生产环境的分界线。这种明确的区分可确保用户的容灾和组织的业务连续性。除了对集群化基础设施进行多项功能性改进外，Kubernetes 还提供开发解决方案的无缝更新和升级。

解决方案架构

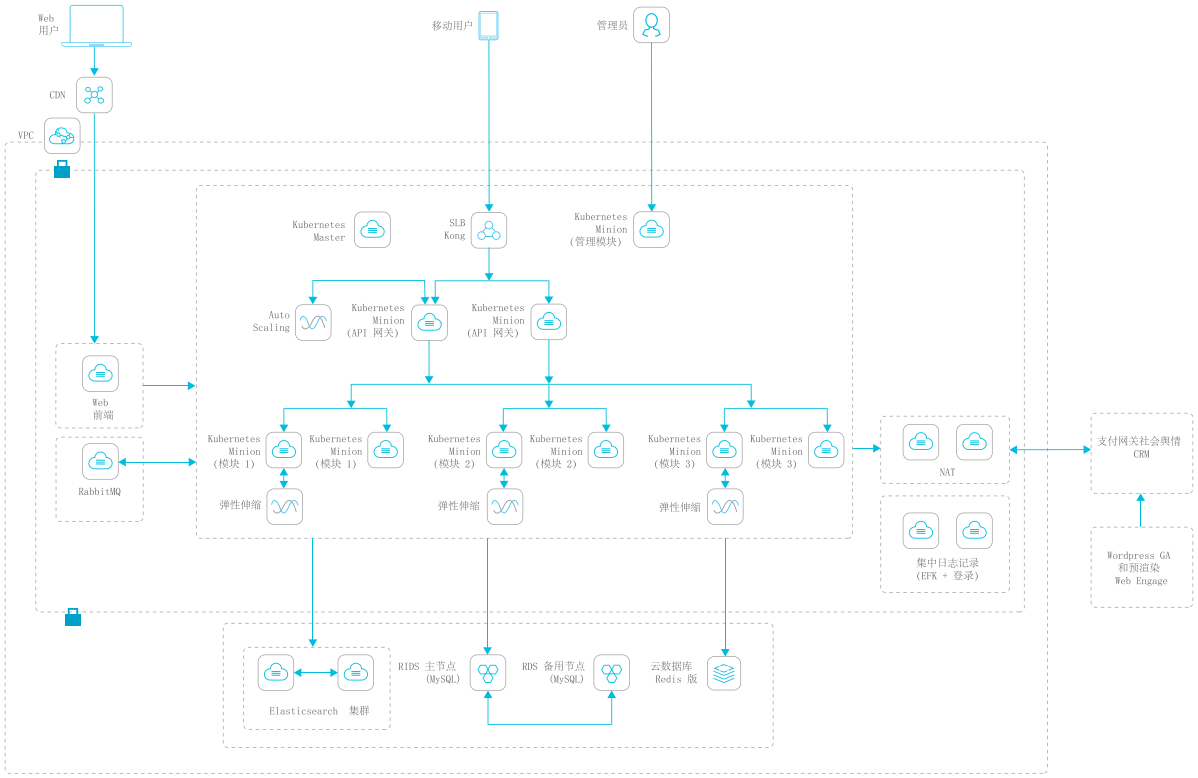


图 1

架构描述：

阿里云实施了面向公众的负载均衡，可将传入的 HTTP 请求自动分发给运行 Kubernetes Pod 的弹性计算服务 (ECS) 网络服务器。这使得使用共享存储/网络以及适用于运行容器的规格成为可能。

此外，负载均衡通过有效地路由请求，还有助于尽可能缩短响应时间并提升用户体验。ECS 实例上部署的 Web/应用服务器可高效处理业务逻辑。该流程提供了一个有利于初创企业的灵活架构，因为 ECS 根据应用的实时需求来工作。

云数据库 RDS 版帮助存储自动数据备份以避免冗余，并分离数据以实现可扩展性。这样不仅可以节省时间，还会显著优化资源使用。云数据库 RDS 版是确保数据安全和高级别内置安全功能（包括网络级别的资源和访问管理）所必需的。

