AWS 白皮书

在 AWS 上使用 WordPress 的最佳实践

在 AWS 上使用 WordPress 的最佳实践: AWS 白皮书

Table of Contents

在 AWS 上使用 WordPress 的最佳实践

发布日期：2021 年 10 月 19 日 (文档修订)

摘要

本白皮书旨在为系统管理员提供如何在 Amazon Web Services (AWS) 上开始使用 WordPress 以及如何提高部署的成本效益和最终用户体验的具体指导。另外，文中还概述了一种满足一般可扩展性和高可用性要求的参考架构。

引言

WordPress 是一种基于 PHP 和 MySQL 的开源博客工具和内容管理系统 (CMS)，可用于为从个人博客到高流量网站的任何内容提供支持。

第一版 WordPress 发布于 2003 年，在当初设计时，还没有弹性和可扩展性的现代云基础设施的概念。随着 WordPress 社区工作的推进和各种 WordPress 模块的发布，该 CMS 解决方案的功能不断扩展。现在，您可以构建一个利用 AWS 云的众多优势的 WordPress 架构。

部署简单

对于没有严格的高可用性要求的低流量博客或网站，简单的单个服务器的部署比较合适。这种部署虽然不是最具弹性或可扩展性的架构，但却是让网站启动并运转起来的最快速且最经济的方法。

主题

注意事项

本讨论从单个 Web 服务器部署开始。有时单个服务器会不够用，例如：

- 部署您 WordPress 网站的虚拟机属于单点故障。这个实例出现问题会导致您的网站服务中断。
- 要扩展资源来提高性能，只能通过“垂直扩展”的办法，也就是，增加运行 WordPress 网站的虚拟机的大小。

可用方法

AWS 提供许多预置虚拟机的不同选项。在 AWS 上托管您自己的 WordPress 网站主要有三种方式：

- Amazon Lightsail
- Amazon Elastic Compute Cloud (Amazon EC2)
- AWS Marketplace

Amazon Lightsail 是允许您快速启动虚拟专用服务器（Lightsail 实例）来托管 WordPress 网站的一种服务。如果您不需要高度可配置的实例类型或访问高级联网功能，Lightsail 是最简单的入门方式。

Amazon EC2 是提供可调整大小的计算容量的 Web 服务，使您可以在几分钟内启动虚拟服务器。与 Lightsail 相比，Amazon EC2 提供更多配置和管理选项，适用更高级的架构。您拥有对您的 EC2 实例的管理访问权限，并且可以安装您选择的任何软件包，包括 WordPress。

AWS Marketplace 是一个在线商店，您可以在此查找、购买在 AWS 上运行的软件，然后快速部署该软件。您可以使用一键式部署，在短短几分钟内，直接在您自己的 AWS 账户下的 Amazon EC2 中启动预配置的 WordPress 镜像。有许多 Marketplace 供应商提供准备就绪的 WordPress 实例。

本白皮书介绍 Lightsail 选项，并建议将其作为单服务器 WordPress 网站的实现方式。

Amazon Lightsail

Lightsail 为开发人员、小型企业、学生以及需要简单虚拟专用服务器 (VPS) 解决方案的其他用户提供一种非常轻松的利用 AWS 的方法。

该服务将基础设施管理中许多较为复杂的元素抽象出来，让用户可以轻松操作。因此，如果您没有太多基础设施经验，或者您需要专注地运行自己的网站，而且简化的产品足以满足您的需求，那么这是一个理想的起点。

通过 Amazon Lightsail，您可以选择 Windows 或 Linux/Unix 操作系统和常用 Web 应用程序（包括 WordPress），然后只需在预配置的模板中轻点一下，就能部署这些应用程序。

当您的需求扩大时，您可以轻松跨越最初的边界，连接额外的 AWS 数据库、对象存储、缓存和内容分配服务。

选择 Amazon Lightsail 定价计划

Lightsail 计划规定了您用于托管 WordPress 网站的 Lightsail 资源的每月费用。有很多计划涵盖多种使用场景，包括不同级别的 CPU 资源、内存、固态硬盘 (SSD) 存储和数据传输。如果您的网站很复杂，则可能需要具有更多资源的更大实例。您可以通过使用 Web 控制台或按照 Amazon Lightsail CLI 文档中的说明将服务器迁移到更大的计划来实现此目的。

安装 WordPress

Lightsail 为 WordPress 等常用应用程序提供了模板。该模板是运行您自己的 WordPress 网站的良好起点，因为它预装了您需要的大多数软件。您可以使用浏览器内终端，或者您自己的 SSH 客户端，或者通过 WordPress 管理 Web 界面来安装其他软件或自定义软件配置。

Amazon Lightsail 与 GoDaddy 的 Pro Sites 产品相结合，来帮助 WordPress 客户轻松免费地管理实例。Lightsail WordPress 虚拟服务器是预配置好的，并且针对快速性能和安全性进行了优化，使您可以轻松快速地让 WordPress 网站启动并正常运行。运行多个 WordPress 实例的客户会发现，更新、维护和管理其所有网站是件费时费力的事情。利用这一集成，您只需单击几下，在几分钟内就可以轻松管理多个 WordPress 实例。

有关在 Lightsail 上管理 WordPress 的更多信息，请参阅从 Amazon Lightsail 实例开始使用 WordPress。自定义 WordPress 网站后，建议您为实例拍摄快照。

快照是创建 Lightsail 实例的备份镜像的一种方法。它不仅是系统磁盘的副本，而且还存储了原计算机的配置（即内存、CPU、磁盘大小和数据传输速率)。当出现部署或升级失败时，可以利用快照还原到已知良好的配置。

此快照不仅使您可以在需要时恢复服务器，还可以启动具有相同自定义设置的新实例。

从故障中恢复

单个 Web 服务器意味着单点故障，所以必须确保备份网站数据。为此，可以采用前面介绍的快照机制。要从故障中恢复，可以从最近的快照还原新实例。要减少还原期间可能丢失的数据量，快照必须尽可能为最新。

要将数据丢失的可能性降至最低，请确保定期拍摄快照。您可以安排自动拍摄 Lightsail Linux/Unix 实例快照。有关步骤，请参阅在 Amazon Lightsail 中启用或禁用实例或磁盘的自动快照。

AWS 建议您使用静态 IP，这是专用于您的 Lightsail 账户的固定公有 IP 地址。如果需要替换实例，可以将静态 IP 重新分配给新实例。这样，在每次需要替换实例时，不必重新配置任何外部系统（如 DNS 记录）以指向新的 IP 地址。

提高性能和成本效益

您最终可能会超出单服务器部署容量。在这种情况下，您可能需要考虑提高网站性能的选项。在迁移到可扩展的多服务器部署（本文稍后将讨论）之前，您可以采取许多改进性能和提高成本效益的方法。这些是您无论如何都应该遵循的最佳实践，即使迁移到多服务器架构也是如此。

以下各节介绍了许多可以改善 WordPress 网站的性能和可扩展性的选项。有些可以应用于单服务器部署，有些则利用了多服务器的可扩展性。其中许多修改都需要使用一个或多个 WordPress 插件。尽管选项很多，但 W3 Total Cache 是一种受欢迎的选择，它将许多修改整合到一个插件中。

主题

- 加快内容交付
- 数据库缓存
- 字节码缓存

加快内容交付

任何 WordPress 网站都需要提供静态和动态混合内容。静态内容包括镜像、JavaScript 文件或样式表。动态内容包括使用 WordPress PHP 代码在服务器端生成的任何内容，例如，从数据库生成的或针对每个查看者提供个性化内容的网站元素。

最终用户体验的一个重要方面是，在向世界各地的用户交付以前的内容时可能出现网络延迟。加快交付以前的内容可改善最终用户的体验，尤其是分布在全球各地的用户的体验。为此，可以借助 Amazon CloudFront 等内容分发网络 (CDN)。

Amazon CloudFront 是一项 Web 服务，它提供通过全球多个边缘站点分发内容的方式，具有低延迟、数据传输快、简单且经济高效的优点。查看者的请求将自动路由到合适的 CloudFront 边缘站点，缩短了延迟时间。如果内容可以缓存（几秒钟、几分钟甚至几天），并且已经存储在特定边缘站点，则 CloudFront 可以很快交付它。如果内容不能缓存，或者已经过期，或者目前不在该边缘站点上，CloudFront 将在 CloudFront 配置中从一个或多个真实来源（在本例中为 Lightsail 实例）检索内容。这种检索通过优化的网络连接完成，因此交付网站上内容的速度加快。除了改善最终用户体验，所讨论的模型还减少了源服务器上的负载，并有可能显著节省成本。

