



优化 Microsoft 工作负载的成本 AWS

AWS 规范性指导



AWS 规范性指导: 优化 Microsoft 工作负载的成本 AWS

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商标和商业外观不得用于任何非 Amazon 的商品或服务，也不得以任何可能引起客户混淆、贬低或诋毁 Amazon 的方式使用。所有非 Amazon 拥有的其他商标均为各自所有者的财产，这些所有者可能附属于 Amazon、与 Amazon 有关联或由 Amazon 赞助，也可能不是如此。

Table of Contents

简介	1
概述	1
受众	1
如何使用本指南	1
目标业务成果	3
成本优化之旅	4
优化成本的首要建议	6
概览	6
首要建议	6
AWS 优化与许可评测	8
概览	8
评测选项	8
全面评测	9
确定工作负载范围	9
收集数据	9
分析数据	10
计划后续步骤	12
评测影响	13
后续步骤	14
其他资源	14
Amazon EC2 上的 Windows	15
自动执行停止和启动计划	15
概述	15
案例研究	16
成本优化方案	17
成本优化建议	18
其他资源	29
调整 Windows 工作负载	29
概述	29
成本优化方案	30
成本优化建议	30
建议	38
其他资源	38
为 Windows 工作负载选择合适的实例类型	39

概述	39
成本优化建议	39
后续步骤	47
其他资源	48
带来适用于 Windows 和 SQL Server 工作负载的许可证	49
概述	49
Amazon EC2 专属主机	49
AWS 许可选项	52
自带 Windows Server 许可证	53
成本优化方案	54
成本优化建议	59
其他资源	59
优化 Amazon EC2 上的 Windows 的支出	60
概述	60
了解节省计划	60
成本优化方案	65
成本优化建议	68
其他资源	70
使用 AWS 工具监控成本	70
概述	70
成本优化建议	70
其他资源	73
SQL Server	75
选择高可用性和灾难恢复解决方案	76
概述	76
SQL Server Always On 可用性组	77
SQL Server Always On 失效转移群集实例	78
SIOS DataKeeper	80
Always On 可用性组	82
分布式可用性组	83
日志传送	84
AWS Database Migration Service	86
AWS Elastic Disaster Recovery	86
成本比较	87
成本优化建议	90
其他资源	91

了解 SQL Server 许可	91
概述	91
AWS 许可选项	91
自带许可证对成本的影响	92
许可证优化	93
成本优化建议	93
其他资源	38
为 SQL Server 工作负载选择合适的 EC2 实例。	99
概述	99
成本比较	99
成本优化方案	100
成本优化建议	101
其他资源	105
整合实例	105
概述	105
成本优化方案	106
成本优化建议	107
其他资源	108
比较 SQL Server 版本	108
概述	108
成本影响	109
成本优化建议	110
其他资源	115
评估 SQL Server 开发人员版	115
概述	115
成本影响	115
其他资源	38
评估 Linux 上的 SQL Server	118
概述	118
成本影响	119
成本优化建议	120
其他资源	121
优化 SQL Server 备份策略	121
概述	121
使用启用 VSS 的快照进行服务器级备份	122
使用 SQL 服务器备份 AWS Backup	124

数据库级备份	126
成本优化建议	134
其他资源	136
现代化 SQL Server 数据库	136
概述	136
数据库产品	137
Amazon RDS 和 Aurora 的比较	137
成本优化建议	139
其他资源	143
优化 SQL Server 的存储	143
概述	143
Amazon EBS 的 SSD 存储类型、性能和成本	144
Amazon EBS 的通用 SSD 成本优化	145
其他资源	147
使用 Compute Optimizer 优化 SQL Server 许可	147
概述	147
成本优化建议	148
配置 Compute Optimizer	148
其他资源	149
使用 Compute Optimizer 优化 SQL Server 的大小	150
概述	150
配置 Compute Optimizer	150
其他资源	151
查看 SQL Server 工作负载 Trusted Advisor 的建议	151
概述	151
成本优化建议	151
配置 Trusted Advisor	152
其他资源	153
容器	154
将 Windows 应用程序移到容器	154
概述	154
成本优势	155
成本优化建议	156
后续步骤	159
其他资源	159
优化 Amazon ECS 上 AWS Fargate 任务的成本	159

概述	159
成本优势	160
成本优化建议	160
后续步骤	165
其他资源	165
了解您的 Amazon EKS 成本	166
概述	166
成本优势	166
成本优化建议	166
后续步骤	170
其他资源	170
使用 App2Container 更换 Windows 应用程序平台	170
概述	170
成本优势	171
成本优化建议	171
后续步骤	171
其他资源	172
仓储服务	173
Amazon EBS	173
将 Amazon EBS 卷从 gp2 迁移到 gp3	173
修改 Amazon EBS 快照	177
删除未附加的 Amazon EBS 卷	179
Amazon FSx	182
选择合适的 SMB 文件存储	182
在 Amazon 中启用重复数据删除 FSx	186
了解 Windows 文件服务器中的 FSx 数据分片	188
了解 Amazon 中的硬盘容量使用情况 FSx	192
使用单可用区	194
AWS Storage Gateway	196
Amazon S3 文件网关	196
Amazon FSx 文件网关	196
成本影响	197
成本优化建议	199
其他资源	201
Active Directory	202
Amazon EC2 上的自行管理的 Active Directory	202

概述	202
成本影响	202
成本优化建议	203
其他资源	207
AWS Managed Microsoft AD	207
概述	207
成本影响	207
成本优化建议	207
其他资源	208
AD Connector	209
概述	209
成本影响	209
成本优化建议	209
其他资源	210
.NET	211
重构为现代的 .NET，并迁移到 Linux 上	212
概述	212
成本影响	212
成本优化建议	213
其他注意事项和资源	214
容器化 .NET 应用程序	214
概述	214
成本影响	214
成本优化建议	216
其他资源	218
使用 Graviton 实例和容器	218
概述	218
成本影响	218
成本优化建议	220
其他资源	221
支持对静态 .NET Framework 应用程序进行动态扩展	221
概述	221
成本影响	226
成本优化建议	227
其他资源	228
利用缓存来降低数据库需求	228

概述	228
成本影响	228
成本优化建议	229
其他资源	234
考虑使用无服务器 .NET	235
概述	235
成本影响	235
成本优化建议	235
其他资源	238
考虑使用专用数据库	239
概述	239
成本影响	241
成本优化建议	243
其他资源	245
后续步骤	246
文档历史记录	247
术语表	248
#	248
A	248
B	251
C	252
D	255
E	258
F	260
G	261
H	262
我	263
L	265
M	266
O	270
P	272
Q	274
R	275
S	277
T	280
U	281

V	282
W	282
Z	283
.....	cclxxxiv

优化微软工作负载的成本 AWS

Bill Pfeiffer、Chase Lindeman 和 Kevin Sookhan , Amazon Web Services (AWS)

2025 年 10 月 ([文档历史记录](#))

概述

本指南提供了建议、最佳实践和策略，以帮助您优化 Microsoft 工作负载的成本 AWS。该指南还包括基础的 AWS 知识、成本优化技术以及参考架构，以帮助您构建和自动执行经济高效、高性能的工作负载，从而满足您的业务目标。本指南统称为 Microsoft on AWS Cost Optimization (MACO)。MACO 指南由行业专家基于实际场景而制定。

本指南涵盖了以下 Microsoft 工作负载：

- 亚马逊弹性计算云 (亚马逊 EC2) 上的 Windows
- SQL Server
- 容器
- 存储
- Active Directory
- .NET

受众

本指南适用于架构师、工程师、管理员 CTOs、主管、技术决策者和 AWS 合作伙伴。有 AWS 计费、Microsoft 技术和 AWS 系统管理的经验和基本了解很有帮助，但不是必需的。

如何使用本指南

您可以依据本指南来规划并实施您的 MACO 云迁移之旅。我们建议你从头到尾阅读本指南，以全面了解优化 Microsoft 工作负载成本的选项和方法 AWS。您可以根据您所在组织的需求来查看以下工作负载部分：

- [亚马逊上的 Windows EC2](#)
- [SQL Server](#)

- [容器](#)
- [存储](#)
- [Active Directory](#)
- [.NET](#)

 Important

本指南中提供的代码示例仅用于演示目的。在将所有代码应用于生产环境之前，先在开发环境中对其进行测试是一种最佳实践。在您实施代码之前，我们建议您先分小批次对代码进行测试，然后通过使用 [AWS Cost Explorer](#) 来审查该代码所导致的成本变化情况。这有助于您排查一些边缘情况以及其他可能会在日后引发问题的问题。

 Important

本指南中的定价示例基于发布价格。价格可能会发生变化。此外，您的费用可能会有所不同 AWS 区域，具体取决于您的 AWS 服务 配额以及与您的云环境相关的其他因素。

目标业务成果

本指南可以帮助您和您的组织实现以下业务目标：

- 了解如何使用 AWS 优化和许可评测（AWS OLA）根据资源利用率、第三方许可和应用程序依赖关系来评测和优化当前的本地和云环境。
- 利用适用于 Microsoft 工作负载的 AWS Modernization Calculator 制定成本优化的业务案例。
- 优化特定 Microsoft 工作负载的成本，包括 Amazon Elastic Compute Cloud（Amazon EC2）、SQL Server、容器、存储、Active Directory 和 .NET 上的 Windows 工作负载。

成本优化之旅

云迁移之旅的范围、时间和具体路径取决于您的业务目标、技术要求及其他因素。本节提供了一个云迁移之旅的示例，重点是[使用 AWS 进行云财务管理](#)，并遵循 MACO 建议和最佳实践。您可以使用此示例来了解如何为 Microsoft 工作负载设计云迁移之旅。

以下高级任务说明了组织在实施 MACO 建议和最佳实践时可以采用的方法：

- 建立标记策略并启用用户定义的成本分配标签。有关更多信息，请参阅 AWS 白皮书 [Best Practices for Tagging AWS Resources](#)。
- 根据应用程序、团队或部门定义预算。有关更多信息，请参阅《AWS 账单与成本管理用户指南》中的 [Managing your costs with AWS Budgets](#)。
- 执行 AWS 优化与许可评测 (AWS OLA) 以加快节省开支。有关更多信息，请参阅 AWS 文档中的 [AWS 优化与许可评测](#)。
- 使用 Amazon Elastic Compute Cloud Dedicated Hosts 自带许可 (BYOL) 运行 Windows 和 SQL Server 工作负载。有关更多信息，请参见本指南的 [带来适用于 Windows 和 SQL Server 工作负载的许可证](#) 部分。
- 在 AWS 上优化您的 SQL Server 许可。有关更多信息，请参见本指南的 [了解 SQL Server 许可](#) 部分。
- 为 Windows 工作负载选择合适的实例类型。有关更多信息，请参见本指南的 [为 Windows 工作负载选择合适的实例类型](#) 部分。
- 为 SQL 工作负载选择合适的实例类型。有关更多信息，请参见本指南的 [为 SQL Server 工作负载选择合适的 EC2 实例](#) 部分。
- 将 Amazon Elastic Block Store (Amazon EBS) 从 gp2 迁移到 gp3。有关更多信息，请参见本指南的 [将 Amazon EBS 卷从 gp2 迁移到 gp3](#) 部分。
- 使用 AWS 上的 EC2 实例调度器控制工作负载。有关更多信息，请参见本指南的 [自动执行停止和启动计划](#) 部分。
- 使用 SQL Server 开发者版来消除非生产工作负载的 SQL Server 成本。有关更多信息，请参见本指南的 [评估 SQL Server 开发人员版](#) 部分。
- 使用适用于 Windows File Server 的 Amazon FSx 的单个可用区来处理开发和测试工作负载。有关更多信息，请参见本指南的 [使用单可用区](#) 部分。
- 使用 AWS Compute Optimizer 来调整您的 Windows 工作负载。有关更多信息，请参见本指南的 [调整 Windows 工作负载](#) 部分。

- 使用节省计划优化 Windows on Amazon EC2 的支出。有关更多信息，请参见本指南的 [优化 Amazon EC2 上的 Windows 的支出](#) 部分。
- 对适用于 Windows File Server 的 FSx 启用重复数据删除 有关更多信息，请参见本指南的 [在 Amazon 中启用重复数据删除 FSx](#) 部分。
- 对适用于 Windows File Server 的 FSx 上的文件系统使用数据分片。有关更多信息，请参见本指南的 [了解 Windows 文件服务器中的 FSx 数据分片](#) 部分。
- 优化您的 SQL Server 备份策略。有关更多信息，请参见本指南的 [优化 SQL Server 备份策略](#) 部分。
- 使静态 .NET 框架应用程序支持动态扩缩。有关更多信息，请参阅本指南的[支持对静态 .NET Framework 应用程序进行动态扩展](#)部分。
- 使用无服务器 .NET 微服务。有关更多信息，请参见本指南的 [考虑使用无服务器 .NET](#) 部分。
- 将您的 Windows 应用程序移到容器。有关更多信息，请参见本指南的 [容器化 .NET 应用程序](#) 部分。
- 使用 [AWS Compute Optimizer](#) 调整在 AWS Fargate 上运行的 Windows 容器，以用于 Amazon Elastic Container Service (Amazon ECS)。有关更多信息，请参见本指南的 [启用 Compute Optimizer](#) 部分。
- 重构为现代的 .NET，并迁移到 Linux。有关更多信息，请参见本指南的 [重构为现代的 .NET，并迁移到 Linux 上](#) 部分。
- 利用 Graviton 实例和容器。有关更多信息，请参见本指南的 [使用 Graviton 实例和容器](#) 部分。
- 现代化 SQL Server 数据库。有关更多信息，请参见本指南的 [现代化 SQL Server 数据库](#) 部分。
- 设计 Active Directory 基础设施。有关更多信息，请参见本指南的 [Active Directory](#) 部分。

有关以 AWS 云财务管理为重点的客户之旅的更多信息，请参阅 AWS 白皮书[云财务管理功能](#)。

优化成本的首要建议

概览

成本优化是 [AWS Well-Architected Framework](#) 的支柱之一，它在您的云迁移计划中起着至关重要的作用。在本指南中，您会发现有关成本优化的各种建议，但本部分重点介绍了最具影响力的建议。您可以迅速实施这些建议，并且它们会对您的组织产生重大影响。这些建议能够帮助为您的整体成本优化工作奠定基础。

首要建议

以下表格列出了最具影响力的成本优化方面的首要建议。“难以实施”这一列会根据从最容易实施（1）到最难实施（5）的量表来对每项优化措施进行评级。“预计节省费用”这一列显示的是基于百分比估算得出的您的组织在每项建议的优化措施上能够节省的金额。

优化	实施难度	预估节省
调整 Windows 工作负载	3	25%
带来适用于 Windows 和 SQL Server 工作负载的许可证	3	30%
评估 SQL Server 开发人员版	2	20%
了解 SQL Server 许可	2	最高 50%
自动执行停止和启动计划	3	最高 40%
为 Windows 工作负载选择合适的实例类型	1	10–30%
重构为现代的 .NET，并迁移到 Linux 上	5	10–20%
优化 Amazon EC2 上的 Windows 的支出	3	最高 20–40%

优化	实施难度	预估节省
将 Amazon EBS 卷从 gp2 迁移到 gp3	4	最高 20%

 Important

上述表格中所预估的节省金额是针对各个技术领域而言的，并非指一个账户内的总体 AWS 支出。例如，您可以针对各种不同的环境类型和规模来实施实例调度器，这将会影响潜在的节省效果。这些估算值专门适用于 Amazon EC2 实例成本，并不意味着其他 AWS 服务可以节省任何总体成本。这些估算值仅作为衡量标准提供，而不是保证。

MACO 的专家们可以就成本优化问题进行更深入的探讨。若要安排一次深入探讨您的使用案例的会议，请联系您的账户团队或发送电子邮件至 optimize-microsoft@amazon.com。

AWS 优化与许可评测

概览

[AWS 优化与许可评测 \(AWS OLA\)](#) 可以帮助您根据资源利用率、第三方许可和应用程序依赖关系来评测和优化您当前的本地和现有云环境。您可以使用 AWS OLA 帮助您的组织构建迁移和许可策略，以在迁移到 AWS 或评测 AWS 上现有的 Microsoft 工作负载时节省成本。AWS OLA 还可以帮助您实现以下目标：

- 了解现有部署、应用程序性能和合同。
- 调整您的资源。
- 制定 AWS 云的路线图。
- 通过使用现有投资且仅按实际用量付费，从而降低或消除成本。

我们建议您将 AWS OLA 作为[成本优化之旅](#)的第一步。您可以与 AWS Partner Network 合作完成 AWS OLA。他们将帮助您收集评测数据，并为您提供优化许可和实例成本的建议。

下图提供了评测过程的概述。



评测选项

您可以为 AWS 上的 Microsoft 工作负载选择以下两个 AWS OLA 选项：

- **Lite 版本**：在此使用案例中，您的所有工作负载都在 VMware 上。您可以为 AWS 提供 [RVTools](#) 的输出。然后，AWS 可以提供 1-5 天的周转时间。此方法使用直接从 VMware vCenter 拉取的时间点信息，来制定规模建议并提供按需定价选项。

- 完整版本：在此使用案例中，您拥有一个混合环境，该环境运行在不同的云提供商、物理服务器和虚拟服务器中。AWS 使用操作系统代理来收集 14 到 30 天的使用情况数据。这使得 AWS 能够根据您的应用程序使用模式做出明智的实例大小调整决策。AWS 使用 Cloudamize 等多个第三方工具来完成分析。AWS 与其 AWS Partner Network 合作，通过多个定价选项（考虑定价模型和不同架构）来帮助提供最终总拥有成本（TCO）评测。

全面评测

完整的 AWS OLA 评测首先会进行一小时的电话沟通。在这次通话中，AWS 将帮助您确定支持迁移的最佳 AWS 基础设施，选择数据收集方法，并制定完成时间线。在组织中实施发现工具取决于数据收集方法、组织规模以及组织用于管理其服务器群的工具。收集使用情况数据通常需要两周时间。

完整的 AWS OLA 流程需要 30-45 天，包括以下几个阶段：

- 确定工作负载范围
- 收集数据
- 分析数据
- 计划后续步骤

确定工作负载范围

首先，AWS 与您和您的团队合作确定评测的范围。这通常按环境类型（例如，非生产环境 and 生产环境）进行细分。范围包括工作负载的位置。这可能是您要迁移到 AWS 的工作负载、已经在 AWS 上运行的工作负载（例如适用于 Amazon EC2 的 AWS OLA），或者是在其他云提供商中运行的工作负载。

收集数据

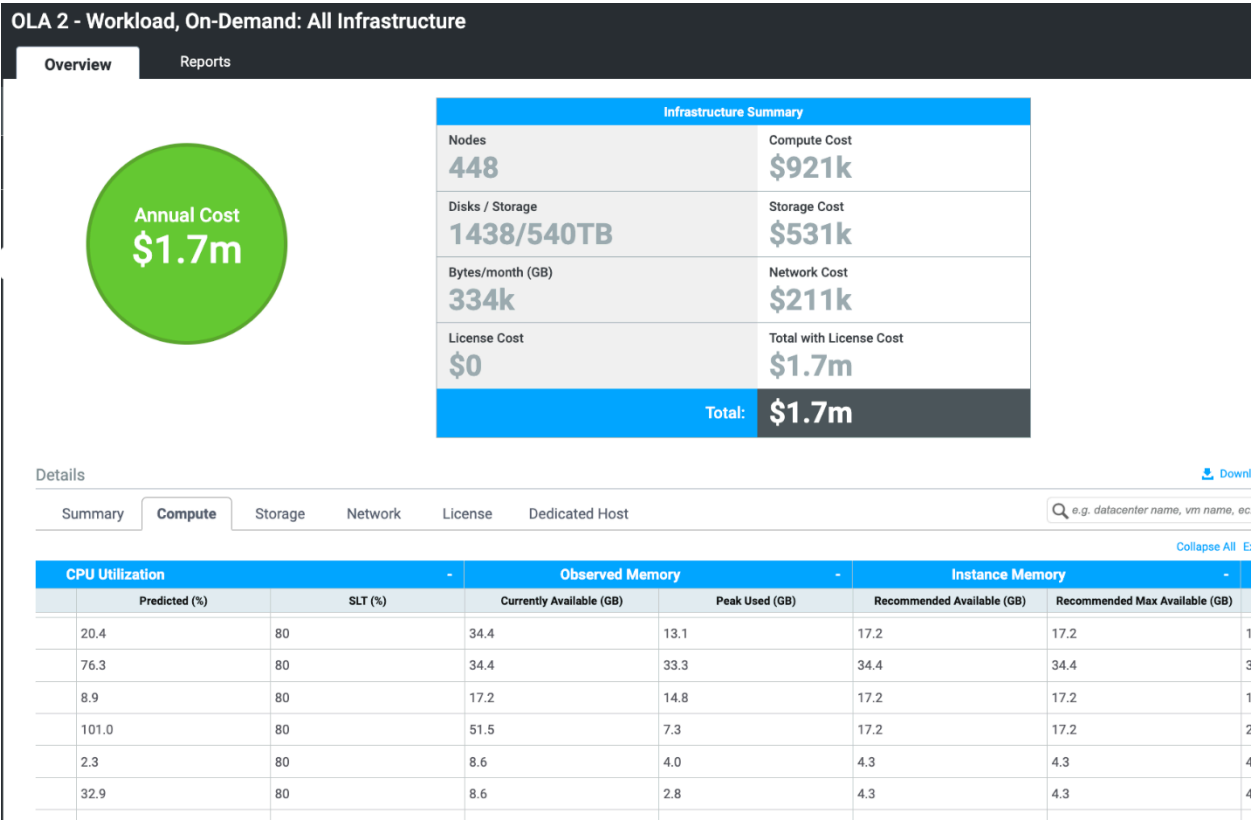
接下来，AWS 部署工具来帮助发现资源并从服务器收集性能数据。此工具有四个部署选项：

- 可以查询虚拟机监控程序的工具（仅需要 VMware vCenter 或 Hyper-V 凭证）
- 可以部署在物理机或虚拟机上的代理
- 根据您的环境和操作系统，使用 SSH、Windows 远程管理（WinRM）或 Windows 管理工具（WMI）进行无代理发现
- 平面文件数据收集和分析

对于您的工具部署，您可以混合搭配各个选项并整合结果。确保无论您选择哪个选项，都必须确保不会给您的 IT 资源造成过大负担，这一点至关重要。AWS 致力于让评测流程尽可能简单便捷。除了简短的电话协助设置外，AWS OLA 团队和 Microsoft 专业解决方案架构师还将准备总拥有成本（TCO）分析和建议以供审查。

分析 CPU 利用率、RAM 利用率、存储吞吐量、IOPS 和网络吞吐量时，数据收集通常需要两到三周的时间。理想情况下，此收集发生在工作月的高峰时段（例如，在月末财务报告期间）。AWS 希望捕捉峰值使用情况，因为这可以提供良好的统计样本，从而确定合适的 AWS 实例大小，同时确保其性能优于本地可用水平。AWS 会将利用率指标与各种处理器世代的性能启发式方法相结合，以精确确定给定工作负载所需的 CPU 和 RAM 数量。这些目标通常低于本地分配的目标。这不仅可以降低实例大小的计算成本，还可以优化许可成本。

以下控制面板视图显示了可通过评测捕获的基础设施成本示例。

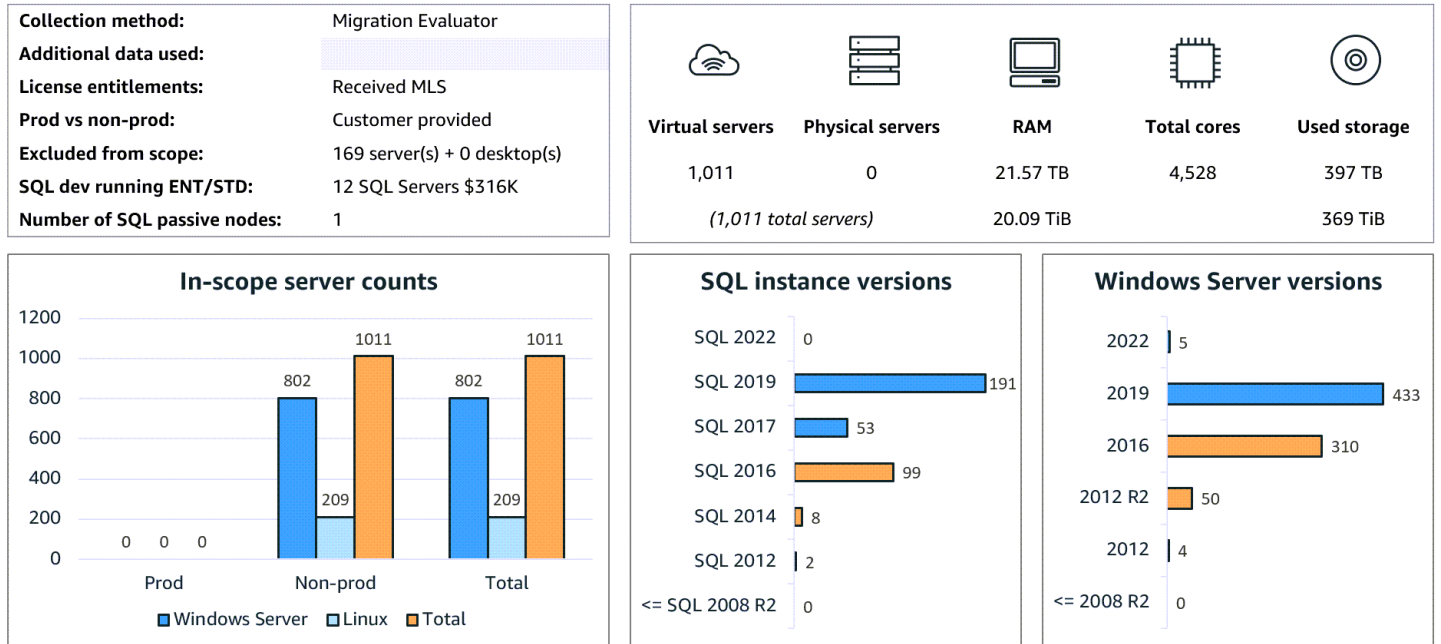


分析数据

AWS 在数据收集完成后提供汇报演示文稿。AWS 审查数据，总结调查发现，然后就本地使用和云迁移提出建议。您可以通过研究整合机会、弹性增益（可以关闭或按季节调整工作负载）、合适的 SKU 机会（例如，正在使用 SQL Server 企业版，但资源要求和功能使用情况表明 SQL Server 标准版更合适）来降低计算和许可成本。对于像 SQL Server 这样按核心数量许可的产品，将工作负载放在更昂

贵的计算实例中通常在经济上是合理的。也就是说，如果 CPU 配置和 RAM 与 vCPU 的比例能够有效减少许可核心的数量，无论是对于包含许可证的使用案例还是自带许可 (BYOL) 的使用案例，都是如此。

下面展示的是基于评测收集的数据的示例分析。



计划后续步骤

最后，AWS 使用收集的性能数据来估算特定的工作负载规模和成本。AWS 还可以汇总查看您的范围环境，并提供定量分析。这可以帮助您确定最佳选项是本地更新还是迁移到 AWS。您可以使用 AWS OLA 末尾提供的 TCO 分析摘要（如以下示例所示）来构建云经济业务案例。

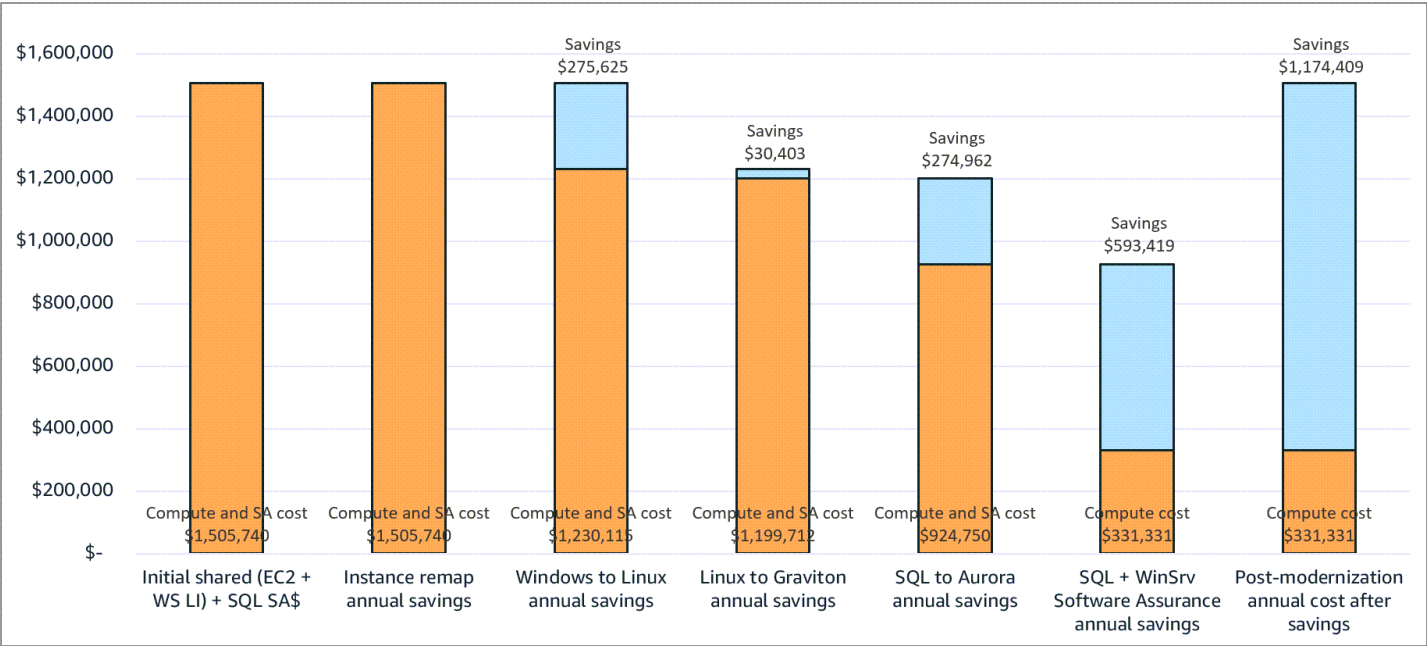
	Option 1: Amazon EC2 shared	Option 1a: Amazon EC2 shared + power management	Option 2: Amazon EC2 mixed	Option 2a: Amazon EC2 mixed + power management
Option details: compute	100% Reserved Instances (RIs)	RIs + on-demand power management	100% RIs	RIs + on-demand power management
Option details: Microsoft licenses	WS LI and SQL BYOL	WS LI and SQL BYOL	WS BYOL or LI+SQL BYOL	WS BYOL or LI+SQL BYOL
Compute costs ¹				
Year 1 compute cost	\$414,546	\$482,623	\$504,019	\$513,941
Year 1 vendor license included cost	\$392,858	\$244,415	\$9,804	\$4,783
	\$807,404	\$727,038	\$513,823	\$518,724
Total compute savings in year 1, compared to Option 1	—	10% (\$80,366)	36% (\$293,581)	36% (\$288,680)
Storage and networking costs ²				
Annual estimated storage cost	\$336,494	\$336,494	\$336,494	\$336,494
Annual estimated networking cost	\$41,455	\$41,455	\$41,455	\$41,455
	\$377,949	\$377,949	\$377,949	\$377,949
Microsoft license costs**				
WS/CIS annual Software Assurance (SA) + current SPLA/Subs cost	\$0	\$0	\$0	\$0
WS/CIS license + SA + SPLA/Subs true-up cost	\$0	\$0	\$0	\$0
SQL annual SA + current SPLA/Subs cost	\$0	\$0	\$0	\$0
SQL license SA + current SPLA/Subs true-up cost	\$0	\$0	\$0	\$0
	\$0	\$0	\$0	\$0
Total estimated costs	\$1,185,353	\$1,104,987	\$891,772	\$896,673
Annual TCO savings in year 1, compared to Option 1	—	7% (\$80,366)	25% (\$293,581)	24% (\$288,680)

¹ Pricing model used: 3-year, no upfront RI
² Software Assurance and true-up costs provided by Microsoft

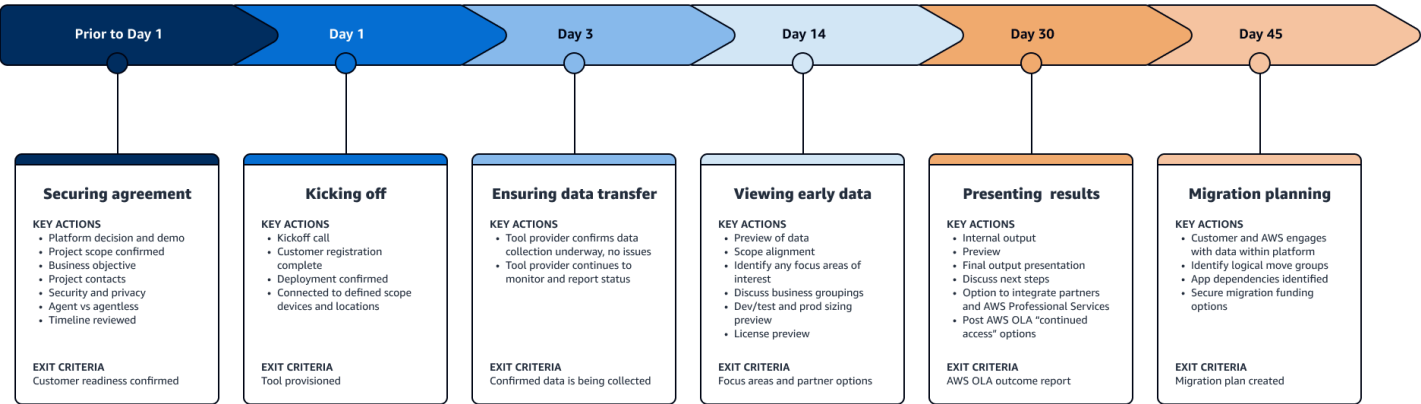
AWS OLA 还通过提出以下建议，深入了解现代化可能对现有工作负载产生的影响：

- 迁移到 Linux 操作系统。
- 添加对 ARM 处理器（AWS Graviton）的应用程序支持。
- 将 SQL Server 工作负载转移到 Amazon Aurora。
- 通过将 Windows 和 SQL Server 工作负载迁移到开源技术来消除软件保障。

下图显示了通过现代化技术可以节省的成本，例如从 Windows 迁移到 Linux 或从 SQL Server 迁移到 Aurora。



完整的 AWS OLA 流程从开始到结束大约需要 45 天。下图显示了一个时间线示例。



如果您有一个纯粹的 VMware 环境且可以提供 RVTtools 的输出，则可以将此时间线缩短到一个工作周。此外，AWS 还可以分析包含资产和利用率数据的平面文件，例如 CPU 平均值、CPU 峰值、RAM 平均值和 RAM 峰值。

评测影响

通过优化规模，普通客户通常可以节省 20-30% 的成本。根据使用情况数据，调整可将源工作负载与大小最佳的 AWS 实例相匹配。这些优化规模的调整不仅可以降低 AWS 环境的每月成本，而且通常还能为组织的其他方面节省开支。例如，Windows 或 SQL Server 许可收入增加 20% 至 30%，可以减少下次与 Microsoft 的结算费用，或者释放许可收入用于其他业务线应用程序。通过整合和调整 SQL Server 工作负载，通常可以实现最显著的经济效益。

AWS 可以帮助您将系统归类为现代化存储桶。有些系统是遗留系统，进行改造在经济上不可行；而另一些系统则可以现代化改造为容器或无服务器应用程序，从而实现最显著的成本节省。与 AWS 团队的对话将从云计算的通用功能等宽泛话题，逐步深入到如何以及为何要对特定工作负载进行现代化改造等更具体的讨论。AWS 还能帮助您探索潜在的创新机会。

后续步骤

如果您正着手优化运行在本地环境或 AWS 上的 Microsoft 工作负载的成本，请联系您的 AWS 客户团队并申请 AWS OLA。AWS 团队成员可以解答您的疑问，并帮助您决定 AWS OLA 是否最终适合您和您的组织。或者，您可以[在线申请 AWS OLA](#)。

其他资源

- [AWS 优化与许可评测](#) (AWS 文档)
- [AWS re:Invent 2022 - How to save costs and optimize Microsoft workloads on \(AWSSENT205 \)](#) (YouTube)

Amazon EC2 上的 Windows

[Amazon Elastic Compute Cloud \(Amazon EC2 \)](#) 是一款高度灵活且可扩展的云计算平台，非常适合运行 Windows 工作负载。您可以使用 Amazon EC2 在 AWS 云的安全、可靠、高度可用且可调整的基础设施上部署、管理和扩展您的 Windows Server 工作负载。考虑一下在 Amazon EC2 上运行 Windows 工作负载的以下主要益处：

- 可扩展性：Amazon EC2 使您能够轻松扩展 Windows 工作负载以适应不断变化的要求。您可以迅速创建新的 EC2 实例以处理增加的需求，同时也能在不再需要这些实例时轻松地将其终止。您仅需为实际使用的资源付费。
- 灵活性：Amazon EC2 上的 Windows 支持多种不同的实例类型，这些类型旨在迎合各种工作负载的要求，包括通用型实例到内存优化型实例或计算优化型实例等。这种灵活性使得您可以根据您特定的基于 Windows 的应用程序选择最合适的实例类型，从而实现性能的最大化和成本的最小化。
- 安全- AWS 为您的 Windows 工作负载提供多层安全保护，包括网络防火墙、数据加密和安全访问控制。这意味着您可以放心，您的应用程序和数据都得到了保护，同时您仍能完全控制自身的安全设置和配置。
- 成本效益 — 一定 pay-as-you-go 价模式使您只需为所使用的资源付费，无需对硬件和软件进行前期投资。该模型还能帮助您优化成本、减少资本支出和提高运营效率。对于各种规模的企业来说，它都是理想的定价模式。