此外，内容分发网络 (CDN) 向全球边缘节点网络分发静态内容，因此全球各地的用户均可访问 MewMe 平台并享受更好的用户体验。这还帮助组织有效地将网站响应时间缩短至几毫秒，从而确保顺利浏览图像并处理海量流量。阿里云的可靠性、可用性和简化的维护帮助组织显著降低基础设施成本，从而使其能够更多地关注应用开发、创新和用户体验

使用的产品：

弹性伸缩、弹性计算服务 (ECS)、负载均衡、专有网络 (VPC)、云数据库 RDS 版（关系数据库系统）、云数据库 Redis 版、CDN（内容分发网络）。

5.2 使用 DevOps 的自动化

鉴于持续集成和持续部署是 DevOps 的核心功能，显然自动化是整个 DevOps 模型的重要贡献因素。自动化的目的不仅仅是改进软件开发机制，而且还要填补软件开发模型中的手动工作造成的漏洞。组织可以采用自动化来处理频繁的回归测试迭代，并设法加快交付过程。此外，开发人员还会发现，在处理微服务架构和处理超大型项目时，自动化简直是种福音。

自动化优势：

自动化已经改变了开发人员行使职责、交付和处理事情的方式。自动化优势可帮助开发人员理解整个机制的影响和重要性：

加快发布速度：

由自动化机制执行的部署流程见证了最低开销。这让组织可以轻松地重复此类流程，从而让开发人员可以加快发布速度。频繁的发布随后又会促成更敏捷的软件开发机制。

部署时错误最少：

由于手动部署基于人的判断和计算，因此总是存在出现人为错误的可能性。在手动部署期间，开发人员可能会忽略关键步骤或关键错误。在完全自动化的部署中，由于部署不需要人为干预，因此不存在这种可能性。配置完成后，每次开发人员启动发布时，部署过程都将保持不变，从而消除了出现错误的可能性。

缩短部署时间：

手动部署过程是一项耗时的任务。该过程需要开发人员投入很长时间，而这些时间本可以用来完成更有意义的任务。进行自动部署时，整个过程只需要极短的时间。验证在后台进行，不需要开发人员参与。这样便可最大程度地缩短开发时间，从而让开发人员能够执行更有影响力的任务。

轻松进行新版本发布：

尽管基础发布是永久性的，但自动化部署允许轻松更改目标环境和机器。在团队需要创建新安装的情况下，自动化部署可确保所涉及的开销为最低。只需配置现有设置，让自动化执行其余任务。

在本白皮书中，我们将讨论两个自动化工具 – Terraform 和 Packer。

5.2.1. Terraform

Terraform 是一个开源工具，它将 API 整理到分析配置文件中。这些文件可在不同的 DevOps 团队成员之间共享，并且可被视为代码进行修改、审阅和版本管理。此外，该工具还能高效、安全地促进基础设施的创建、编辑和版本管理。Terraform 由 HashiCorp 提供，可帮助用户创建解决方案来完成创建完整数据中心的编码。它会生成一个执行计划，其中概述了用于获取所需状态的计划。而且，该工具还可确定所做的更改并创建适用的增量执行计划。

凭借管理低级别组件（如存储、计算实例和网络组件）的功能，Terraform 有了自己的不同语言，称为 HashiCorp 配置语言 (HCL)。除具有机器友好特性外，该语言还可促进人类可读与人类可编辑之间的平衡。

5.2.1.1 Terraform 的功能

自动化 DevOps 工具随附多项功能，除了定制的内部解决方案外，这些功能还可帮助管理常见的现有服务提供商。其功能使 Terraform 能够以独立和声明的方式整理和组合多个服务提供商的资源。下面是对其功能的简要说明：

基础设施即代码 (IaC)：

指的是使用高级配置语法配置基础设施，而不是定义物理硬件配置。使用此类语法可实现数据中心的版本管理，并且可以像创建其他任何代码一样创建数据中心。Terraform 还允许用户在一段时间内重用以及与其他团队成员共享基础设施。

执行计划：

Terraform 的一个重要功能是包含“规划”步骤。当用户应用此步骤时，此步骤会生成一个可向用户显示 Terraform 操作的执行计划。此类显示可确保用户了解 Terraform 执行的基础设施操作。

资源图：

Terraform 会为所有资源创建一个图表，并冻结任何非相关资源的创建和编辑。这样会创建一个高效的基础设施，它让用户能够全面洞悉基础设施中的依赖项。

促进自动更改：

Terraform 支持对基础设施应用复杂的更改集，以最大限度减少人工干预。通过不同的执行计划和资源图，用户可以跟踪 Terraform 实现的更改以及更改顺序。这种自动化更改机制不需要人工干预，可降低出现人为错误的概率。