静态内容卸载

这包括 CSS、JavaScript 和镜像文件，不管它们是属于 WordPress 主题还是属于内容管理员上传的媒体文件。所有这些文件都可以使用 W3 Total Cache 之类的插件存储在 Amazon Simple Storage

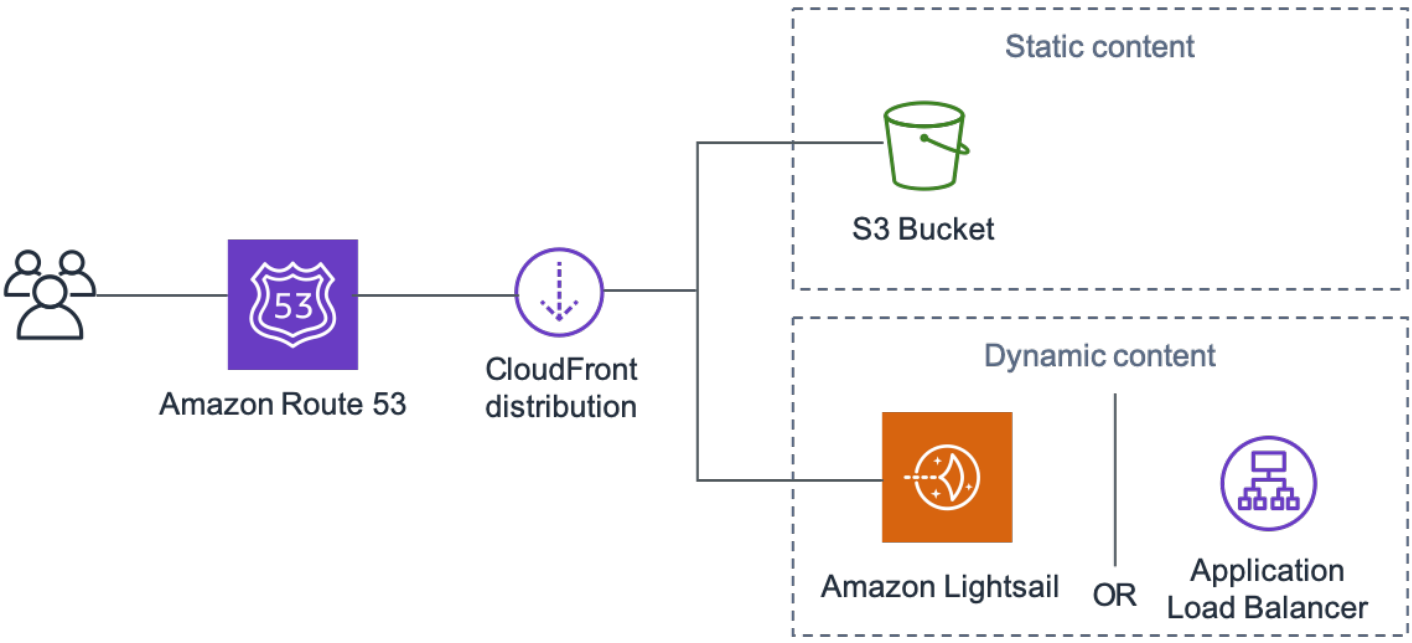
Service (Amazon S3) 中，并且可以采用可扩展且高度可用的方式提供给用户。Amazon S3 以低成本提供高度可扩展、可靠且低延迟的数据存储基础设施，可通过 REST API 访问该设施。Amazon S3 以冗余方式存储您的对象，不仅存储在多台设备上，还会跨一个 AWS 区域的多个设施存储，这带来了极高的持久性。

此举的积极效果是，从 Lightsail 实例上卸载了这一工作负载，让它专注于动态内容的生成。这减轻了服务器负载，是迈向创建无状态架构（实现自动扩展的先决条件）的重要一步。

随后，您可以将 Amazon S3 配置为 CloudFront 的源，以改进向全球用户交付这些静态资产的过程。尽管 WordPress 未与 Amazon S3 和 CloudFront 整合到开箱即用的程度，但各种插件增加了对这些服务（例如 W3 Total Cache）的支持。

动态内容

动态内容包括服务器端 WordPress PHP 脚本的输出。将 WordPress 网站配置为源后，也可以通过 CloudFront 提供动态内容。因为动态内容包含个性化内容，所以您需要将 CloudFront 配置为将某些 HTTP Cookie 和 HTTP 标头作为请求的一部分转发到自定义源服务器。CloudFront 使用转发的 Cookie 值作为标识其缓存中唯一对象的密钥的一部分。为确保最大限度提高缓存效率，应将 CloudFront 配置为仅转发那些真正改变内容的 HTTP Cookie 和 HTTP 标头（不是仅在客户端使用的或被第三方应用程序使用的 Cookie，例如用于网络分析的 Cookie）。



通过 Amazon CloudFront 交付整个网站

前图包括两个源：一个静态内容源，一个动态内容源。有关实现的详细信息，请参阅附录 A：CloudFront 配置和附录 B：插件的安装和配置。

CloudFront 使用标准缓存控制标头来确定是否应缓存特定 HTTP 响应以及缓存多长时间。Web 浏览器也使用相同的缓存控制标头来决定何时在本地缓存内容以及缓存多长时间，以获得更好的最终用户体验（例如，在下载一个 `.css` 文件后，每次回访者查看页面时都不会重新下载该文件）。您可以在 Web 服务器级别配置缓存控制标头（例如，通过 `.htaccess` 文件或修改 `httpd.conf` 文件），或安装 WordPress 插件（例如 W3 Total Cache）来指示如何为静态和动态内容设置这些标头。

数据库缓存

数据库缓存可以显著减少延迟和提高读取密集型应用程序工作负载（如WordPress）的吞吐量。通过将经常访问的数据片段存储在内存中以实现低延迟访问（例如，I/O 密集型数据库查询的结果），可以提高应用程序的性能。当很大一部分查询由高速缓存处理时，需要访问数据库的查询数量就会减少，从而降低与运行数据库相关的成本。

尽管 WordPress 开箱即用的缓存功能有限，但各种插件都支持与 Memcached（一种广泛采用的内存对象缓存系统）集成。W3 Total Cache 插件就是一个很好的例子。

在最简单的情况下，您可以在 Web 服务器上安装 Memcached，并将结果捕获为新快照。在本例中，您负责与运行缓存相关的管理任务。

另一种选择是利用 Amazon ElastiCache 等托管服务来避免这种运营负担。ElastiCache 让您在云环境中轻松部署、操作和扩展分布式内存中的缓存。您可以在 Amazon ElastiCache 文档中找到有关如何连接到 ElastiCache 集群节点的信息。

如果您正在使用 Lightsail 并希望私下访问您的 AWS 账户中的 ElastiCache 集群，可以使用 VPC 对等连接。有关启用 VPC 对等连接的说明，请参阅设置 Amazon VPC 对等连接以使用 Amazon Lightsail 之外的 AWS 资源。

字节码缓存

每次运行一个 PHP 脚本时，都会解析和编译该脚本。通过使用 PHP 字节码缓存，PHP 编译的输出存储在 RAM 中，这样就无需反复编译相同的脚本。这减少了与执行 PHP 脚本相关的开销，改进了性能并降低了 CPU 要求。

字节码缓存可以安装在任何托管 WordPress 的 Lightsail 实例上，因此大大减少了其负载。对于 PHP 5.5 及更高版本，AWS 建议使用 OpCache，这是该 PHP 版本随附的一个扩展功能。