指南的本部分将介绍以下主题：

- [自动执行停止和启动计划](#)
- [调整 Windows 工作负载](#)
- [为 Windows 工作负载选择合适的实例类型](#)
- [带来适用于 Windows 和 SQL Server 工作负载的许可证](#)
- [优化 Amazon EC2 上的 Windows 的支出](#)
- [使用 AWS 工具监控成本](#)

自动执行停止和启动计划

概述

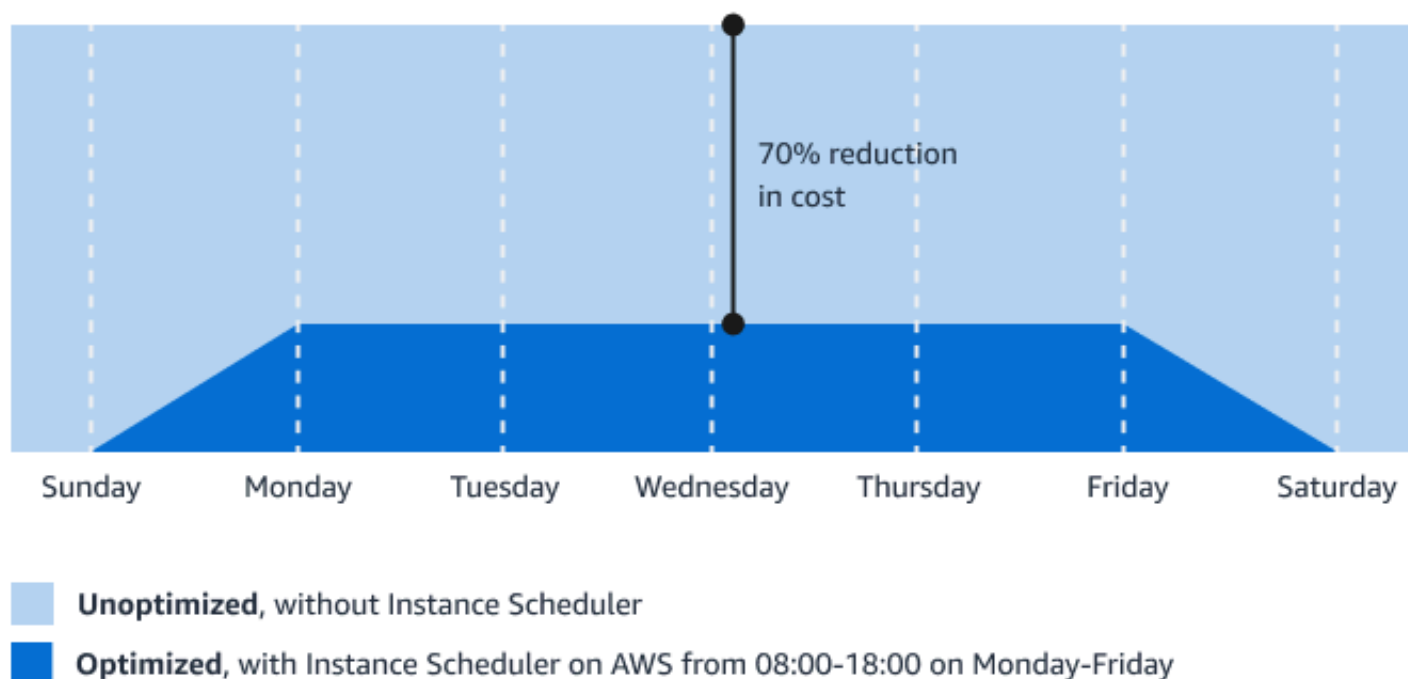
[AWS 上的实例调度器](#) 能够通过自动控制 [Amazon EC2](#) 和 [Amazon Relational Database Service \(Amazon RDS \)](#) 实例的启动与停止，帮助您降低运营成本。如果您让所有实例一直以满负荷

状态运行，则可能会为那些实际上并未被使用的资源支付费用。开启的实例调度器 AWS 使您能够在不需要实例的时间关闭实例，例如在非工作时间、周末或其他使用率较低的时段。这能够随着时间的推移带来显著的成本节省。

上的 Instance Scheduler AWS 还提供跨账户实例调度、自动标记以及使用命令行界面或维护窗口配置计划或周期的功能。[AWS Systems Manager](#) 这些功能可以帮助您更有效地管理实例，并准确地在不同的项目或团队之间跟踪和分配成本。

案例研究

以一家公司为例，该公司在生产环境 AWS 中使用实例调度器在每天工作时间以外自动停止实例。如果该公司让其所有实例都保持满负荷运行状态，则对于那些仅在正常工作时间内才需要使用的实例，他们能够最高实现 70% 的成本节省。以下图表展示了每周的使用时长是如何从 168 小时减少到 50 小时的。

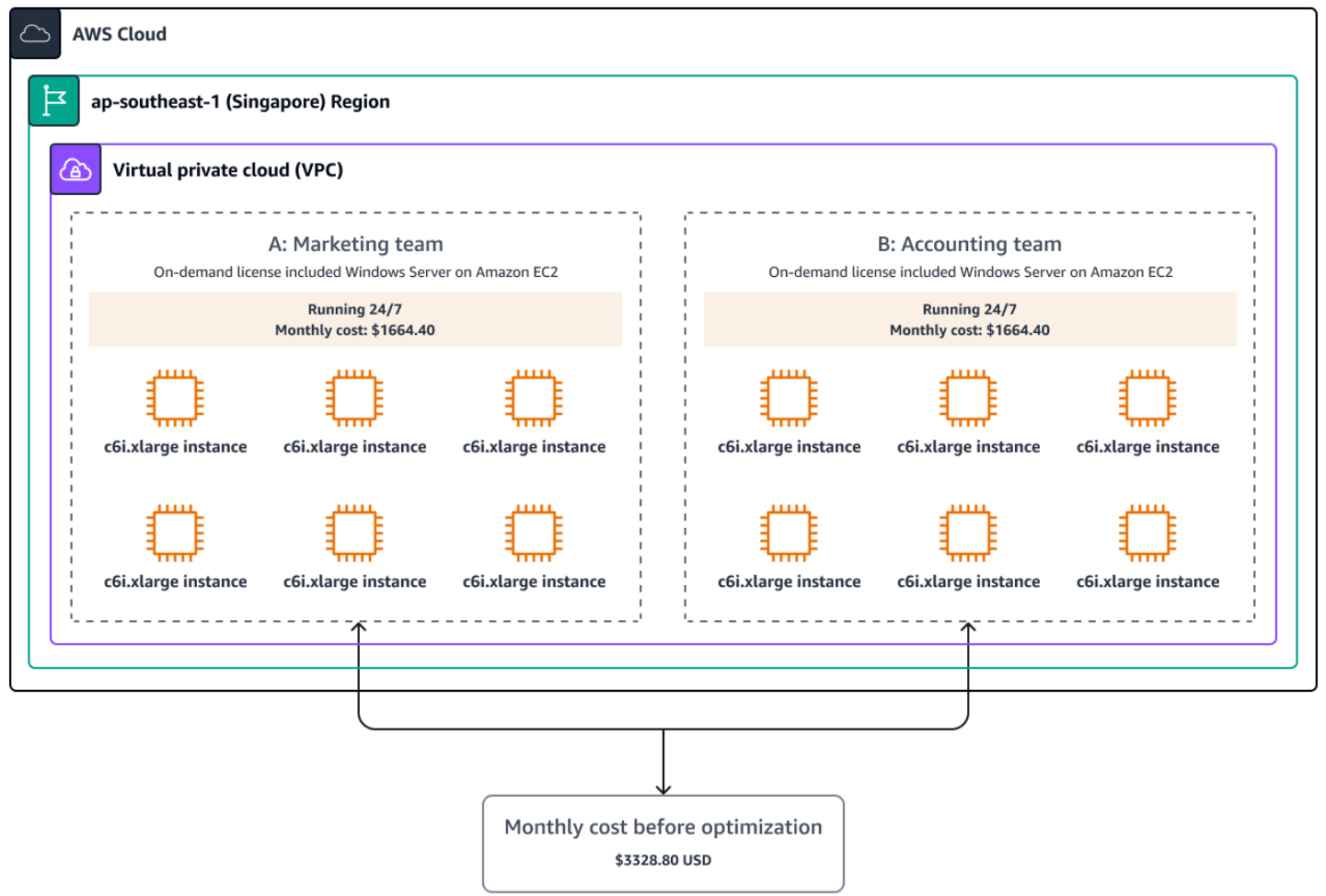


再举一个例子。电力公司牙买加公共服务有限公司 (JPS) 将其数据库迁移到 Amazon RDS。现在，JPS 使用 Amazon EC2 来托管 API 服务并运行其他应用程序。对于 JPS 来说，开启的实例调度器 AWS 成为管理非生产环境的关键工具。JPS 使用了实例调度器 AWS 来降低开发成本，并根据团队需求和工作计划管理 EC2 实例。这帮助 JPS 将成本降低了 40%。有关更多信息，请参阅 AWS 案例研究 [《牙买加公共服务高效迁移到云端，AWS 使用实例调度器将成本降低40%》](#)。

成本优化方案

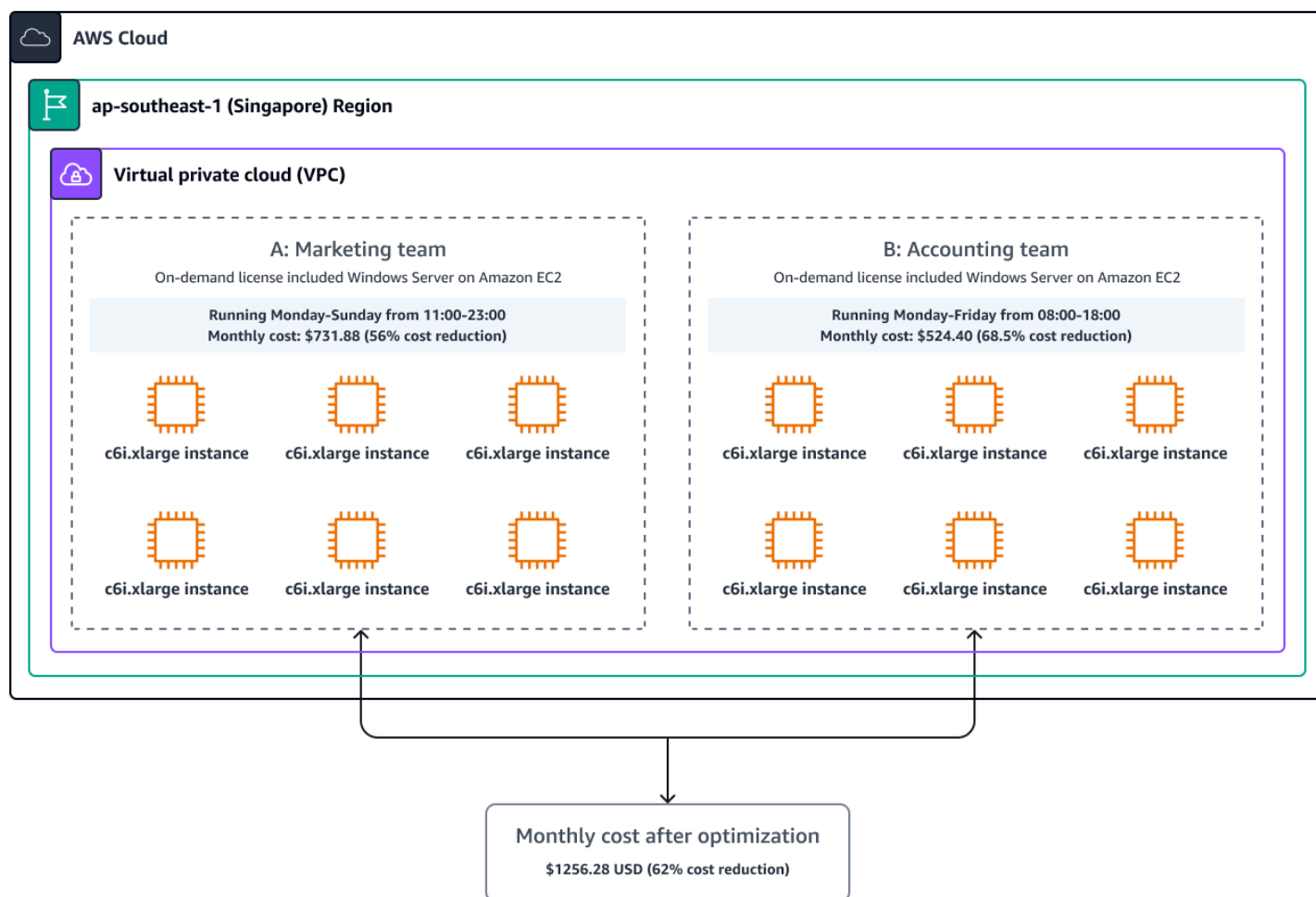
以下示例场景有助于说明在上 AWS使用实例调度器的成本优势。在这种场景下，新加坡一家大型零售公司在 Amazon EC2 上部署了两个 Windows 环境。第一个环境被称为工作负载 A，被营销团队用于在门店营业期间实时分析店内交易情况。第二个环境被称为工作负载 B，为会计团队预留，仅在正常工作时间内工作。目前这两个环境的运行时间表（全天候运行）并不理想，考虑到当前的使用模式，需要进行优化以降低公司的运营成本。

以下图表显示了优化前的每月成本。



例如，三月份有 31 天，其中 23 天是工作日。如果营销团队使用实例调度器 AWS 并仅在需要时才运行其实例（即每月 321 小时而不是每月 730 小时），则他们每月有可能节省 932.52 美元。这意味着运营成本降低了 56%。会计团队也能获得显著的收益，他们每月的实例使用时间从 730 小时减少到了 230 小时。这导致费用减少了 1140 美元，即 68.5%。该公司每月总共可以节省 2072.52 美元（相当于减少 62%），或每年节省 24870.24 美元。

以下图表显示了优化后的每月成本。



Note

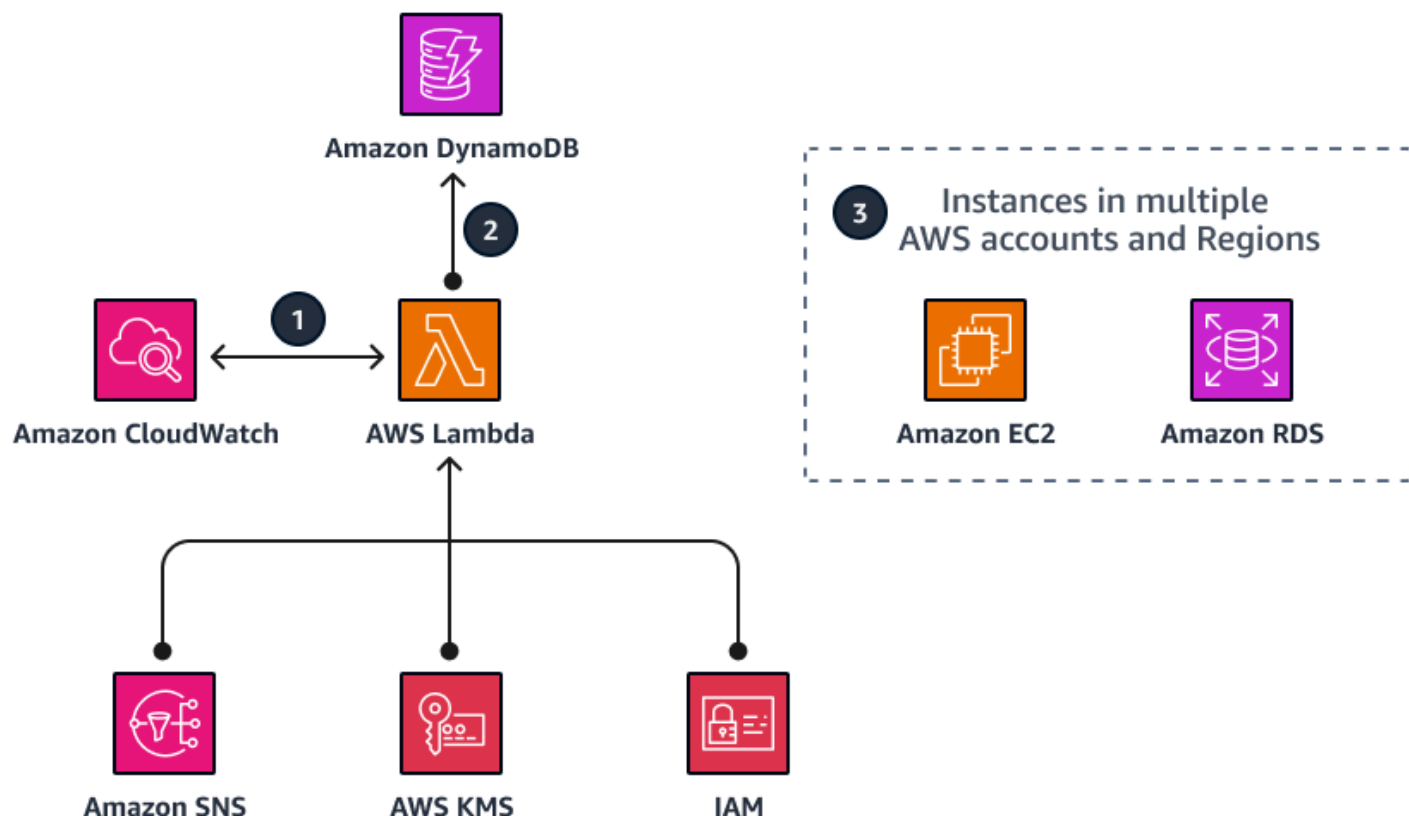
此示例的定价是在 2023 年 3 月通过使用 [AWS 定价计算器](#) 确定的。

成本优化建议

本部分基于前一部分成本优化场景中所介绍的示例场景，说明如何部署和配置 AWS 上的实例调度器。我们建议您采取以下后续步骤，通过在上 AWS 使用实例计划程序来优化成本：

1. 启动实例调度器堆栈
2. 配置周期
3. 配置计划
4. 标记实例。

以下架构图显示了实例调度器堆栈中创建的内容。AWS 云



该图表显示了以下工作流程步骤：

1. AWS CloudFormation 模板按照您定义的时间间隔设置 Amazon CloudWatch 事件。此事件调用一个函数。AWS Lambda 在配置过程中，您可以定义 AWS 区域 和帐户。您还可以定义一个自定义标签，实例调度器 AWS 使用该标签将计划与适用的 Amazon EC2 实例、Amazon RDS 实例和集群关联起来。
2. 计划配置值存储在 Amazon DynamoDB 中，Lambda 函数在每次运行时都会从中获取这些值。然后，您可以将自定义标签应用于适用的实例。
3. 在实例调度器的初始配置过程中，您可以定义一个标签键来识别适用的 Amazon EC2 和 Amazon RDS 实例。创建计划时，您指定的名称将用作标签值，用于标识要应用于已标记资源的计划。

启动实例调度器堆栈

本节介绍如何在上 AWS 启动实例调度程序的 CloudFormation 堆栈。

Note

在上运行实例调度器时 AWS 服务 使用的费用由您承担。AWS截至 2023 年 1 月，使用该解决方案的默认设置在 us-east-1 区域运行的费用约为每月 9.90 美元（这是 Lambda 的费用），如果您的 Lambda 每月使用有免费服务抵扣金可用，则费用会更低。有关更多信息，请参阅 AWS 解决方案库中《[实例计划程序 AWS 实施指南](#)》中的“成本”部分。

要启动实例调度器堆栈，请完成以下步骤。

1. 登录[AWS 管理控制台](#)并选择 [Launch 解决方案](#)（可下载的模板）以启动instance-scheduler-on-aws.template CloudFormation模板。

Note

您也可以[下载模板](#)作为自己实施的起点。

2. 默认情况下，该模板在美国东部（弗吉尼亚州北部）区域启动。要在其他区域中启动实例调度器，请使用控制台导航栏中的区域选择器。

Note

此示例使用亚太地区（新加坡）区域。

3. 在创建堆栈页面的先决条件 - 准备模板部分，确认已选中模板已准备就绪选项。在模板来源部分，确认已选中 Amazon S3 URL 选项。
4. 验证 Amazon S3 URL 文本框中显示了正确的模板 URL，然后选择下一步。
5. 在指定堆栈详细信息页面上，为您的解决方案堆栈分配一个名称。有关命名字符限制的信息，请参阅 AWS Identity and Access Management ([IAM](#)) [文档中的 IAM 和 STS 限制](#)。本指南中示例的堆栈名称被称为 MyInstanceScheduler。

Note

堆栈名称不能超过 28 个字符。

6. 在参数下，检查模板的参数，并根据需要进行修改。
7. 选择 Next(下一步)。在配置堆栈选项页面上，请选择下一步。
8. 在审核页面上，审核并确认设置。选中确认模板将创建 IAM 资源的复选框。

9. 选择创建以部署堆栈。

配置周期

部署 CloudFormation 模板后，该解决方案会创建一个 DynamoDB 表，其中包含示例周期规则和计划，您可以将其用作创建自己的自定义周期规则和计划的参考。有关周期配置的示例，请参阅 AWS 文档中的实例调度器中的[示例计划](#)。

要完成此场景的步骤，必须生成与每个工作负载相对应并满足其特定需求的周期。例如：

```
Period 1 (Workload A):  
  Name: retail-hours  
  Days: Monday to Sunday  
  Hours: 1100 - 2300  
Period 2 (Workload B):  
  Name: office-hours  
  Days: Monday to Friday  
  Hours: 0800 - 1800
```

要配置周期，请完成以下步骤：

1. 登录 [DynamoDB](#) 控制台，确保您所在的区域与启动实例计划程序模板 CloudFormation 的区域相同。AWS
2. 在导航窗格中，选择表，然后选择名为的表ConfigTable。
3. 选择浏览表项目。
4. 要创建办公时间时段，请为办公时间项选择时段。
5. 在编辑项目页面上，将 begintime 的值更改为 0800，将 endtime 的值更改为 1800。将工作日的默认值保持不变。

Note

begintime 和 endtime 值决定何时启动和停止实例，而工作日值则决定该计划适用于一周中的哪几天（本示例中为星期一至星期五）。

6. 选择保存更改。
7. 要复制办公时间时段并使用它来创建新的零售时间段，请为办公时间项选择时段。然后，从操作菜单中选择复制项目。
8. 修改属性以满足您的需求。以下属性用于满足示例场景的要求：

```
type: period
name: retail-hours
begintime: 11:00
description: Retail hours
endtime: 23:00
weekdays: mon-sun
```

9. 选择创建项目。

10. 在 DynamoDB 中，确定项目列表中列出的您刚刚创建的两个时段。

配置计划

在“开启实例调度器”的上下文中 AWS，计划是指应用一个或多个时段以及相关时区。随后，这些时间表会被作为标签分配到您的实例中。本部分将向您展示如何创建两个计划（如下所示），以适应两个示例工作负载所呈现出的不同时间模式，然后将这些计划与您在上一部分中创建的时段进行关联。

```
Schedule 1:
  Name: singapore-office-hours
  Period: office-hours
  Timezone: Asia/Singapore
Schedule 2:
  Name: singapore-retail-hours
  Period: retail-hours
  Timezone: Asia/Singapore
```

要创建和配置计划，请完成以下步骤：

1. 登录 [DynamoDB](#) 控制台，确保您所在的区域与启动实例计划程序模板 CloudFormation 的区域相同。AWS
2. 在导航窗格中，选择表，然后选择名为的表 ConfigTable。
3. 选择浏览表项目。
4. 要复制英国办公时间表并使用它为您的办公时间（本例为新加坡办公时间）创建新的时间表，请为该 uk-office-hours 项目选择时间表。然后，从操作菜单中选择复制项目。
5. 修改属性以满足您的需求。以下属性用于满足示例场景的要求：

```
type: schedule
name: singapore-office-hours
description: Office hours in Singapore
```

```
periods: office-hours
timezone: Asia/Singapore
```

6. 选择创建项目。

7. 重复步骤 4–6，以使用以下属性值为新加坡零售时间创建计划：

```
type: schedule
name: singapore-retail-hours
description: Retail hours in Singapore
periods: retail-hours
timezone: Asia/Singapore
```

8. 在 Dynam ConfigTableoDB 中，确定您创建的两个计划和两个周期。

标记 实例。

建立计划后，必须使用标签将计划分配给您要使用的特定实例。您可以在 [AWS Resource Groups](#) 中使用标签编辑器生成标签并将其分配给您的 Amazon EC2 实例。

1. 登录[AWS 管理控制台](#)并确保您位于之前启动 CloudFormation 模板的同一区域。
2. 打开[资源组控制台](#)。在导航窗格中，展开标记，然后选择标签编辑器。
3. 在查找要标记的资源部分中，对于区域，选择您的区域。对于资源类型，请选择 Amazon EC2 或 Amazon RDS。此场景主要针对工作负载 A 中的 Amazon EC2 实例。营销团队正在新加坡区域使用工作负载 A。用于此工作负载的资源已经标记有部门键和营销值。您可以使用此标签来搜索实例。
4. 选择搜索资源。
5. 从搜索结果列表中选择要包含在计划中的实例，然后选择管理所选资源的标签。
6. 在编辑所有选定资源的标签部分中，选择添加标签，以将实例调度器计划标签添加到您的 EC2 实例。您可以使用与 schedulea（之前在 DynamoDB 中创建）匹配的标签键和值。
7. 对于标签键，请添加计划。在“标签值”中，输入singapore-retail-hours。
8. 选择检查并应用标签更改。
9. 要将标签应用于您选择的所有 EC2 实例，请选择将更改应用于所有选定实例。
- 10对于要应用的任何其他计划，请重复步骤 3–9。

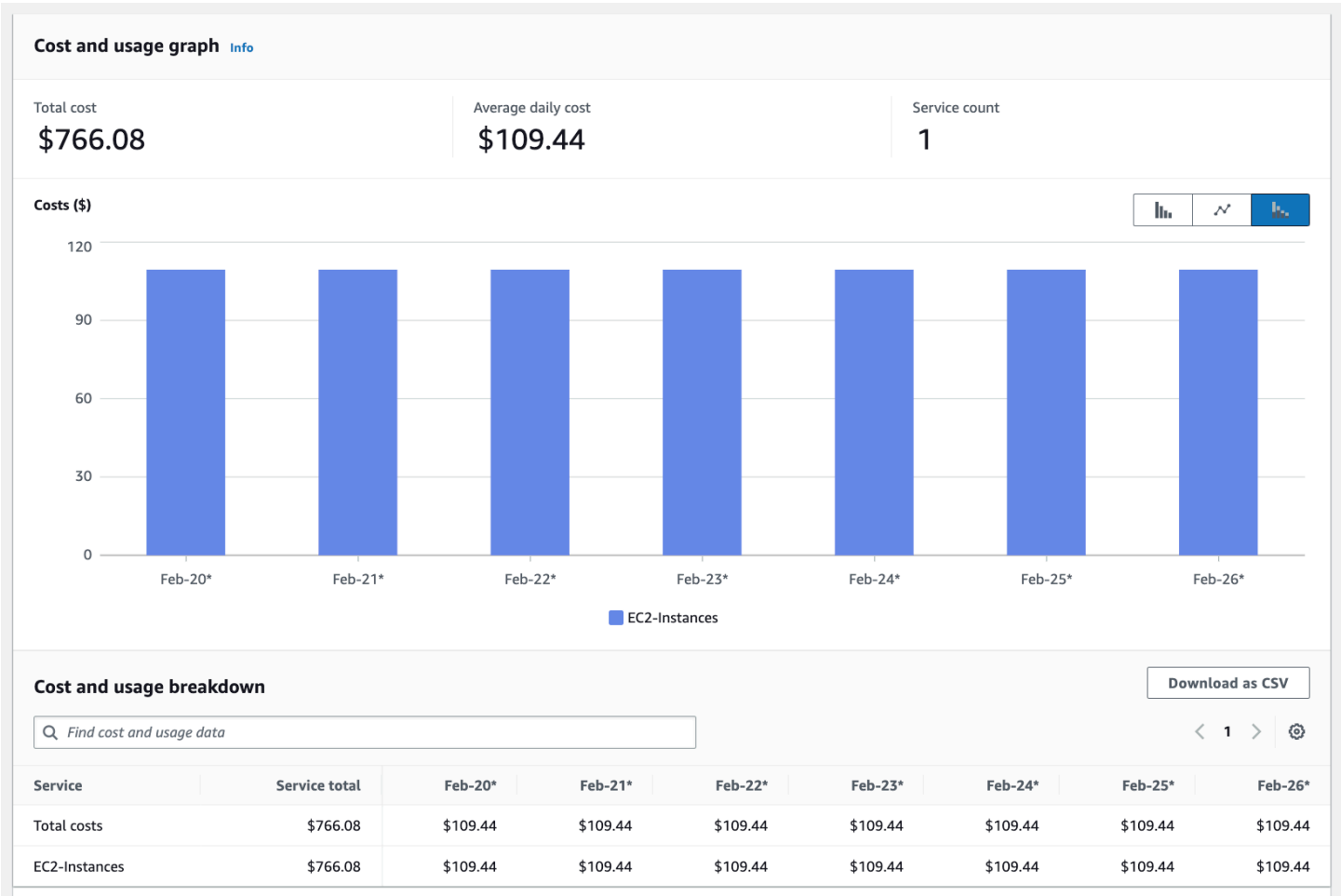
验证结果

我们建议您使用[AWS Cost Explorer](#)来衡量在上 AWS使用实例调度程序的成本收益。您可以使用 Cost Explorer 成本管理服务等来执行以下操作：

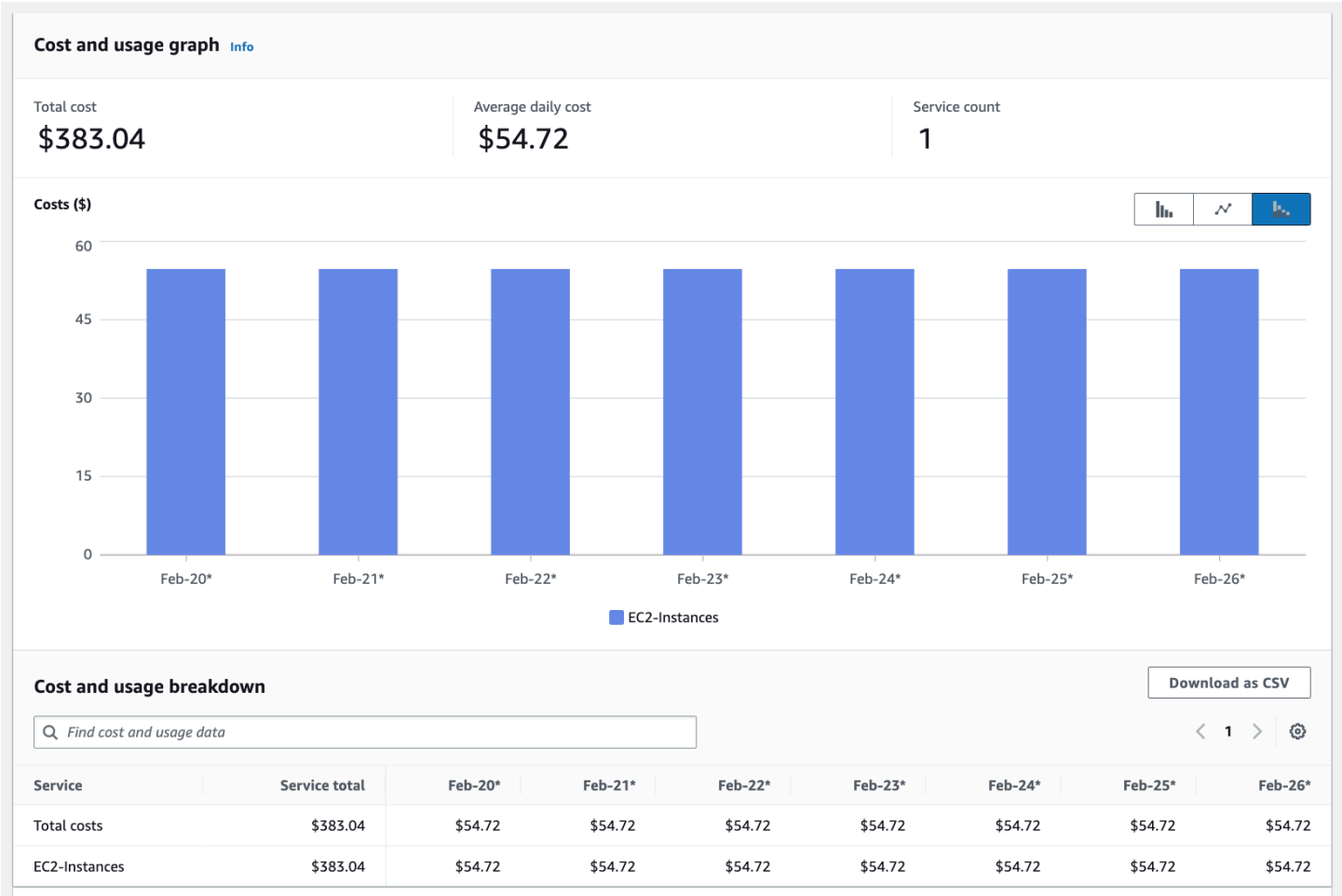
- 查看和分析与您的 EC2 实例相关的成本，包括由实例调度器管理的实例。
- 按标签筛选您的 Cost Explorer 成本管理视图，以便您能够专注于特定的工作负载，并详细了解通过使用实例调度器所实现的成本节省情况。
- 深入了解使用实例调度器的财务影响。
- 找出进一步优化成本的机会，并做出以数据为依据的决策以优化 AWS 支出。