5.2.1.2 业务场景

要求：

一家领先的直播卫星电视提供商设置了用于负载测试的新环境。但是，他们不想浪费资源和时间来手动完成此任务，而是希望借助自动化工具来帮助提供所有资源。

挑战：

方便与云提供商集成

自动创建新环境

自动配置资源

Terraform 为何成为首选工具

组织面临的主要挑战是设置用于加载的环境，而不想花太多时间。如果没有 Terraform，这将是一个手动且又耗时的过程。此外，只要他们需要创建新环境，就必须配置资源。组织不想使用实施起来很复杂且需要广泛技能集和知识的工具。

组织中的开发人员决定选用 Terraform，因为通过该工具可以轻松、方便地配置所有资源。当不再需要资源时，开发人员还可以销毁已配置的资源。

解决方案架构

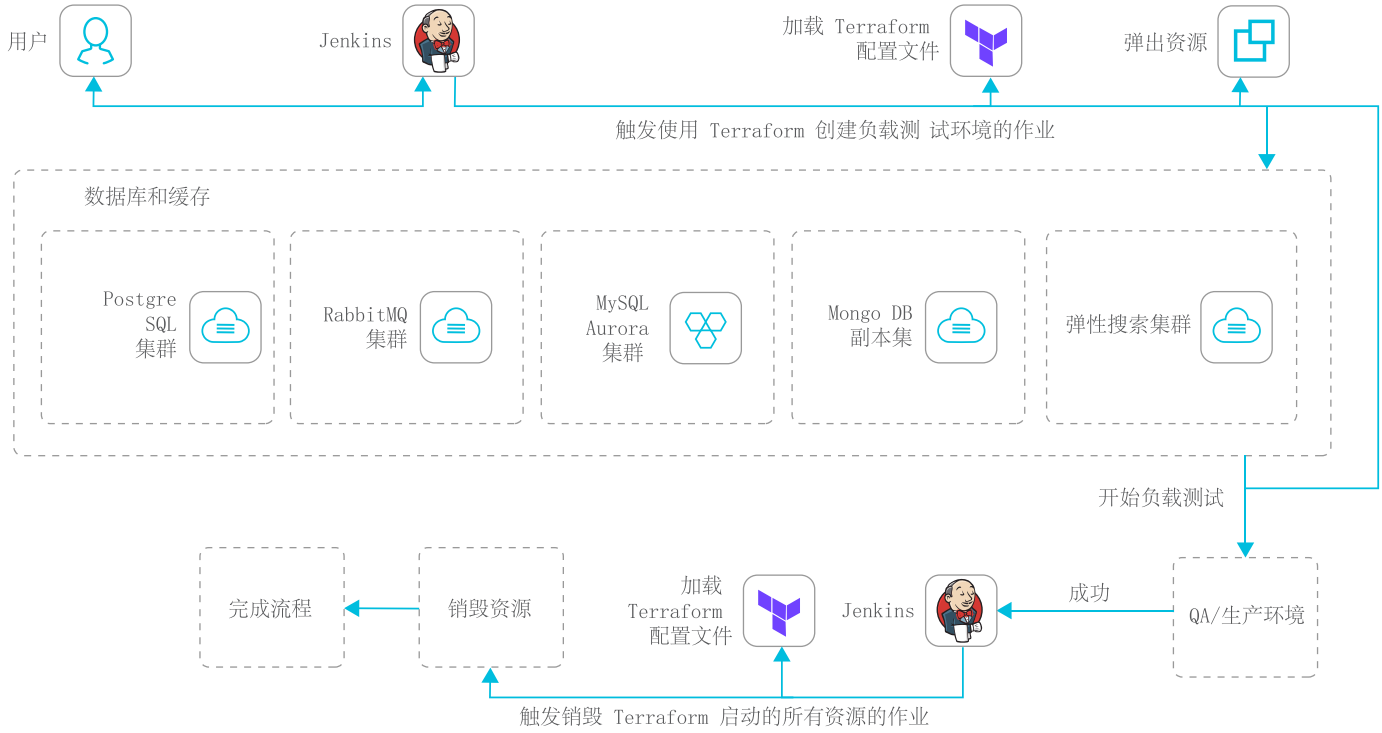


图 2

架构描述：

为了加载环境，开发人员在负载测试每次需要资源时都使用 PostgreSQL 集群、RabbitMQ 集群、MongoDB 副本和 Elasticsearch 集群。同样，开发人员还部署了 Terraform，以确保正确配置这些资源。为此，开发人员为每个资源创建一个 Terraform 模板，完成该过程后再运行资源。这些资源运行完成后，开发人员运行 chef 配方来确保对配置进行管理。