请注意，Bitnami WordPress Lightsail 模板中默认启用 OpCache，无需进一步操作。

弹性部署

很多时候，单服务器部署可能无法满足您的网站需求。这时候就需要多服务器的可扩展架构。

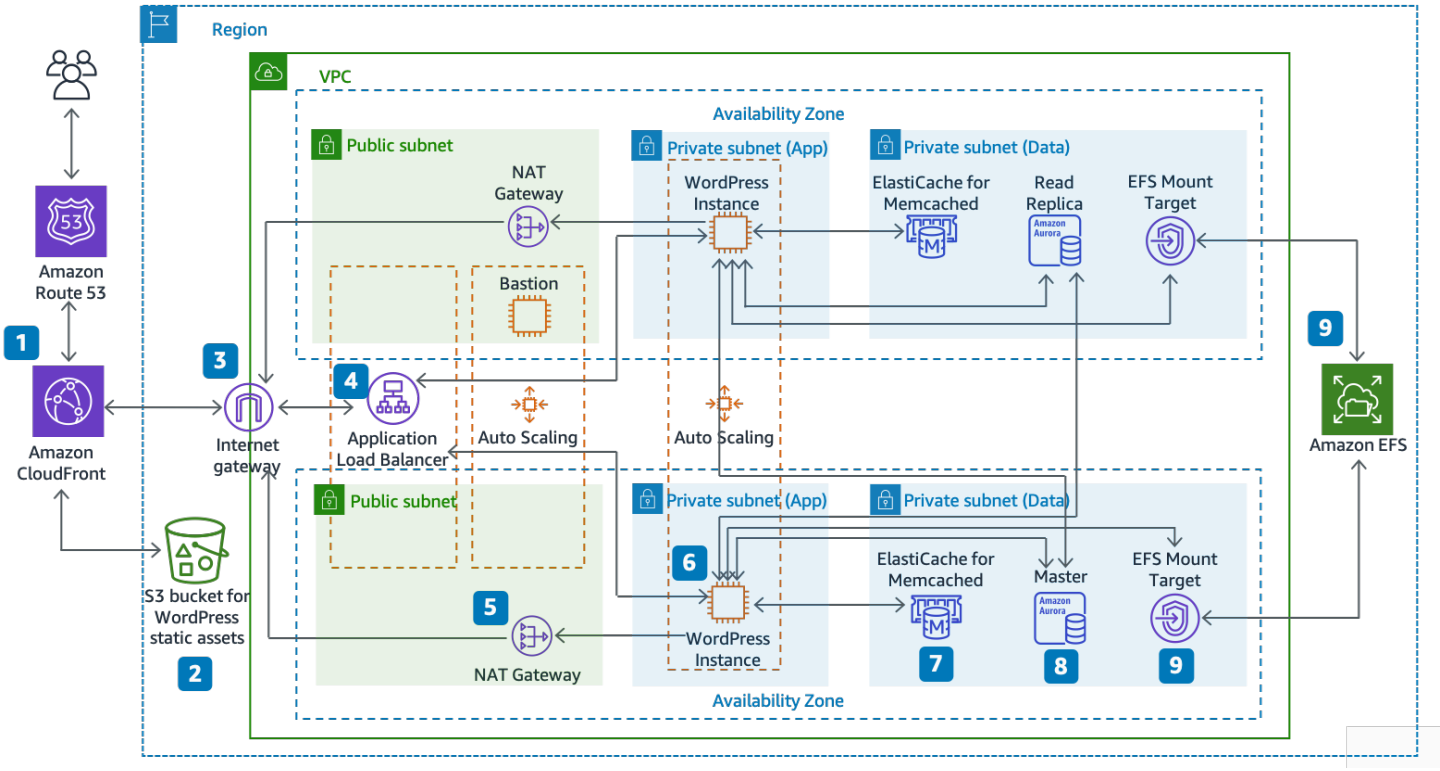
主题

- 参考架构
- 扩展 Web 层
- 无状态 Web 层
- WordPress high availability by Bitnami on AWS Quick Start

参考架构

GitHub 上提供的在 AWS 上托管 WordPress 的参考架构概述了在 AWS 上部署 WordPress 的最佳实践，并包括一组 AWS CloudFormation 模板来帮助您快速启动和运行。以下架构基于该参考架构。本节的其余部分将回顾架构选择背后的原因。

2021 年 7 月，GitHub 中的基础 AMI 从 Amazon Linux1 改为 Amazon Linux2。但是，S3 中的部署模板尚未更改。如果使用 S3 中的模板部署参考架构有问题，建议使用 GitHub 上的模板。



在 AWS 上托管 WordPress 的参考架构

架构组件

此参考架构展示了在 AWS 上部署 WordPress 网站的完整最佳实践。

- 首先是 Amazon CloudFront (1) 中的边缘缓存，旨在将内容缓存到靠近最终用户的地方以加快 交付。
- CloudFront 从 Web 实例前面的 S3 存储桶 (2) 拉取静态内容，从 Application Load Balancer (4) 拉取动态内容。
- Web 实例在 Amazon EC2 实例 (6) 的 Auto Scaling 组中运行。
- ElastiCache 集群 (7) 缓存频繁查询的数据以加快响应速度。

  Amazon Aurora MySQL 实例 (8) 托管 WordPress 数据库。
- WordPress EC2 实例通过每个可用区中的 EFS 挂载目标 (9) 访问 Amazon EFS 文件系统上的共享 WordPress 数据。
- 互联网网关 (3) 允许 VPC 中的资源与互联网进行通信。
- 每个可用区中的 NAT 网关 (5) 允许私有子网（应用程序和数据）中的 EC2 实例访问互联网。

在 Amazon VPC 中，存在两种子网：公有（公有 子网）和私有（应用程序子网和数据子网）。部署到公有子网的资源将获得公有 IP 地址，并在互联网上公开可见。此处部署了 Application Load Balancer (4) 和用于管理的堡垒主机。部署到私有子网的资源只接收私有 IP 地址，因此在互联网上对外不可见，这提高了这些资源的安全性。WordPress Web 服务器实例 (6)、ElastiCache 集群实例 (7)、Aurora MySQL 数据库实例 (8) 和 EFS 挂载目标 (9) 都部署在私有子网中。

本节的其余部分将更详细地介绍这些注意事项。

扩展 Web 层

要将单服务器架构发展成为可扩展的多服务器架构，必须使用五个关键组件：

- Amazon EC2 实例
- Amazon Machine Image (AMI)
- 负载均衡器
- 自动扩展
- 运行状况检查

AWS 提供各种 EC2 实例类型，因此您可以选择兼顾性能和成本的最佳服务器配置。一般来说，针对计算而优化（例如 C4）的实例类型对于 WordPress Web 服务器来说可能是一个不错的选择。您可以跨一个 AWS 区域内的多个可用区部署实例，以提高整体架构的可靠性。

因为您能完全控制自己的 EC2 实例，所以可以使用根访问权限登录，来安装和配置运行 WordPress 网站所需的所有软件组件。之后，您可以将该配置另存为 AMI，将其用于启动具有您的所有自定义设置的新实例。

要将最终用户请求分发到多个 Web 服务器节点，需要一种负载均衡解决方案。AWS 通过 Elastic Load Balancing（一种将流量分发到多个 EC2 实例的高度可用的服务）来提供此功能。因为您的网站通过 HTTP 或 HTTPS 向用户提供内容，所以建议您利用 Application Load Balancer，这是一种应用程序层的负载均衡器，具有内容路由功能，并且能够在不同的域上运行多个 WordPress 网站（如果需要）。

Elastic Load Balancing 支持在一个 AWS 区域内的多个可用区之间分配请求。您还可以配置运行状况检查，允许 Application Load Balancer 自动停止向个别失败（例如，因硬件问题或软件崩溃导致失败）的实例发送流量。AWS 建议使用 WordPress 管理员登录页面 (`/wp-login.php`) 进行运行状况检查，因为此页面可确认 Web 服务器是否在运行，以及是否配置为正确提供 PHP 文件。

您可以选择构建自定义运行状况检查页面来检查其他依赖资源，例如数据库和缓存资源。有关更多信息，请参阅 Application Load Balancer 指南中的目标组的运行状况检查。

弹性是 AWS 云的一个关键特征。您可以根据需要启动更多或更少的计算容量（例如 Web 服务器）。AWS Auto Scaling 是一项 AWS 服务，可帮助您自动执行此预置，以根据您定义的条件扩展或缩减 Amazon EC2 容量，而无需手动干预。您可以配置 AWS Auto Scaling，在需求高峰时无缝增加所使用的 EC2 实例数量来保持性能，并在流量减少时自动减少实例数量来最大限度降低成本。

Elastic Load Balancing 还支持在负载均衡轮换中动态添加和删除 Amazon EC2 主机。Elastic Load Balancing 本身也能动态增减负载均衡容量，以适应流量需求，无需人工干预。

无状态 Web 层

为了在自动扩展配置中利用多个 Web 服务器，您的 Web 层必须是无状态的。无状态应用程序是不需要知道之前的交互也不存储任何会话信息的应用程序。就 WordPress 而言，这意味着无论哪个 Web 服务器处理最终用户的请求，所有终端用户都会收到相同的响应。无状态应用程序可以横向扩展，因为任何可用的计算资源（即 Web 服务器实例）都能处理任何请求。当不再需要该容量时，可以安全地终止任何单个资源（在耗尽运行的任务后）。这些资源不需要知道其他资源是否存在 — 所需要的只是一种将工作负载分发给它们的方法。

当涉及到用户会话数据存储时，WordPress 核心是完全无状态的，因为它依赖存储在客户端 Web 浏览器中的 Cookie。除非您安装了任何依赖原生 PHP 会话的自定义代码（例如 WordPress 插件），否则会话存储不是一个问题。