以下图表展示了在使用实例调度器进行优化前的七天时间段（周一至周日）内，工作负载 A 和工作负载 B 的运行成本情况。

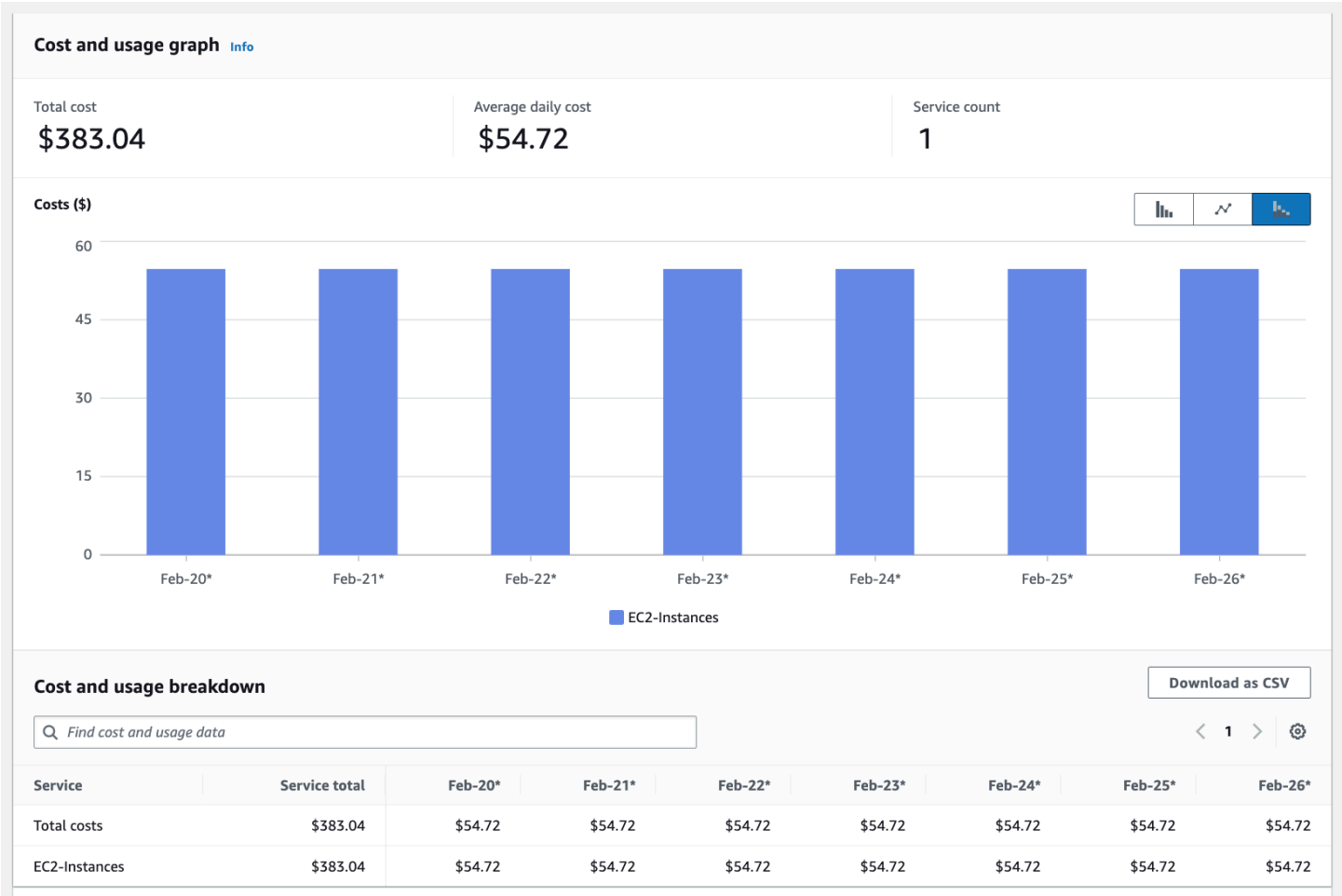
工作负载 A 和 B 的总费用之和



工作负载 A 费用

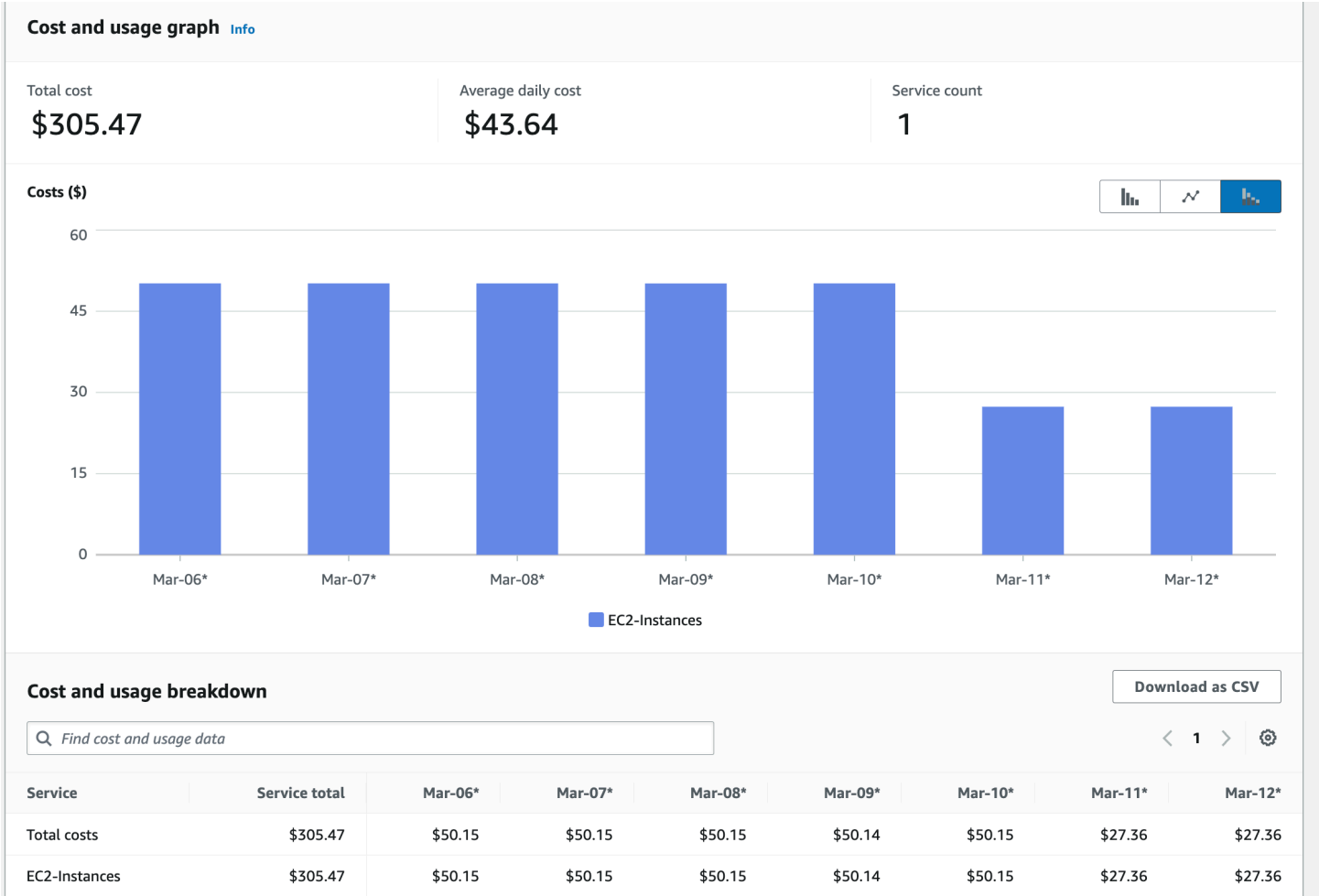


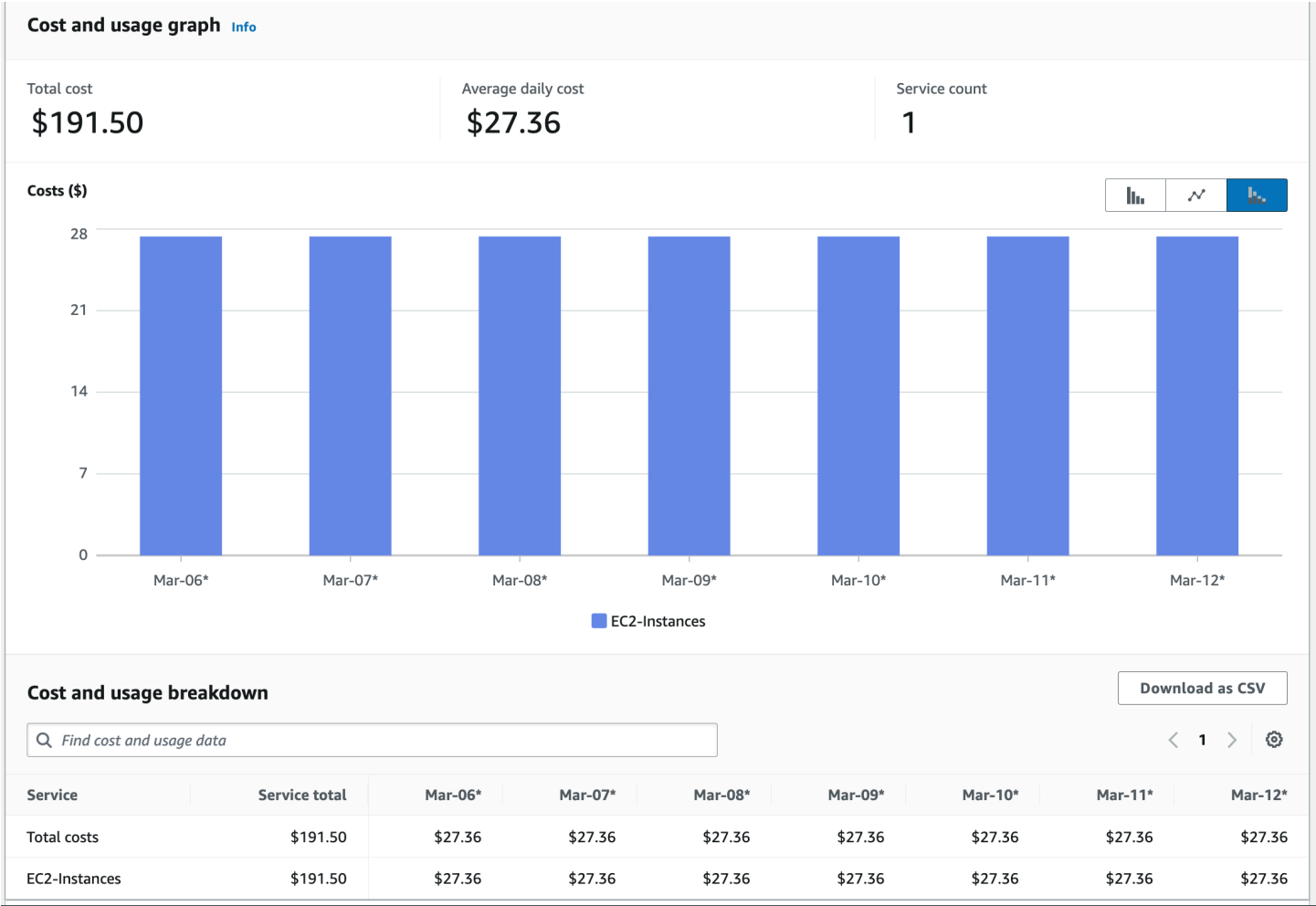
工作负载 B 费用



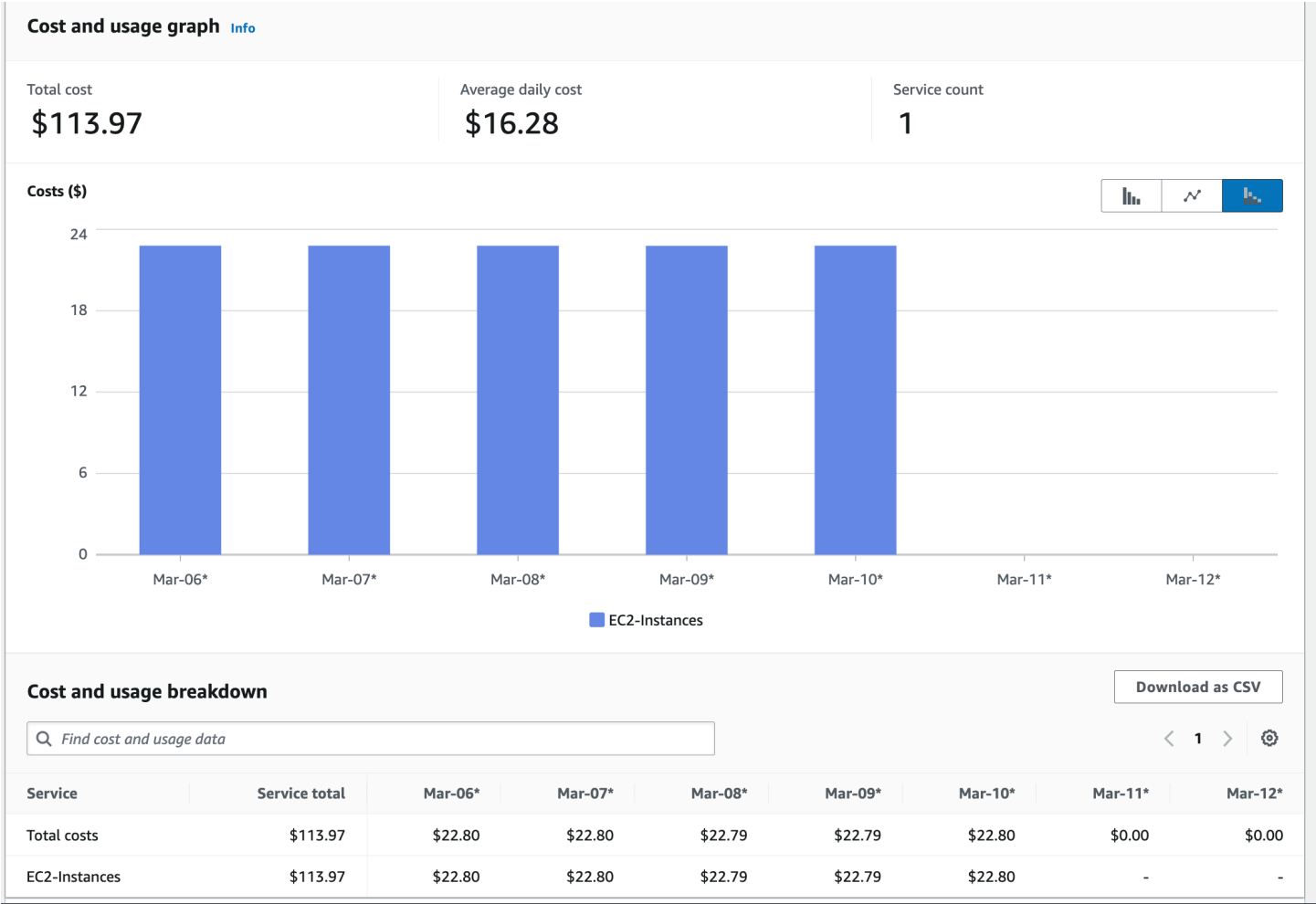
在此场景中，Cost Explorer 成本管理服务显示了通过实施 AWS 上的实例调度器而降低的成本。以下图表描述了优化后七天（星期一至星期日）内工作负载 A 和工作负载 B 的运行成本情况。

工作负载 A 和 B 的总费用之和





工作负载 B 费用



其他资源

- [自动启动和停止 AWS 实例](#) (AWS 文档中的实例调度器)
- [回归基础知识：使用实例计划程序控制 Amazon EC2 和 Amazon RDS 资源成本](#) () YouTube
- 为@@ [AWS 资源添加标签](#) (标记 AWS 资源用户指南)
- [使用 AWS Cost Explorer](#) ([AWS 账单与成本管理 文档](#)) 分析您的成本

调整 Windows 工作负载

概述

合适的尺寸是最有效的节省成本的工具之一。AWS 提供了各种收集正确规模信息的方法，从使用[AWS 优化和许可评估 \(O AWS LA\)](#) 审查潜在的工作负载，到使用查看现有工作负载[AWS Cost Explorer](#)。

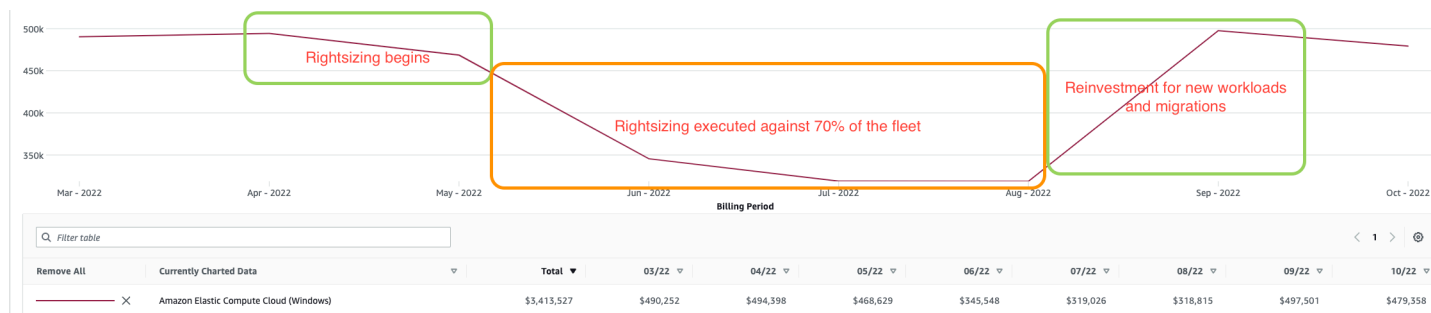
本部分向您展示如何使用 [AWS Compute Optimizer](#) 来识别 Amazon EC2 调整的机会。Compute Optimizer 有助于防止以下类型的资源出现过度配置和配置不足的情况：AWS

- [Amazon Elastic Compute Cloud \(Amazon EC2 \)](#) 实例类型
- [Amazon Elastic Block Store \(Amazon EBS \)](#) 卷
- [亚马逊弹性容器服务 \(Amazon ECS\)](#) Container Service 服务在 AWS Fargate
- [AWS Lambda](#) 基于 [Amazon](#) 提供的使用率数据的函数 CloudWatch

成本优化方案

衡量调整措施的有效性可能颇具挑战性，因为调整工作既可以针对特定的应用程序、团队，也可以针对整个组织。例如，假设一个组织将数千个实例迁移到 AWS，其中 90% 的队列包含 Windows 工作负载。该组织可以采用 Compute Optimizer 来分析其实例集，发现其账户和 AWS 区域中存在严重的过度预调配。然后，他们可以使用 [AWS Systems Manager Automation](#) 通过多个维护窗口调整其实例集。因此，该组织成功为 70% 的实例集配置了大小合适的实例类型，并实现了 35% 的成本节省。

以下控制面板展示了该组织在数月内所实现的节省成效，这是因为在该组织的策略性实施下，其充分利用了 Compute Optimizer 所提供的调整建议。他们的目标是尽可能高效地运行现有的工作负载，以便重新启动从一家即将到期合同的主机托管数据中心进行的已停滞的迁移工作。



成本优化建议

我们建议您采取以下后续步骤以通过使用 Compute Optimizer 优化成本：

- 启用 Compute Optimizer
- 为 Windows 节点启用内存指标收集
- 使用 Compute Optimizer 建议
- 标记实例以进行调整
- 启用成本分配标签以使用 AWS 计费工具

- 使用 AWS Systems Manager 自动化实施合适的尺寸建议
- 考虑其他调整方法
- 在 Cost Explorer 成本管理服务中查看成本前后

启用 Compute Optimizer

您可以在 AWS Organizations 中，在组织级别或单个账户级别启用 [Compute Optimizer](#)。组织范围的配置为所有成员账户的整个实例集中的新实例和现有实例提供持续报告。这样一来，正确调整大小就可以成为重复的活动，而不是一项 point-in-time 活动。

组织级别

对于大多数组织来说，使用 Compute Optimizer 的最有效方式是在组织层面。这为您的组织提供了多账户和多区域可见性，并将数据集中到一个来源供审查。要在组织层面启用此功能，请执行以下操作：

1. 使用具有[所需权限](#)的角色登录您的 [Organizations 管理账户](#)，然后选择加入该组织内的所有账户。您的组织必须[已启用所有功能](#)。
2. 启用管理账户后，您可以登录该账户，查看所有其他成员账户，并浏览他们的推荐。

Note

最佳做法是为 Compute Optimizer 配置[委托管理员账户](#)。这使您能够行使最低权限原则。这样，您就能最大程度地限制对组织管理账户的访问权限，同时仍能提供对组织内各项服务的访问。

单个账户级别

如果您的目标账户成本高昂，但又没有 AWS Organizations 的访问权限，您仍然可以为该账户和区域启用 Compute Optimizer。要了解选择加入流程，请参阅 Compute Optimizer 文档中的 [AWS Compute Optimizer 入门](#)。

为 Windows 节点启用内存指标收集

内存指标为 Compute Optimizer 提供了制定您组织内明智的调整建议所必需的关键指标。这是因为在给出建议之前，会对 CPU、内存、网络 and 存储等方面进行分析。

要将内存指标从 Windows EC2 实例传递给 Compute Optimizer，您必须启用 CloudWatch 代理并将内存指标配置为每 60 秒收集一次。将内存指标与一起使用不会产生额外费用 CloudWatch。

启用 CloudWatch 代理并配置内存指标

下载 [ComputeOptimize.yml 文件](#)。您可以使用此文件为您账户中的所有实例启用内存收集功能。模板文件会生成以下组件：

- [AWS Systems Manager 参数存储](#)-用于存储收集内存指标所需的 CloudWatch 代理配置。
- AWS Identity and Access Management (IAM) 角色 AWS Systems Manager 附带 [AWS 托管策略 — 此角色适用于](#) Systems Manager Automation 文档。
- [AWS Systems Manager documents](#) — 这将安装和配置 CloudWatch 代理（替换任何现有 CloudWatch 配置）。
- [AWS Systems Manager 状态管理器](#) 关联 — 这使得 Systems Manager 文档能够在您的账户中的所有实例上运行。

Important

运行此模板会覆盖实例上的任何现有 CloudWatch 配置。

接下来执行以下操作：

1. 登录 AWS 管理控制台 并打开 [CloudFormation 控制台](#)。
2. 在导航窗格中，选择 Stacks（堆栈）。
3. 选择创建堆栈，然后选择使用现有的资源（导入资源）。
4. 选择下一步。
5. 对于模板来源，选择上传模板文件。
6. 选择文件，然后上传 ComputeOptimize.yml 文件。
7. 选择下一步。
8. 在指定堆栈详细信息页面上，对于堆栈名称字段，输入堆栈的名称，然后选择下一步。
9. 在识别资源页面上，输入要导入的资源的标识符值。
10. 选择导入资源。
11. 部署堆栈后，选择输出选项卡以查找关联的键、值和描述。

监控关联的进度

1. CloudFormation 堆栈部署完成后，打开 [Systems Manager 控制台](#)。

2. 在导航窗格中的节点管理部分，选择状态管理器。
3. 在关联页面上，选择关联 ID。
4. 选择 Execution history (执行历史记录) 选项卡。
5. 在执行 ID 列中，选择您的关联的执行 ID。状态应为成功。

在中查看指标 CloudWatch

我们建议您至少等待五分钟，让指标填充。 CloudWatch

1. 打开 [CloudWatch 控制台](#)。
2. 在导航窗格中，展开指标部分，然后选择所有指标。
3. 确认指标显示在CWAgent命名空间下。

Note

要将设置应用于任何新实例，请重新运行关联。

使用 Compute Optimizer 建议

举一个例子，该示例侧重于在单个账户和单个区域内进行调整更改。在本示例中，Compute Optimizer 在组织级别为所有账户启用。请记住，调整属于破坏性过程，在大多数情况下，应用程序所有者会在几周的计划维护时段内精确地执行此过程。

如果您从组织的管理账户中导航到 Compute Optimizer (如以下步骤所示)，则可以选择要调查的账户。在此示例中，us-east-1 区域的单个账户中有六个实例在运行。所有六个实例都被过度预调配。目标是根据 Compute Optimizer 提供的建议调整实例。

识别过度预调配的实例并导出建议详细信息

1. 登录 AWS 管理控制台 并打开 [Compute Optimizer 控制台](#)。
2. 在导航窗格中，选择控制面板。
3. 在控制面板页面的搜索框中，输入区域=美国东部 (弗吉尼亚州北部)。然后，输入调查发现=过度预调配。这些筛选条件能让您查看 us-east-1 区域中所有过度预调配的实例。
4. 要查看针对过度预调配的 EC2 实例的详细建议，请向下滚动到 EC2 实例卡片，然后选择查看建议。

5. 选择导出，并保存文件以备将来使用。
6. 对于 S3 存储桶，输入您希望作为导出文件目标的 Amazon S3 存储桶的名称。

Note

要保存建议以备将来查看，您必须有一个可用的 S3 存储桶供每个区域的 Compute Optimizer 写入数据。有关更多信息，请参阅 Compute Optimizer 文档中的 [Amazon S3 bucket policy for AWS Compute Optimizer](#)。

7. 在导出筛选条件部分中，选中包括组织中所有成员账户的建议复选框。
8. 对于资源类型，选择 EC2 实例。
9. 在要包括的列部分中，选中全选复选框。
10. 选择导出。

根据建议选择实例

实例建议基于 Compute Optimizer 收集和分析的性能指标。必须清楚了解实例上正在运行的工作负载情况，这样才能确保您选择到最适合的实例。此示例假设您可以从最新一代的 Amazon EC2 [R6i](#)、[R5](#) 和 [T3](#) 实例中进行选择。T3 实例是爆发性实例，并且网络带宽能力较低。R5 和 R6 实例的每小时成本相同，而且几乎相同。但是，R6 实例具有更高的网络带宽容量，配备了最新一代的英特尔处理器，并且其计算占用空间与 R5 相同。在此示例中，R6 是用于调整的最佳选择。

1. 在 [Compute Optimizer 控制台](#) 中，从导航栏中选择适用于 EC2 实例的建议。此页面向您展示了当前实例类型与推荐替代方案的对比情况。
2. 要获取您想要调整的实例的 ID，请从 AWS Organizations 中的管理账户打开 [Amazon S3 控制台](#)。
3. 在导航窗格中，选择存储桶，然后选择您用于存储导出结果的存储桶。
4. 在对象选项卡上，从对象列表中选择您的导出文件，然后选择下载。
5. 要从文件中提取实例信息，可以使用 Microsoft Excel 中数据选项卡上的文本到列按钮。

Note

实例以 IDs Amazon 资源名称 (ARNs) 表示。请务必将分隔符设置为“/”并提取实例 ID。或者，您可以编写脚本或使用集成式开发环境 (IDE) 来修剪 ARN。

6. 在 Excel 中，筛选调查发现列以仅显示 OVER_PROVISIONED 实例。这些是您需要针对其进行调整的实例。

7. 将实例保存在文本编辑器 IDs 中，便于日后访问。

标记实例以进行调整

标记工作负载是组织 AWS 资源的强大工具。您可以使用标签来更精细地了解成本并帮助退款。有关向资源添加标签的策略和方法的更多信息，请参阅 AWS 白皮书为 AWS 资源添加[标签 AWS 的最佳实践](#)。对于此示例，您可以使用 [AWS 标签编辑器](#) 对您希望在维护窗口期间进行调整的过度预调配实例进行标记调整。您也可以使用此标签来查看更改前后的费用。

1. 登录 AWS 管理控制台 并打开包含要调整大小的实例的账户的 [AWS Resource Groups 控制台](#)。
2. 在导航栏的标记部分，选择标签编辑器。
3. 对于区域，请选择您的目标区域。
4. 对于资源类型，请选择 AWS::EC2::Instance。
5. 选择搜索资源。
6. 在资源搜索结果页面上，选择要调整的所有实例，然后选择管理所选资源的标签。
7. 选择 Add tag (添加标签)。
8. 对于标签键，输入调整。对于标签值，输入已启用。然后，选择检查并应用标签更改。

Note

您可以包括诸如团队或业务部门等的额外元数据，以帮助后续在 Cost Explorer 成本管理服务中进行筛选。

在创建用户定义的标签并将其应用到您的资源之后，标签最多可能需要 24 小时才能显示在“成本分配标签”页面上进行激活。选择要激活的标签后，可能再需要 24 小时才能激活标签。

对于高级用户，您可以在目标账户和区域内使用 [AWS CloudShell](#) 来标记多个实例。例如：

```
bash
#!/bin/bash
# Set variables
TAG_KEY="rightsizing"
TAG_VALUE="type-m5"
# Get a list of instance IDs
INSTANCE_IDS=$(aws ec2 describe-instances --query
  "Reservations[].Instances[].InstanceId" --output text)
# Loop through each instance ID and add the tag
```

```
for INSTANCE_ID in $INSTANCE_IDS; do
  aws ec2 create-tags --resources $INSTANCE_ID --tags Key=$TAG_KEY,Value=$TAG_VALUE
done
```

启用成本分配标签以使用 AWS 计费工具

建议激活用户定义的成本分配标签。这样就可以在 AWS 计费工具（例如，Cost Explorer 和）中识别和筛选 Right sizing 标签。AWS 成本和使用情况报告如果您不启用此选项，则标签筛选选项和数据将不可用。有关使用成本分配标签的更多信息，请参阅 AWS 账单与成本管理 文档中的 [Activating user-defined cost allocation tags](#)。

1. 登录 AWS 管理控制台 并打开 [AWS Billing 控制台](#)。
2. 在导航窗格中的账单部分，选择成本分配标签。
3. 在用户定义的成本分配标签选项卡上，输入调整。
4. 选择调整标签键，然后选择激活。

24 小时后，标签应显示在 Cost Explorer 成本管理服务中。

使用 Systems Manager Automation 实施调整建议

调整是一种需要停止和启动实例的场景。在这种情况下，您可能需要在维护窗口内处理这一中断问题，并且需要不同的团队分别负责各自的调整工作。在更改实例类型之前，请查看 Amazon EC2 文档中[兼容实例类型的考虑因素](#)。

本节中的示例步骤使用名为 [AWS-](#) 的 Systems Manager Automation 文档，为每个账户和地区实施合适的规模建议ResizeInstance。这种方法在大多数组织中都很常见，因为大多数组织出于不同目的需要不同类型的实例。您也可以使用相同的 AWS-ResizeInstance 自动化文档来针对单账户和多账户部署。

1. 登录 AWS 管理控制台 并打开 [Systems Manager 控制台](#)。
2. 在导航窗格的共享资源部分，选择文档。
3. 在搜索栏中输入 AWS-ResizeInstance，然后ResizeInstance 从搜索结果中选择 AWS-。
4. 选择执行自动化。
5. 在执行自动化运行手册页面上，选择简单执行。
6. 在输入参数部分中，输入InstanceId和InstanceType。保留其余的默认值。
7. 选择执行，然后等待自动化程序完成更改实例类型的各项步骤。

考虑其他调整方法

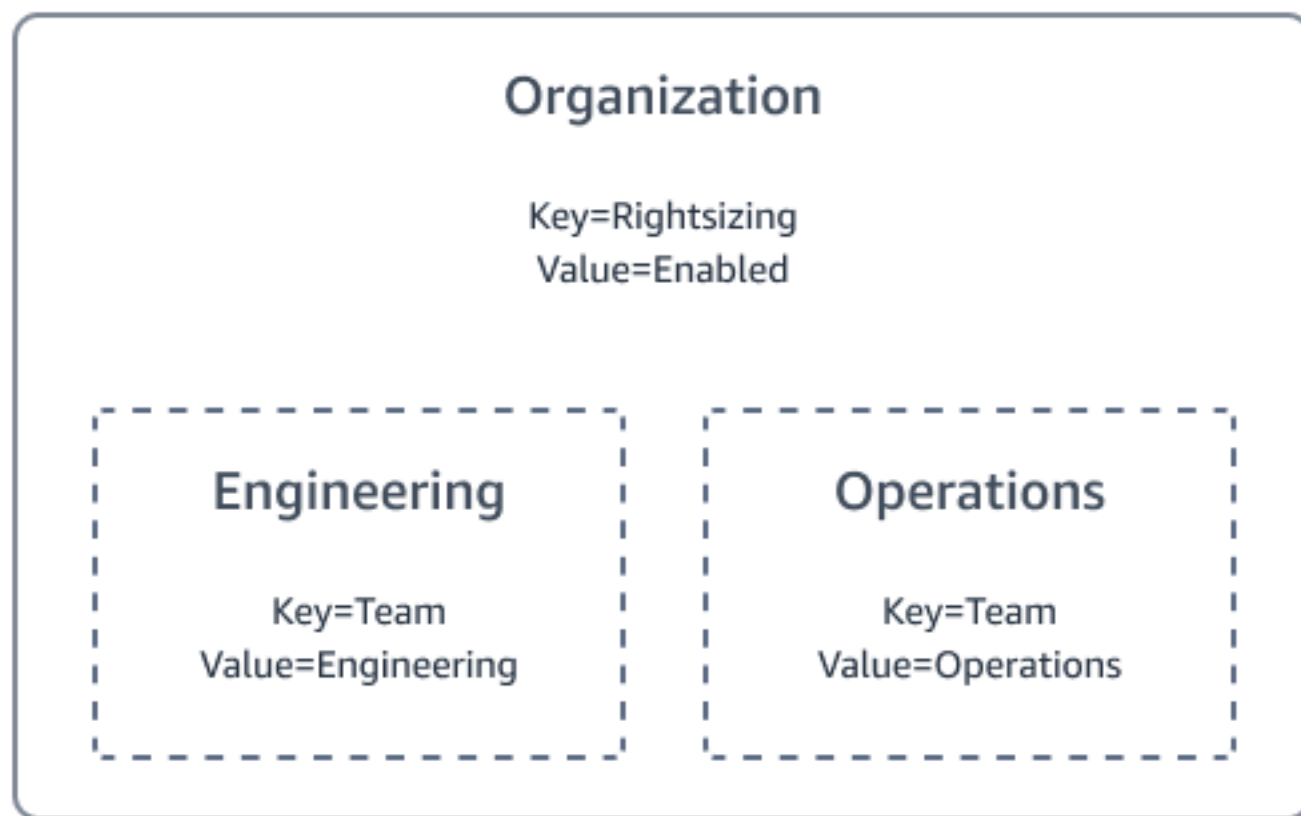
如果您使用启动模板来部署实例，则可以使用调整过的实例类型更新启动模板，然后执行实例刷新，以用调整过的版本替换现有实例。

如果您计划在多个账户和区域中使用调整流程，则必须创建自定义 Systems Manager Automation 文档。此文档允许您将多个实例作为参数输入，并且目标实例会迁移到相同的目标实例类型（例如，所有实例都会转换为 t3a.medium 类型，不论源实例类型为何）。

在 Cost Explorer 成本管理服务中查看成本前后

调整资源后，您可以使用 Cost Explorer 成本管理服务通过调整标签显示调整前后的成本。请记住，您可以使用[资源标签](#)来跟踪成本。通过使用多层标签，您可以精细地了解成本。在本指南中介绍的示例中，调整标签用于将通用标签应用于所有目标实例。然后，使用团队标签来进一步整理资源。下一步是引入应用程序标签，以进一步显示运行特定应用程序的成本影响。

以下图表显示了一个组织的标签结构。



以调整运营团队拥有的生产 Web 服务器的企业为例。在 Cost Explorer 成本管理服务中，调整标签被设置为启用，团队标签被设置为运营。在此示例中，调整工作可将运营成本从一小时 0.89 美分降低到 0.28 美分。假设每月 744 小时，调整前的年度成本为 7945.92 美元。调整后，年度成本降为 2499.84

美元。这意味着每年的工作负载成本降低了 68.5%。想象一下这对大型组织的影响。请记住，这是在模拟环境中进行的操作，而且这些实例大多处于闲置状态。在生产环境中，您可以看到 10-35% 的节省。

现在，考虑一下调整工程团队拥有的生产堡垒主机会产生什么影响。在 Cost Explorer 成本管理服务中，调整标签被设置为启用，团队标签被设置为工程。在此示例中，调整工作可将成本从一小时 0.75 美分降低到 0.44 美分。假设每月 744 小时，调整前的年度成本为 6696.00 美元。调整后，年度成本降为 3928.32 美元。

如果您使用多个标签，则可以将数据筛选到具体的成本明细层面。在此示例中，团队标签减少了干扰，让您能够从团队层面查看其影响情况。由于调整标签已启用，因此您还可以筛选具有该标签且值为启用或不存在任何值的任何实例。这可以让您全面了解您的调整工作，尤其是在 Cost Explorer 成本管理服务层面的管理账户（付款人）中查看时。通过此视图，您可以查看所有账户和实例。

举一个单账户级别的示例，其中调整标签设置为启用。运营成本从一小时 1.64 美元降至一小时 0.72 美分。假设每月 744 小时，调整前的年度成本为 14641.92 美元。调整后，年度成本降为 6428.16 美元。这意味着该账户的计算成本降低了 56%。

在开始调整旅程之前，考虑以下事项：

- AWS 提供了许多降低成本的选项。这包括 [AWS OLA](#)，AWS 它会在迁移到本地实例之前先检查您的本地实例 AWS。O AWS LA 还为您提供合适的规模建议和许可指南。
- 在购买[节省计划](#)之前，完成所有调整。这可以帮助您避免超额购买节省计划承诺。

建议

我们建议您完成以下后续步骤：

1. 查看您的现有格局，并考虑将 Amazon EBS gp2 卷转换为 gp3 卷。
2. 查看[节省计划](#)。

其他资源

- [AWS Compute Optimizer](#) (AWS 文档)
- [标记 AWS 资源的最佳实践](#) (AWS 白皮书)
- [如何从 AWS Compute Optimizer 您的 AWS Organizations\(YouTube\) AWS Trusted Advisor 中收集数据](#)

- [优化性能并降低许可成本：将 AWS Compute Optimizer 用于 Amazon EC2 SQL Server 实例](#) (AWS 上的 Microsoft 工作负载博客)

为 Windows 工作负载选择合适的实例类型

概述

云环境中运行的工作负载与本地环境中的工作负载之间的一个重要区别在于过度预调配的做法。在购置用于本地的物理硬件时，您会进行一笔资本支出，这笔支出预计将持续一定期限，通常为 3 至 5 年。为了适应硬件在其使用寿命期间的预期增长需求，所购置的硬件所配备的资源比当前的工作负载所需资源要多。因此，物理硬件的配置往往会超出实际工作负载的需求，存在过度预调配的情况。

虚拟机 (VM) 技术已成为有效利用过剩硬件资源的一种手段。管理员过度配置了 VMs v CPUs 和 RAM，这使虚拟机管理程序能够通过向每台 VM 分配未使用的资源来管理繁忙和空闲服务器之间的物理资源使用情况。在管理时 VMs，分配给每个虚拟机的 vCPU 和 RAM 资源更多地起到资源调控器的作用，而不是实际使用情况的指标。VM 资源过度分配可能会轻易超出可用计算资源的三倍之多。

[Amazon Elastic Compute Cloud \(Amazon EC2\)](#) 避免在底层硬件上进行 VMs 过度配置，因为这是不必要的。云计算是一项运营费用，而不是资本支出，您只需为所使用的资源付费。如果未来您的工作负载需要更多的资源，请在实际需要时再预调配这些资源，而不是提前一次性全部准备好。

在选择正确的 [Amazon EC2 实例类型](#) 方面，有数百种选项可用。如果您计划将 Windows 工作负载迁移到云端，请 AWS 提供 O [AWS L A](#)，以帮助您更好地了解当前的工作负载并举例说明其运行性能 AWS。AWS OLA 分析旨在将合适的 EC2 实例类型和大小与您的实际本地使用量相匹配。

如果您已经在 Amazon EC2 上运行工作负载，并且希望寻找成本优化策略，则本指南的这一部分将帮助您了解 Amazon EC2 实例之间的差异以及它们对典型 Windows 工作负载的适用性。

成本优化建议

要优化 EC2 实例类型的成本，我们建议您执行以下操作：

- 为工作负载选择合适的实例系列
- 了解处理器架构之间的价格差异
- 了解各代 EC2 实例的性价比差异
- 迁移到较新的实例
- 使用可爆发实例

为工作负载选择合适的实例系列

为工作负载选择适合的实例系列很重要。

Amazon EC2 实例分为以下各组：

- 通用型
- 计算优化型
- 内存优化型
- 加速计算型
- 存储优化
- HPC 优化型

大多数 Windows 工作负载属于以下类别：

- 通用型
- 计算优化型
- 内存优化型

为了进一步简化这一过程，考虑为每个类别设定一个基准的 EC2 实例：

- 计算优化型：C6i
- 通用型：M6i
- 内存优化型：R6i

上一代 EC2 实例在处理器类型上存在细微差异。例如，C5 计算优化型实例的处理器比 M5 通用实例或 R5 内存优化型实例更快。最新一代的 EC2 实例 (C6i、M6i、R6i、C6a、M6a 和 R6a) 在不同的实例系列中使用相同的处理器。由于在最新一代实例中均使用一致的处理器，因此实例系列之间的价格差异现在更多地取决于 RAM 的量。实例拥有的 RAM 越多，其成本就越高。