使用的产品：

PostgreSQL、RabbitMQ、MongoDB 和 Elasticsearch

5.2.2 Packer

对于需要通过工具自动创建机器镜像的组织，HashiCorp 提供了一款名为 Packer 的自动化工具。它采用现代配置管理，帮助用户使用自动化脚本在 Packer 创建的镜像中安装和配置软件。Packer 能够将机器镜像投射到现代环境，以创造多种可能性和开启新的机会，从而从现有 DevOps 自动化工具中脱颖而出。它采用的方法是允许用户使用 Chef 或 Puppet 等框架在 Packer 创建的镜像中安装和配置软件。

5.2.2.1 Packer 的功能

加快基础设施部署：

Packer 镜像让用户只需几秒钟便可启动完全配置和调配的机器，而之前的工具完成这个过程通常需要几分钟甚至几小时。由于 Packer 可以立即启动开发虚拟机，而无需等待冗长的配置时间，这对生产以及开发团队非常有利。

多提供商可移植性：

Packer 为能够在私有云中运行的多个平台以及桌面虚拟化解决方案中的开发环境创建相同的映像。每个环境都运行相同的机器镜像，从而为用户提供可移植性优势。

增强的稳定性：

Packer 可为用户提供增强的稳定性。它允许在镜像形成期间为机器安装和配置所有软件。用户可以即时追踪和解决可能出现的任何错误。

高级可测试性：

Packer 允许在机器镜像构建完成后立即执行冒烟测试，从而实现优异的可测试性。测试中的积极结果表示任何其他机器镜像均可正常工作。

5.2.2.2 业务场景

要求：

一家专业提供可定制镜像的领先组织需要自动化开发模型来确保镜像独立于基础设施。

在进行大量研究后，DevOps 经理决定使用 Packer 来满足该组织的各种需求。现有的机器镜像创建和管理起来过于麻烦，而 Packer 则能轻松、快速地处理此任务，同时促进操作和业务的敏捷性。

Packer 为何成为首选工具

Packer 是一个易于使用的自动化 DevOps 工具，可自动创建任何机器镜像。它采用现代配置管理方法，鼓励用户部署 Chef 或 Puppet 等框架，以便在组织内由 Packer 创建的镜像中安装和配置软件。此处讨论的组织使用 Packer 利用相同配置同时创建多个镜像。这让他们的客户在所有环境中都拥有具有相同配置的镜像。