但是，WordPress 最初是为在单个服务器上运行而设计的。因此它会在服务器的本地文件系统中存储一些数据。在多服务器配置中运行 WordPress 时，这就造成了问题，因为 Web 服务器之间不一致。例如，当用户上传一个新镜像时，该镜像只存储在其中一台服务器上。

这解释了为什么我们需要改进默认 WordPress 运行配置，以将重要数据移动到共享存储的原因。这种最佳实践架构将数据库作为 Web 服务器之外的单独层，并利用共享存储来存储用户上传内容、主题和插件。

共享存储（Amazon S3 和 Amazon EFS）

默认情况下，WordPress 将用户上传的内容存储在本地文件系统上，因此它不是无状态的。所以我们需要将 WordPress 安装和所有用户自定义项（例如配置、插件、主题和用户生成的上传内容）迁移到共享数据平台，以帮助减少 Web 服务器上的负载，并使 Web 层成为无状态的。

Amazon Elastic File System (Amazon EFS) 提供可扩展的网络文件系统用于 EC2 实例。Amazon EFS 文件系统分布在数量不受约束的存储服务器上，因此它们能够弹性扩展，并允许从 EC2 实例大规模并行访问。Amazon EFS 采用分布式设计，这克服了传统文件服务器固有的瓶颈和限制。

通过将整个 WordPress 安装目录移到 EFS 文件系统上，并挂载到每个启动的 EC2 实例中，您的 WordPress 网站及其所有数据将自动存储在不依赖任何 EC2 实例的分布式文件系统中，从而使您的 Web 层处于完全无状态下。这种架构的好处是，您不需要在每个新实例启动时安装插件和主题，因此可以显著加快 WordPress 实例的安装和恢复速度。如本文档的部署注意事项部分所述，您还可以在 WordPress 中更加轻松地部署插件和主题更改。

为确保从 EFS 文件系统运行网站时获得最佳性能，请在适用于 WordPress 的 AWS 参考架构中检查 Amazon EFS 和 OpCache 的推荐配置设置。

您还可以选择将所有静态资产（例如镜像、CSS 和 JavaScript 文件）卸载到前面的具有 CloudFront 缓存的 S3 存储桶中。如本白皮书的静态内容部分所述，在多服务器架构中执行此操作的机制与在单服务器架构中完全相同。其好处与在单服务器架构中相同 — 您可以将与提供静态资产相关的工作卸载给 Amazon S3 和 CloudFront，从而让您的 Web 服务器专注于仅生成动态内容，并针对每个 Web 服务器处理更多用户请求。

数据层（Amazon Aurora 和 Amazon ElastiCache）

WordPress 安装存储在分布式、可扩展、共享的网络文件系统中，并且从 Amazon S3 提供静态资产，使您可以将注意力集中在剩余的有状态组件上，即数据库上。与存储层一样，数据库不应依赖于任何单个服务器，所以不能托管在其中一台 Web 服务器上，相反，应在 Amazon Aurora 上托管 WordPress 数据库。

Amazon Aurora 是一种与 MySQL 和 PostgreSQL 兼容的关系数据库，专为云环境而打造，既具有高端商用数据库的性能和可用性，又具有开源数据库的简单性和成本效益。Aurora MySQL 将数据库引擎与 SSD 支持的专门构建的分布式存储系统紧密集成，提高了 MySQL 的性能和可用性。它具有容错和自我修复能力，可在三个可用区中复制六个数据副本，可用性高于 99.99%，并能持续备份您在 Amazon S3 中的数据。Amazon Aurora 可以自动检测数据库崩溃并重新启动，无需进行崩溃恢复或重新构建数据库缓存。

Amazon Aurora 提供了多种实例类型以适应不同的应用程序配置文件，包括内存优化型实例和突发实例。要提高数据库的性能，您可以选择大型实例类型，以提供更多 CPU 和内存资源。

Amazon Aurora 可自动处理主实例和 Aurora 副本之间的故障转移，以便您的应用程序可以尽快恢复数据库操作，而无需人工管理干预。故障转移通常耗时不到 30 秒。

创建至少一个 Aurora 副本后，使用集群端点连接到主实例，以允许应用程序在主实例发生故障时自动进行故障转移。最多可以在三个可用区中创建 15 个低延迟只读副本。

随着数据库的扩展，数据库缓存也需要扩展。正如之前在数据库缓存部分所讨论的，ElastiCache 具有跨越一个 ElastiCache 集群中的多个节点和跨越一个区域中的多个可用区扩展缓存来提高可用性的功能。在扩展 ElastiCache 集群时，请确保将缓存插件配置为使用配置端点进行连接，以便在添加新的集群节点时，WordPress 可以使用它们，而当旧的集群节点被移除时，WordPress 可以停止使用它们。您还必须将 Web 服务器设置为使用 ElastiCache Cluster Client for PHP，并更新您的 AMI 来存储此更改。

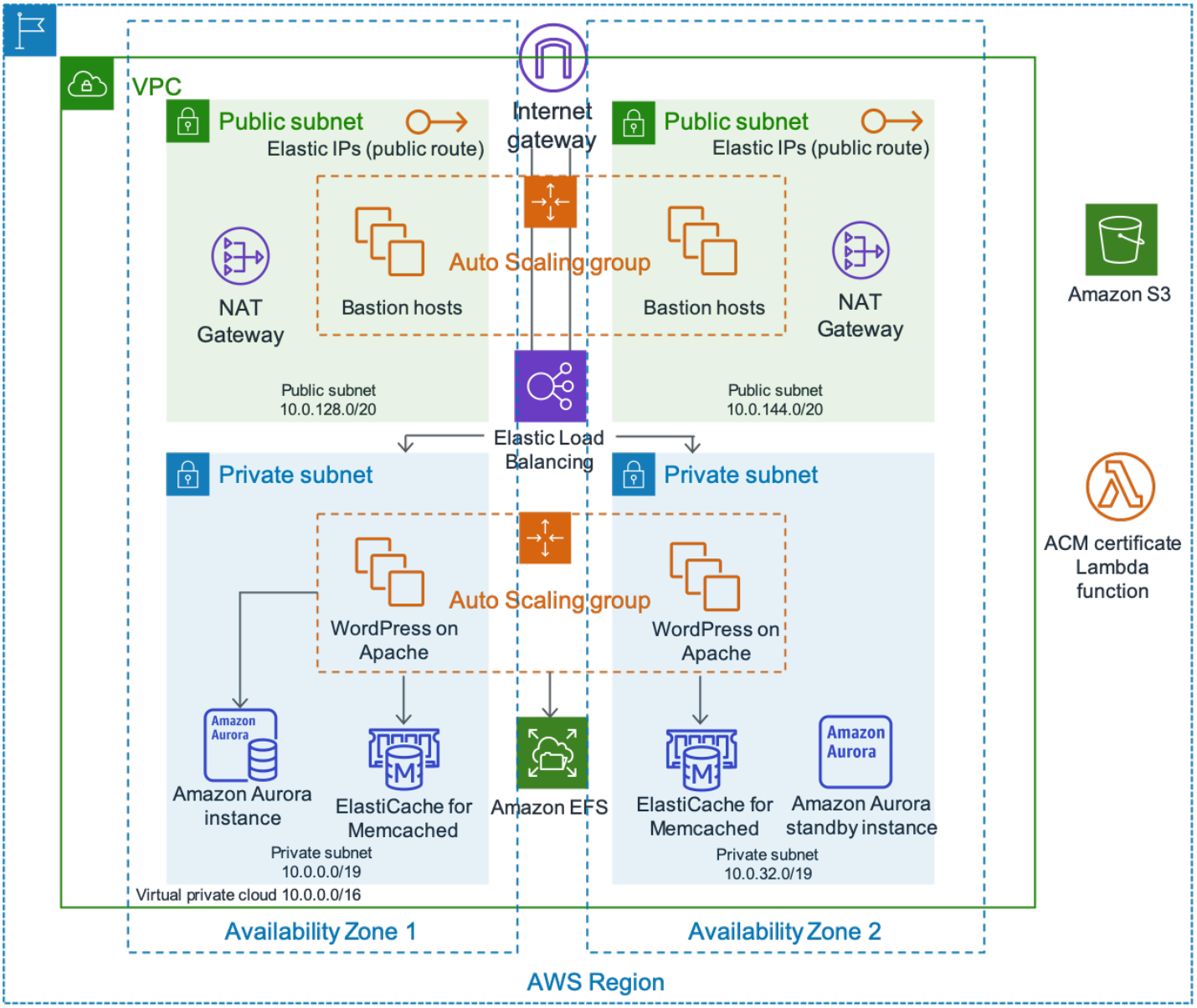
WordPress high availability by Bitnami on AWS Quick Start