以下示例说明了在 us-east-1 区域运行的基于 Intel 的 4 vCPU 实例的每小时定价。

实例	v CPUs	RAM	每小时价格
c6i.xlarge	4	8	0.17 美元

实例	v CPUs	RAM	每小时价格
m6i.xlarge	4	16	0.19 美元
r6i.xlarge	4	32	0.25 美元

 **Note**
定价基于 us-east-1 区域的每小时按需定价。

可爆发实例

在云计算中，通常会将未使用的计算资源关闭以避免产生费用，但并非所有工作负载都能在每次需要时随时开启或关闭。有些工作负载会长时间处于闲置状态，但必须能够全天 24 小时随时访问。

可爆发实例 (T3) 提供了一种解决方案，可使具有波动性或低利用率的工作负载全天保持在线运行，同时还能将计算成本控制在较低水平。可爆发 EC2 实例具有在短时间内可使用的最大 vCPU 资源量。这些实例使用基于[可爆发 CPU 服务抵扣金](#)的系统。这些服务抵扣金是在一天中的闲置时间段内逐步累积起来的。Burstable 实例提供不同的 vCPU-to-RAM比率，这使得它们在某些情况下是计算优化实例的替代方案，而在另一些情况下则是其他通用实例的替代方案。

以下示例说明了在 us-east-1 区域运行的 T3 实例 (即可爆发实例) 的每小时定价。

实例	v CPUs	RAM (GB)	每小时价格
t3.nano	2	0.5	0.0052
t3.micro	2	1	0.0104
t3.small	2	2	0.0208
t3.medium	2	4	0.0416
t3.large	2	8	0.0832
t3.xlarge	4	16	0.1664 美元
t3.2xlarge	8	32	0.3328 美元

Note

定价基于 us-east-1 区域的每小时按需定价。

了解处理器架构之间的价格差异

自 EC2 实例问世以来，[Intel](#) 处理器一直是其标准配置。前几代 EC2 实例（例如 C5、M5 和 R5）并未将 Intel 指定为处理器架构（因为它是默认架构）。新一代的 EC2 实例（例如 C6i、M6i 和 R6i）都包含一个“i”，用于表示使用 Intel 处理器。

处理器架构注释的变化是由于引入了其他处理器选项。与 Intel 最具可比性的处理器是 [AMD](#)（用“a”表示）。AMD EPYC 处理器使用相同的 x86 架构，其性能与 Intel 处理器相似，但价格更低。如以下定价示例所示，AMD 的 EC2 实例在计算成本方面相较于 Intel 同类产品大约可节省 10% 的费用。

Intel 实例	每小时价格	AMD 实例	Price	差异 %
c6i.xlarge	0.17 美元	c6a.xlarge	0.153 美元	10%
m6i.xlarge	0.192	m6a.xlarge	0.1728 美元	10%
r6i.xlarge	0.252 美元	r6a.xlarge	0.2268 美元	10%

Note

定价基于 us-east-1 区域的每小时按需定价。

第三个主要的处理器架构选项是 EC2 实例上的 [AWS Graviton 处理器](#)（用“g”表示）。Graviton 处理器由设计 AWS，在 Amazon EC2 上具有最佳的性价比。当前的 Graviton 处理器不仅比 Intel 同类产品便宜 20%，而且性能提升幅度也达到了 20% 或更高。下一代 Graviton 处理器预计将进一步扩大这种性能差异，测试结果表明其性能将再提高 25%。

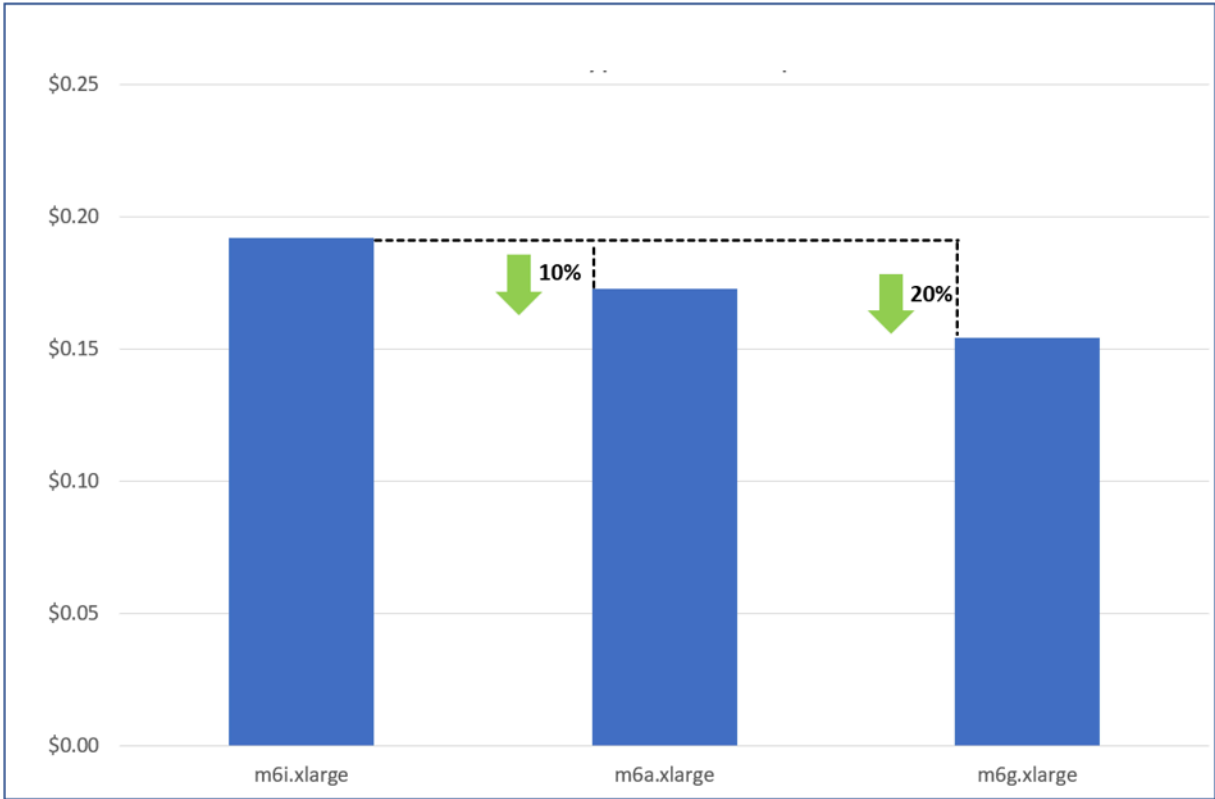
Windows Server 无法在基于 ARM 架构的 Graviton 处理器上运行。实际上，Windows Server 只能在 x86 处理器上运行。虽然使用基于 Graviton 架构的 Windows Server 实例无法实现 40% 的性价比提升，但您仍可以将 Graviton 处理器用于特定的 Microsoft 工作负载。例如，[较新版本的 .NET 可以在 Linux 上运行](#)。这意味着这些工作负载可以使用 ARM 处理器，并能从更快、更经济实惠的 Graviton EC2 实例中获益。

以下示例说明了在 us-east-1 区域运行的 Graviton 实例的每小时定价。

Intel 实例	每小时价格	Graviton 实例	每小时价格	差异 %
c6i.xlarge	0.17 美元	c6g.xlarge	0.136 美元	20%
m6i.xlarge	0.192	m6g.xlarge	0.154	20%
r6i.xlarge	0.252 美元	r6g.xlarge	0.2016	20%

 **Note**
定价基于 us-east-1 区域的每小时按需定价。

下图比较了 M 系列实例的价格。



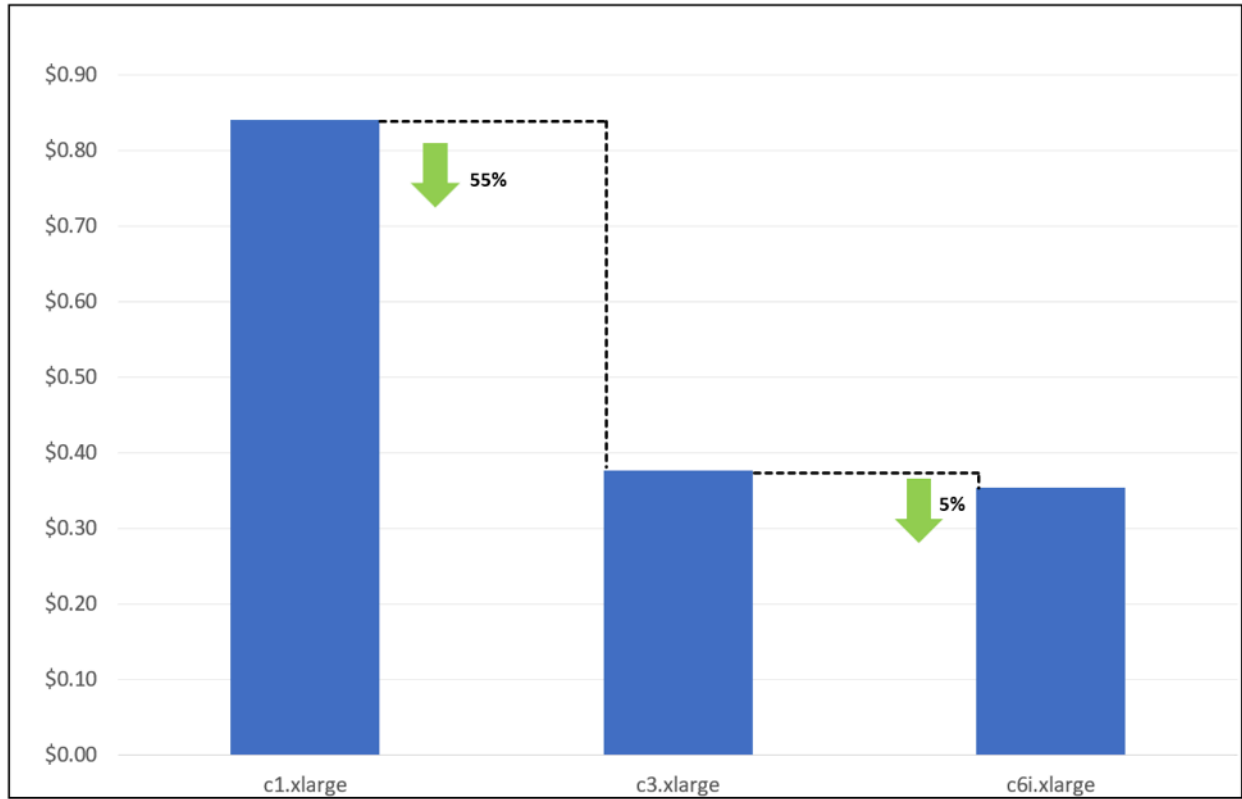
了解各代 EC2 实例的性价比差异

Amazon EC2 的一个最为一致的特点是，每一代新产品都比前一代具有更优的性价比。如下表所示，新一代 EC2 实例的价格会随着每个后续版本的发布而降低。

计算优化型实例	每小时价格	通用实例	每小时价格	内存优化型实例	每小时价格
C1.xlarge	0.52 美元	M1.xlarge	0.35 美元	r1.xlarge	不适用
C3.xlarge	0.21 美元	M3.xlarge	0.266 美元	r3.xlarge	0.333 美元
C5.xlarge	0.17 美元	M5.xlarge	0.192	r5.xlarge	0.252 美元

 **Note**
定价基于 us-east-1 区域的每小时按需定价。

下图比较了不同代 C 系列实例的成本情况。



但是，第 6 代实例的价格与第 5 代实例的价格相同，如下表所示。

计算优化型实例	每小时价格	通用实例	每小时价格	内存优化型实例	每小时价格
C5.xlarge	0.17 美元	M5.xlarge	0.192	r5.xlarge	0.252 美元
C6i.xlarge	0.17 美元	M6i.xlarge	0.192	r6i.xlarge	0.252 美元

 **Note**
定价基于 us-east-1 区域的每小时按需定价。

尽管成本相同，但新一代产品凭借更快的处理器、更高的网络吞吐量以及更大的 Amazon Elastic Block Store (Amazon EBS) 吞吐量和 IOPS，提供更出色的性价比。

最显著的性价比改进之一是 [X2i 实例](#) 的增强。与上一代实例相比，这一代实例的性价比最高可提高 55%。如下表所示，x2iedn 在各项性能指标上均有所提升（且价格与前一代产品相同）。

实例	每小时价格	v CPUs	RAM	处理器速度	实例存储	Networking	Amazon EBS 吞吐量	EBS IOPS
x1e.2xlarge	1.66 美元	8	244	2.3 GHz	237GB SSD	10 Gbps	125 MB/s	7400
x1iedn.2xlarge	1.66 美元	8	256	3.5 GHz	240GB 固态硬盘 NVMe	25 Gbps	2500 MB/s	65000

 **Note**
定价基于 us-east-1 区域的每小时按需定价。

应用场景示例

以一家负责追踪运输车辆并希望提升其 SQL Server 性能的分析公司为例。在 MACO SME 审查了该公司的性能瓶颈后，该公司从 x1e.2xlarge 实例转换到 x2iedn.xlarge 实例。新实例大小较小，但对 x2 实例所做的改进通过使用缓冲池扩展，提高了 SQL Server 的性能和优化效果。这使该公司能够从 SQL Server 企业版降级到 SQL Server 标准版。它还使该公司能够将其 SQL Server 许可从 8 v 减少到 4 v CPUs。

优化前：

服务器	EC2 实例	SQL Server 版本	月度成本
Prod DB1	x1e.2xlarge	Enterprise	3,918.64 美元
Prod DB2	x1e.2xlarge	Enterprise	3,918.64 美元
总计			7,837.28 美元

优化后：

服务器	EC2 实例	SQL Server 版本	月度成本
Prod DB1	x2iedn.xlarge	标准	1,215.00 美元
Prod DB2	x2iedn.xlarge	标准	1,215.00 美元
总计			2,430.00 美元

总而言之，从 x1e.2xlarge 实例到 x2iedn.xlarge 实例的这一更改使示例场景中的公司每月在其生产数据库服务器上节省了 5407 美元的费用。这使工作负载的总成本降低了 69%。

Note

定价基于 us-east-1 区域的每小时按需定价。

迁移到较新的实例

Amazon EC2 的早期版本在 Xen 虚拟机监控程序上运行，而新一代则在 [AWS Nitro System](#) 上运行。Nitro System 将主机硬件的几乎所有计算和内存资源都提供给您实例。这会提高整体性能。[从 Xen 迁移到基于 Nitro 的实例时](#)，需要考虑一些特殊因素。例如，[AWS Windows AMIs](#) 使用微软安装媒体使用的默认设置和自定义项进行配置。定制包括支持最新一代实例类型（[在 Nitro System 上构建的实例](#)）的驱动程序和配置。

如果您要从自定义 Windows AMIs 或亚马逊 AMIs 提供的 Windows 启动在 2018 年 8 月之前创建的实例，我们建议您完成 Amazon EC2 文档中从[迁移到最新一代实例类型的](#)步骤。

使用可爆发实例

虽然可爆发实例是节省计算成本的好方法，但我们建议您在以下情况下避免使用这些实例：

- 具有桌面体验的 [Windows Server 的最低规格](#) 要求有 2GB 的 RAM。避免将 t3.micro 或 t3.nano 实例与 Windows Server 结合使用，因为它们缺少最低的 RAM 量。
- 如果您的工作负载波动较大，但又无法长时间保持闲置状态以积累突增积分，则使用普通 EC2 实例要比使用可爆发实例更高效。我们建议[监控您的 CPU 积分](#)以验证这一点。
- 在大多数情况下，我们建议您避免在 SQL Server 中使用可爆发实例。SQL Server 的许可基于 CPUs 分配给实例的 v 数。如果 SQL Server 一天中大部分时间处于闲置状态，则您需要为未充分利用的 SQL 许可证支付费用。在这些情况下，我们建议您将多个 SQL Server 实例整合到一个更大的服务器上。

后续步骤

我们建议您采取以下后续步骤来优化 Amazon EC2 Windows 实例的成本：

- 使用最新一代的 EC2 实例以获得最佳性价比。
- 使用搭载 AMD 处理器的 EC2 实例将计算成本降低 10%。
- 通过选择与您的工作负载相匹配的 EC2 实例类型，最大限度地提高资源利用率。

下表列出了适用于 Windows 工作负载的典型起始点示例。还有其他选项可供选择，例如用于增强 SQL Server 工作负载的实例存储卷或具有更大 vCPU-to-RAM 比率的 EC2 实例。我们建议您对工作负载进行全面测试，并使用 AWS Compute Optimizer 之类的监控工具来帮助进行必要的调整。

工作负载	典型	可选
Active Directory	T3、M6i	R6i
文件服务器	T3、M6i	C6i
Web 服务器	T3、C6i	M6i、R6i
SQL Server	R6i	x2iedn、X2iezn

如果您必须更改 EC2 实例类型，通常只需对服务器进行一次重启即可完成这一操作。有关更多信息，请参阅 Amazon EC2 文档中的[更改实例类型](#)。

在更改您的实例类型之前，我们建议您考虑以下内容：

- 您必须先停止由 Amazon EBS 支持的实例，然后才能更改其实例类型。当实例停止时，请务必计划停机时间。停止实例并更改其实例类型可能需要几分钟时间，重新启动实例所用的时间则由应用程序的启动脚本决定。有关更多信息，请参阅 Amazon EC2 文档中的[停止和启动您的实例](#)。
- 当您停止并启动实例时，AWS 会将该实例移至新硬件。如果您的实例有公有 IPv4 地址，则 AWS 释放该地址并为您的实例提供新的公有 IPv4 地址。如果您需要一个不变的公共 IPv4 地址，请使用[弹性 IP 地址](#)。
- 您不能更改已启用了[休眠](#)的实例的实例类型。
- 您无法更改[竞价型实例](#)的实例类型。
- 如果您的实例处于自动扩缩组中，则 Amazon EC2 Auto Scaling 会将已停止的实例标记为运行状况不佳，可能会终止它并启动替换实例。为防止出现此情况，您可以在更改实例类型时，为组暂停扩展流程。有关更多信息，请参阅 Amazon EC2 Auto Scaling 文档中的[暂停和恢复自动扩缩组的进程](#)。
- 当您更改带有 NVMe 实例存储卷的实例的实例类型时，更新的实例可能会有额外的实例存储卷，因为即使未在 Amazon 系统映像 (AMI) 或 NVMe 实例块储存设备映射中指定，所有实例存储卷都可用。否则，已更新的实例通常具有您在启动原始实例时指定的相同实例存储卷数。

其他资源

- [亚马逊 EC2 实例类型](#) (AWS 文档)
- [AWS 优化与许可评测](#) (AWS 文档)

带来适用于 Windows 和 SQL Server 工作负载的许可证

概述

如果你在 Microsoft 工作负载和现有企业许可协议方面有大量投资，则可以从多个 AWS 选项中进行选择来支持这些工作负载，包括[随附许可证（由提供 AWS）](#)和[自带许可（BYOL）](#)选项。您可以使用[Amazon EC2 专属主机](#)来充分利用现有的 Microsoft 许可协议，将 Windows Server 带入 AWS。这可以为您节省最多 50% 的 Amazon EC2 实例成本。由于 Windows 许可证约占实例成本的一半，因此将 Windows Server 置于 AWS 于专用主机上可以节省大量成本。由于无法将 Windows Server 置于[默认（共享）租期](#)，因此，如果您想在上 AWS 面使用现有的 Windows Server 许可证，专用主机是理想的选择。

专属主机不仅仅适用于 Windows Server BYOL 实例。它们还为您提供为现有 SQL Server 工作负载匹配本地许可的灵活性。专属主机公开底层服务器的物理核心，使您能够在物理核心级别上许可 SQL Server。在默认（共享）租赁中，这是不可能的，其中 SQL Server 许可基于 CPUs 分配给实例的虚拟数量。此功能使您能够以符合本地许可策略 AWS 的方式对 SQL Server 工作负载进行许可。因此，通过使用符合条件的 Windows 许可证，您在 SQL Server 的许可费用方面可节省最高 50% 的成本（相较于默认（共享）租赁模式），此外在实例费用方面也能实现成本节省。有关此场景的更多信息，请参阅本指南的[了解 SQL Server 许可](#)部分。

Amazon EC2 专属主机

Amazon EC2 专用主机与用于运行其 EC2 计算产品的 EC2 主机基本相同。AWS 其区别在于，这些主机完全专用于单个客户，并为客户提供对底层物理基础设施的独占访问权限。您可以使用专属主机在完全专供您使用的硬件上运行您的实例，而不必与其他 AWS 客户共享资源。这能让您更好地控制云资源，并且通过自带 Windows Server 和 SQL Server 之类的软件许可证到 AWS 中，帮助您降低成本。

记住以下内容：

- 专属主机是一种专供单一客户的物理服务器。您能够查看专属主机的套接字和物理核心，从而能够满足许可合规要求，例如每个套接字、每个核心或每个 VM 的软件许可协议。
- 可以支持同一实例系列的多个实例大小的专属主机被称为异构专属主机。这些[实例系列](#)包括 T3、A1、C5、M5、R5、C5n、R5n 和 M5n。相比之下，其他实例系列在相同专属主机上仅支持一种实例大小。它们被称为同构专属主机。
- 专属主机按每台主机计费。这意味着，无论在专属主机上运行多少实例，您都是按每台专属主机付费。专属主机的定价因实例系列、区域和所选的付款选项而异。您可以为工作负载选择最佳配置，以实现所需的性能和成本结果。

此图表展示了共享租赁实例和专属主机之间的区别。



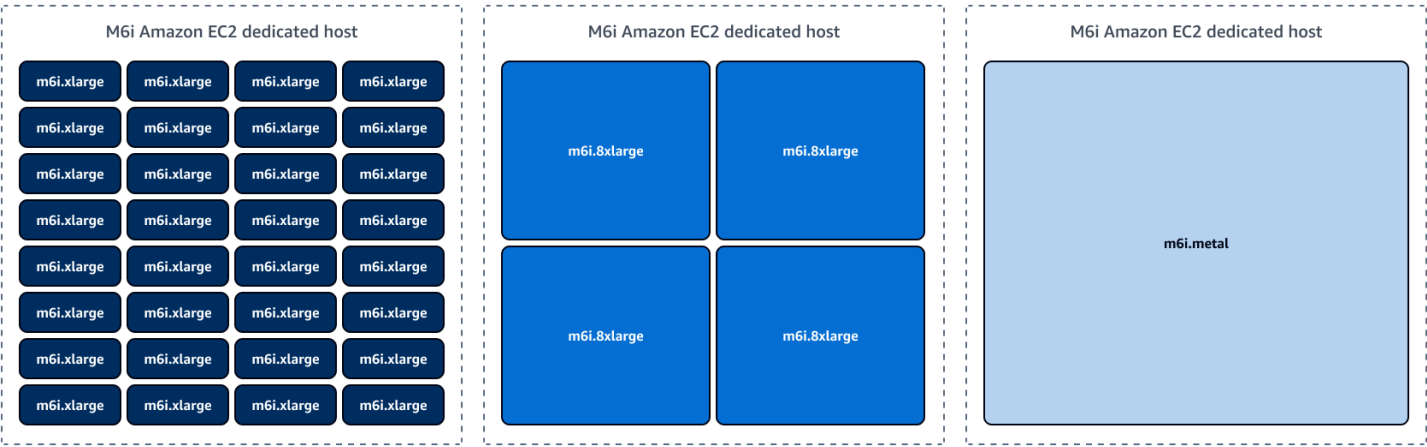
同构专属主机

以使用 M6i 专属主机的场景为例。M6i 和 R6i 专属主机有两个套接字、64 个物理核心，并支持相同大小的实例类型。它们被称为同构专属主机。这意味着，在单个 M6i 专属主机上能够启动的实例数量取决于实例的大小。

例如：

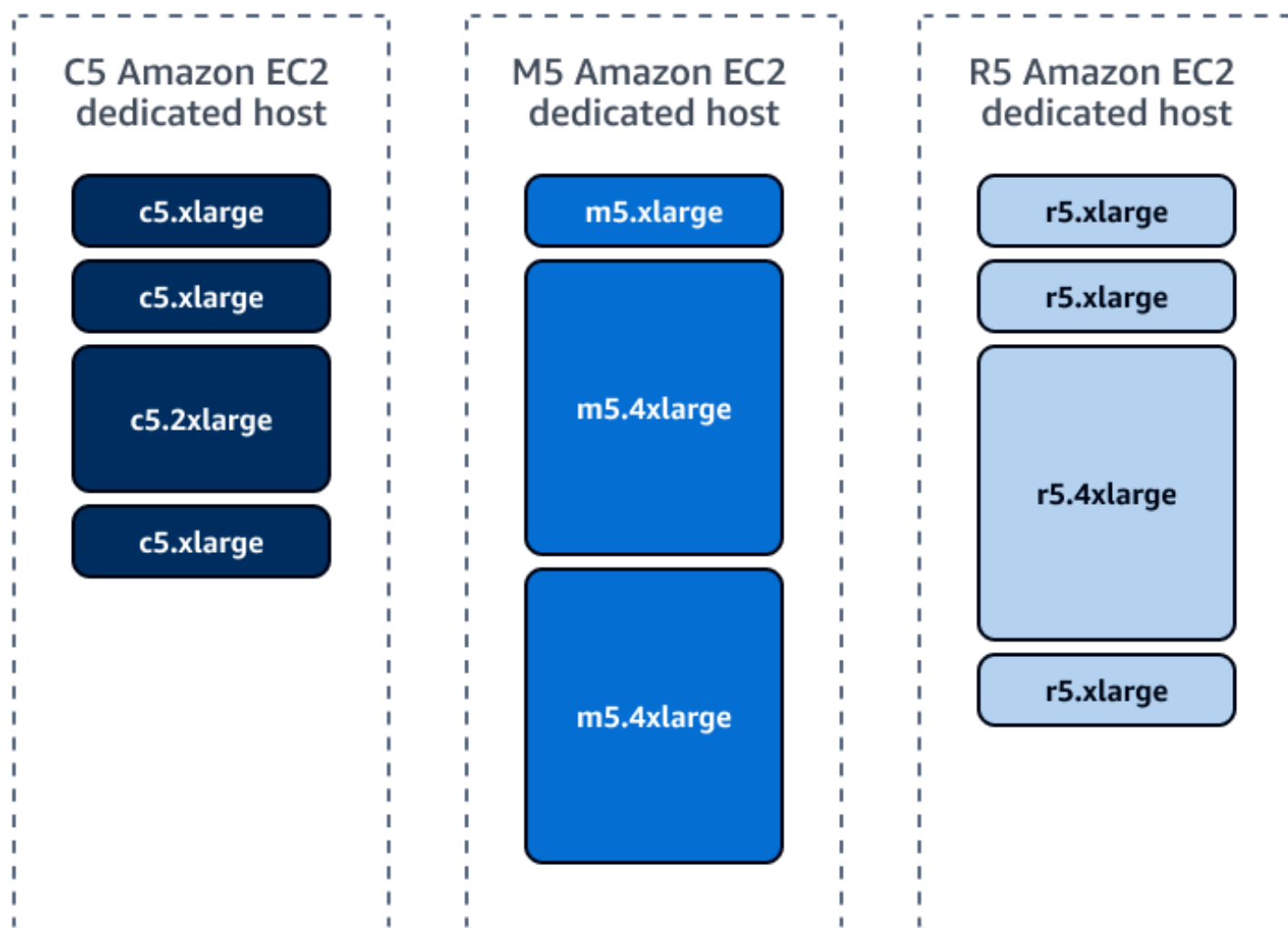
- 对于 xlarge (4 vCPUs)，您最多可以在此专用主机上启动 32 个 m6i.xlarge 实例。
- 对于 8xlarge (32 vCPUs)，您最多可以在此专用主机上启动 4 个 m6i.8xlarge 实例。
- 对于金属 (128 VCPUs)，您最多可以在此专用主机上启动 1 m6i.metal 实例。

以下图表显示了适用于 M6 实例的专属主机选项。



异构专属主机

在同一主机上支持多种实例大小的专属主机被称为异构 Amazon EC2 专属主机。以下图表显示了具有各种实例大小（例如 2xlarge、xlarge 和 4xlarge）的 C5、M5 和 R5 专属主机的示例。



专属主机管理

我们建议您在管理 Amazon EC2 专属主机时考虑以下几点：

- 要充分利用专属主机，您可以在[组织内的多个账户之间共享一台主机](#)。主机共享可实现资源优化，并且通过充分利用主机上的每一个可用槽位，还能节省成本。通过在业务部门之间共享专属主机，您可以集中管理您的 IT 基础设施并提高资源利用率，同时仍能保持工作负载之间的分离。如果您是组织中的一员，AWS Organizations 并且您的组织内启用了共享，则会自动授予组织中的消费者访问共享专用主机的权限。否则，使用者会收到加入资源共享的邀请，并在接受邀请后获得对共享专用主机的访问权限。

- 您可以在随附许可证模型下，在专属主机上运行 Windows Server 2022，因为 Windows Server 2019 是您可以 BYOL 的最新版本。如果您想在专属主机上使用 Windows Server 2022，则必须使用 Windows Server 2022 随附许可证的实例。
- [AWS License Manager](#) 是一款全面的解决方案，用于管理来自不同供应商的跨本地环境 AWS 的软件许可证。如果您[使用 License Manager](#)，则可以更好地了解和控制软件许可证的使用方式，从而节省成本并提高合规性。您可以使用 License Manager 来设定规则，以模拟您特有的许可条款。这使得您能够执行这些规则，并防止许可证被滥用。这可以降低不合规的风险，并改善许可证管理流程。
- 您可以使用 License Manager 通过[主机资源组](#)来自动执行主机的放置、释放和恢复。这可以提高工作效率并减少管理开销。License Manager 还提供基于许可规则的跨 AWS 本地环境的许可证使用情况的集中视图，从而可以轻松管理整个组织的增量许可购买、合规性和供应商审计。此外，License Manager 还与 AWS Organizations 和 AWS Resource Access Manager (AWS RAM) 集成，可跨账户和地区共享许可证配置。这使您能够根据计划为整个环境创建报告，并在一个 AWS 账户中集中管理许可规则。最终，这可以改善治理并降低复杂性。
- 在为单个区域内的专属主机设计高可用性时，请务必确保为生产关键型工作负载在至少两个可用区中分配了最少两个专属主机。有关更多信息，请参阅[适用于 AWS 上的 Microsoft Windows 的 Amazon EC2 专属主机](#)参考部署。
- 对于每个专属主机实例系列，每个实例大小可以运行的实例数量都有限制。有关更多信息，请参阅 Amazon EC2 文档中的[专属主机配置表](#)。

AWS 许可选项

许可证主要分为以下几类：

- 随附许可证：此许可选项允许您按需购买并使用许可证，仅需按实际使用量付费。它非常适合您希望在许可使用方面具有灵活性，并且希望避免前期费用的使用案例。您可以从各种 Windows Server、SQL Server 和其他 Microsoft 产品中进行选择。
- 具有许可证移动性的 BYOL 产品：如果您已拥有现有许可证并希望将其用于云中，则此许可选项允许您通过[Microsoft 的许可证移动性计划](#)将自己的许可证带入云中使用。可以将具有许可证移动性的产品（例如带软件保障（SA）的 SQL Server）带到共享或专用租赁中。这降低了 AWS 实例成本。
- 没有许可证移动性的 BYOL 产品 — 对于缺少许可证移动性的微软产品，如 Windows Server，AWS 提供了在云中使用这些产品的专用选项。此外，专属主机还能够在物理核心层面实现许可，从而能够节省 50% 或以上运行工作负载所需的许可证费用。专属主机对于那些大部分时间都能保持稳定且可预测的工作负载是一个出色的选择。

自带 Windows Server 许可证

自带 Windows 许可证是最有效的许可证优化策略之一，因为它使您能够利用现有投资并减少 AWS 开支。特定的 BYOL 场景不需要 SA 或许可证移动性权益，但是 Amazon EC2 的专属基础设施始终是必需的。要符合资格，您必须已在 2019 年 10 月 1 日之前购买了永久许可证，或者在 2019 年 10 月 1 日之前已激活的企业注册中将其作为调整项添加进来。在这些特定的 BYOL 场景中，您只能将许可证升级到 2019 年 10 月 1 日之前可用的版本。例如，如果您在 2017 年放弃了 SA，则最高只能部署到 Windows Server 2016，而不是 2019。但是，2019 年是最后一个有资格使用 BYOL 的 AWS 版本。有关更多信息，请参阅 AWS 文档中的[许可-Windows 服务器](#)。

自带许可证可能会对在 AWS 上运行 Microsoft 工作负载的成本产生显著影响。如果您自带许可证，则无需为在云中运行的实例支付额外的许可费用，这会带来相当可观的成本节省。

下表展示了在各种配置上全天候运行单个 c5.xlarge 实例的按需每月成本。

配置	每月成本 (美元)
Windows Server + SQL Server 企业版	1353.00 美元 (LI)
Windows Server + SQL Server 标准版	609.00 美元 (LI)
仅限 Windows Server	259.00 美元 (LI)
仅限计算 (Linux)	127.00 美元

您可以使用现有的许可证来降低许可成本并节省总 AWS 账单。

要获得 Amazon EC2 专属主机上的 BYOL 资格，您必须自带软件许可证，例如 Windows Server 和 SQL Server 的软件许可证。BYOL 允许您在上使用现有许可证 AWS，从而节省成本。要自带许可证，您必须拥有软件供应商提供的许可证授权，并且还必须提供该软件的安装介质或映像。安装介质或映像可用于在专属主机上启动实例。要了解有关创建 BYOL AMI 的更多信息，请参阅博客上的 Microsoft 工作负载中的[如何使用虚拟机导入/导出 Bring-Your-Own-License AMIs 从本地创建 Windows 服务器](#)。AWS

Note

设置为自动的许可证类型相当于 [AWS 随带许可证选项](#)。此选项可能会导致不必要的按需支出。您将需要切换[许可类型](#)。

成本优化方案

调整和优化许可证是 AWS 上的成本优化的关键组成部分。如果您实施适当的策略，则可以通过 Amazon EC2 专属主机和 BYOL 选项降低许可成本，维护合规性，并从您的许可投资中获得最佳效益。

此部分涵盖以下示例场景：

- 使用 T3 专属主机节省成本
- 将共享租赁与使用 SQL Server BYOL 的专属主机进行比较
- 高可用性的 SQL Server 部署

使用 T3 专属主机节省成本

T3 专属主机与传统上提供固定 CPU 资源的其他 Amazon EC2 专属主机有所不同。相比之下，T3 专属主机支持可爆发实例，这些实例能够共享 CPU 资源，提供基准 CPU 性能，并在需要时爆发。共享 CPU 资源（也称为超额订阅）使得单个 T3 专属主机能够支持的实例数量是同类通用专属主机的四倍之多。

T3 专属主机通过提供比其他任何 Amazon EC2 专属主机更高的实例密度，实现更低的 TCO。Burstable T3 实例使您能够在比以往任何时候更少的主机上整合 low-to-moderate 更多具有平均 CPU 利用率的实例。与其他 Amazon EC2 专属主机相比，T3 专属主机还能在更多的 vCPU 和内存组合中提供更小的实例大小。较小的实例大小有助于降低 TCO，并能够实现与本地主机相当或更高的整合率。

T3 专用主机最适合运行具有 low-to-moderate CPU 利用率和符合条件的每插槽、每内核或每虚拟机软件许可证的 BYOL 软件，包括微软 Windows 桌面、Windows Server、SQL Server 和 Oracle 数据库。

使用 T3 专属主机减少 Windows Server 数据中心许可证（每个核心）

在本地环境中，您可以利用这样一个事实，即您可以轻松地 CPUs 在 VMware 主机上超额订阅物理设备并实现高水平的整合。

考虑以下示例。您目前在本地环境中使用的是 10x36 核心、384 GB RAM VMware 主机。此外，每台主机都运行 96x2 vCPU、4GB RAM 的 Windows Server 虚拟机，且其平均 CPU 利用率较低。

现在，您可以通过将虚拟机迁移到 T3 专用主机来实现更高的整合级别，T3 专用主机的 RAM 量是当前本地 VMware 主机的两倍。您可以在 T3 专属主机上运行相同数量的服务器，同时将主机成本降低

50%。这可以帮助您将 Windows 服务器的许可成本降低 33%。以下表格重点介绍使用 T3 专属主机所带来的节省情况。

	本地 VMware 主机	T3 专属主机	节省成本
物理服务器	10	5	
每台主机的物理核心	36	48	
每台主机的 RAM (GB)	384	768	
2 个 vCPU , 每台主机 4 GB 内存 VMs	96	192	
的总数 VMs	960	960	
Windows Server 数据中心许可证总数 (每个核心) = (服务器数量 * 物理核心数量)	10 * 36 = 360	5 * 48 = 240	33%

将共享租赁与使用 SQL Server BYOL 的专属主机进行比较

以一个实际的示例来说明 Amazon EC2 专属主机的价值。在此情境中，一个组织在拥有 240 个核心的本地环境中运行着一个 SQL Server 工作负载，并且希望以经济高效的方式将同样的工作负载部署到 AWS上。如果该组织自带许可证（BYOL），则他们仍需支付 SA 费用，并且减少核心数量会直接影响其成本。