Packer 部署首先创建一个实例，然后借助 SSH 将其配置为“活动”。与现有工具不同，Packer 不需要解压和重新打包系统镜像。

解决方案架构

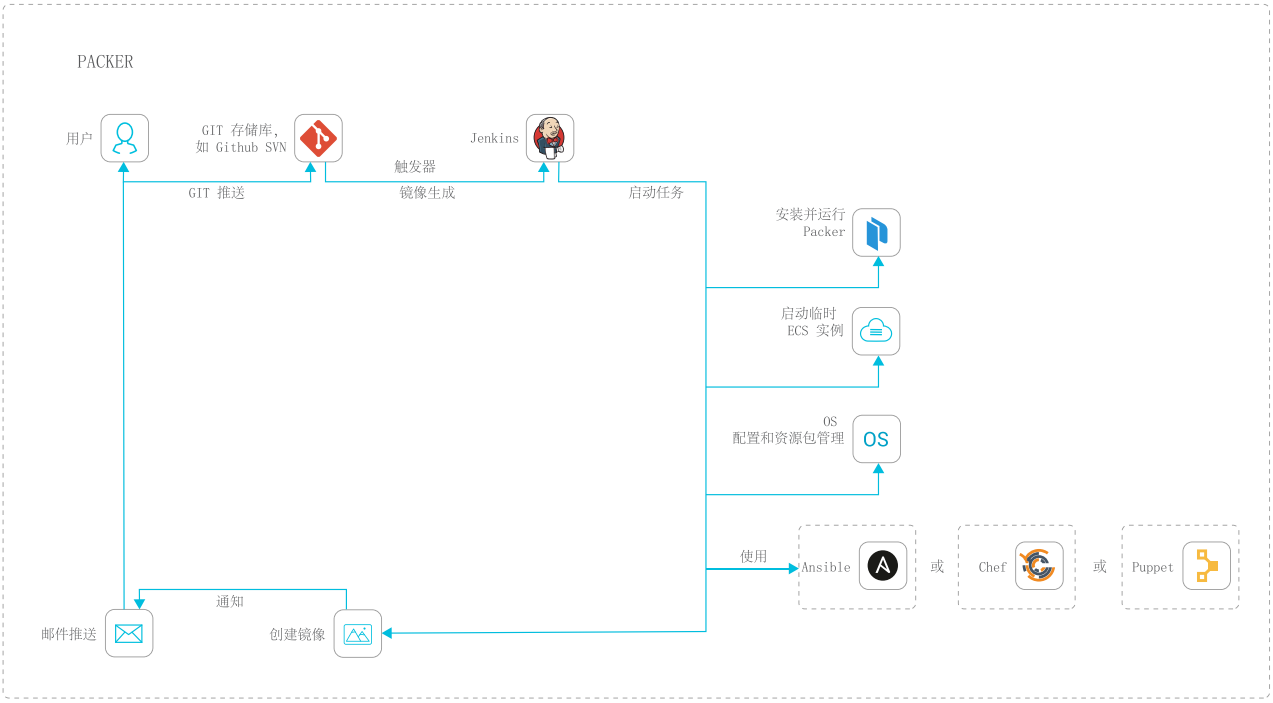


图 3

解决方案描述：

首先，系统在 Git 存储库中推送 Packer 文件。然后，Git 触发 Jenkins，后者部署在 ECS 实例上或镜像生成任务中。Jenkins 用于测试和生成代码。接下来，Jenkins 运行 Packer，后者反过来使用 Packer 配置文件中提供的基本镜像来启动 ECS 实例。然后，Jenkins 根据每个配置文件安装任何包，并根据 Packer 配置文件中提供的配置创建镜像。最后，它通过阿里巴巴的直邮服务通知用户。

5.2.2.3 将 Packer 和 Terraform 与阿里云结合使用

阿里云支持 Packer 和 Terraform，以用于核心封装和基础设施配置。用户可以使用这些工具在阿里云上快速部署基础设施和应用。企业业务应用和基础设施的快速迭代加上持续不断的开发可确保增强运营和最大限度降低维护成本。而且，阿里云还提供一系列灵活的服务，旨在帮助客户使用阿里云和 DevOps 实践快速、可靠地构建和交付产品。

凭借对 Terraform 和 Packer 的支持，阿里云客户能够拥有强大的工作流，轻松管理其全球基础设施。这样一来，用户就可以节省时间，并专注于满足关键业务需求。

Packer 用户可以在阿里云上轻松构建和配置定制镜像，所用工作流和配置与在其他平台上管理镜像所用的完全相同。

同样地，Terraform 用户也可以在阿里云上配置计算、网络和存储资源，所用工作流和配置与在其他云上管理基础设施所用的完全相同。

06 阿里云 – 兼具云和 DevOps 的双重优势

毫无疑问，企业将从云技术与 DevOps 的结合中受益。Bernard Golden 是云计算领域卓越的思想领袖之一，他认为敏捷开发和 DevOps 都需要基础设施立即可用，而云能以最完美的形式满足这一要求。在云上托管 DevOps 可以帮助组织从被动反应式方法变为更具主动性的方法。

为了满足对基础设施配置自动化日益增长的需求，许多云提供商提供了专门的服务，旨在帮助 DevOps 团队生成全新的环境，该环境可能是现有环境的副本。利用云中的 DevOps，组织可以通过连续集成、可扩展性、测试和部署确保完成交付。

下面介绍了以更优方式利用云中 DevOps 的方法。

更高可扩展性：

由于云提供的可扩展性，组织可以选择将云计算用于 DevOps 模型。更高的可扩展性让用户只需单击一下按钮即可增加容量。通过与 DevOps 结合，应用可自动集成可扩展性。而且，在实现所有这些功能的同时，还可最大程度地降低总体基础设施成本。