Quick Start 由 AWS 解决方案构架师和合作伙伴共同构建，旨在帮助您基于 AWS 最佳实践在 AWS 上部署热门技术，以实现安全性和高可用性。这些加速器将手动操作从数百个减少为只需几个步骤，使您可以快速构建生产环境并立即开始使用。每个 Quick Start 都包括用于自动执行部署的 AWS CloudFormation 模板和一个探讨架构并提供分步部署指导的指南。

WordPress High Availability by Bitnami on AWS Quick Start 在 AWS 上设置了以下可配置环境：

- 跨越两个可用区的具有高可用性的架构。*
- 一个根据 AWS 最佳实践配置了公有和私有子网的 Virtual Private Cloud (VPC)。这为您的部署提供了网络基础设施。*
- 一个用于提供互联网访问权限的互联网网关。此网关供堡垒主机用来发送和接收流量。*
- 在公有子网中设置托管 NAT 网关，以允许对私有子网中的资源进行出站互联网访问。*
- 在公有子网内的 Auto Scaling 组中设置 Linux 堡垒主机，以允许对公有和私有子网中的 EC2 实例进行入站 Secure Shell (SSH) 访问。*
- Elastic Load Balancing 可跨多个 WordPress 实例分发 HTTP 和 HTTPS 请求。
- 在私有子网中，配置在 Apache 上托管 WordPress 应用程序的 EC2 实例。这些实例在 Auto Scaling 组中预置，以确保高可用性。
- 在私有子网中，配置由 Amazon Relational Database Service (Amazon RDS) 管理的 Amazon Aurora 数据库实例。
- 在私有子网中配置 Amazon Elastic File System (Amazon EFS) 以跨 WordPress 实例共享资产（例如插件、主题和镜像）。
- 在私有子网中配置 Amazon ElastiCache for Memcached 节点以缓存数据库查询。

* 将 Quick Start 部署到现有 VPC 的模板将跳过带有星号的任务，并提示您现有的 VPC 配置。



WordPress High Availability by Bitnami 架构

有关部署 WordPress High Availability by Bitnami on AWS 的详细说明不在本文档的讨论范围之内。相关配置和选项，请参阅 WordPress High Availability by Bitnami on AWS。

总结

AWS 提供了许多用于运行 WordPress 的架构选项。最简单的选项就是安装一台服务器，这适用于低流量网站。对于更高级的网站，网站管理员可以添加其他几个选项，每个选项都代表了可用性和可扩展性方面的持续改进。管理员可以选择最符合其要求和预算的功能。

贡献者

本文档的贡献者包括：

- Paul Lewis，Amazon Web Services 解决方案构架师
- Ronan Guilfoyle，Amazon Web Services 解决方案构架师
- Andreas Chatzakis，Amazon Web Services 解决方案架构经理
- Jibril Touzi，Amazon Web Services 技术客户经理
- Hakmin Kim，Amazon Web Services 迁移合作伙伴解决方案构架师

文档修订

要获得有关此白皮书的更新通知，请订阅 RSS 源。

更新-历史记录-更改	更新-历史记录-描述	更新-历史记录-日期
已更新白皮书	更新以修改参考架构和 AWS for WordPress 插件。	2021 年 10 月 19 日
已更新白皮书	更新以包括新的部署方法和 AWS for WordPress 插件。	2019 年 10 月 30 日
已更新白皮书	更新以澄清 Amazon Aurora 产品讯息。	2018 年 2 月 1 日
已更新白皮书	更新以包含首次发布以来启动的 AWS 服务。	2017 年 12 月 1 日
初次发布	第一次发布。	2014 年 12 月 1 日

附录 A：CloudFront 配置

在将 Amazon CloudFront 与 WordPress 网站结合使用时，要获得最佳性能和效率，重点是针对不同类型的内容正确配置网站。

主题

源和行为

源是 CloudFront 发送内容请求以便通过边缘站点分发的地方。根据您的实现，您可以有一到两个源。一个是使用自定义源的动态内容源（单服务器部署选项中的 Lightsail 实例，或弹性部署选项中的 Application Load Balancer）。另一个是引导 CloudFront 获取静态内容的源。在前面的参考架构中，这是一个 S3 存储桶。如果将 Amazon S3 作为分配源，需要使用存储桶策略使内容公开可访问。

行为允许您设置控制 CloudFront 如何缓存内容的规则，进而确定缓存的有效性。行为允许您控制访问您的网站所借助的协议和 HTTP 方法。它们还允许您控制是否将 HTTP 标头、Cookie 或查询字符串传递到后端（如果是，传递哪些）。行为适用于特定 URL 路径模式。

创建 CloudFront 分配

创建 CloudFront Web 分配，通过遵循此分配，自动创建的默认源和行为将用于动态内容。创建四个额外的行为，以进一步自定义处理静态和动态请求的方式。下表总结了这五种行为的配置属性。您也可以跳过此手动配置步骤，使用附录 B：插件的安装和配置中介绍的 AWS for WordPress 插件，这是配置 CloudFront 来加快您的 WordPress 网站速度的最简单方法。

表 1：CloudFront 行为的配置属性摘要

属性	静态	动态（管理员）	动态（前端）
路径（行为）	`wp-content/*` `wp-includes/*`	`wp-admin/*` `wp-login.php`	默认值 (*)
协议	HTTP 和 HTTPS	重定向到 HTTP	HTTP 和 HTTPS

属性	静态	动态（管理员）	动态（前端）
HTTP 方法	GET，HEAD	ALL	ALL
HTTP 标头	NONE	ALL	Host CloudFront-Forwarded-Proto CloudFront-Is-Mobile-Viewer CloudFront-Is-Tablet-Viewer CloudFront-Is-Desktop-Viewer
Cookie	NONE	ALL	comment_* wordpress_* wp-settings-*
查询字符串	是（失效）	是	是

对于默认行为，AWS 建议采用以下配置：

- 允许源协议策略匹配查看者，这样，如果查看者使用 HTTPS 连接到 CloudFront，CloudFront 也会使用 HTTPS 连接到您的源，从而实现端到端加密。请注意，这需要在负载均衡器上安装受信任的 SSL 证书。有关详细信息，请参阅需要 HTTPS 来实现 CloudFront 和自定义源之间的通信。
- 允许所有 HTTP 方法，因为网站的动态部分同时需要 GET 和 POST 请求（例如，支持 POST 用于评论提交表）。
- 仅转发改变 WordPress 输出的 Cookie，例如 >`wordpress_*`、`wp-settings-*` 和 `comment_*`。如果您安装了依赖列表中未包含的其他 Cookie 的插件，则必须扩展该列表。
- 仅转发影响 WordPress 输出的 HTTP 标头，例如 `Host`、`CloudFront-Forwarded-Proto`、`CloudFront-is-Desktop-Viewer`、`CloudFront-is-Mobile-Viewer` 和 `CloudFront-is-Tablet-Viewer`：

  - `Host` 允许多个 WordPress 网站托管在同一源上。
  - `CloudFront-Forwarded-Proto` 允许缓存页面的不同版本，具体取决于通过 HTTP 还是 HTTPS 访问的页面。
  - `CloudFront-is-Desktop-Viewer`、`CloudFront-is-Mobile-Viewer`、`CloudFront-is-Tablet-Viewer` 允许根据最终用户的设备类型自定义主题的输出。
- 转发所有查询字符串以根据其值进行缓存，因为 WordPress 依赖这些值，它们也可以用于使缓存的对象失效。

如果您想在自定义域名（也就是非 `*.cloudfront.net`）下提供网站，请在“分配设置”的备用域名下输入相应的 URI。在这种情况下，您的自定义域名还需要 SSL 证书。可以通过 AWS Certificate Manager 请求 SSL 证书，然后根据 CloudFront 分配配置证书。

现在，可以为动态内容再创建两种缓存行为：一种针对登录页面（路径模式：`wp-login.php`），一种针对管理控制面板（路径模式：`wp-admin/*`）。这两种行为具有完全相同的设置，如下所示：

- 强制执行仅限 HTTPS 的查看者协议策略。
- 允许所有 HTTP 方法。
- 基于所有 HTTP 标头进行缓存。
- 转发所有 Cookie。
- 基于所有查询字符串转发并缓存。

采取这种配置背后的原因是，网站的这一部分是高度个性化的，通常只有少数用户，因此缓存效率不是首要考虑的问题。重点是保持配置简单，将所有 Cookie 和标头都传递到源，确保最大限度兼容任何已安装的插件。