下图比较了微软授权和 SQL Server 之间 AWS 节省的费用。

Microsoft entitlements (Enterprise Agreements)		SQL Server savings with AWS	
	Number of cores	AWS shared vCPUs	AWS BYOL/Dedicated Hosts cores
SQL Server Enterprise edition	208	120	96
SQL Server Standard edition	32	20	-
Total SA cost	\$341,000	\$197,418	\$151,355

通过在 AWS 共享租赁上调整实例的大小，您可以将 SQL Server 许可证减少到 140 个内核。这导致产生 197000 美元的 SA 费用。

Amazon EC2 专属主机使您能够在物理核心级别上许可 SQL Server。在 SQL Server 许可基于 CPUs 分配给实例的 v 数量的共享租赁中，这是不可能的。因此，通过使用两台 R5 专用主机，每台 48 个内核，您只需要覆盖 96 个内核，而不是共享租赁 CPUs 所需的 140 v。通过部署 R5 专属主机并在物理层面许可工作负载，您可以将所需数量的 SQL Server 企业版许可证减少到 96 个核心。这意味着您可以部署多达 192 个核心（考虑到超线程）的 SQL Server 工作负载，同时仍能满足许可要求并节省大量成本。

在此案例中，组织每年支付约 34.1 万美元的 SA 费用。在调整共享租赁后，他们使用 140 个 vCPU 将成本降低到 197000 美元。Amazon EC2 专属主机将成本进一步降低至 15.1 万美元（减少了约 56%）。

高可用性的 SQL Server 部署

此示例通过各种许可考虑因素分析成本如何影响 AWS 上的 SQL Server 的部署。假设一个组织需要部署六台 SQL Server Enterprise 服务器 AWS 来支持三个应用程序。这些服务器需要高可用性，并且每台服务器都有 16 v CPUs 和 256 GB 的 RAM。请查看以下方案详细信息：

- 服务器：SQL Server
- 操作系统版本：Windows Server Datacenter 2019
- SQL Server 版本：SQL Server Enterprise 2019
- vCPU：16
- 内存（GB）：256
- 数量：6

为了在 AWS 不牺牲性能的情况下优化成本，我们建议您根据 CPU、内存、网络 and 磁盘 (IOPS/BW) 利用率调整实例的大小。正确调整工作负载大小后，将其放在提供 16 v 的 x2iedn.4xlarge 实例类型上。CPUs 但是，此实例类型还包括工作负载所需的两倍内存。进一步的优化仍是可以实现。

方案 1

一个组织使用适用于 Windows 和 SQL Server 的“包含许可证”选项，在 AWS 共享租约上部署了六台 SQL Server Enterprise 服务器。使用此选项，Windows 和 SQL Server 许可证的成本将包含在实例价格中。请查看以下方案详细信息：

- 共享租赁（实例）：x2iedn.4xlarge
- 每小时成本（美元）：10.0705 美元
- 每月每单位成本（美元）：7351.47 美元

- 服务器数量：6
- CPU：16
- 内存：512
- 6 台服务器的每月成本：44108 美元

方案 2

一个组织在共享租赁模式下为 SQL Server 配备了 SA 和 BYOL。这意味着该组织使用适用于 Windows 的“包含许可证”选项，但会根据 CPUs 分配给实例的 v 数量提供自己的 SQL Server 许可证。由于该组织有六台 SQL Server Enterprise 服务器，CPUs 每台 16 v，CPUs 因此总共需要 96 v。请查看以下方案详细信息：

- 共享租赁（实例）：x2iedn.4xlarge
- 每小时成本（美元）：4.0705 美元
- 每月每单位成本（美元）：2971.47 美元
- 服务器数量：6
- CPU：16
- 内存：512
- BYOL 核心：96
- 6 台服务器的每月成本：17828 美元

通过自带具有 SA 的 SQL Server 许可证，该组织在此情况下与使用 SQL Server 的随附许可证选项相比可实现成本节省。确切的成本节省因特定的许可协议的定价和条款而有所不同。在这种情况下，将 SQL Server Enterprise 许可证引入后，每月 AWS 的成本将减少 26,280 美元。AWS

方案三

一个组织在 Amazon EC2 专属主机上同时为 Windows 和 SQL Server BYOL。这意味着该组织将在物理核心级别分配许可证，使他们能够仅许可主机的物理核心。在物理核心级别进行许可可能让您在不影响所需许可证数量的情况下部署最多的实例。此许可模式通常用于 Windows Server 数据中心版和 SQL Server 企业版。

此方案使用两台 X2iezn Amazon EC2 专属主机。每台主机有 24 个物理内核和 48 v CPUs。这为六台 SQL Server Enterprise 服务器提供了足够的容量，每台服务器具有 16 v CPUs 和 256 GB 的内存。请查看以下方案详细信息：

- 专属主机数量：2
- 实例系列：x2iezn
- 每小时成本 (美元)：11.009 美元
- 每月每单位成本 (美元)：8036 美元
- 物理核心：48
- 可用的 vCPU：96
- 所需的 Windows Server 核心许可证数量：24
- SQL Server 企业版核心所需的许可证数量：24
- 每月成本：16073

两台 X2iezn 系列 Amazon EC2 专属主机的总成本为每月 16073 美元。有关定价的更多信息，请参阅此场景的 AWS 定价计算器 [估算值](#)。该组织在此情况下可通过自带 Windows 许可证每月节省 1755.65 美元的费用。如果他们使用 Amazon EC2 专属主机，他们还可以减少所需的 SQL Server 许可证数量。在共享租赁中，他们需要 96 个 SQL Server 企业版许可证才能覆盖六台各有 16 个 vCPU 的 SQL Server 企业版服务器。但是，通过使用 Amazon EC2 专属主机和在物理核心级别进行许可，他们可以将所需的许可证数量减少到 48 个核心。

以下详细信息比较了示例 3 中的成本，并显示了与其他方案相比，使用 BYOL 选项在 Amazon EC2 专属主机上部署工作负载可以节省多少成本。

- 本地服务器：SQL Server
- vCPU：16
- 内存：256
- 服务器数量：6
- 方案 1 的每月费用：Windows (LI) + SQL Server 企业版 (LI)：44108 美元
- 方案 2 的每月费用：Windows (LI) + SQL Server 企业版 (BYOL)：17828 美元
- 方案 3 的每月费用：Amazon EC2 专属主机上的 Windows (LI) + SQL Server 企业版 (BYOL)：16073 美元

Note

费用基于按需定价。您可以使用节省计划或专属预留实例进一步降低成本。这些选项提供了一种灵活的定价模式，与按需定价相比，能显著节省成本。有了这些计划，您可以承诺签订为期

一年或三年的合同。有关更多信息，请参阅本指南的[优化 Amazon EC2 上的 Windows 的支出](#)部分。

针对 Amazon EC2 专属主机，考虑以下付款选项：

- [专属主机](#) (Amazon EC2 文档)
- [专属主机预留](#) (Amazon EC2 文档)
- [节省计划](#) (Amazon EC2 文档)

[AWS 定价计算器](#) 现在支持专属主机定价。这可以帮助您选择合适的底层专属主机。

成本优化建议

我们建议您通过使用 AWS Cost Explorer 采取以下后续步骤优化成本：

1. [启用 Cost Explorer 成本管理](#)服务。
2. 使用 Cost Explorer 成本管理[服务查看和分析您的 Amazon EC2 专属主机部署的成本和使用情况](#)。
3. 验证您是否正在运行 BYOL。您可以在 Amazon EC2 控制台的实例或 AMI 页面上显示以下平台详细信息和使用情况操作值，或者将其显示在由 describe-images 或 describe-instances 命令返回的响应中。
 - 平台详情：Windows，使用操作:: 0002 RunInstances (包括许可证)
 - 平台详情：Windows BYOL，使用操作:: 0800 RunInstances

其他资源

- [符合许可证类型转换条件的许可证类型](#) (AWS License Manager 文档)
- [AWS License Manager 和专用主持人研讨会](#) (AWS License Manager 研讨会)
- [亚马逊 EC2 专用主机 FAQs](#) (AWS 文档)
- [如何使用虚拟机导入/导出 Bring-Your-Own-License AMIs 从本地创建 Windows 服务器](#) (博客上的 AWS 微软工作负载)
- [虚拟机导入/导出 \(文档 \)](#) AWS
- [亚马逊 Web Services 和微软：常见问题解答](#) (AWS 文档)
- License [Manager 中的许可证类型转换](#) (AWS License Manager 文档)
- 在 [Amazon EC2 专用主机上部署高度可用的 SQL Server](#) (AWS 云运营和迁移博客)

优化 Amazon EC2 上的 Windows 的支出

概述

将服务器迁移到的首要问题之一 AWS 是基础架构成本。诚然，云技术的一个优点在于按需付费使用资源，但有些生产工作负载则需要全年无休地持续运行。Savings Plans 旨在为您在 EC2 实例的 AWS 稳定状态下节省资金 AWS Lambda、和。 AWS Fargate

Savings Plans 提供灵活的定价模式，可以帮助您降低亚马逊 EC2、Fargate、Lambda 和 Amazon A SageMaker I 使用量的定价，以换取对稳定的使用量（例如 10 美元/小时）的承诺。您需承诺在一年或三年内保持固定的每小时计算费用水平，作为交换，您将获得相应的使用折扣。

节省计划有三种不同的付款方式可供选择：

- 无费用预付选项不需要任何预付款，并且您的承诺费用完全按月收取。
- 预付部分费用选项为节省计划提供更低的价格。您需要先支付至少一半的费用，其余部分则按月支付。
- 预付全部费用选项提供最低的价格，您的全部承诺费用均通过一次付款收取。

您可以在 AWS Cost Explorer 中跟踪节省计划的到期日期，以及即将购买的已加入队列的节省计划。您可以使用节省计划提醒在计划到期日前 1、7、30 或 60 天或在将承诺加入队列等候购买时收到电子邮件提醒。这些通知还会在到期日期向您发出提醒。您最多可以向 10 个电子邮件收件人发送通知。

了解节省计划

每种计算使用类型都有一个按需计费费率和一个节省计划费率。如果您承诺每小时使用计算资源 10 美元，那么您在所有不超过 10 美元的使用量上都将按照节省计划费率享受节省计划的价格优惠。超出计算支出承诺范围的任何使用均根据固定的按需费率收费。您可以通过在 AWS 管理控制台中使用 Cost Explorer 成本管理服务开始使用节省计划。

通过使用 [Cost Explorer](#) 成本管理服务中提供的建议，您可以轻松做出节省计划承诺，从而实现最大的节省。建议的每小时承诺使用量取决于您的历史按需使用量以及您选择的计划类型、期限和付款选项。节省计划首先应用于购买该计划的账户，然后与整合账单系列中的其他账户共享。

Note

中的 Savings Plans 共享选项默认 AWS Organizations 处于启用状态。您可以在付款人账户的 AWS Billing 控制台中拒绝此选项。您可以访问 [“推荐”](#) 页面，查看推荐的 Savings P AWS

lans，以帮助您节省符合条件的使用量。这些建议可随时更新，以便您能更便捷地选择最佳节省计划。

计算类节省计划

计算类节省计划提供最大的灵活性，有助于您降低成本。这些计划可自动应用于 EC2 实例的用量，无论实例系列、大小、可用区、区域、操作系统或租期如何。它们也适用于您的 Fargate 和 Lambda 用量。例如，借助计算类节省计划，您可以随时从 C4 更改为 M5 实例，将工作负载从欧洲（爱尔兰）转移到欧洲（伦敦），或者随时将工作负载从 EC2 迁移到 Fargate 或 Lambda。您将自动继续按照节省计划价格付款。

EC2 实例节省计划

EC2 实例节省计划提供最优惠的折扣，但需要您承诺在某一区域使用特定的实例系列（例如，承诺在弗吉尼亚州北部区域持续保持 M5 用量）。这会自动为您在该区域选定的实例系列提供按需价格折扣，无论其可用区、大小、操作系统还是租赁模式如何。EC2 实例节省计划使您可以灵活地更改该区域某个系列中的实例之间的用量。例如，您可以从运行 Windows 的 c5.xlarge 实例迁移到运行 Linux 的 c5.2xlarge 实例，然后自动从节省计划的价格中获益。

计算类和 EC2 实例节省计划都适用于属于 Amazon EMR、Amazon Elastic Kubernetes Service（Amazon EKS）和 Amazon Elastic Container Service（Amazon ECS）集群一部分的 EC2 实例。节省计划不涵盖 Amazon EMR、Amazon EKS 和 Amazon ECS 费用，但底层 EC2 实例包括在内。EC2 实例节省计划在计算类节省计划之前应用，因为计算类节省计划适用范围更广。

Note

一旦做出承诺后，您将无法轻易更改节省计划。我们建议您在承诺使用任一节省计划选项之前仔细规划。节省计划的价格低于按需定价，但需要您做出一定的承诺，并且在合约期间内不可取消。

每小时承诺示例

如果您购买了节省计划，则需要在计划期限内按小时支付相应的费用。如果您承诺每小时使用计算资源 10 美元，则节省计划的定价将自动适用于每小时最高 10 美元的所有用量。超出承诺范围的任何用量均根据固定按需费率收费。您可以使用 Cost Explorer 成本管理服务中的节省计划购买推荐工具来获取推荐的承诺，从而最大限度地节省费用。对于某一特定计划而言，其每小时的财务承诺在计划期限内不可更改。如果您想在分析用量后增加承诺用量，则可以购买额外的节省计划来支付超额使用量。

节省计划优惠

与预留实例相比，节省计划提供了更灵活的定价模式，能帮助您节省费用，同时还能让您充分利用节省计划所提供的更多计算选项。即使您的计算需求发生变化，节省计划也提供折扣。这能够帮助您跟上不断变化的动态环境，同时又不会产生任何额外的管理开销。以下是使用节省计划的其他一些好处：

- 易于使用：获得自动折扣以换取资金承诺。
- 灵活性：适用于多种使用类型的单一承诺。
- 潜在的节省：有多种节省方式。考虑以下示例：
 - 使用计算类节省计划在 Windows 服务器工作负载上节省 60% 的费用 ([d2.8xlarge](#) , 3 年 , 预付全部费用 , Windows , 共享租赁 , us-east-2)
 - 使用 EC2 实例节省计划在 Windows 服务器工作负载上节省 73% 的费用 ([dd2.8xlarge](#) , 3 年 , 预付全部费用 , Windows , 共享租赁 , us-east-2)
 - 非特殊实例类型可节省 28–41 % 的费用 ([t3 系列](#) , 3 年 , 预付全部费用 , Windows , 共享租赁 , us-east-2)
- Windows Server 平均可节省 25-40% 的费用

 Note

由于灵活性降低，EC2 实例节省计划提供的折扣比计算类节省计划更大。您承诺以折扣价使用。

每种计算使用类型都有一个节省计划费率和一个按需计费费率。下表显示了每种操作系统类型的节省计划费率和按需计费费率。您需要根据承诺用量按照节省计划费率付费，超出承诺用量的任何使用均按固定按需费率收费。

实例名称	节省计划费率	按需节省	按需费率	操作系统	Region	付款选项	期限
x2iedn.xlarge	0.32 美元	61%	0.83 美元	Linux	美国东部 (弗吉尼亚州北部)	无预付款	3

实例名称	节省计划费率	按需节省	按需费率	操作系统	Region	付款选项	期限
x2iedn.xlarge	2.01 美元	50%	1.02 美元	Windows	美国东部（弗吉尼亚州北部）	无预付款	3
x2iedn.xlarge	1.02 美元	20%	2.52 美元	Windows 随附许可证 + SQL Server 企业版	美国东部（弗吉尼亚州北部）	无预付款	3
x2iedn.xlarge	0.32 美元	61%	0.83 美元	BYOL	美国东部（弗吉尼亚州北部）	无预付款	3

节省计划包括操作系统，它们对 BYOL 有单独的折扣。它们都已在[计算类节省计划计算器](#)中进行了详细分解。

预留实例定价模型

AWS 还有另一种基于承诺的定价模式，称为预留实例。如果在您做出承诺之后，您的计算发生变化，导致预留实例无法得到充分利用，则这种模式就可能会出现~~问题~~。节省计划旨在提供与[标准和可转换预留实例](#)类似的成本降低，但灵活性要大得多。无论实例系列、大小、操作系统、租赁或区域为何，计算类节省计划都能提供更低的 EC2 实例使用价格。它们还可实现最大的灵活性。

下表可帮助您在节省计划或预留实例之间进行选择。

	Reserved Instance	EC2 实例节省计划	计算类节省计划
1 年的平均折扣	最高 38%	最高 29%	最高 29%
3 年的平均折扣	最高 58%	最高 73%	最高 60%
实例系列	Fixed	Fixed	灵活

	Reserved Instance	EC2 实例节省计划	计算类节省计划
实例大小	固定 (非 Linux)	灵活	灵活
地理位置	1 个区域	1 个区域	灵活
操作系统	Fixed	灵活	灵活
服务	Amazon EC2 或 Amazon RDS	Amazon EC2	Amazon EC2、Fargate、Lambda
付款选项	全部、部分、无预付款	全部、部分、无预付款	全部、部分、无预付款
实例限制	每个可用区 20 个	无限制	无限制

 Note

节省计划的工作原理是根据您每小时的资金承诺为您提供折扣。在计划期限内，每小时财务承诺不可取消或更改，但您可以通过购买额外的节省计划来覆盖额外使用量。这使您能够在实例集扩大时仍能保持一致的每小时承诺量。

您可以使用 [AWS Cost Explorer](#) 或 [AWS 云 智能控制面板](#)之类的工具来跟踪您的承诺。Cost Explorer 成本管理服务提供了一条覆盖目标线，这有助于您的组织规划其节省计划的覆盖策略。如果您 75% 的工作负载处于稳定状态，则 75% 是一个不错的目标。这就使得 25% 的支出根据动态的工作负载而有所变化/具有灵活性。如果您需要将覆盖率提高到 85%，您可以再购买一项节省计划承诺，以此来增加每小时的资金承诺额度。

 Note

我们建议您购买节省计划而不是预留实例，但如果您已经购买了预留实例，则两种承诺模式可以一起使用。

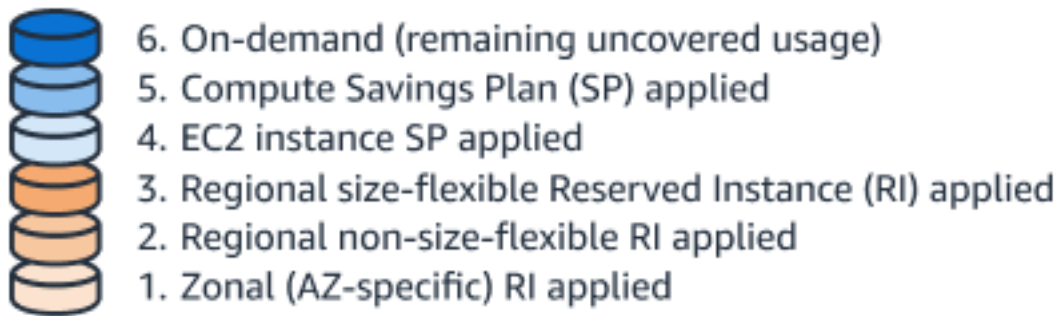
举例来说，您已经购买了一个预留实例，但您现在想尝试一下节省计划选项。这种组合在您的最终账单中是合乎逻辑的。以下是您可以应用于 AWS 账户的层次结构：

1. 可用区预留实例适用于拥有该实例的账户。如果某个预留实例还有剩余的使用时间，则这些时间将适用于组织的其余部分。
2. 适用于 Windows 的非可变大小区域预留实例适用于其所属账户的相应使用情况。剩余的所有内容都会传递给组织的其余部分。
3. 可变大小的区域预留实例适用于拥有该实例的账户（首先是该系列中的最小实例，然后依次是更大的实例），并且还适用于该组织的其余部分。
4. 区域预留实例适用于任何未使用的按需容量预留。
5. EC2 实例节省计划适用于购买该计划的账户。
6. 计算类节省计划适用于购买该计划的账户。

Note

折扣从最高折扣对应的用量开始，依次递减至最低折扣。通常情况下，对于大多数常见的实例类型（例如 T3、M6 和 C5），Windows 实例的折扣潜力要低于 Linux 实例。这意味着，在大多数情况下，Linux 实例比 Windows 实例的优势更大。

下图展示了在将预留实例与节省计划区分开来后的价格。计算类和 EC2 实例节省计划都首先适用于正在运行的实例，然后适用于未使用的按需容量预留。



成本优化方案

本部分介绍 Amazon EC2 专属主机与使用随附许可证计费模式的 Amazon EC2 实例的成本优化方案。

Amazon EC2 专属主机

设想这样一个场景：您打算将本地的 Windows 工作负载迁移到 AWS。您的数据中心有以下服务器：

- 拥有 16 个 vCPU 和 128GB RAM 的两台服务器

- 拥有 32 个 vCPU 和 164GB RAM 的两台服务器
- 拥有 8 个 vCPU 和 64GB RAM 的一台服务器
- 拥有 vCPU 和 32GB RAM 的 16 台服务器

此外，假设你可以自带许可证，AWS 因为你有足够的许可证可以带过来。下表显示了您可以在 AWS 中使用的服务器实例。

实例类型	CPU	RAM	使用额
r5.4xlarge	16	128	2
r5.8xlarge	32	256	2
r5.2xlarge	8	64	1
r5.xlarge	4	32	16
			21

分析显示，这 21 台虚拟机可以分布在两台具有 R5 实例系列主机的专属主机上。下表显示了这两台专属主机的费用。

专属主机按需场景	预付费用	1 个月	1 年	3 年	AWS 定价计算器
按需	无	10,123 美元	121,475 美元	364,392 美元	AWS 定价计算器 估计
1 年期节省计划	无	7,447 美元	89,362 美元	—	AWS 定价计算器 估计
3 年期节省计划	无	5,476 美元	65,712 美元	197,128 美元	AWS 定价计算器 估计
需预付款的 3 年期节省计划	84,438 美元	2,755 美元	117,499 美元	183,618 美元	AWS 定价计算器 估计

如果您有想要迁移到 AWS 的服务器，则 1 年期 Savings Plans 的最终价格为 89,362 美元，而不是按需价格计算的 121,475 美元。这意味着一年后可享受 26.5% 的折扣。如果你正在考虑延长居留时间，那么你可以选择 AWS 为期 3 年的 Savings Plans，以进一步节省成本。三年结束后，您支付的是 197,128 美元，而不是 364,392 美元。这使得三年后节省下来的金额达到总金额的 46%。

随附许可证的 Amazon EC2 实例

假设您要将单个三层应用程序迁移到 AWS，并希望使用提供的 AWS 许可证。此外，假设您的应用程序适用于以下服务器：

- 两台带有两个 v CPUs 和 4 GB 内存的 Web 服务器
- 两台具有 8 v CPUs 和 16 GB 内存的应用程序服务器
- 两台具有 16 v CPUs 和 64 GB 内存的数据库服务器（使用 SQL Server 标准版）

下表显示了您可以在 AWS 中使用的服务器实例。

实例类型	CPU	RAM	使用额
c5.large	2	4	2
c5.2xlarge	8	16	2
r5.2xlarge	8	64	2
			6 个服务器

下表显示了这些服务器在中的成本 AWS。

包括的许可证 AWS	预付费用	1 个月	1 年	3 年	AWS 定价计算器
按需	无	3,912 美元	46,950 美元	140,849 美元	AWS 定价计算器 估计
1 年期节省计划	无	3,466 美元	41,952 美元		AWS 定价计算器 估计

包括的许可证 AWS	预付费用	1 个月	1 年	3 年	AWS 定价计 算器
3 年期节省计 划，无需预付 款	无	3,189 美元	38,264 美元	114,804 美元	AWS 定价计 算器 估计
需预付款的 3 年期节省计划	112,110 美元	无	无	无	AWS 定价计 算器 估计

如果您想将这些服务器用于生产环境（全天候运行）并采用按需计费模式，则您每月需支付 3912 美元的费用。如果您按月支付费用，一年下来总计将花费 46950 美元，三年下来则总计需 140849 美元。

如果您选择无需预付款的 1 年期节省计划，则每月费用将降至 3466 美元。在第一年结束时，您需要支付 41952 美元。总折扣率为 11%。如果您选择无需预付款的 3 年期节省计划，则每月费用将降至 3189 美元。三年结束时，您需要支付 114804 美元。这能为您节省 18.5% 的费用。

成本优化建议

当您计划和预测 AWS 中工作负载时，这两种情况都可以帮助您节省费用。需要明确的是，在第二种情况下所给予的折扣要低于第一种情况。在第二种情况下，许可价格包含在云服务器的价格中。AWS 不提供许可价格折扣，但您可以随时携带许可证（在特定情况下），并且 AWS 可以随时保证最 compute/instance 优惠的价格。

我们建议您采取以下措施来控制计算和实例资源 AWS 支出：

- 访问建议
- 根据自身需求定制建议
- 查看每小时承诺

访问建议

您可以使用 [Amazon EC2 控制台](#) 访问有关您的节省计划的建议。您甚至可以将建议下载为 CSV 格式的文件，以在稍后查看。有关更多信息，请参阅节省计划文档中的[监控您的节省计划](#)。

根据自身需求定制建议

打开 [Amazon EC2 控制台](#)，展开实例部分，然后选择节省计划。此页面向您展示在提出建议前后的实例和计算定价情况。您也可以针对建议调整以下因素：

- 期限：例如，1-3 年
- 付款选项：例如，预付款、预付部分费用或不预付
- 历史记录：例如，过去 7、30 或 60 天

查看每小时承诺

使用同样的示例，假设您有一个全天候运行的实例。建议使用节省计划。根据大小，您的按需价格为每小时 120 美元。您可以选择承诺每小时 90 美元，但这可能会因区域、实例和购买选项而异。在此示例中，与按需成本相比，您可以节省 25% 的费用。您还可以跟踪利用率和覆盖范围（如果它们低于您定义的阈值），并在预算即将结束时配置提醒。

查看建议

我们建议您仔细查看节省计划建议。未经您的许可，AWS 不会更改任何内容。这些只是建议，是否应用它们由您自己决定。

购买计划

打开 [Amazon EC2 控制台](#)，展开实例部分，然后选择节省计划。然后，选择购买节省计划。根据您的要求，您可以选择以下选项：期限、区域、实例系列、每小时承诺、付款选项，甚至是开始日期。您可以从 Compute Savings 计划、EC2 实例储蓄计划和 SageMaker AI Savings 计划中进行选择。有关更多信息，请参阅节省计划文档中的[购买节省计划](#)。

获取利用率报告

购买节省计划后，您可以获得利用率报告。该报告能帮助您检查利用率，查看所购买的计划是否足以满足需求并实现最大折扣优惠，并且还能让您取消或添加新的折扣。此报告可以导出为其他格式，例如 CSV。有关更多信息，请参阅节省计划文档中的[使用利用率报告](#)。

遵循最佳购买实践

我们建议您在购买节省计划之前遵循以下最佳实践：

- 使用 [AWS Trusted Advisor](#) 移除闲置的 EC2 资源。

- 在购买节省计划之前，请进行任何调整。
- 确定持续保持 30-60 天的小时费率。
- 购买一份服务承诺，其内容应涵盖您所在组织所能接受的尽可能多的固定小时费率。考虑需求或季节的波动情况。
- 选择季度审查节省计划预算以保持一致的费率（例如，节省计划覆盖范围的覆盖率目标 70%）。如果该费率低于所需的覆盖范围，请购买额外的节省计划来进行调整，以达到您的覆盖目标。

其他资源

- [Amazon EC2 预留实例的储蓄计划](#) (AWS 白皮书)
- [了解 Savings Plans 如何适用于您的 AWS 用量](#) (Savings Plans 文档)
- [宣布 EC2 Windows 服务器和 SQL Server 实例的每秒计费](#) (AWS 文档)
- [AWS 成本优化系列：Savings Plans 视频 | Amazon Web Services](#) () YouTube

使用 AWS 工具监控成本

概述

成本可见性是优化成本的关键因素 AWS。AWS 有许多工具可用于可视化成本并针对这些成本创建警报。这些工具包括可 AWS Budgets 帮助您跟踪和报告支出的工具。本部分介绍监控 AWS 上的 Windows 支出的具体方法，以便您可以跟踪支出情况并根据预算要求相应地做出反应。这包括为您的 Windows EC2 资源添加必要的标签。这些标签使您能够使用 AWS Budgets 适当监控 Windows EC2 和其他 Microsoft 服务。

通过监控支出并使用 AWS 工具创建警报，您可以更详细地了解当前支出、预计支出和支出异常情况。如果您使用 [节省计划](#) 来帮助降低 EC2 实例的每小时定价，我们建议您查看节省计划的总体利用率和覆盖范围。这可以帮助您确保不断实现节省。您可以使用 AWS Cost Explorer 来查看 Savings Plans 库存，并根据以前的使用情况获取有关其他储蓄计划的建议。您还可以通过使用 [AWS Budgets](#) 和设置 [AWS Cost Anomaly Detection](#) 来跟踪特定支出。

成本优化建议

我们建议您采取以下后续步骤，通过使用 AWS Budgets Cost Explorer 和异常检测来优化成本：

- 标记 Windows EC2 资源
- 使用设置警报 AWS Budgets

- 启用成本异常检测
- 获取实时支出分析
- 使用 Cost Explorer 成本管理服务查看 Windows 的随附许可证支出

标记 Windows EC2 资源

要有效监控 AWS 支出，必须为要监控的工作负载[制定标签策略](#)。这一点非常重要，因为它能让您明确划分资源，并能收到有关具体支出的警报通知，而不是收到关于一般性支出的警报。您可以使用这些标签资源，它们不仅能降低成本，还能用于其他目的，比如 [AWS Systems Manager automation](#)。此外，我们建议您对[所需的标签](#)进行一些管理。

要跟踪您在 AWS Budgets“Cost Explorer”和“成本异常检测”中的支出，必须确保设置了适当的标签。您可以使用标签来为符合这些标签的项目设定特定预算，这样当支出增加时，您就会收到提醒。

例如，您可以使用一个简单的标签，比如 Key=OS Value=Windows。这会将您的所有 Windows 实例整合到一个组中，以便您跟踪其支出。您也可以将标签用于其他项目，例如 Systems Manager。创建标签后，必须激活标签以进行成本跟踪。考虑添加[一条AWS Config 规则来监控附加到某些资源的标签](#)。AWS Config 如果正在运行的资源不包含相应标签，则可以提醒您，这些标签可以准确地显示您的 Windows EC2 支出。

有了标签后，您可以在 AWS Billing 中创建自定义预算。这能让您了解您的 Windows EC2 支出。您可以设置每日预算或每月预算。

使用设置警报 AWS Budgets

在此示例场景中，您将为 Windows EC2 创建每日预算。这是一项经常性预算，它采用自动调整选项来跟踪您的支出情况，并据此对预算进行相应调整。如果您的环境是静态的，则可以改用固定预算。请务必选择基准时间范围（例如，30 天）。

1. 登录 AWS 管理控制台 并打开[AWS Cost Management 控制台](#)。
2. 在导航窗格中，选择预算。
3. 在页面顶部，选择创建预算。
4. 在预算设置下，选择自定义（高级）。
5. 在预算类型下，选择成本预算。然后选择下一步。
6. 在详细信息下，对于预算名称，输入预算的名称。例如，Windows EC2 支出。
7. 在设置预算金额下，为周期选择每日。
8. 对于预算续订类型，请在预算周期之后重置的预算选择经常性预算。

9. 对于起始日期，请选择起始日期或期间，以开始跟踪预算金额。
10. 对于预算方法，请选择自动调整（新建）。
11. 对于基准时间范围，请选择自定义范围，然后输入 30 天。
12. 选择下一步。
13. 在预算范围部分，选择筛选特定 AWS 成本维度。这是使用标签来创建适当尺寸的地方。AWS Budgets 在其筛选器中不支持“平台类型”作为选项。因此，您必须应用操作系统标签。
14. 选择添加筛选条件，然后从维度中选择标签选项。
15. 选择操作系统标签，然后为此选择 Windows 值，以便为标签创建预算。
16. 选择下一步。
17. 在配置提醒页面上，选择添加提醒阈值。在这里，您可以设置两个提醒：一个用于 50% 的阈值，另一个用于 100% 的阈值。如果在该月中间阶段之前突破了 50% 的阈值提醒，将会发出警告。这样，您就可以检查支出是否超出预期，并在月底之前及时采取措施。
18. 在阈值中，输入 50，然后选择预算金额的 %。
19. 对于触发器，选择实际。
20. 对于电子邮件收件人，输入电子邮件地址。添加另一个针对阈值 100 的提醒。

Note

此示例中使用了电子邮件通知来发送提醒，但您也可以采用其他方式，例如 [Slack](#)。

启用成本异常检测

您可以使用成本标签来设置异常支出提醒。例如，您可以使用 [AWS Cost Anomaly Detection](#) 创建支出监控器，并在系统检测到您的账户中存在异常支出时收到提醒。

要为之前创建的 Key=OS and Value=Windows 标签设置监控器和提醒，请执行以下操作：

1. 登录 AWS 管理控制台 并打开 [AWS Cost Management 控制台](#)。
2. 在导航窗格中，选择成本异常检测。
3. 选择成本监控器选项卡，然后选择创建监控器。
4. 在步骤 1 中，选择成本分配标签作为您的监控器类型。
5. 对于成本分配标签键，请选择 Windows EC2 支出。
6. 对于成本分配标签值，请选择 Windows。

7. 对于为监控器命名，输入 Windows EC2 支出。
8. 选择下一步。
9. 要为提醒创建订阅，请选择创建新订阅。如果您有现有的订阅，请选择 Choose an existing subscription (选择现有的订阅)。
10. 对于订阅名称，输入 Windows EC2 支出异常。
11. 对于提醒频率，请选择每日摘要。
12. 对于提醒收件人，输入您的电子邮件地址。
13. 选择添加阈值。对于阈值，输入 10，然后选择高于预期速度的百分比。
14. 选择 Create monitor (创建监控)。

实时查看支出

提醒是监控您在 Windows EC2 上的支出的有用工具，但如果您想要实时查看情况，就必须使用 Cost Explorer 成本管理服务。观看此视频，了解 Cost Explorer 成本管理服务如何使您能够分析和降低 EC2 成本。有关更多信息，请观看上的 [“AWS 支持您 | 了解并降低您的 EC2 成本”](#) 视频 YouTube。

查看 Windows 的随附许可证支出

您可以使用 Cost Explorer 成本管理服务查看账户中的 EC2 Windows 支出。要查看 Windows 的随附许可证支出，您必须在 Cost Explorer 成本管理服务中设置以下正确的 [筛选条件](#)：

- 对于平台，请选择 Windows (Amazon VPC)。对于 API 操作，请选择 RunInstance:0002。这是包含许可证的 Windows EC2 实例的 AWS Billing 代码。
- 如果您想查看您的 BYOL 实例支出，请将:0002 更改为:0800 RunInstance。RunInstance 这是 Windows EC2 BYOL 的账单代码。

借助 Cost Explorer 成本管理服务中的这种可见性，您可以快速筛选出您在 Windows EC2 上花费的成本。如果您想更深入地了解自己的 AWS 支出，可以使用向下筛选 AWS 成本和使用情况报告 到单个实例级别的支出。您还可以生成可在 Amazon Quick 中可视化的报告，并构建自定义控制面板。

欲了解更多信息，请观看上的 [“AWS 支持您-可视化您的成本和使用情况报告”](#) YouTube 视频。

其他资源

- [使用 AWS Config \(AWS Config 文档 \) 设置必需的标签](#)
- [AWS Budgets 教程-为 AWS Billing | Amazon Web Services \(YouTube\) 设置提醒](#)

- [AWS 成本和使用情况报告 查询库](#) (Well-Archit AWS ected Labs)

SQL Server

客户运行 Microsoft 工作 AWS 负载的时间已超过 15 年，比任何其他云提供商都要长。这主要是因为云端使用微软应用程序方面 AWS 拥有最丰富的经验，并且在以下领域为 Windows Server 和 Microsoft SQL Server 提供了最佳平台：

- 更高的性能和可靠性
- 更好的安全和身份服务
- 更多的迁移支持
- 最广泛、最深层次的能力
- 降低总拥有成本 (TCO)
- 灵活的许可选项

AWS 支持构建和运行依赖于 SQL Server 的 Windows 应用程序所需的一切，包括 Active Directory、.NET、SQL Server、Windows 桌面即服务以及所有支持的 Windows 服务器版本。凭借久经考验的专业知识，AWS 可以帮助您轻松地提升、转移、重构甚至现代化 Windows 工作负载。

指南的本部分将介绍以下主题：

- [选择高可用性和灾难恢复解决方案](#)
- [了解 SQL Server 许可](#)
- [为 SQL Server 工作负载选择合适的 EC2 实例。](#)
- [整合实例](#)
- [比较 SQL Server 版本](#)
- [评估 SQL Server 开发人员版](#)
- [评估 Linux 上的 SQL Server](#)
- [优化 SQL Server 备份策略](#)
- [现代化 SQL Server 数据库](#)
- [优化 SQL Server 的存储](#)
- [使用 Compute Optimizer 优化 SQL Server 许可](#)
- [使用 Compute Optimizer 优化 SQL Server 的大小](#)
- [查看 SQL Server 工作负载 Trusted Advisor 的建议](#)

选择高可用性和灾难恢复解决方案

概述

我们建议您为 AWS 上的 SQL Server 部署设计一种架构，既要满足您的业务需求，又能满足[灾难恢复 \(DR\) 目标](#)，包括恢复时间目标 (RTO) 和恢复点目标 (RPO)。以下这些解决方案能够帮助您在 Amazon Elastic Compute Cloud (Amazon EC2) 上运行的 SQL Server 设计合适的架构，同时优化您的 SQL Server 工作负载的成本。