高效的恢复机制：

与数据中心相比，云在执行备份还原时更加容易。云提供商会公开其命令行 API，以帮助用户自动执行多个任务。此外，用户还可以将这些脚本与持续集成工具相集成。使用命令行，用户可以配置自动备份、自动还原最新备份以及删除旧备份。

消除停机时间：

凭借基于云的连续运营，用户可以消除系统的停机时间。自动开发模型的实现将允许开发人员创建无状态应用。这些应用有助于提高业务可靠性和客户满意度。

凭借转变组织运作、开发和交付方式的多个成功案例，阿里云成为向 DevOps 模型发展的组织的一站式解决方案。访问 www.alibabacloud.com 可了解有关阿里云的更多信息。

References

1. https://www.atlassian.com/devops

2. https://www.mongodb.com/what-is-devops

3. http://www.webopedia.com/TERM/D/devops_development_operations.html

4. http://www.tothenew.com/devops-automation-consulting

5. http://www.tothenew.com/blog/10-mistakes-to-avoid-while-implementing-devops/

6. http://www.tothenew.com/blog/6-barriers-to-devops-and-how-to-avoid-them/

7. http://www.tothenew.com/blog/10-key-devops-practices-to-improve-it-efficiency/

8. https://eternalsunshineoftheismind.wordpress.com/2013/03/10/problems-with-traditional-methods-of-software-development/

9. https://techbeacon.com/devops-automation-4-simple-steps-prioritizing

10. https://dzone.com/articles/what-is-devops-and-how-automation-helps-achieve-itdevops

11. https://www.alibabacloud.com/help/doc-detail/53751.htm

12. https://www.packer.io/docs/commands/index.html

13. https://dzone.com/articles/microservices-with-kubernetes-and-docker

14. https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

15. http://www.infoworld.com/article/3118345/cloud-computing/why-kubernetes-is-winning-the-container-war.html

16. https://www.ubuntu.com/containers/kubernetes

17. https://www.ubuntu.com/containers/kubernetes

18. https://www.alibabacloud.com/forum/read-830

19. https://www.terraform.io/devops

20. https://www.packer.io/

21. 2017-state-of-devops-report.pdfoperations.html

22. https://newrelic.com/devops/benefits-of-devops

23. https://betanews.com/2016/11/25/devops-benefits/

24. https://www.forbes.com/sites/mikekavis/2014/12/18/11-common-devops-bottlenecks/#621dcd2a7737

25. https://www.forbes.com/sites/danielnewman/2017/04/18/agility-is-the-key-to-accelerating-digital-transformation/#1b8de5397277

26. https://www.futurum.xyz/devops-the-key-to-your-companys-success-in-2017/

27. https://www.rightscale.com/press-releases/rightscale-2017-state-of-the-cloud-report-uncovers-cloud-adoption-trends

28. https://dzone.com/articles/what-is-continuous-integration

29. http://www.drdobbs.com/architecture-and-design/top-10-practices-for-effective-devops/240149363

30. https://martinfowler.com/bliki/ContinuousDelivery.html

31. https://www.digitalocean.com/community/tutorials/an-introduction-to-kubernetes#what-is-kubernetes

32. http://www.zdnet.com/article/kubernetes-leads-container-orchestration/

33. https://insights.hpe.com/articles/why-cloud-makes-agile-and-devops-more-important-1701.html

34. http://www.infoworld.com/article/3173266/containers/4-reasons-you-should-use-kubernetes.html

35. https://www.infoq.com/news/2014/08/terraform

36. https://www.packer.io/intro/why.html

37. http://www.devopsdigest.com/devops-advantages-3

38. http://www.tothenew.com/blog/infographic-8-key-benefits-of-using-container-technology/

39. http://www.tothenew.com/blog/why-cios-should-adopt-containers-technology/

40. http://www.decisiondesign.com/approach/methodology-problem/

41. https://dzone.com/articles/jez-humble-why-software

42. https://www.red-gate.com/blog/database-lifecycle-management/5-big-benefits-automated-deployment

43. https://www.forbes.com/sites/forbestechcouncil/2017/07/21/the-relationship-between-the-cloud-and-devops/2/#4783fa5c3428

阿里云

阿里云