附录 B 中介绍的 AWS for WordPress 插件会自动创建符合上述配置的 CloudFront 分配。

默认情况下，WordPress 将所有内容都本地存储在 Web 服务器上，对于单服务器部署来说，这是数据块存储 (Amazon EBS)，对于弹性部署来说，这是文件存储 (Amazon EFS)。除了降低存储和数据传输成本，将静态资产移到 Amazon S3 上还能提高可扩展性、数据可用性、安全性和性能。有几个插件可以使静态内容移动到 Amazon S3 上变得轻松；其中之一是 W3 Total Cache，附录 B：插件的安装和配置中也对此插件进行了介绍。

附录 B：插件的安装和配置

主题

AWS for WordPress 插件

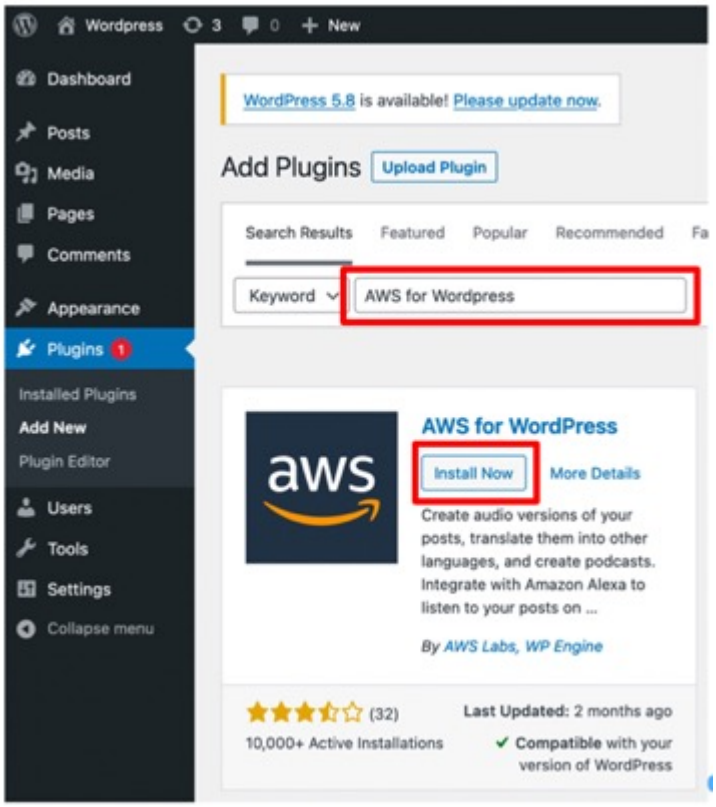
AWS for WordPress 插件是 AWS 编写并积极维护的唯一 WordPress 插件。它允许客户轻松为 WordPress 网站配置 Amazon CloudFront 和 AWS Certificate Manager (ACM)，以增强性能和安全性。该插件使用 Amazon Machine Learning 服务，将内容翻译成一种或多种语言，并可生成每种翻译的音频版本，通过 Amazon Alexa 设备朗读 WordPress 网站。

该插件已安装在 WordPress High Availability by Bitnami on AWS Quick Start 中。

插件的安装和配置

安装插件：

1. 要使用 AWS for WordPress 插件，必须为其创建一个 IAM 用户。IAM 用户是 AWS 账户下有权对 AWS 服务进行 API 调用的人员或应用程序。
2. 您需要一个 AWS Identity and Access Management (IAM) 角色或 IAM 用户来控制您 AWS 账户的身份验证和授权。要防止未经授权的用户获取这些权限，请保护 IAM 用户的凭证。像密码一样对待秘密访问密钥；将其存储在安全的位置，并且不要与任何人共享。像密码一样，定期轮换访问密钥。如果秘密访问密钥意外泄露，应立即将其删除。然后，您可以创建一个新的访问密钥来用于 AWS for WordPress 插件。
3. 在 WordPress 管理面板的插件菜单中，搜索 AWS for WordPress，然后选择立即安装。



4. 如果插件安装未运行，可能是用户权限有问题。连接到 WordPress Web 服务器，完成以下操作来解决问题。

   a. 打开 WordPress 安装目录中的 `wp-config.php` 文件，在 `wp-config.php` 文件末尾写入以下代码：

   ```
   define('FS_METHOD','direct');
   ```

   b. 启动以下命令以授予写入权限：

   ```
   chmod 777 <WordPress install directory>/wp-content
   ```

   警告：保留写入权限为 777 存在风险。如果权限保留为 777，任何人都可以编辑或删除此文件夹。完成插件工作后，将写入权限改为 755 或以下。

   c. 如果使用参考架构，则 WordPress 安装目录为“`/var/www/wordpress/<site directory>`”。

所有 AWS for WordPress 设置的详细说明均不在本文档的讨论范围之内。有关配置和选项，请参阅 AWS for WordPress 插件入门。

Amazon CloudFront 和 AWS Certificate Manager

设置 CloudFront 和 AWS Certificate Manager：

1. 在插件菜单上，选择 CloudFront 并输入以下参数：
   - 源域名：CloudFront 从中获取您网站内容的 HTTP 源服务器的 DNS 域（例如 `example.com`）。
   - 备用域名 (CNAME)：您的访客用于加速网站体验的域名。AWS 建议在域名前面使用“www”（例如 `www.example.com`）。
2. 选择启动设置以启动配置。

该插件会通过 ACM 自动为别名记录请求 SSL 证书，一旦您通过使用别名记录条目更新 DNS 记录来验证 ACM 令牌后，该插件将创建符合附录 A 中定义的最佳实践的 CloudFront 分配。

> **Note**
>
> AWS for WordPress 插件需要 HTTPS 来完成 CloudFront 和您的自定义源之间的通信。确保您的源拥有对源域名有效的 SSL 证书。有关更多信息，请参阅需要 HTTPS 来实现 CloudFront 和自定义源之间的通信。

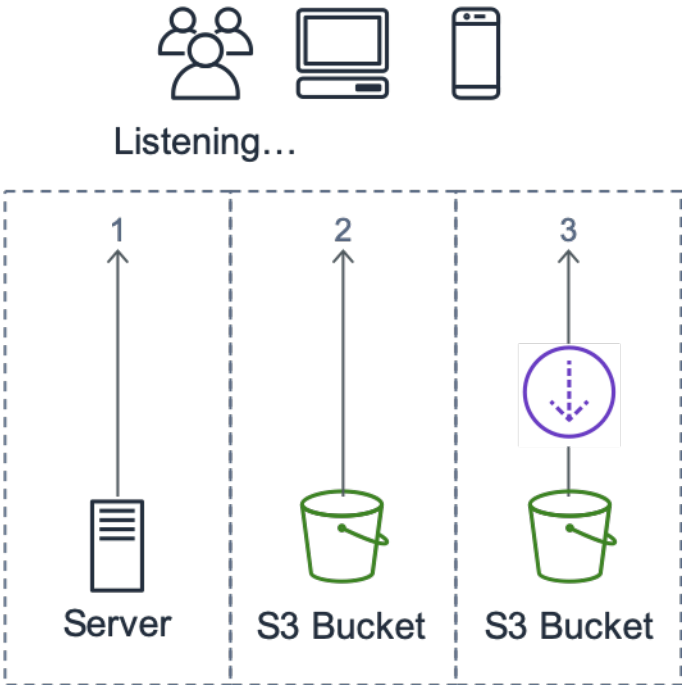
翻译内容并创建音频

您可以使用 AWS for WordPress 插件自动翻译不同语言的文本，并将书面内容转换为多语言音频格式。这些功能由 Amazon Machine Learning 服务提供支持。

Amazon Polly 是将文本转换为逼真语音的服务。从多种语言的几十种语音中，您可以选择中意的语音，构建在许多国家/地区适用的支持逼真语音功能的应用程序。使用此插件可创建 Amazon Polly 支持的任何语音和语言的音频文件。访客可以按需使用内联音频播放器和移动应用程序流式播放音频。

默认情况下，该插件将新的音频文件存储在您的 Web 服务器上。您可以选择将文件存储在 Amazon S3 或 Amazon CloudFront 上。不管音频文件存储在何处，用户都能获得同样的收听体验。仅广播位置发生改变：

- 如果音频文件存储在 WordPress 服务器上，则直接从服务器上广播文件。
- 如果文件存储在 S3 存储桶中，则从此存储桶广播文件。
- 如果您使用 CloudFront 并且文件存储在 Amazon S3 上，则通过 CloudFront 广播。