- SQL Server Always On 可用性组 — SQL Server Always On 可用性组提供高可用性和灾难恢复 (HA/DR) solutions for SQL Server databases. An availability group consists of a set of user databases that fail over together. Always On availability groups also provide redundancy at the database level, but don't require shared storage—each replica has its own local storage. You can deploy this feature as an HA/DR解决方案。有关更多信息，请参阅 Microsoft 文档中的[什么是 Always On 可用性组？](#)。
- SQL Server 始终开启故障转移群集实例 (FCI) — SQL Server Always On FCIs 使用 Windows 服务器故障转移群集 (WSFC) 在 SQL Server 实例级别提供 HA。FCIs 需要共享存储空间来托管数据库。您可以使用共享块存储或者共享文件存储。例如，您可以使用适用于 Windows 文件服务器 FSx 的亚马逊或 FSx 适用于 NetApp ONTAP 的亚马逊作为具有多个可用区的共享存储解决方案。有关更多信息，请参阅 Microsoft 文档中的[Always On 失效转移群集实例 \(SQL Server\)](#)。
- SIOS DataKeeper — SIOS DataKeeper 可以通过启用跨可用区和可用区的 SQL Server FCI 来帮助同时满足高可用性和灾难恢复要求。AWS 区域 SIOS 使用本地 Amazon Elastic Block Store (Amazon EBS) 卷 DataKeeper 创建集群虚拟 SAN，并在可用区之间使用同步复制实现 HA，同时使用区域间的异步复制和灾难恢复。有关更多信息，请参阅 SIOS 文档中的[Windows 应用程序的高可用性保护](#)。
- 分布式可用性组：分布式可用性组是一种特殊类型的可用性组，它跨越两个单独的 Always On 可用性组。可用性组可以位于两个不同的区域（例如 us-east-1 和 us-west-1）。您可以将分布式可用性组视为由多个可用性组组成的可用性组，因为其底层的 Always On 可用性组分别配置在两个不同的 WSFC 集群中。部署分布式可用性组需要 SQL Server 企业版。有关更多信息，请参阅 Microsoft 文档中的[分布式可用性组](#)。
- 日志传送：您可以实施日志传送来保护您在多个区域间的数据库，以防某个区域出现故障且不可用的情况发生。根据事务和日志传送频率，您能够在几分钟内实现 RPO 和 RTO。有关更多信息，请参阅 Microsoft 文档中的[关于日志传送 \(SQL Server\)](#)。
- AWS Elastic Disaster Recovery— Elastic 灾难恢复是一种软件即服务 (SaaS) 应用程序，用于管理服务器从任何基础设施复制到 AWS 灾难恢复。您还可以使用 Elastic Disaster Recovery 跨区域复制 SQL Server。Elastic Disaster Recovery 是一种基于代理的解决方案，它可以将整个虚拟机（包括

操作系统、所有已安装的应用程序以及所有数据库) 复制到一个暂存区域。有关更多信息，请参阅 Elastic Disaster Recovery 文档中的[什么是 Elastic Disaster Recovery ?](#)。

- AWS Database Migration Service (AWS DMS) — AWS DMS 支持数据进出实时迁移 AWS，包括不同区域。您可以利用此功能在不同的区域中设置一个独立的 SQL Server 实例，将其用作灾难恢复数据库。有关更多信息，请参阅[什么是 AWS Database Migration Service ?](#) 在 AWS DMS 文档中。

SQL Server Always On 可用性组

如果您将 SQL Server 企业版仅用于高可用性的 [Always On 可用性组](#)，则可以利用基本可用性组降级到 SQL Server 标准版。通过使用基本的可用性组而非 Always On 可用性组，您可以将成本降低 65% 至 75%。

Note

有关不同 SQL Server 版本之间成本差异的其他信息，请参阅本指南的[比较 SQL Server 版本部分](#)。

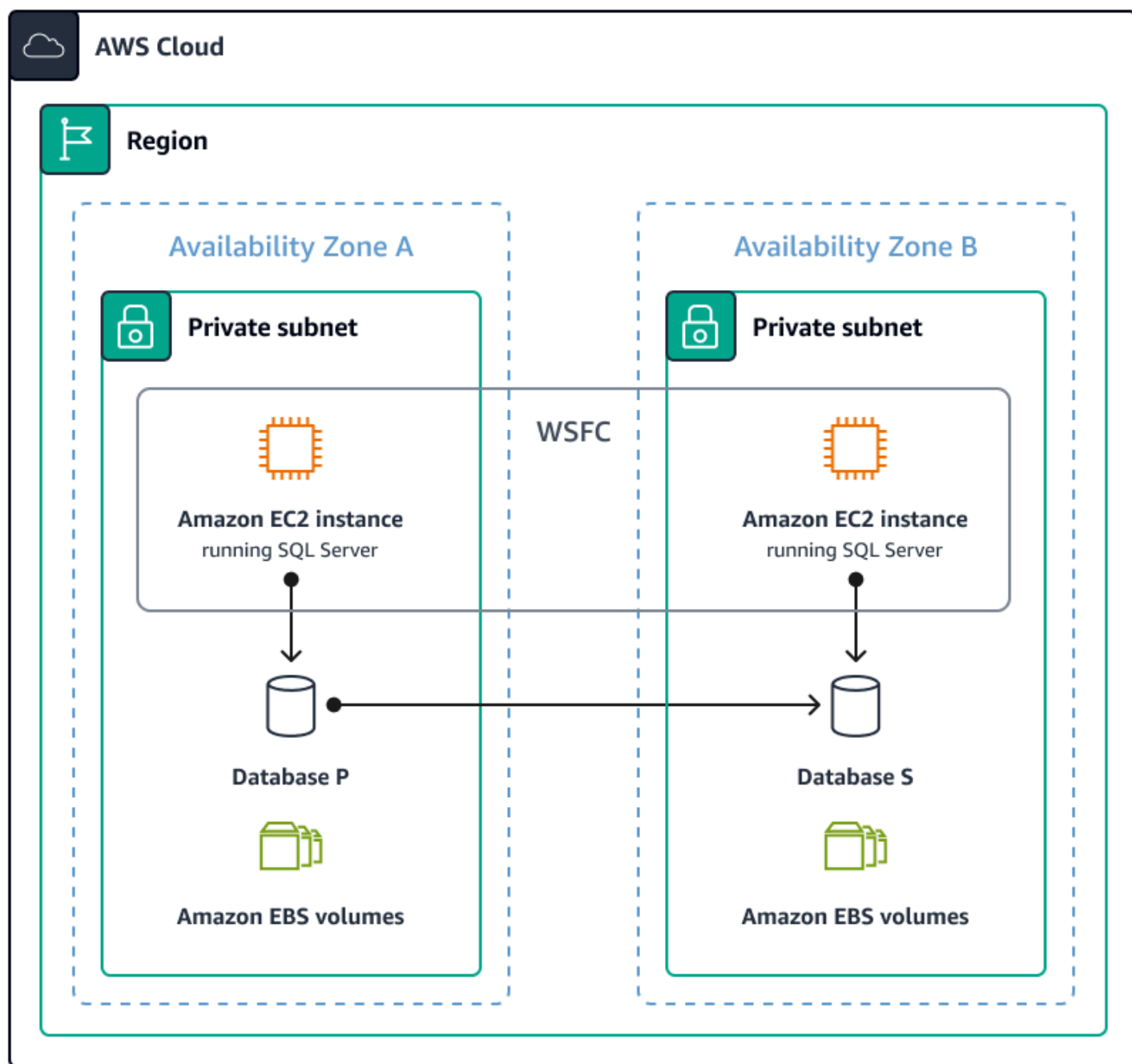
功能

- 在 SQL Server 标准版中可用
- 限制两个副本 (主副本和辅助副本)
- 对辅助副本没有读取访问权限
- 对辅助副本没有进行完整性检查

限制

- 仅为每个可用性组支持一个可用性数据库
- 基本可用性组不能成为分布式可用性组的一部分

下图显示了 Windows Server 失效转移群集解决方案的示例架构。



SQL Server Always On 失效转移群集实例

您可以使用故障转移群集实例 (FCIs) 来确保数据库的持续运行，同时最大限度地减少停机时间并降低数据丢失的风险。FCIs 如果您希望在不配置只读副本的情况下实现 SQL Server 数据库的高可用性，则可以提供可靠的解决方案。

与可用性组不同，无需 FCIs 使用 SQL Server 企业版即可提供可靠的故障转移解决方案。相反，只 FCIs 需要 SQL Server 标准版许可。您可以使用 FCIs 将 SQL Server 的许可成本降低 65-75%。

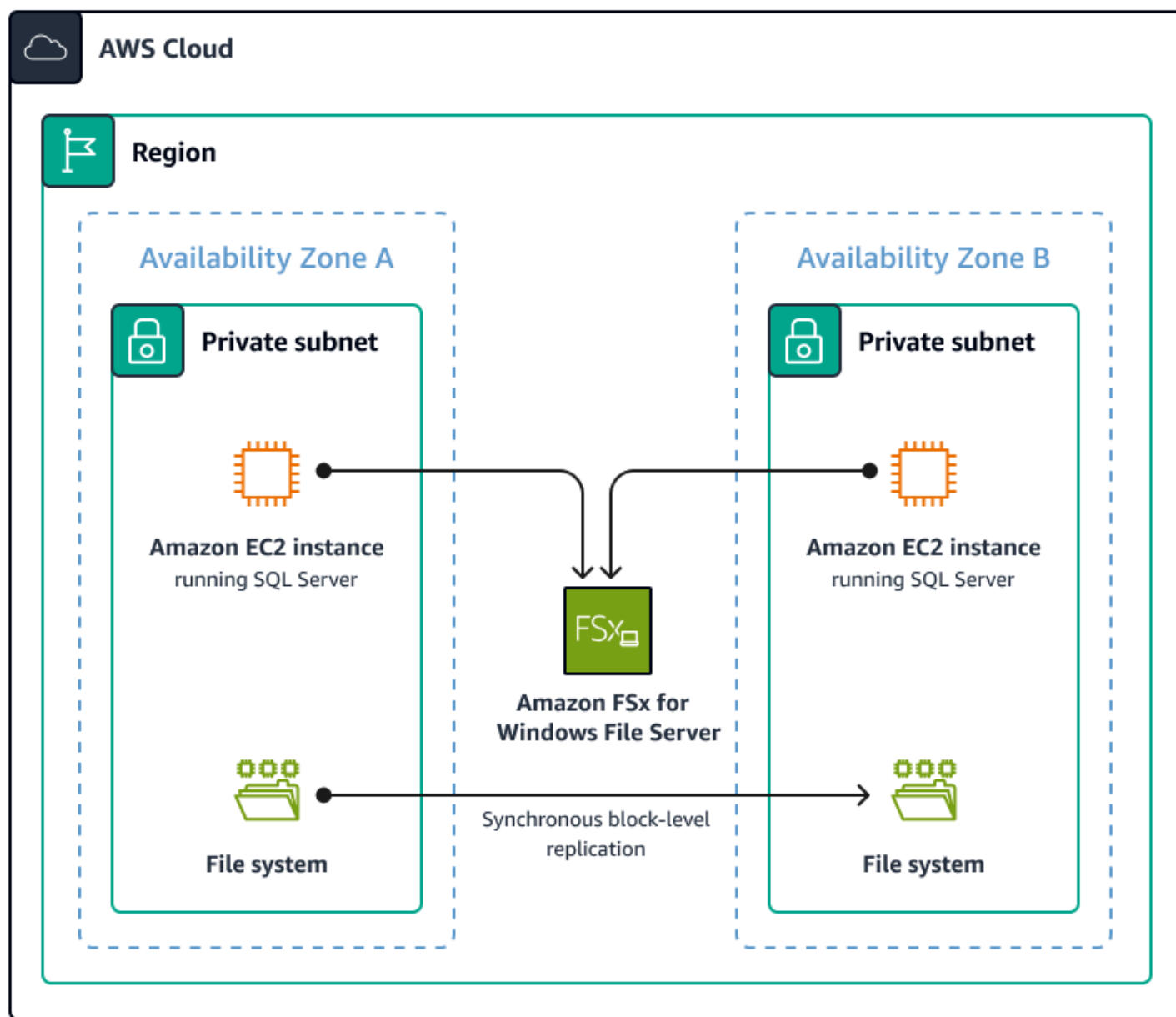
Note

有关各 SQL Server 版本之间成本差异的其他信息，请参阅本指南的[比较 SQL Server 版本](#)部分。

请考虑以下事项：

- Amazon FSx for Windows File Server 提供了一个强大的解决方案，可以满足你的 SQL Server FCI 共享存储需求。您可以在 Windows File Server 上使用 FSx，这样就无需购买存储复制解决方案的许可证，也无需自己管理共享存储。这可能会带来 30% 至 40% 的显著成本节省。有关更多信息，请参阅 AWS 存储博客上的[“使用亚马逊 FSx Windows 文件服务器简化微软 SQL Server 高可用性部署”](#)一文。
- 借助[软件保障权益摘要](#)（可下载的 PDF）和自带许可（BYOL）模型，只要辅助服务器处于被动状态，您就可以利用被动失效转移带来的好处。这可以节省 SQL 许可的成本，因为您无需向集群中的被动节点提供许可证。

下图显示了使用于 Windows 文件服务器的 SQL Server FCI FSx 的示例架构。



SIOS DataKeeper

如果您计划在 FCIs 上部署 SQL Server，我们建议您考虑共享存储需求 AWS。传统的本地安装通常使用存储区域网络 (SAN) 来满足共享存储需求，但这在 AWS 上不是一个可行的选项。Amazon FSx for Windows File Server 是 SQL Server FCI 的推荐存储解决方案 AWS，但它存在一些限制，无法在不同的 AWS 区域服务器中添加群集服务器。

您可以使用 [SIOS DataKeeper](#) 创建涵盖可用区和区域的 SQL Server FCI，同时将成本降低 58-71%。SIOS DataKeeper 可以帮助您实现 FCI 的高可用性优势。这使得 SIOS DataKeeper 成为组织经济实惠且可靠的解决方案。

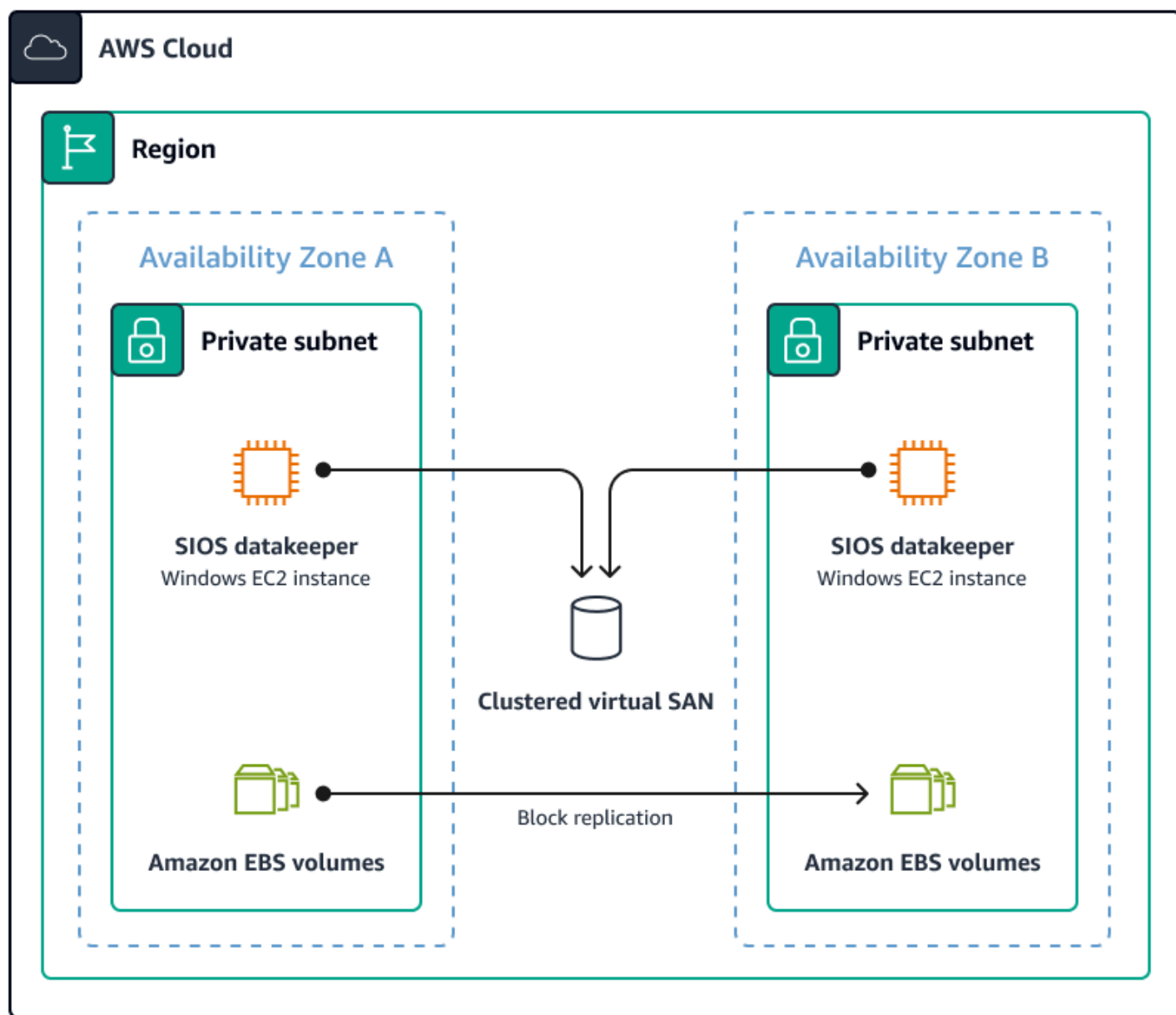
请考虑使用 SIOS DataKeeper 的以下其他好处：

- SIOS 使用本地 EBS 卷 DataKeeper 创建群集虚拟 SAN，并在可用区之间使用同步复制来实现高可用性。为了进行灾难恢复，SIOS DataKeeper 使用区域间的异步复制。
- SIOS 使用 SQL Server 标准版 DataKeeper 提供企业级群集功能。与使用 SQL Server 企业版的 Always On 可用性组来实现高可用性相比，这能将 SQL Server 的许可成本降低 65% 至 75%。借助 SIOS DataKeeper，您可以创建高度可用、灵活且经济实惠的 SQL Server 环境，以满足您的组织需求。

Note

有关各 SQL Server 版本之间成本差异的其他信息，请参阅本指南的[比较 SQL Server 版本](#)部分。

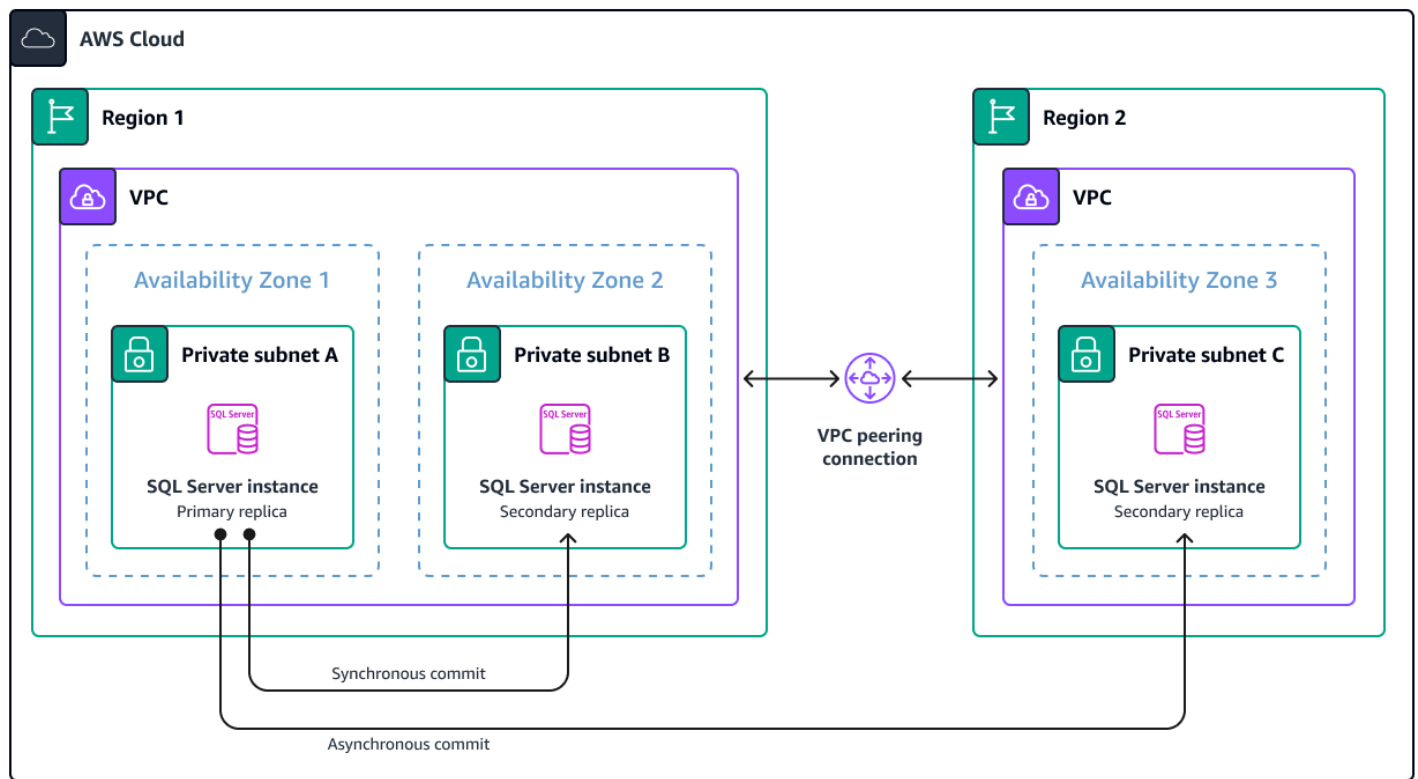
以下图表展示了使用集群式虚拟 SAN 解决方案的 SQL Server FCI 的一个示例架构。



Always On 可用性组

您可以使用 Always On 可用性组来实现高可用性和灾难恢复的目的。您可以通过跨一个区域内的两个可用区部署 SQL Server 来实现高可用性。您可以通过跨区域扩展可用性组来实现灾难恢复。

以下图表展示了基于 Always On 可用性组的解决方案的示例架构。图中区域 1 中的副本采用了同步提交方式，这种方式能够实现可用性组的自动失效转移。区域 2 中的副本采用的是异步提交方式，这将需要对可用性组进行手动失效转移。

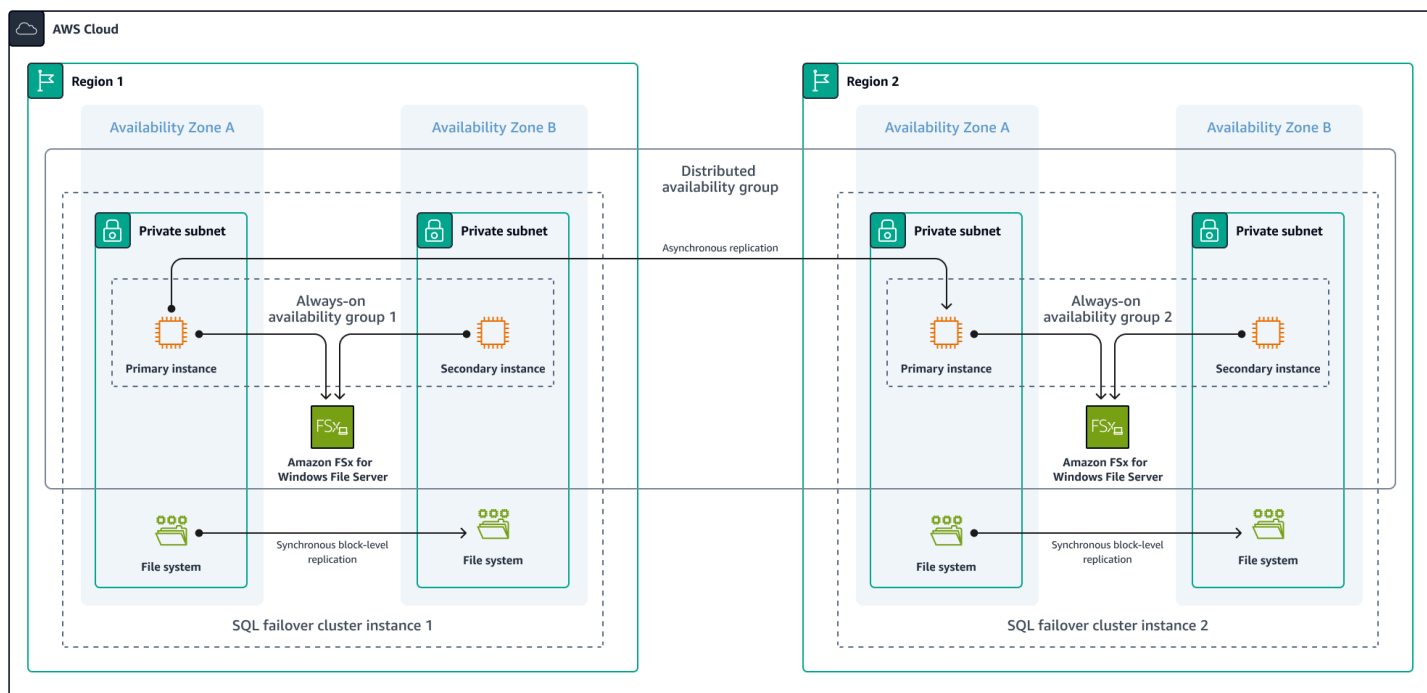


分布式可用性组

对于那些对可靠性或灾难恢复有着极高标准要求的任务关键型 SQL Server 部署而言，我们建议采用跨区域部署的方式。将您的可用性组分布到多个区域，是确保业务连续性、最大限度减少停机时间的最可靠解决方案。

该架构充分利用了 Amazon FSx for Windows 文件服务器的功能，包括共享存储、同步块级复制和 SQL Server。FCIs 这些功能使您能够创建一个跨多个可用区的高可用性 SQL Server 环境。通过在另一个区域复制这一设置，您将能够获得一个完全冗余的系统，该系统能够应对任何最严重的中断情况。该解决方案的与众不同之处在于其提供的灵活性和安全性。分布式可用性组独立于域的架构使得底层的 Windows 集群服务器能够加入不同的 Active Directory 域，而基于证书的身份验证则能为您的 SQL Server 环境提供最大程度的保护，并满足多区域 DR 策略的高 RTO 和 RPO 要求。有关构建多区域架构的信息，请参阅架构博客中的[现场注释：使用 FCI 和分布式可用性组为 SQL Server 构建多区域架构](#)。AWS

以下图表展示了采用分布式可用性组的多区域解决方案的示例架构。



日志传送

日志传送是一种经过验证、可靠且经济高效的解决方案，可在出现意外停机情况时保障您在不同区域中的数据库安全。几十年来，各组织一直采用日志传送的方式来保护其数据。

如果您启用日志传送 AWS，则可以在几分钟内实现 RPO 和 RTO，具体取决于事务和日志传送任务的频率。在极不可能出现的极端情况下，如果某个区域突然无法访问，则日志传送能确保您的数据安全且可恢复。

以下列举了使用日志传送的一些额外优势，请予以考虑：

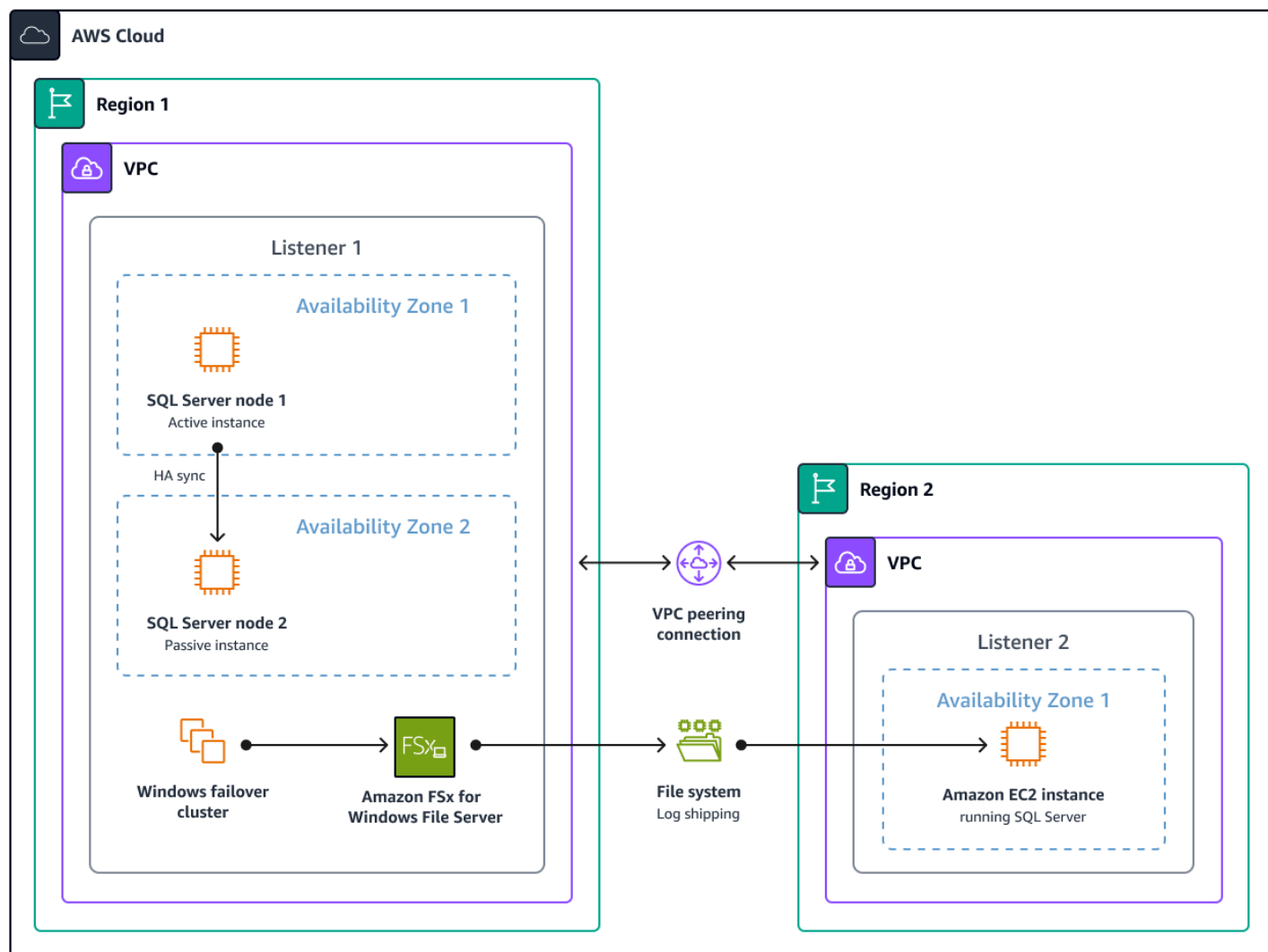
- 使用日志传送实现跨区域的灾难恢复韧性，从而降低成本并满足您的业务需求。日志传送能降低您的总拥有成本，因为您只需购买 SQL Server 标准版或 SQL Server Web 版的许可证即可。
- 通过使用带有有效[软件保障](#)的日志传送，消除灾难 recovery/passive 服务器的许可成本。当您使用带软件保障的日志传送时，只需要授权 primary/active SQL Server。
- 通过消除在不同区域之间设置分布式可用性组的 SQL Server 企业版需求，将 SQL Server 的许可成本降低 65% 至 75%。为此，您可以将 SQL Server 标准版和 SQL Server 与日志传送 FCIs 结合使用，以满足您的灾难恢复要求。

Note

有关各 SQL Server 版本之间成本差异的其他信息，请参阅本指南的[比较 SQL Server 版本](#)部分。

有关更多信息，请参阅 AWS 架构博客中的[使用日志传送扩展 SQL Server 灾难恢复和亚马逊 FSx Windows 版 Windows 配置的 SQL Server FCI](#)。

以下图表展示了日志传送解决方案的一个示例架构。

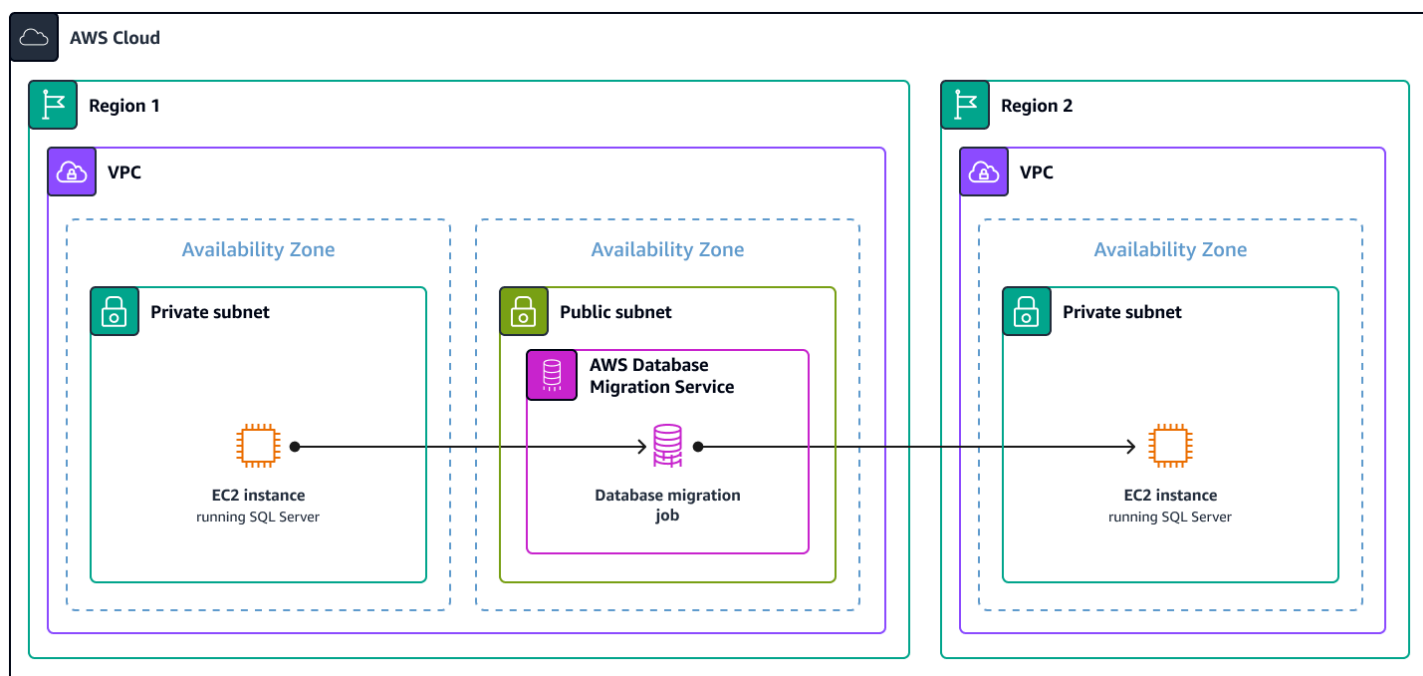


AWS Database Migration Service

您可以使用 AWS Database Migration Service (AWS DMS) 根据您的应用程序需求设计 HA/DR 解决方案。AWS DMS 使您能够轻松地将数据复制到同一区域 (HA) 或跨区域 (DR) 的辅助 SQL Server 数据库。这种方法在技术上是可行的，它能让您在 AWS 基础设施方面最大化您的投资，同时还能优化资源的利用效率。

AWS DMS 是一项具有成本效益的服务。您只需为传输过程中所使用的 CPU 资源以及任何额外的日志存储付费。这意味着，您可以从该解决方案中获益，同时无需承担过多额外费用。您可以使用 AWS DMS 来确保数据可用且可访问，同时最大限度地降低与许可和资源使用相关的成本。