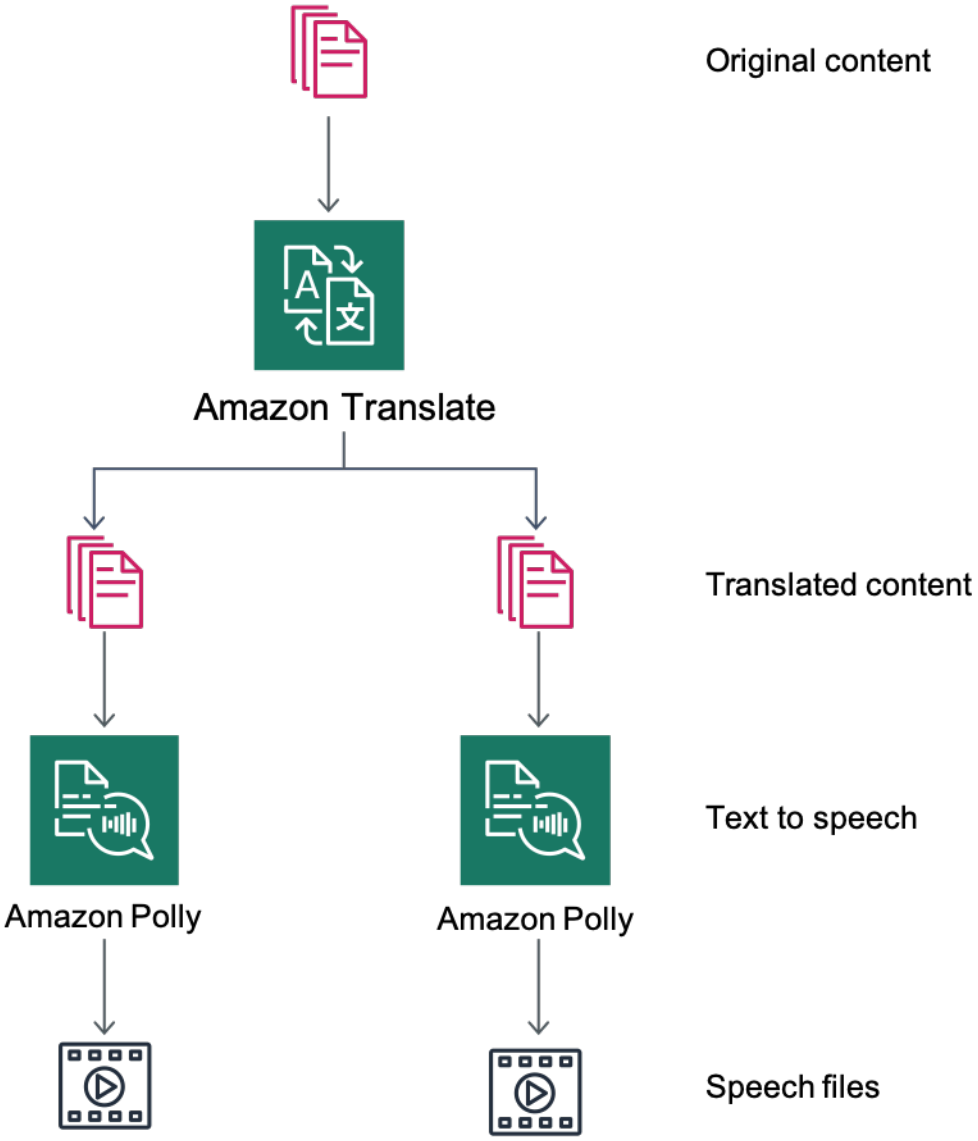


广播位置

Amazon Translate 是一种机器翻译服务，可提供快速、高质量且经济实惠的语言翻译。提供多语言内容给网站拥有者带来了巨大机会。虽然英语是网络上的主要语言，但母语为英语的人仅占网络受众总数的 26%。

以多种语言提供书面和音频版本的 WordPress 内容，可以满足更多国际受众的需求。可以配置此插件来执行以下操作：

- 自动翻译为不同的语言，并在新内容发布时为其每种翻译语言创建录音，或者选择翻译单独的帖子并创建录音
- 翻译成不同的语言，并为存档内容的每种翻译语言创建录音
- 使用 Amazon Pollycast RSS 源广播音频内容



内容翻译和文本转语音概述

使用 Amazon Pollycast 广播

借助 Amazon Pollycast 源，您的访客可以使用标准播客应用程序收听您的音频内容。符合 RSS 2.0 标准的 Pollycast 源提供常用移动播客应用程序（如 iTunes）和播客目录合成播客所需的 XML 数据。

在安装 AWS for WordPress 插件时，您会在播客配置选项卡中找到启用生成 XML 源功能的选项。此外，您还可以找到配置多个可选属性的选项。启用该功能后，您将收到生成源的链接。

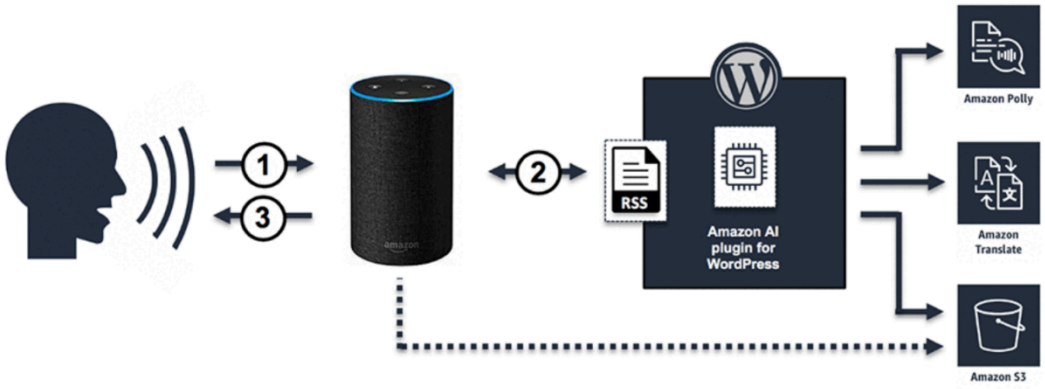
通过 Amazon Alexa 设备朗读您的内容

您可以通过 Alexa 设备扩展 WordPress 网站和博客。这为网站的创建者和作者接触更广泛的受众提供了新的可能性。人们只需要求 Alexa 朗读，就可以更轻松地收听他们喜欢的博客。

要向 Alexa 公开 WordPress 网站，必须启用：

- AWS for WordPress 插件。
- 文本转语音和 Amazon Pollycast 功能。这些功能会在您的 WordPress 网站上生成 RSS 源并供 Amazon Alexa 使用。
- 要使 Amazon S3 作为文本转语音文件的默认存储，重要的是您的网站必须使用安全的 HTTPS 连接向 Alexa 公开其源。

下图显示了通过 Alexa 公开网站所需的交互流程和组件。



通过 Alexa 公开 WordPress 网站所需的交互流程

1. 用户调用一项新的 Alexa 技能，例如说道：Alexa，我需要演示博客的最新更新。 这项技能本身是使用 Alexa Skill Blueprints 之一创建的。它允许您即使不具备深厚的技术知识，也可以通过 Alexa 设备展示您的技能。
2. Alexa 技能会分析呼叫和由 AWS for WordPress 插件生成的 RSS 源，然后返回最新文章的音频版本的链接。
3. 根据源提供的链接，Alexa 通过播放存储在 Amazon S3 上的音频文件来朗读文章。

请参阅 WordPress Marketplace 上的插件页面，获取安装和配置插件及其功能的详细分步指南。

静态内容配置

默认情况下，WordPress 将所有内容都本地存储在 Web 服务器上，对于单服务器部署来说，这是数据块存储 (Amazon EBS)，对于弹性部署来说，这是文件存储 (Amazon EFS)。除了降低存储和数据传输成本，将静态资产移到 Amazon S3 上还能提升可扩展性、数据可用性、安全性和性能。

在本例中，W3 Total Cache (W3TC) 插件用于在 Amazon S3 上存储静态资产。但是，其他一些插件也具有相同的功能。如果您想使用替代方案，则可以相应地调整以下步骤。这些步骤仅涉及与本例相关的功能或设置。所有设置的详细说明均不在本文档的讨论范围之内。有关更多信息，请参阅 wordpress.org 上的 W3 Total Cache 插件页面。

创建 IAM 用户

您需要为 WordPress 插件创建一个 AWS Identity and Access Management (IAM) 用户，以便在 Amazon S3 中存储静态资产。相关步骤，请参阅在您的 AWS 账户中创建 IAM 用户。