以下图表展示了基于 AWS DMS 的解决方案的一个示例架构。



AWS Elastic Disaster Recovery

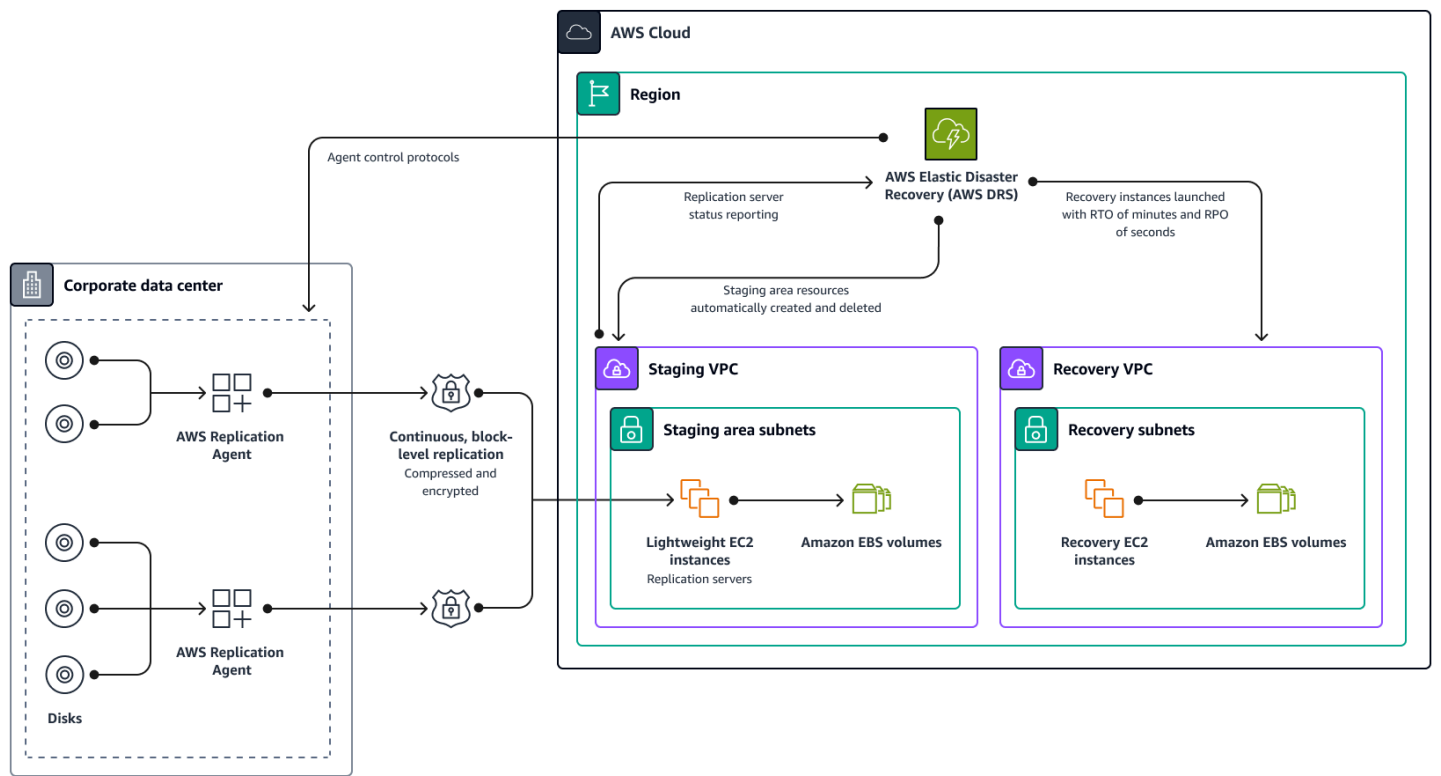
一些组织必须确保所有关键业务应用程序都已实施灾难恢复计划。过去，这些组织中有许多都大量投入资金用于传统的灾难恢复解决方案。这类方案要求您预先构建并维护一个完整的复制基础设施。这种方法成本高昂、耗时长且难以扩展。

现在，您可以使用 AWS Elastic Disaster Recovery 无需预先构建灾难恢复基础架构。在 Elastic Disaster Recovery 中，灾难恢复设备仅在必要时才会启动。因此，您只需按实际使用量在您需要它时付费。这意味着您能够显著降低软件许可费用和高性能计算成本。

此外，灾难恢复解决方案的暂存区域还包含低成本的 Amazon Elastic Block Store (Amazon EBS) 卷。EBS 卷进一步降低了预调配重复资源的成本。这使您能够降低总体的灾难恢复成本，同时仍能维护一个强大且可靠的灾难恢复方案，满足您的业务需求。您可以使用 Elastic 灾难恢复来专注于核心业务活动，AWS 同时为您的灾难恢复解决方案处理底层基础架构。

对于 SQL Server，您可以将 Elastic Disaster Recovery 用作经济高效的灾难恢复选项。如果您使用主动式软件保障服务，那么在具有容错能力且高度可用的 SQL Server 架构中，被动节点的许可费用将包含在内。然而，您仍需支付计算费用才能让该被动服务器保持在线状态。借助 Elastic Disaster Recovery，主服务器可以复制到灾难恢复环境中，而无需保持软件保障的活跃状态，也无需支付灾难恢复的计算费用。这种节省组合可以将您的 SQL Server 灾难恢复成本降低 50% 或更多。

以下图表展示了基于 Elastic Disaster Recovery 的解决方案的一个示例架构。



有关更多信息，请参阅[AWS 博客上的 Microsoft 工作负载 AWS Elastic Disaster Recovery 上如何为灾难恢复站点的 SQL Server 设置高可用性](#)。

成本比较

下表比较了本节所涵盖的 HA/DR 解决方案的成本。为进行此次比较，我们做出了以下假设：

- 实例类型：r5d.xlarge

- 许可证类型：包括适用于 Windows 和 SQL Server 的许可证
- 区域 - us-east-1

解决方案	高可用性	灾难恢复	企业版	标准版	成本
日志传送	否	是	是	是	SQL Server 企业版： 32674.8 美元 (2 个节点) SQL Server 标准版： 14804.4 美元 (2 个节点)
Always On 可用性组	支持	是	是	是，但为基本可用性组 (2 个节点)	SQL Server 企业版： 32674.8 美元 (2 个节点) SQL Server 标准版： 14804.4 美元 (2 个节点)
永远开启 FCIs	是	否	是	是 (2 个节点)	SQL Server 标准版： 14804.4 美元
分布式可用性组	支持	是	是	否	SQL Server 企业版： 65349.6 美元 (4 个节点)
弹性灾难恢复	否	是	是	是	复制 1 个实例和 1TB 存储空间的费用约

解决方案	高可用性	灾难恢复	企业版	标准版	成本
					为 107.48 美元/月 注意：Elastic Disaster Recovery 根据每台复制服务器按小时计费。无论磁盘数量、存储大小、执行演练或恢复操作的次数，还是您正在复制的区域如何，费用都是相同的。
SIOS Data Keeper	支持	是	是	是	具有软件保障的 Always On 可用性组（2 个节点，24 个内核）：213480 美元 在 SQL Server 标准版上运行的 2 节点 SQL Server 群集，具有 SIOS DataKeeper 和软件保障：61,530 美元（2 个节点）

解决方案	高可用性	灾难恢复	企业版	标准版	成本
AWS DMS	否	是	是	是	r5.xlarge 实例和 1TB 存储费用为 745.38 美元/月

成本优化建议

我们建议您采取以下后续步骤来选择符合组织要求的 HA/DR 解决方案：

- 请查看本指南的[为 SQL Server 工作负载选择合适的 EC2 实例](#)部分。
- 通过在工作负载高峰期运行性能计数器，来确定您的工作负载的 IOPS 和吞吐量要求：
 - $IOPS = \text{磁盘 reads/sec} + \text{磁盘写入次数/秒}$
 - $\text{吞吐量} = \text{磁盘读取 bytes/sec} + \text{磁盘写入字节/秒}$
- 请使用以下存储卷类型以获得更好的性能和成本节省：
 - NVMe 缓冲池扩展的实例存储空间 tempdb 和缓冲池扩展
 - 用于数据库文件的 io2 卷
- 使用 [AWS Trusted Advisor](#) 获取有关 Amazon EC2 上的 SQL Server 成本优化的建议。您无需安装代理即可进行 SQL Server 优化检查。Trusted Advisor 检查您的 Amazon EC2 SQL Server 包含许可证的实例配置，例如虚拟 CPUs (vCPUs)、版本和版本。然后，根据最佳实践 Trusted Advisor 提出建议。
- 适用 AWS Compute Optimizer 于 Amazon EC2 实例和 Amazon EBS 的正确大小建议。
- [AWS 定价计算器](#)用于设计成本估算 HA/DR 策略。
- 要确定从 SQL Server 企业版降级到 SQL Server 标准版是否是一个可行的选择，可使用 [sys dm_db_persisted_sku_features](#) 动态管理视图来识别在当前数据库中处于活动状态的版本特定功能。

Note

Side-by-side 使用包含许可证的 EC2 实例时，需要迁移才能更改 SQL Server 版本。

- 每半年或每年进行一次灾难恢复演练，以便更好地制定设计方案，确保能够按照既定的 RTO 和 RPO 恢复数据库。这还能帮助您发现任何可能存在的架构缺陷。

其他资源

- [使用 FSx 适用于 Windows 的亚马逊文件服务器简化微软 SQL Server 高可用性部署](#) (AWS 存储博客)
- [现场说明：使用 FCI 和分布式可用性组为 SQL Server 构建多区域 AWS 架构](#) (架构博客)
- [为 SQL Server 设计灾难恢复架构 AWS：第 1 部分](#) (AWS 数据库博客)
- [微软 SQL 借助 FSx 适用于 Windows 的亚马逊实现高可用性](#) (YouTube)
- [利用 Amazon EBS 最大程度提高 Microsoft SQL Server 性能](#) (AWS 存储博客)
- [将您的本地存储模式与 AWS 存储服务进行比较](#) (AWS 存储博客)
- [计划用 Amazon FSx 文件网关替换数据中心 NAS](#) (AWS 存储博客)
- [优化高可用性 SQL Server 部署的成本 AWS](#) (AWS 存储博客)
- [如何使用 AWS Elastic Disaster Recovery 为 SQL Server Always On 可用性组设置灾难恢复](#) (AWS 上的 Microsoft 工作负载)
- [如何在使用恢复的灾难恢复站点上为 SQL Server 设置高可用性 AWS Elastic Disaster Recovery](#) ([开启](#) Microsoft 工作负载 AWS)

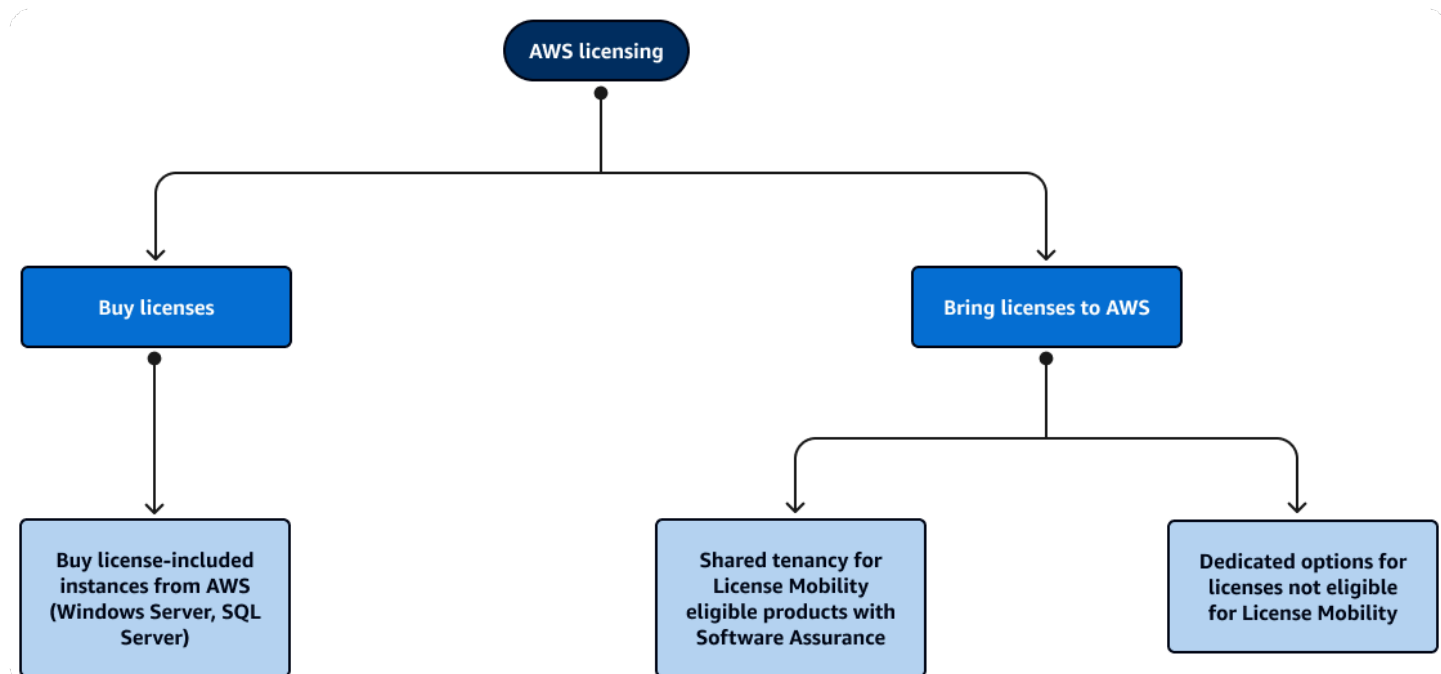
了解 SQL Server 许可

概述

随着越来越多的企业将工作负载迁移到云中，优化云平台的成本已成为首要任务。许可是与运行 Microsoft 工作负载相关的最重要成本之一 AWS。本节介绍如何 AWS 通过优化 SQL Server 的 Microsoft 许可来优化成本。

AWS 许可选项

AWS 为许可提供了一系列灵活的成本优化选择。这些许可选项旨在帮助您降低成本、确保合规性并满足您的业务需求。



AWS 将许可证分为三种主要类型：

1. 包含许可证：此许可选项允许您按需购买并使用许可证，仅需按实际使用量付费。包含许可证选项非常适合那些您希望在使用许可方面具有灵活性，并且又不想承担前期费用的情况。您可以从一系列 Windows Server、SQL Server 和其他 Microsoft 产品中进行选择。
2. 自带许可证 (BYOL) 产品，具有许可证移动性 — 此许可选项专为已有许可证并希望在云中使用的场景而设计。AWS 允许客户通过 Microsoft 的许可证[移动性计划将自己的许可证](#)带到云端。您可以将具有许可证移动性的产品（例如带软件保障的 SQL Server (SA)）带到共享或专用租赁，以降低 AWS 实例成本。
3. 没有许可证移动性的 BYOL 产品 — 对于没有许可证移动性的微软产品，例如 Windows Server，AWS 提供了在云中使用这些产品的专用选项。此外，专属主机还提供在物理核心级别进行许可的机会。这能让您在运行工作负载时节省 50% 或更多的许可证费用。专属主机对于那些大部分时间都能保持稳定且可预测的工作负载是一个非常理想的选择。

自带许可证对成本的影响

自带许可证可能会对在 AWS 上运行 Microsoft 工作负载的成本产生显著影响。如果您自带许可证，则无需为在云中运行的实例支付额外的许可费用。这能够带来显著的成本节省。

以下对比数据展示了全天候运行单个 c5.xlarge 实例的按需每月成本：

- Windows Server + SQL Server 企业版：1353 美元/月（包含许可证）

- Windows Server + SQL Server 标准版：609 美元/月（包含许可证）
- 仅 Windows Server：每月 259 美元（包含许可证）
- 仅计算（Linux）：每月 127 美元

归根结底，自带许可证可能会对在 AWS 上运行 Microsoft 工作负载的成本产生显著影响。如果您使用现有的许可证，则可以降低许可成本并节省总 AWS 账单。

许可证优化

AWS 优化和许可评估 (AWS OLA) 可以通过降低计算和许可成本来帮助您优化许可。AWS OLA 旨在评估您对运行的工作负载 AWS 或计划迁移的工作负载的许可要求。AWS OLA 提供了有关优化许可证使用情况的建议。

优化许可使用的关键策略之一是[合理调整实例大小](#)。适当调整大小涉及到根据工作负载对 CPU、内存和存储的需求来选择合适的实例类型。通过选择适当的实例大小，您可以确保以经济高效的方式使用资源。这能够带来显著的成本节省。

使用 Microsoft 软件许可时，软件运行所在的核心数量是决定软件许可费用的关键因素。例如，Windows Server 和 SQL Server 许可证通常根据核心数量进行许可。通过适当调整实例大小，您可以减少 Microsoft 软件运行所在的内核数量，进而降低实例的成本并减少所需的许可证数量。

成本优化建议

优化许可证是 AWS 上的成本优化的关键组成部分。通过实施适当的策略，您可以降低许可成本，维护合规性，并从您的许可投资中获得最佳效益。本部分概述了几种许可证优化的策略。

自带符合资格的 Windows Server 许可证

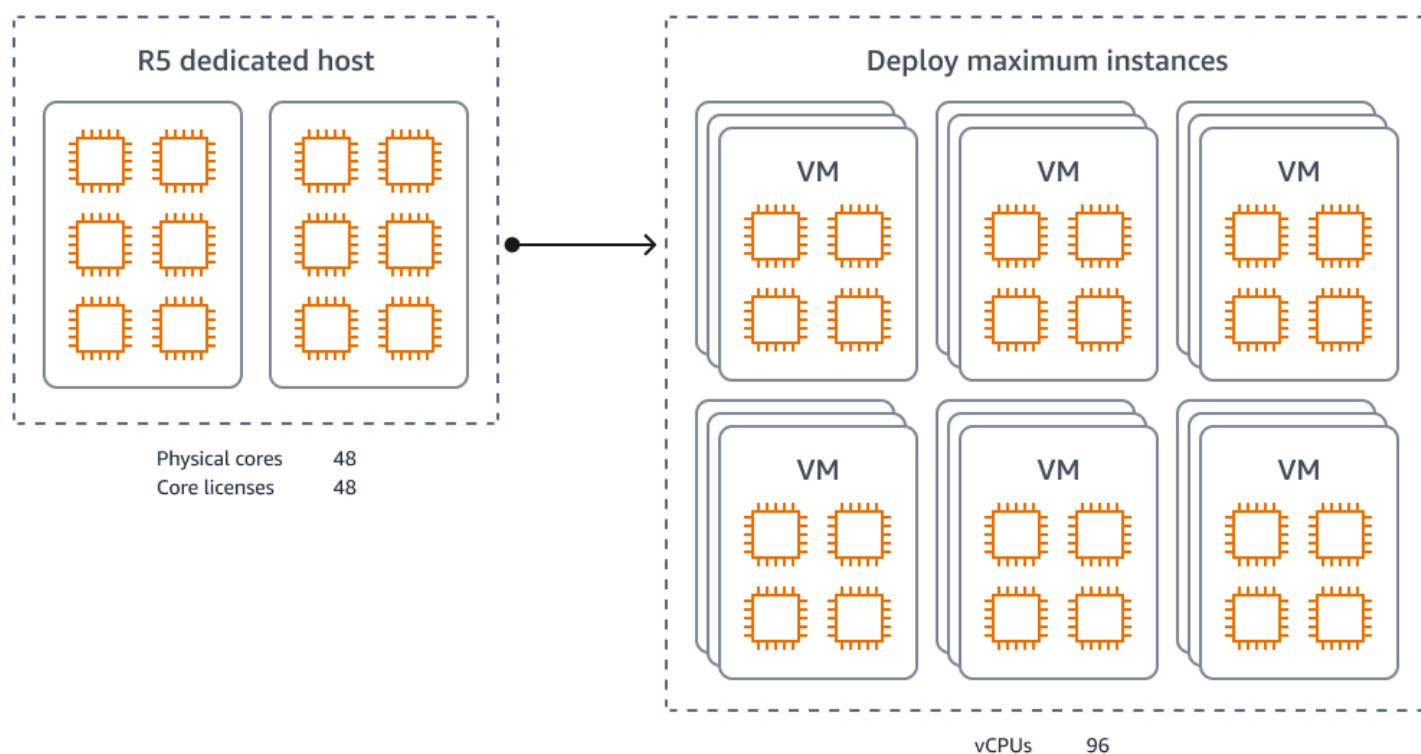
自带 Windows Server 许可证是最有效的许可证优化策略之一。该策略使您能够利用现有投资来减少 AWS 支出。

例如，如果您在 2019 年 1 月 10 日之前购买了许可证，或者在该日期之前根据有效的企业协议进行了许可证的追加购买，则可以在[Amazon EC2 专属主机](#)上部署 Windows Server 2019 及更早版本。该规则基于微软在 2019 年对没有许可移动性的产品（例如 Windows Server）的许可条款和条件所做的更改，这些产品部署在[上市提供商](#)（例如阿里巴巴或谷歌云）上。AWS 根据新条款，你不能将自己的 Windows Server 许可证带到 AWS 而必须使用包含许可证的实例。然而，如果您在此之前已购买了永久许可证，那么您仍然可以将这些 Windows Server 许可证部署到 Amazon EC2 专属主机。

物理级别的许可证

在物理核心级别进行许可，允许您仅许可主机的物理核心，这样一来，您就可以在不影响所需许可证数量的情况下部署最多数量的实例。这通常通过使用 Windows Server 数据中心和 SQL Server 企业版来完成。

举个例子，假设一台具有 48 个内核的 R5 专用主机，相当于 96 vCPU。如果您使用 Windows 服务器数据中心版，则只需要 48 个许可证。这使您能够部署高达 96 v 的实例组合 CPUs，如下图所示。



如果您有足够的工作负载，能够最大限度地利用一台主机来运行多个实例，那么这种方法将会特别具有成本效益。通过在物理核心级别进行许可，您可以避免为每个实例支付额外的许可费用，并实现您在许可方面的最大投资效益。

在 SQL Server 的物理核心级别进行许可

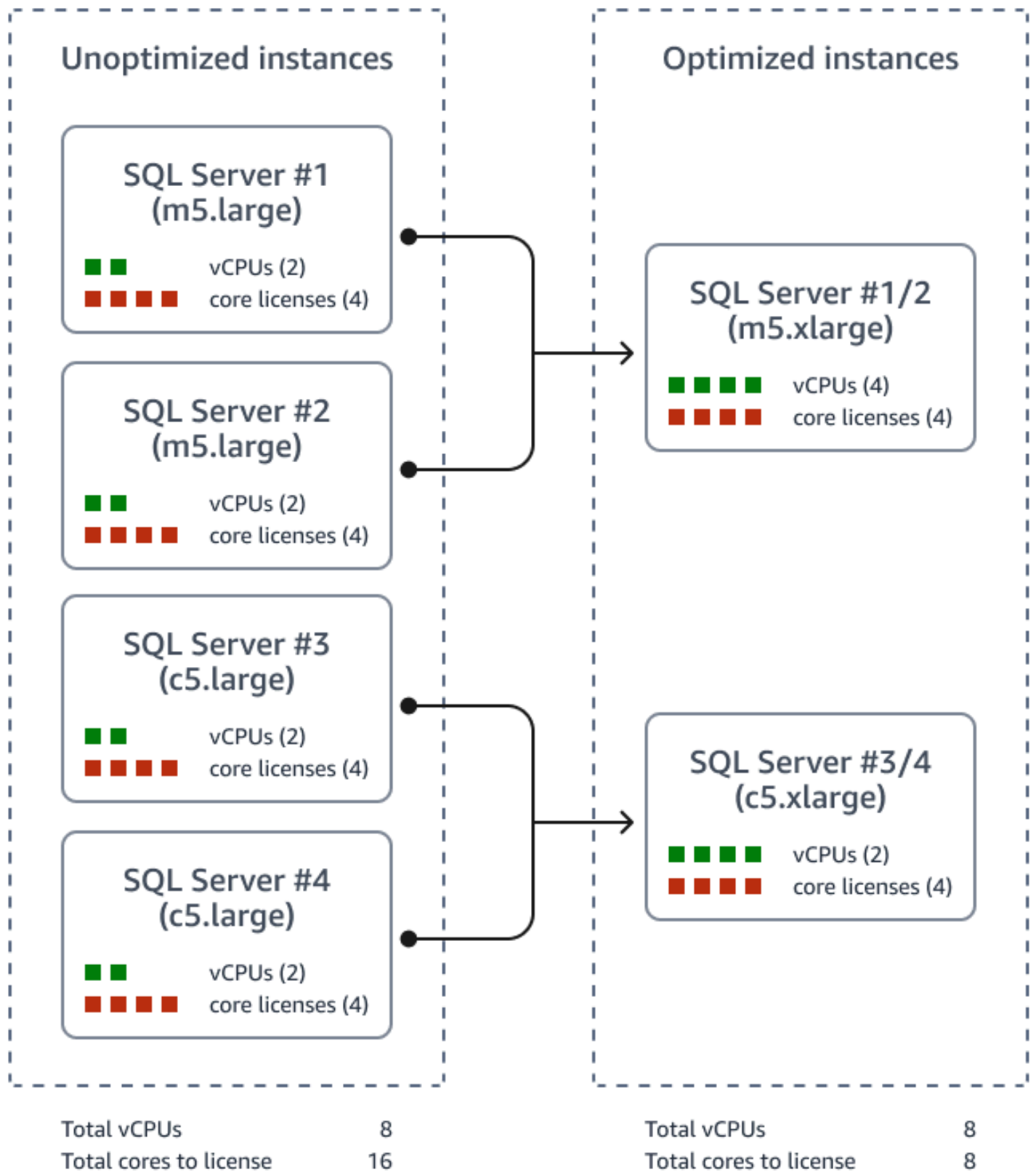
在共享租赁中，SQL Server 许可基于 CPUs 分配给实例的 v 数。相比之下，对于专属主机，您可以按物理核心级别或 vCPU 级别来许可 SQL Server 企业版。

与之前提到的 R5 专属主机的示例一样，如果您在物理核心级别许可了 SQL Server 企业版，那么您只需 48 个 SQL Server 企业版许可证即可为这些主机进行许可。相比之下，在共享租赁模式中（在这种模式下，唯一的选项是按 vCPU 许可使用），要完成相同的工作负载，您必须拥有 96 个 SQL Server

企业版许可证。因此，与共享租赁相比，专属主机最多可以为您节省 50% 的 SQL Server 许可成本。此外，通过自带符合条件的 Windows 许可证，还可以节省实例成本。

整合 SQL Server 实例

[SQL Server 整合](#)是将多个 SQL Server 实例合并到一台服务器的过程。SQL Server 要求每个实例至少有四个核心许可证，即使该实例只有两个 v CPUs。这意味着，在少于四个内核的服务器上运行 SQL Server 可能会导致您过度许可这些实例并使用超出必要数量的许可证。



例如，将两个 CPUs 各有两个 v 的实例整合为一个带有 4 v 的实例，CPUs 可以将许可要求降低 50%。这是因为只需要四个核心许可证，而不是八个。

有关整合的更多信息，请参阅本指南的 [SQL Server 整合](#) 部分。

降低 SQL Server 版本级别

[更改 SQL Server 版本](#) 是优化许可使用情况和降低成本的关键策略之一。SQL Server 的企业版价格要比标准版贵得多，因此降级可以节省大量费用。

透明数据加密 (TDE) 和 Always On 可用性组是 SQL Server 企业版中的两个常用功能。但是，如果您不需要 SQL Server 企业版所包含的全部功能，则还有一些经济高效的替代方案可供您考虑。例如，从 SQL Server 2019 开始，您就可以在 SQL Server 标准版中使用 TDE。在 SQL Server 标准版中，您可以使用开启共享存储的故障转移群集来 FSx 取代 Always On 可用性组，以实现高可用性。

通过从 SQL Server 企业版降级到 SQL Server 标准版，您能够显著降低许可费用。有关更多信息，请参阅 AWS 存储博客 [上 AWS 发布的“优化高可用性 SQL Server 部署的成本”](#)。

除了降低许可成本之外，降低 SQL Server 版本的级别还能帮助您减少软件保障费用，并避免未来可能出现的调整费用。如果您将未使用的许可证放回原处，则能避免额外的许可费用，并最大程度地发挥您的许可投资的价值。

重要的是要仔细评估您的 SQL Server 工作负载，并确定哪些功能对于您的业务需求而言至关重要。有关更多信息，请参阅 AWS 规范性指南中的 [评估您的环境](#)，并确定您的 Microsoft SQL Server 数据库是否使用特定于 SQL Server 企业版的功能。

如果您选择正确的 SQL Server 版本并使用 SQL Server 企业版功能的替代方案，则能够实现显著的成本节省，同时还能确保符合规定并满足业务需求。有关降级选项的更多信息，请参阅本指南的 [比较 SQL Server 版本](#) 部分。

在非生产环境中使用 SQL Server 开发人员版

在非生产环境中，您可以通过在本地环境中使用 MSDN 订阅来部署可授权使用的 SQL Server 版本，例如企业版或标准版。但是，MSDN 订阅不具有许可证移动性。因此，如果您迁移到 AWS，则无法将这些许可证带过来。您必须改用 SQL Server 开发人员版。

SQL Server 开发人员版是 SQL Server 的全功能版本，免费提供。此版本适用于 SQL Server 版本 2016 及更高版本。您可以从 Microsoft 网站下载它。SQL Server 开发人员版旨在用于所有非生产环境，例如开发、测试和暂存，前提是它未连接到实时生产数据。

如果您在非生产环境中使用 SQL Server 开发人员版，则可以避免额外的许可费用。有关更多信息，请参阅本指南的[评估 SQL Server 开发人员版](#)部分。

针对 SQL Server 工作负载优化 CPU

在某些情况下，由于 RAM 或网络限制等其他因素，您可能需要选择 CPUs 超过工作负载所需数量的实例类型。但是，AWS 提供了一种解决方案来帮助您在这些情况下优化许可成本。

您可以像大多数带有 SQL Server 核心许可证的客户一样，在 EC2 实例上禁用超线程或关闭 CPU 以限制主机可用的 CPUs 数量。此选项使您能够利用其他实例功能，例如 RAM，同时还能节省购买额外许可证的费用。

例如，如果您部署一个 r5.4xlarge 实例，因为您的工作负载需要 128 GB 的内存，但您只需要八个 SQL Server 内核，则可以在只有八个内核处于活动状态的情况下禁用实例的超线程处理。CPUs 通过这样做，您可以在所需的 SQL Server 许可证上节省 50% 的费用，因为您只需为正在实际使用的八个核心进行许可即可。

实例类型	总计 v CPUs	具有“优化”功能的主动 vCPU CPUs	SQL Server 许可证节省费用
r5.4xlarge	16	8	50%
r5.12xlarge	48	8	83%

优化 CPU 功能可以在 Amazon EC2 启动配置期间进行配置，也可以通过修改现有实例来配置。它也可以应用于 BYOL 和附带许可证的 Amazon EC2 实例。这种灵活性可以帮助您根据工作负载需求调整 CPU 的大小，同时还可以减少 Windows Server 和 SQL Server 许可。对于包含许可证的 Amazon EC2 实例，减少 CPUs 可以立即节省许可成本。

如果您调整了实例的大小，则可以确保为您的工作负载选择最经济高效的实例类型。随着新实例类型的 AWS 推出，评估这些新实例能否以更少的内核满足工作负载要求非常重要。

其他资源

- [亚马逊 Web Services 和微软：常见问题解答](#) (AWS 文档)

为 SQL Server 工作负载选择合适的 EC2 实例。

 Important

在阅读本部分之前，我们建议您先阅读本指南的[了解 SQL Server 许可](#)和[为 Windows 工作负载选择合适的实例类型](#)部分。

概述

微软 SQL Server 在亚马逊弹性计算云 (Amazon EC2) 实例上运行已有 15 年多了。AWS 借鉴了这种经验，并利用它来帮助开发 Amazon EC2 实例，以适应从最低规格一直运行到高性能、多区域集群的 SQL Server 工作负载。

为 SQL Server 选择合适的 EC2 实例在很大程度上取决于您的工作负载。了解 SQL Server 的许可方式、其内存使用情况以及 SQL Server 的功能与 Amazon EC2 产品之间的对应关系，能够帮助您为您的应用程序选择最合适的 EC2 实例。

本部分介绍了各种 SQL Server 工作负载，以及如何将这些工作负载与特定的 EC2 实例相结合，从而将您的许可费用和计算成本控制在最低水平。

成本比较

Amazon EC2 使您能够以自带许可 (BYOL) 或按实际使用量付费的方式使用 Windows Server 和 SQL Server 许可。对于 pay-as-you-go 许可，Windows Server 和 SQL Server 许可证的许可成本包含在 EC2 实例的每小时成本中。例如，您可以选择不同的价格 AMIs 和不同的价格。AMI 的价格取决于该 AMI 所运行的 SQL Server 版本。

Windows Server 和 SQL Server 定价未逐项列出。您将找不到 [AWS 定价计算器](#) 等工具的分项定价。如果您选择包含许可证产品的不同组合，则可以计算出许可费用，如下表所示。

EC2 实例	AMI	计算价格	Windows 许可证价格	SQL 许可证价格	总价格
r5.xlarge	Linux (计算定价)	183.96 美元	-	-	183.96 美元
r5.xlarge	Linux + SQL 开发人员版	183.96 美元	\$0	\$0	183.96 美元

EC2 实例	AMI	计算价格	Windows 许可证价格	SQL 许可证价格	总价格
r5.xlarge	Windows Server (LI)	183.96 美元	134.32 美元	-	318.28 美元
r5.xlarge	Windows + SQL 开发人员版	183.96 美元	134.32 美元	\$0	318.28 美元
r5.xlarge	Windows + SQL Web 版 (LI)	183.96 美元	134.32 美元	49.64 美元	367.92 美元
r5.xlarge	Windows + SQL 标准版 (LI)	183.96 美元	134.32 美元	350.4 美元	668.68 美元
r5.xlarge	Windows + SQL 企业版 (LI)	183.96 美元	134.32 美元	1095 美元	1413.28 美元

 Note

上述表格中的定价基于 us-east-1 区域的按需定价确定。

运行 SQL Server 的最经济有效的方法是先使用较低版本，直到您需要使用更高版本所具备的功能时。有关更多信息，请参阅本指南的[比较 SQL Server 版本](#)部分。从 SQL Server Web 版升级到 SQL Server 标准版，其所需支付的许可费用是 SQL Server Web 版的七倍以上，并且是从标准版升级到企业版的许可费用的三倍以上。许可费用的差异是一个需要考虑的重要因素，在本节的后续部分中对此进行了详细探讨。

成本优化方案

以一个示例场景为例，一家负责追踪运输车辆的分析公司正试图提升其 SQL Server 的性能。在 MACO 专家审查了公司的性能瓶颈后，该公司从 x1e.2xlarge 实例转换到 x2iedn.xlarge 实例。尽管实例大小较小，但对 x2 实例所做的改进通过使用缓冲池扩展，提高了 SQL Server 的性能和优化效果。

这使该公司能够从 SQL Server 企业版降级到 SQL Server 标准版，并将其 SQL Server 许可从 8 v CPUs 减少到 4 v CPUs。

优化前：

服务器	EC2 实例	SQL Server 版本	月度成本
Prod DB1	x1e.2xlarge	Enterprise	3,918.64 美元
Prod DB2	x1e.2xlarge	Enterprise	3,918.64 美元
总计			7,837.28 美元

优化后：

服务器	EC2 实例	SQL Server 版本	月度成本
Prod DB1	x2iedn.xlarge	标准	1,215.00 美元
Prod DB2	x2iedn.xlarge	标准	1,215.00 美元
总计			2,430.00 美元

从 x1e.2xlarge 实例到 x2iedn.xlarge 实例的这一合并更改使该客户每月在其生产数据库服务器上节省了 5407 美元的费用。这使工作负载的总成本降低了 69%。

Note

上述表格中的定价基于 us-east-1 区域的按需定价确定。

成本优化建议

内存优化型实例

SQL Server 的最重要方面之一在于要明白其对内存的依赖程度。SQL Server 尝试使用操作系统未使用的所有可用 RAM (默认安装最多为 2TB)。这样做是出于性能方面的考虑。处理内存中的数据要比频

频繁地从磁盘拉取数据、进行更改然后再写回磁盘要高效得多。相反，SQL Server 会尽可能多地从所连接的数据库中加载数据，并将这些数据保存在 RAM 中。对数据所做的更改发生在内存中，稍后会固化到磁盘中。

Note

有关 SQL Server 如何写入更改的详细说明，请参阅 Microsoft 文档中的[写入页面](#)。

由于 SQL Server 在 RAM 较大时性能更好，因此我们通常建议从 [Amazon EC2 内存优化型](#) 实例类型开始。内存优化型实例功能多样，提供了多种不同的选项。R 系列的 vCPU-to-RAM 比例为 1 比 8，可选择英特尔处理器、AMD 处理器、增强的网络、增强的 EBS 性能、实例存储和增强的处理器速度。对于内存密集型工作负载，还有一个 X 系列，它结合了许多相同的选项，并将比率扩展到 1 vCPU-to-RAM 比 32。由于内存优化型实例具有多种功能，因此您可以将其应用于各种性状和大小的 SQL Server 工作负载。

低于最低资源的工作负载 (小于 4 vCPUs)

虽然某些使用案例在使用可爆发 (T3) 实例时运行效果良好，但我们建议您一般不要将可爆发实例用于 SQL Server 工作负载。SQL Server 的许可基于 CPUs 分配给实例的 v 数。如果 SQL Server 大部分时间处于闲置状态，并且正在获取爆发配额，则您需要为未充分利用的 SQL 许可证支付费用。此外，SQL Server 的最低许可证要求为每台服务器 4 个内核。这意味着，如果您的 SQL Server 工作负载不需要 4 v CPUs 的计算能力，则需要支付未使用的 SQL Server 许可。在这些情况下，最好将[多个 SQL Server 实例整合](#)到一个更大的服务器上。

使用最少资源 (小于 64GB RAM) 的工作负载

许多 RAM 在 64GB 以下的 SQL Server 工作负载并不优先考虑高性能或高可用性。对于这些类型的工作负载，如果应用程序受 Microsoft 的许可规定所约束，则 SQL Server Web 版可能会是一个合适的选择。

Important

根据 Microsoft 的许可条款，SQL Server Web 版的使用案例受限。SQL Server Web 版的许可仅支持公用且可通过互联网访问的网页、网站、Web 应用程序和 Web 服务。它不能用于支持 line-of-business 应用程序 (例如，客户关系管理、企业资源管理和其他类似的应用程序)。

SQL Server 网络版可扩展到 32 v CPUs 和 64 GB 内存，比 SQL Server 标准版便宜 86%。对于资源需求较低的工作负载，使用像 r6a 这样的 AMD 内存优化型实例也是不错的选择，这种实例的计算价格比其对应的 Intel 产品低 10%，而这样做能够将计算和 SQL 许可费用控制在最低水平。