注意：虽然 IAM 角色为管理对 AWS 资源的访问提供了一种更好的方式，但是在撰写本白皮书时，W3 Total Cache 插件尚不支持 IAM 角色。

请记下用户安全凭证并妥善保存以供稍后使用。

创建 Amazon S3 存储桶

1. 首先，在您选择的 AWS 区域中创建 Amazon S3 存储桶。有关步骤，请参阅创建存储桶。按照教程：在 Amazon S3 上配置静态网站，为存储桶启用静态网站托管。
2. 创建允许之前创建的 IAM 用户访问指定 S3 存储桶的 IAM 策略，并将该策略附加到该 IAM 用户。有关创建以下策略的步骤，请参阅管理 IAM 策略。

```
{
"Version": "2012-10-17",
"Statement": [
      {
                "Sid": "Stmt1389783689000",
                "Effect": "Allow",
                "Principal": "*",
                "Action": [
                       "s3:DeleteObject",
                       "s3:GetObject",
                       "s3:GetObjectAcl",
                       "s3:ListBucket",
```

```
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::wp-demo",
                "arn:aws:s3:::wp-demo/*"
            ]
        }
    ]
}
```

3. 从 WordPress 管理面板安装并激活 W3TC 插件。
4. 浏览到插件配置的“常规设置”部分，确保浏览器缓存和 CDN 均已启用。
5. 从 CDN 配置的下拉列表中，选择源推送：Amazon CloudFront（此选项将 Amazon S3 作为源）。
6. 浏览到插件配置的“浏览器缓存”部分，启用过期、缓存控制和实体标签 (ETag) 标头。
7. 另外，还需激活防止设置更改后缓存对象选项，以便无论何时更改设置，都会生成新的查询字符串并附加到对象。
8. 浏览至插件配置的“CDN”部分，输入之前创建的 IAM 用户的安全凭证以及 S3 存储桶的名称。
9. 如果您通过 CloudFront URL 提供网站，请在相关框中输入分配域名。否则，为自定义域名输入一个或多个别名记录。
10. 最后，导出媒体库并使用 W3TC 插件将 wp-includes、主题文件和自定义文件上传到 Amazon S3。这些上传功能可在 CDN 配置页面的常规部分找到。

创建静态源

静态文件存储在 Amazon S3 上后，返回到 CloudFront 控制台中的 CloudFront 配置，将 Amazon S3 配置为静态内容的源。为此，需要添加指向为此目的创建的 S3 存储桶的第二个源。然后，再创建两种缓存行为，分别对应两个文件夹（`wp-content` 和 `wp-includes`）之一，这两个文件夹应将 S3 源而非默认源用于动态内容。以同样的方式配置两种行为：

- 仅服务 HTTP GET 请求。
- Amazon S3 不会根据 Cookie 或 HTTP 标头改变其输出，因此，您不用通过 CloudFront 将它们转发到源就可以提高缓存效率。
- 尽管事实上这些行为仅服务静态内容（不接受任何参数），但您仍会将查询字符串转发给源。于是，您可以使用查询字符串作为版本标识符，让旧的 CSS 之类的文件在新版本部署时立即失效。有关更多信息，请参阅 Amazon CloudFront 开发人员指南。

> **Note**
>
> 将静态源行为添加到 CloudFront 分配后，检查顺序，确保 `wp-admin/*` 和 `wp-login.php` 的行为具有比静态内容的行为更高的优先级。否则，当您访问管理面板时，可能会看到奇怪的行为。

附录 C：备份和恢复

与传统托管环境相比，在 AWS 中，故障恢复更快且更容易。例如，您可以在几分钟内启动替换实例来应对硬件故障，或者，在我们的许多托管式服务中利用自动故障转移来消除由于例行性维护导致的重启影响。

但是，为了成功恢复数据，仍需要确保备份了正确的数据。要重新建立 WordPress 网站的可用性，必须能够恢复以下组件：

- 操作系统 (OS) 以及服务安装和配置（Apache、MySQL 等）
- WordPress 应用程序代码和配置
- WordPress 主题和插件
- 上传（例如，帖子的媒体文件）
- 数据库内容（帖子、评论等）

AWS 提供了多种方法来备份和还原 Web 应用程序数据及资产。

本白皮书之前讨论了如何利用 Lightsail 快照来保护存储在实例本地存储中的所有数据。如果您的 WordPress 网站仅在 Lightsail 实例上运行，那么定期拍摄的 Lightsail 快照足够恢复您的整个 WordPress 网站。但是，如果您从某个快照还原，则仍会丢失自上次拍摄快照以来应用于网站的所有更改。

在多服务器部署中，需要使用不同的机制备份前面讨论的每个组件。每个组件可能对备份频率的要求不同，例如，操作系统和 WordPress 的安装和配置会比用户生成的内容的改动频率低很多，因此，备份频率低一点也不会在恢复时丢失数据。

要备份操作系统和服务的安装和配置以及 WordPress 应用程序代码和配置，您可以创建正确配置的 EC2 实例的 AMI。AMI 有两个用途：一是作为实例状态的备份，二是在启动新实例时作为模板。

要备份 WordPress 应用程序代码和配置，需要利用 AMI 和 Aurora 备份。

要备份网站上安装的 WordPress 主题和插件，应备份 Amazon S3 存储桶或存储它们的 Amazon EFS 文件系统。

- 对于存储在 S3 存储桶中的主题和插件，您可以启用跨区域复制，使上传到主存储桶的所有对象都自动复制到另一个 AWS 区域的备份存储桶。跨区域复制要求在源存储桶和目标存储桶上都启用版本控制，这提供了一个额外保护层，并允许您还原到存储桶中任何给定对象的某个之前版本。

- 对于存储在 EFS 文件系统中的主题和插件，您可以创建一个 AWS Data Pipeline，以便将数据从生产 EFS 文件系统复制到另一个 EFS 文件系统，如备份 Amazon EFS 文件系统文档页面中所述。您还可以使用任何熟悉的备份应用程序来备份 EFS 文件系统。
- 要备份用户上传的内容，则应遵循前述的备份 WordPress 主题和插件的步骤。
- 要备份数据库内容，需要使用 Aurora 备份。Aurora 自动备份您的集群卷，并在备份保留期内一直保留还原数据。Aurora 备份是连续且递增的，所以您可以快速还原到备份保留期内的任何时间点。在写入备份数据时，数据库服务的性能不会受影响或中断。您可以指定备份保留期为 1 到 35 天。您还可以创建手动数据库快照，它们会保留到您删除它们为止。手动数据库快照对于长期备份和存档很有用。

附录 D：部署新插件和主题

很少有网站会一直处于静态。大多数情况下，您需要定期添加公开可用的 WordPress 主题和插件，或升级到更新的 WordPress 版本。在其他情况下，您需要从头开始开发自己的自定义主题和插件。

每当您对 WordPress 安装进行结构性改动时，都会存在引入不可预见问题的风险。因此，在应用任何重大更改（例如安装新插件）前，至少应备份应用程序代码、配置和数据库。对于具有商业价值或其他价值的网站，请先在隔离的暂存环境中测试这些更改。借助 AWS，可以轻松复制生产环境的配置，并以安全的方式运行整个部署过程。完成测试后，只需删除测试环境并停止为这些资源付费即可。本白皮书稍后会讨论一些特定于 WordPress 的注意事项。

有些插件将配置信息写入 `wp_options` 数据库表（或引入数据库架构更改），有些则在 WordPress 安装目录中创建配置文件。因为我们已将数据库和存储迁移到共享平台，所以这些更改能立即供您的所有正在运行的实例使用，无需执行更多操作。

在 WordPress 中部署新主题时，可能需要额外执行一些操作。如果您仅使用 Amazon EFS 来存储所有 WordPress 安装文件，那么新主题将立即可用于所有正在运行的实例。但是，如果您将静态内容卸载到 Amazon S3 中，则必须将这些内容的副本置于正确的存储桶位置。像 W3 Total Cache 这样的插件为您提供了一种手动启动该任务的方法。您也可以在构建过程中自动执行此步骤。

由于主题资产可以缓存在 CloudFront 和浏览器中，所以，您需要一种方法来使旧版本在部署更改时失效。实现此目标的最佳方法是在对象中包含某种版本标识符。此标识符可以是带有日期时间戳的查询字符串，也可以是随机字符串。如果您使用 W3 Total Cache 插件，则可以更新附加到媒体文件的 URL 的媒体查询字符串。

声明

客户负责对本文档中的信息进行独立评估判断。本文档：(a) 仅供参考；(b) 代表当前提供的 AWS 产品和实践，如有更改，恕不另行通知；并且 (c) AWS 及其附属机构、供应商或许可方不做任何承诺或保证。AWS 产品或服务“按原样”提供，不提供任何形式的保证、陈述或条件，无论是明示还是暗示。AWS 对其客户的责任和义务由 AWS 协议决定，本文档与 AWS 和客户之间签订的任何协议无关，亦不影响任何此类协议。