资源需求为平均值的工作负载（小于 128GB RAM）

大多数 RAM 最多为 128GB 的 SQL Server 工作负载采用 SQL Server 标准版。SQL Server 标准版比 SQL Server 企业版便宜 65-75%，并且可以扩展到 48 v CPUs 和 128 GB 内存。由于 128GB 的 RAM 限制通常会在 48 个 vCPU 的限制之前就被耗尽，因此这是大多数希望避免升级到 SQL Server 企业版的客户所关注的重点。

SQL Server 有一个称为[缓冲池扩展](#)的功能。此功能使 SQL Server 能够利用磁盘的一部分空间来充当 RAM 的扩展部分。缓冲池扩展与超快速存储（如[Amazon EC2 实例存储中 NVMe SSDs 使用的存储](#)）结合使用时效果很好。包含实例存储的 Amazon EC2 实例在实例名称中用“d”表示（例如 r5d、r6id 和 x2iedn）。

缓冲池扩展不能替代普通 RAM。但是，如果您需要超过 128GB 的 RAM，您可以将缓冲池扩展功能与 EC2 实例（如 r6id.4xlarge 和 x2iedn.xlarge）结合使用来推迟升级到企业版许可。

高性能工作负载（RAM 超过 128GB）

对于那些对性能要求高的 SQL Server 工作负载而言，进行成本优化颇具挑战性，因为它们依赖大量的资源。然而，了解不同 EC2 实例之间的差异能够帮助您避免做出错误的选择。

下表显示了各种内存优化型 EC2 实例及其性能限制。

	r5b	R6idn	r7iz	x2iedn	x2iezn
处理器	3.1 GHz	3.5 GHz	3.9 GHz	3.5 GHz	4.5 GHz
	第二代 Intel Xeon 处理器	第三代 Intel Xeon 处理器	第四代 Intel Xeon Scalable 处理器	第三代 Intel Xeon 处理器	第二代 Intel Xeon 处理器
CPU:RAM 比率	1:8	1:8	1:8	1:32	1:32
最大 vCPU	96	128	128	128	48

	r5b	R6idn	r7iz	x2iedn	x2iezn
最大 RAM	768 GB	1,024 GB	1,024 GB	4,096 GB	1,536 GB
实例存储	–	NVMe SSD (4x 1900GB)	–	NVMe SSD (2x 1900GB)	–
io2 Block Express	支持	支持	支持	支持	–
Max EBS IOPS	260000	350,000	160000	260000	80,000
Max EBS 吞吐量	60 Gbps	80 Gbps	40Gbps	80 Gbps	19 Gbps
最大网络带宽	25 Gbps	200 Gbps	50 Gbps	100 Gbps	100 Gbps

每个实例都有不同的用途。了解您的 SQL Server 工作负载有助于您选择最适合您的实例类型。

有关属性的详细信息：

- r5b：r5b 中的“b”属性表示此实例类型侧重于高 EBS 性能。在第五代内存优化型实例中，r5b 是首选。它是第一个利用 io2 Block Express 卷并达到最大存储 IOPS 260000 的实例类型。r5b 实例类型对于需要高 EBS 性能的情况来说，仍是一种经济实惠的选择。
- r6idn：与上一代相比，第六代内存优化型实例提供了相当大的改进。与 r5b 相比，r6idn 在 EBS 性能方面的改进更进一步，将最大 IOPS 提升到了 350000。r6idn 还为 tempdb 和缓冲池扩展预留了一个实例存储卷，以进一步提升 SQL Server 的性能。
- x2iedn：x2iedn 与 r6idn 相似。它提供类似级别的增强型 EBS、增强型联网和 NVMe SSD 实例存储，但对于高内存工作负载和低 CPU 数量（较低的 SQL Server 许可成本），其 vCPU-to-RAM 比例为 1:32。
- x2iezn：x2iezn 中的“z”属性表示此实例类型侧重于高处理器性能。Cascade Lake 处理器的全核睿频最高可达 4.5 GHz。我们建议您在希望将 vCPU 数量保持在较低水平的场景中使用此 EC2 实例，并采用 1:32 的 vCPU-to-RAM 比例。这反过来又可以降低 SQL Server 许可费用。
- r7iz：r7iz 中的“z”属性表示此实例类型侧重于高处理器性能。Sapphire rapids 处理器的全核涡轮频率高达 3.9 GHz。与 x2iezn 实例一样，r7iz 优先考虑高频处理器性能，但比例为 1:8。vCPU-to-RAM

其他资源

- [通用型 Amazon EC2 实例](#) (AWS 文档)
- [比较工具](#) (Vantage)
- [许可 — SQL Server](#) (AWS 文档)

整合实例

本部分重点介绍将多个 SQL Server 实例整合到同一台服务器上的成本优化方法，旨在降低许可成本并最大限度地提高资源利用率。

概述

创建实例是安装 SQL Server 数据库引擎这一过程的一部分。SQL Server 实例是一个完整的安装版本，包含其自身的服务器文件、安全登录信息以及系统数据库 (master、model、msdb 和 tempdb)。因为每个实例都有其自身的文件和服务，所以您可以在同一操作系统上安装多个 SQL Server 实例，而这些实例之间不会相互干扰。但是，由于这些实例都安装在同一台服务器上，因此它们共享相同的硬件资源，包括计算能力、内存和网络连接。

在生产环境中，通常每个服务器只会使用一个 SQL Server 实例，这样“繁忙”的实例就不会过度占用共享硬件资源。为每个 SQL Server 实例配备其专属的操作系统，并为其分配独立的资源，这比依赖资源治理方式要好得多。对于那些需要大量 RAM 和 CPU 资源的高性能 SQL Server 工作负载来说尤其适用。

但是，并非所有 SQL Server 工作负载都会消耗大量资源。例如，一些组织会为其每位客户分配专属的 SQL Server 实例，以满足合规或安全方面的要求。对于规模较小的客户或者那些通常不太活跃的客户而言，这意味着以最少的资源来运行 SQL Server 实例。

正如 [Microsoft SQL Server 2019：许可指南](#) 中所述，每台运行 SQL Server 的服务器必须至少占用四个 CPU 许可证。这意味着，即使你运行的服务器只有两个 vCPUs，你仍然必须为四个 v 的 SQL Server 授予许可 CPUs。根据 [微软的公开 SQL Server 定价](#)，如果你使用 SQL Server 标准版，则相差 3,945 美元。对于那些使用单一 SQL Server 实例来运行多台服务器且资源使用量最低的组织而言，由于必须为未使用的资源支付许可费用，这可能会产生相当大的成本。

成本优化方案

本节探讨了一个示例场景，该场景对比了运行四台 Windows Server 服务器（每台服务器仅配备一个 SQL Server 实例）与运行一台更大规模的 Windows Server 服务器（该服务器同时运行多个 SQL Server 实例）之间的差异。

如果每个 SQL Server 实例只需要两个 v CPUs 和 8 GB 内存，则每台服务器的 SQL Server 许可证总成本为 7,890 美元，此外每小时的计算成本为 0.096 美元。

EC2 实例	v CPUs	RAM	Price	v CPUs 获得许可	SQL Server 许可总成本
m6i.large	2	8	0.096	4	7,890 美元

如果将服务器数量扩大到四台，那么总的费用将是 31560 美元，其中包括 SQL Server 的许可费用，以及每小时 0.384 美元的计算费用。

EC2 实例	v CPUs	RAM	Price	v CPUs 获得许可	SQL Server 许可总成本
4x m6i.large	2	32	0.384	16	31,560 美元

如果将全部四个 SQL Server 实例整合到一个 EC2 实例中，则总的计算资源量和计算能力将保持不变。但是，通过消除不必要的 SQL Server 许可成本，您能够将运行该工作负载的总成本降低 15780 美元。

EC2 实例	v CPUs	RAM	Price	v CPUs 获得许可	SQL Server 许可总成本
m6i.2xlarge	8	32	0.384	8	15,780 美元

Note

在前面的表格中，计算成本显示了在 us-east-1 区域运行 Windows Server 的 Amazon EC2 服务器的每小时按需定价。SQL Server 标准版的许可费用指的是 [Microsoft 的 SQL Server 公开定价](#)。

成本优化建议

如果您正在考虑整合 SQL Server 实例，那么您最关注的问题就是您想要整合的每个实例所消耗的资源情况。为了更好地了解每台服务器上的工作负载模式，收集一段时间内的性能指标很重要。一些用于监控资源消耗的常用工具包括[亚马逊 CloudWatch](#)、[Windows 性能监控器](#) (perfmon) 和 SQL Server 的[本机监控工具](#)。

我们建议您在分析您的 SQL Server 工作负载是否能够合并以使用同一服务器资源（同时确保这些资源不会相互干扰）时，考虑以下问题：

- 稳定运行状态下消耗了哪些资源（CPU、内存和网络带宽）？
- 峰值状态下消耗了哪些资源（CPU、内存和网络带宽）？
- 峰值多久发生一次？峰值是否一致？
- 一台服务器的资源峰值是否与另一台服务器的资源峰值重合？
- SQL Server 使用的存储 IOPS 和吞吐量分别是多少？

如果您打算继续实施将 SQL Server 实例合并的计划，请参阅 AWS 云运营与迁移博客中的[在一个 Amazon EC2 实例上运行多个 SQL Server 实例](#)一文。这篇文章详细介绍了如何在 SQL Server 中进行配置更改以添加更多实例。开始操作之前，请先考虑一下在同一台服务器上安装多个实例时可能出现的细微差异：

- 默认的 SQL Server 数据库实例名为 MSSQLSERVER 并使用端口 1433。
- 安装在同一台服务器上的每个其他实例都是一个“命名的”数据库实例。
- 每个命名的实例都有一个唯一的实例名称和一个唯一的端口。
- [SQL Server 浏览器](#)必须运行，以便协调对命名实例的访问流量。
- 每个实例都可以为数据库数据文件和单独的登录信息分别指定不同的位置。
- SQL Server 的[最大服务器内存设置](#)必须根据每个实例的性能需求进行配置，其组合总内存还会为底层操作系统留出足够的内存。

- 您可以利用 SQL Server 的[本机备份和恢复](#)功能，或[AWS DMS](#)进行迁移或整合。

其他资源

- [SQL Server 许可数据表](#) (AWS 云运营和迁移博客)
- [SQL Server 多实例设置博客文章](#) (AWS 云操作和迁移博客)

比较 SQL Server 版本

概述

Microsoft SQL Server 许可费用是 Windows 工作负载环境中的最大费用之一。SQL Server 的许可费用可能会远远超出运行工作负载所需的计算成本。如果选错了版本，您可能会为一些您根本不用或根本不需要的功能付费。本部分比较了以下 SQL Server 版本，包括其功能和相关成本：

- 企业版：SQL Server 企业版提供数据中心功能，包括高性能、无限虚拟化和多种商业智能 (BI) 工具。
- 标准版：SQL Server 标准版为规模较小的组织和部门提供基本的数据管理和商业智能服务。
- Web — SQL Server 网络版适用于作为网络托管商或网络增值提供商的公司 (VAPs)。此版本具有较低的总拥有成本，并且为从小型到大型规模的 Web 应用程序提供可扩展性和管理能力。

Important

您可以使用 SQL Server Web 版支持公用且可通过互联网访问的网页、网站、Web 应用程序和 Web 服务。您不能使用 SQL Server Web 版来支持 line-of-business 应用程序 (例如客户关系管理或企业资源管理应用程序) 。

- 开发人员版：SQL Server 开发人员版包含企业版的所有功能，但仅用于开发目的。
- 精简版：SQL Server 精简版是一款免费的数据库软件，可用于学习或构建桌面应用程序。您可以将精简版更新到其它版本。

Note

SQL Server 评估版提供 180 天的试用期。

成本影响

您可以从 Microsoft 的经销商处购买 SQL Server 许可证，并使用软件保障将其带到 AWS。或者，您可以将 SQL Server 许可证与包含许可证的 Amazon EC2 pay-as-you-go 型号一起使用。AMIs

如果您从 Microsoft 经销商处购买 SQL Server 许可证，则核心许可证以两份为一包进行出售，每台服务器至少需要购买四份核心许可证。下表显示了企业版和标准版的成本比较。

版本	SQL Server 企业版 (2 个核心一包)	SQL Server 标准版 (2 个核心一包)	节省成本
2022	15,123 美元	3,945 美元	74%
2019	13,748 美元	3,586 美元	74%

Note

上表中的定价基于 Microsoft 对 [SQL Server 2022](#) 和 [SQL Server 2019](#) 的公开定价。

以下成本比较显示了使用包含许可证的 Amazon EC2 托管不同版本的 SQL Server。AMIs 在此比较中，SQL Server 托管在 us-east-1 区域的 r6i.xlarge (4 个 vCPU) 上。

实例	计算成本	Windows 许可证成本	SQL Server 许可证成本	Total
R6i.xlarge (Linux)	183.96 美元	–	–	183.96 美元
R6i.xlarge + Windows	183.96 美元	134.32 美元	–	318.28 美元
R6i.xlarge + SQL Server Web 版	183.96 美元	134.32 美元	49.35 美元	367.63 美元
R6i.xlarge + SQL Server 标准版	183.96 美元	134.32 美元	350.4 美元	668.68 美元

实例	计算成本	Windows 许可证成本	SQL Server 许可证成本	Total
R6i.xlarge + SQL 企业版	183.96 美元	134.32 美元	1,095 美元	1,413.28 美元

通过为您的工作负载选择适当的 SQL Server 版本，您最多可以节省 95% 的 SQL Server 许可费用。下表比较了 r6i.xlarge 实例上 SQL Server 许可证的费用。

Edition	节省成本 %
标准版与企业版对比	68%
Web 版与标准版对比	86%
Web 版与企业版对比	95%

在大多数情况下，组织会从企业版切换到标准版，但也有一些情况下可以从标准版或企业版切换到 Web 版。

成本优化建议

您可以根据扩展限制、高可用性、性能和安全性等因素，为您的工作负载选择最适合的版本。以下表格列出了各 SQL Server 版本所支持的功能。这能帮助您确定使用哪个版本。这种比较适用于 [SQL Server 2016 SP1 及更高版本](#)。

扩展限制

以下表格对比了不同 SQL Server 版本的扩展限制情况。

功能	企业版	标准版	Web 版	精简版
单个 SQL Server 数据库引擎、SQL Server 分析服务 (SSAS	操作系统最大值	最多只能选择 4 个插槽或者 24 个核心中的较少者	最多只能选择 4 个插槽或者 16 个核心中的较少者	最多只能选择 4 个插槽或者 4 个核心中的较少者

功能	企业版	标准版	Web 版	精简版
) 或 SQL Server Reporting Services (SSRS) 实例所使用的最大计算容量				
每个 SQL Server 数据库引擎实例的缓冲池最大内存	操作系统最大值	128 GB	64 GB	1410 MB
每个 SQL Server 数据库引擎实例的缓冲池扩展的最大容量	已配置最大内存的 32 倍	已配置最大内存的 4 倍	不适用	不适用
最大关系数据库大小	524 PB	524 PB	524 PB	10 GB
Columnstore 缓存或内存优化型数据的最大内存	操作系统最大值	32 GB	16 GB	352 MB

如果您的应用程序需要少于 16 个内核 (32 vCPUs) 和 64 GB 的内存，则可以从 SQL Server 网页版开始评估。如果您的工作负载需要超过 64 GB 的内存或者其他高可用性选项，则您必须升级到 SQL Server 标准版。

您可以使用 SQL Server Web 版来支持面向公众和互联网的网页、网站、Web 应用程序以及 Web 服务，但不能使用 SQL Server Web 版来支持业务线应用程序。有关 SQL Server Web 版使用案例的更多信息，请联系 [Microsoft 许可支持部门](#) 或您的 Microsoft 经销商。

您可以将 SQL Server 标准版用于内存不超过 24 个内核 (48 vCPUs) 和 128 GB 内存的工作负载。但是，您可以使用[缓冲池扩展](#)来使 SQL Server 标准版能够利用[本地实例存储](#)，就像 r6id EC2 实例中所具备的那种存储方式一样。这样可以将内存最高扩展到最大内存配置的四倍。这种组合功能能够在内存需求开始增加时，推迟服务器升级至企业版的过程。

您可以通过在缓冲池和[页生命周期](#)计数器中查找数据库页面来识别内存利用率。页生命周期是指页面在内存中停留的时间（即数据在内存中保存多久后才会被刷新回磁盘）。该计数器的默认值为 300。如果页面在内存中停留数小时或数天，则有降低内存分配量的可能性。

高可用性

以下表格对比了不同 SQL Server 版本的高可用性功能。

功能	企业版	标准版	Web 版	精简版
服务器核心支持 1	支持	是	是	是
日志传送	支持	是	是	否
数据库镜像	是	完全安全模式	仅作为见证	仅作为见证
备份压缩	支持	是	否	否
Always On 失效 转移群集实例	16 个节点	2 个节点	否	否
Always On 可用 性组	最多 8 个辅助副本，包括 2 个同步辅助副本	否	否	否
基本可用性组	否	2 个节点	否	否
在线页面和文件 恢复	是	否	否	否
在线索引	是	否	否	否
在线架构更改	是	否	否	否
快速恢复	是	否	否	否
镜像备份	是	否	否	否
热添加内存和 CPU	是	否	否	否

功能	企业版	标准版	Web 版	精简版
加密备份	支持	是	否	否
混合备份到 Microsoft Azure (备份到 URL)	支持	是	否	否
用于灾难恢复的失效转移服务器	支持	是	否	否
用于实现高可用性的失效转移服务器	支持	是	否	否

其他常见功能

以下表格对比了不同 SQL Server 版本的最常见功能。有关功能的详尽列表，请参阅 Microsoft 文档中的 [SQL Server 2019 的版本和支持的功能](#)。

功能	企业版	标准版	Web 版	精简版
(性能) 资源调控器	是	否	否	否
(安全) 透明数据库加密 (TDE)	支持	是	否	否
(安全) 可扩展密钥管理 (EKM)	是	否	否	否
(复制) Oracle 出版物	是	否	否	否

功能	企业版	标准版	Web 版	精简版
(复制) 对等事务复制	是	否	否	否
更改数据捕获	支持	是	否	否

SQL Server 开发人员版

所有非生产工作负载 (例如开发、QA、测试、暂存和 UAT 环境) 都可以使用 SQL Server 开发人员版来节省 100% 的 SQL Server 许可费用。[下载 SQL Server](#) 后，您可以通过使用共享租赁的方式在 EC2 实例上安装 SQL Server 开发人员版。SQL Server 开发人员版无需专用基础设施。有关更多信息，请参阅本指南中的对 [SQL Server 开发人员版](#) 的建议。

切换版本

对于现有的工作负载而言，从一个版本切换到另一个版本需要进行大量的测试。遵循最佳实践是检查运行在企业版或标准版上的工作负载，以查看是否使用了特定于版本的功能，以及是否有针对这些功能的替代解决方案。例如，如果您想查看您的数据库是否使用了任何企业级功能，可以像下面的示例命令所示，在所有数据库上运行[动态管理视图 \(DMV \)](#)。

```
SELECT feature_name FROM sys.dm_db_persisted_sku_features; GO
```

有些企业版的功能无法通过 T-SQL 来实现，比如作为 SQL 维护作业一部分的在线重新索引功能。这些功能必须手动验证。

迁移注意事项

您对 SQL Server 的许可方式将决定您切换版本的选项。包括 SQL Server AMI 在内的 AMI 将许可费用包含在 EC2 实例的价格中，即将许可费用与 AMI 绑定。您可以使用 [AWS 账单代码](#) 来验证您的 AMI 中包含的 SQL Server 版本。对于 AWS 包含许可证的实例，在操作系统内更改 SQL Server 版本并不会改变与 AMI 相关的账单。您必须将您的数据库迁移至一个新的 EC2 实例，并使用运行新版本 SQL Server 的 AMI 来完成迁移操作。

如果您自带许可证，则可以拥有更大的灵活性。通常仍建议您迁移到运行新版本的另一个 EC2 实例上。这样一来，如果出现未按预期进行的情况，也能方便地进行失效自动恢复。但是，如果必须使用现有服务器，则仍然可以 side-by-side 安装 SQL Server 并在实例之间迁移数据库。有关 side-by-side 版本降级的更多详细步骤，请参阅网站上的 [SQL Server 版本升级和降级](#)。MSSQLTips

其他资源

- [SQL Server 2022 的版本和支持功能](#) (Microsoft Learn)
- [sys.dm_db_persisted_sku_features](#) (Transact-SQL) (Microsoft Learn)
- [您应该使用哪个 SQL Server 版本？](#) (Brent Ozar Unlimited)
- [AWS 定价计算器](#) (AWS)

评估 SQL Server 开发人员版

概述

[SQL Server 开发人员版](#)是 SQL Server 的免费版本，包含了企业版的所有功能，并且可以在任何非生产环境中使用。在无法使用 Microsoft 开发人员网络 (MSDN) 许可的云中，SQL Server 开发人员版是节省成本的好方法，无需为开发和测试工作负载提供许可证。这对于那些运行庞大的开发和测试环境并试图降低不必要的开支的团队来说尤其适用。

生产环境是指由应用程序 (如互联网网站) 的最终用户所访问的环境，其用途不仅限于收集反馈或对该应用程序进行验收测试。其他构成生产环境的场景包括：

- 连接到生产数据库的环境
- 能够为生产环境提供灾难恢复或备份功能的环境
- 至少部分时间用于生产的环境，例如在活动高峰期被轮换至生产模式使用的服务器。

有关更多许可信息，请参阅 AWS 文档中的 [Amazon Web Services](#) 和 [Microsoft：常见问题](#)。

成本影响

如果您将 SQL Server 开发人员版用于非生产工作负载，那么您就可以 100% 节省目前用于开发和测试环境的 SQL Server 许可费用。

SQL Server 版本	SQL Server 企业版 (2 个核心一包)	SQL Server 标准版 (2 个核心一包)	SQL Server 开发人员版
2022	15,123 美元	3,945 美元	免费
2019	13,748 美元	3,586 美元	免费

Note

上表中的定价基于 Microsoft 对 [SQL Server 2022](#) 和 [SQL Server 2019](#) 的公开定价。

下表比较了在该 us-east-2 地区运行 4 v CPUs 和使用按需定价的不同 SQL Server 版本的成本。这适用于依赖中包含许可证的实例的场景。AWS

EC2 实例	AMI	计算价格	Windows 许可证价格	SQL Server 许可证价格	总价格
r5.xlarge	Linux (计算定价)	183.96 美元	–	–	183.96 美元
r5.xlarge	Linux + SQL Server 开发人员版	183.96 美元	\$0	\$0	183.96 美元
r5.xlarge	Windows Server (LI)	183.96 美元	134.32 美元	–	318.28 美元
r5.xlarge	Windows + SQL Server 开发人员版	183.96 美元	134.32 美元	\$0	318.28 美元
r5.xlarge	Windows + SQL Server Web 版 (LI)	183.96 美元	134.32 美元	49.64 美元	367.92 美元
r5.xlarge	Windows + SQL Server 标准版 (LI)	183.96 美元	134.32 美元	350.4 美元	668.68 美元
r5.xlarge	Windows + SQL Server 企业版 (LI)	183.96 美元	134.32 美元	1095 美元	1413.28 美元

成本优化方案

一家数据完整性公司完成了新一轮的收购之后，打算将新收购的工作负载从其当前托管于一家服务提供商的位置迁移出来，以便与 AWS 云中的其他工作负载进行整合。初步定价显示，该公司的 SQL Server 工作负载运行成本将 AWS 比当前托管服务提供商高 60%。一家 MACO SME 对这一估算结果进行了评估，发现客户实际上是在向托管服务提供商支付用于其开发和测试环境的 SQL Server 许可费用。通过在迁移过程中将非生产工作负载切换到 SQL Server 开发人员版，该公司将其 SQL Server 许可费用减低了 40%。

Amazon EC2 上包含的 SQL Server 许可证

如果您在 EC2 实例上安装了 SQL Server AMIs，并且使用[附带许可证](#)，则无法直接从企业版转换为开发者版。包含许可证的实例的许可费用与 AMI 相关。即使从操作系统内部卸载了 SQL Server，EC2 实例仍需支付许可费用。

要转换为开发人员版，必须[下载 SQL Server 开发人员版](#)，将其安装到一个新的 EC2 实例上，然后迁移您的数据库。您可以通过多种方式在 EC2 实例之间迁移 SQL Server 数据库。有关更多信息，请参阅将 Microsoft SQL Server 数据库迁移到 AWS 云指南中的[SQL Server 数据库迁移方法](#)。您还可以使用[自动化 SQL Server 开发人员解决方案](#)来准备您计划迁移到的新实例。

Amazon EC2 上的 SQL Server BYOL

如果您有使用 BYOL 的 SQL Server 实例，则可以从以下就地转换或 side-by-side 降级选项中进行选择：

- 从 Microsoft 网站下载[SQL Server 开发人员版](#)。有关手动或自动安装的说明，请参阅 AWS 博客上的[自动化 SQL Server 开发人员部署](#)一文。
- 使用[SQL Server 本机备份和恢复](#)将数据库或 detach/attach 数据库从一个 SQL 实例迁移到另一个 SQL 实例。
- 使用[自动化工具](#)进行批量部署。

Note

SQL Server 开发人员版仅适用于非生产环境。

其他资源

- [自动部署 SQL Server 开发人员以在 EC2 上部署 SQL Server 开发人员版](#) (AWS 博客)

- [SQL 2022 定价](#) (Microsoft)
- [SQL 2019 定价](#) (Microsoft)
- [许可选项](#) (Amazon EC2 上的 SQL Server)
- [AWS 定价计算器](#) (Amazon EC2 上的 SQL Server 文档)
- [Microsoft SQL Server 2019 许可指南](#) (从 Microsoft 中下载)
- [SQL Server 2022 开发人员版](#) (从 Microsoft 中下载)

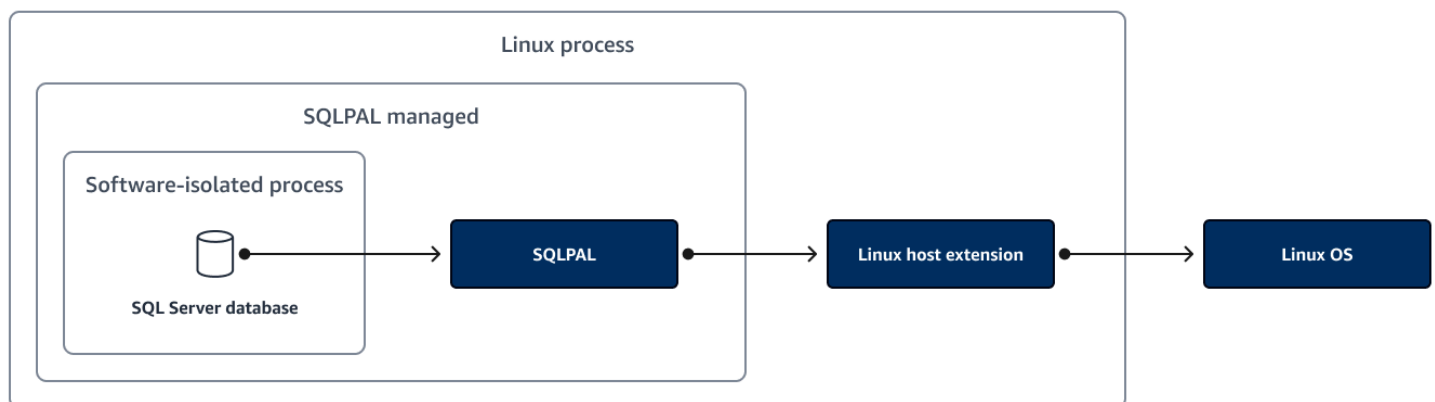
评估 Linux 上的 SQL Server

概述

自 SQL Server 2017 开始，可以在 Linux 操作系统上安装 SQL Server。Linux 上的 SQL Server 已为企业做好准备，具有灵活性、高性能、安全 HA/DR 功能、降低的总体拥有成本、功能和出色的用户体验。您可以从 Windows Server 上的 SQL Server 切换到 Linux 上的 SQL Server，以节省 Windows Server 的许可费用。

对于 Linux，SQL Server 可以在 Red Hat Enterprise Linux (RHEL)、SUSE Linux Enterprise Server (SLES)、Ubuntu 和 Amazon Linux 2 上部署。SQL Server 数据库引擎在 Windows Server 和 Linux 上的运行方式是相同的，但在使用 Linux 系统时，某些任务存在一些根本性的变化。在 Linux 和 Windows 上运行 SQL Server Always On 应用程序的一个关键区别与失效转移群集有关。如果您在 Windows Server 主机上部署 Always On 可用性组，则可以利用 [Windows Server 失效转移群集 \(WSFC \)](#) 和 Active Directory 作为支持失效转移群集的内置功能。但是，WSFC 和 Active Directory 都无法在 Linux 上支持失效转移群集。如果要在 Linux 上启动 SQL Server 的故障转移群集，则可以使用 [ClusterLabs Pacemaker](#) 或 [AWS Launch Wizard](#) 来简化集群设置和 Linux 实例上的 SQL 安装。

Windows 和 Linux 上的 SQL Server 共享一个共同的代码库。也就是说，SQL Server 核心引擎根本没有任何更改即可在 Linux 上运行。SQL Server 引入了平台抽象层 (SQLPAL)，如下图所示。



SQLPAL 负责对 SQL Server 与底层操作系统之间的调用及通信进行抽象处理。主机扩展只是一个原生的 Linux 应用程序。低级操作系统函数是针对优化 I/O、内存和 CPU 使用情况而进行的原生调用。当主机扩展启动时，它会加载并初始化 SQLPAL，随后便会启动 SQL Server。SQLPAL 会启动独立的软件进程，这些进程会为其余的代码提供所需的转换服务。在 SQL Server 架构中新增这一层意味着，那些使 SQL Server 在 Windows 系统上表现如此强大的企业级核心功能和优势，在任何操作系统环境下都能得以实现。

成本影响

对于 r5.2xlarge 实例，在每种情况下，Windows Server 的许可费用均降低约 268 美元。与使用价格较低的 SQL Server 版本相比，此次成本削减占服务器总成本的比例更高。下表显示成本节省情况。

实例	Edition	Windows 上的 SQL Server 的每月费用	Linux 上的 SQL Server 的每月费用	节省成本
r5.2xlarge	Web	735 美元	466 美元	37%
r5.2xlarge	标准	1,337 美元	1,068 美元	20%
r5.2xlarge	Enterprise	2,826 美元	2,558 美元	10%

 Note

上表中的价格估算基于 us-east-1 区域的按需定价，可以直接在 [AWS 定价计算器](#) 中查看。

以一个示例场景为例，一家处于 SMB 细分领域的 ISV 客户希望在其开发环境方面节省成本。他们已经在 一组 Windows 服务器上使用了 SQL Server 开发人员版。通过从使用 SQL Server 开发人员版的 Windows 切换到使用 SQL Server 开发人员版的 Linux，ISV 客户可以节省 33% 的开发工作负载费用。以下表格列出了此情景下的相关估算成本。

Estimate	月度成本
Windows + SQL Server	9,307.72 美元
Linux + SQL Server	6,218.36 美元

Estimate	月度成本
预估的成本节省	3089.36 美元 (33%)

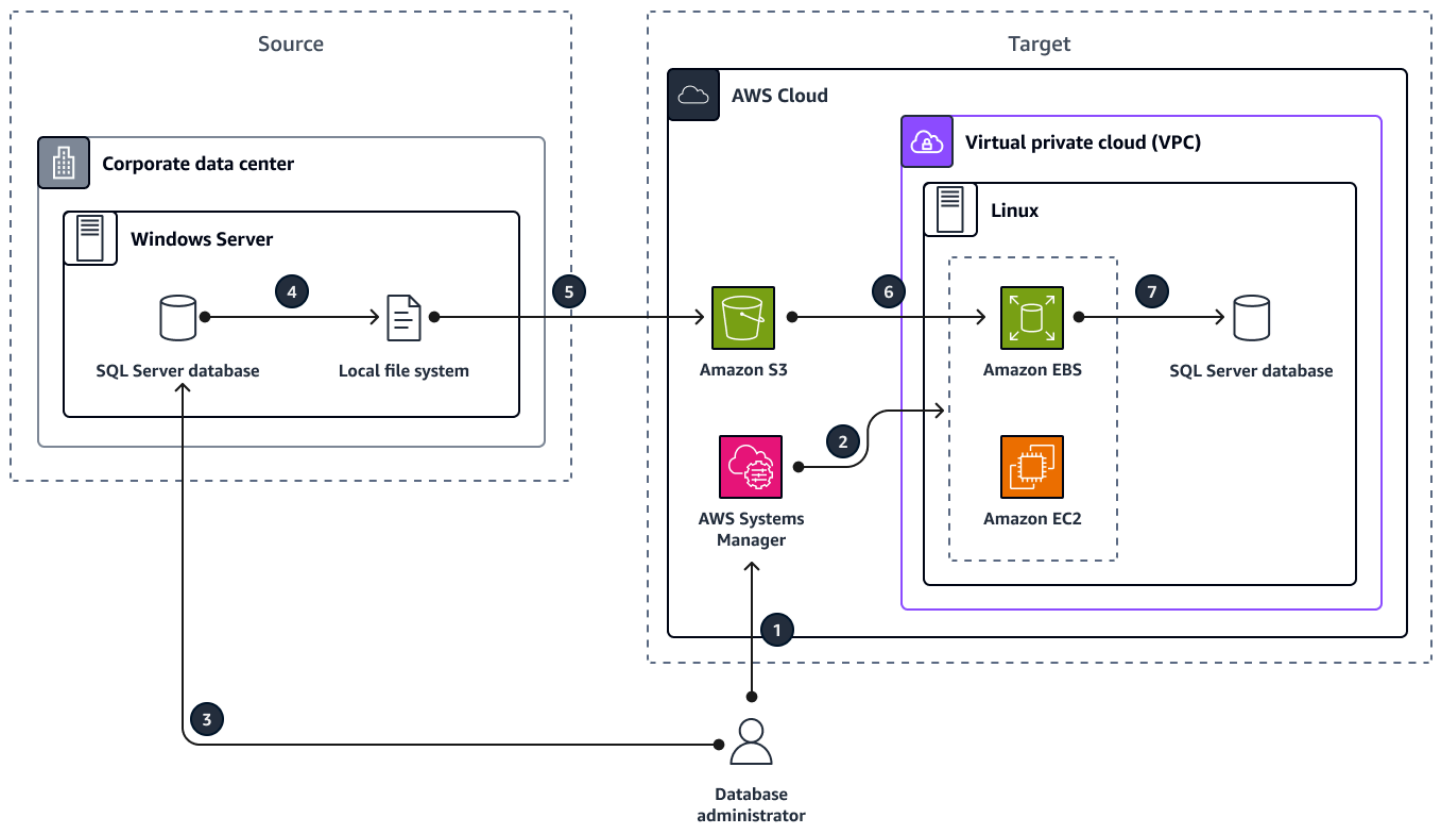
在另一个示例场景中，一家公司将包含许可证的 SQL Server EC2 实例从 Windows 迁移到了 Linux。该公司每年在 Windows Server 的许可费用上节省了总计 30 万美元——约占其整个 AWS 费用的 20%。

成本优化建议

我们建议您考虑以下几点：

- 从 SQL Server 2017 开始，Linux 上的 SQL Server 受到支持。
- 为了帮助进行切换，您可以使用[适用于 Microsoft SQL Server 数据库的 Windows 到 Linux 平台更换助手](#)。平台更换助手是一款脚本工具，能够帮助您将现有的 SQL Server 工作负载从 Windows 迁移到 Linux 操作系统。它会检查常见的不兼容性，从 Windows 主机中导出数据库，然后将这些数据库导入到在 Ubuntu 16.04 上运行 Microsoft SQL Server 2017 的 EC2 实例中。
- 您也可以使用 SQL Server 中的[备份和还原](#)功能从 Windows 上的 SQL Server 切换到 Linux。
- 您可以使用 [AWS Launch Wizard](#) 轻松快速地部署到 Linux 或 Ubuntu 上的 SQL Server。Launch Wizard 可以根据您的应用程序需求，在独立部署和高可用性场景下，在 Linux 或 Ubuntu 系统上部署 SQL Server。有关更多信息，请参阅 Microsoft 工作负载博客上的[“始终在 Linux 上部署到 SQL Server”](#)的 AWS 文章。AWS Launch Wizard

下图展示了采用 Microsoft SQL Server 数据库的 Windows 到 Linux 更换平台助手的解决方案架构。



其他资源

- [Linux 上的 SQL Server 概述](#) (Microsoft Learn)
- [Linux 上的 SQL Server 的安装指南](#) (Microsoft Learn)
- 使用 AWS Launch Wizard (AWS 博客@@ [上的微软工作负载](#)) 在 Linux 上部署到 SQL Server
- [Linux 上的高可用性 SQL Server](#) (AWS 开源博客)

优化 SQL Server 备份策略

概述

大多数组织都在寻找合适的解决方案，以确保其在 [Amazon EC2](#) 上的 SQL Server 的数据安全，从而满足其当前对于恢复点目标 (RPO) (即自上一次备份以来的最大可接受时间间隔) 以及恢复时间目标 (RTO) (即服务中断与恢复服务之间最大可接受延迟时间) 的要求。如果在 EC2 实例上运行 SQL Server，您可以通过多种方式创建数据备份和还原数据。保护 Amazon EC2 上的 SQL Server 数据的备份策略包括以下内容：

- 使用启用了 Windows 卷影复制服务 (VSS) 的 [Amazon Elastic Block Store \(Amazon EBS \)](#) 快照或 [AWS Backup](#) 进行服务器级备份
- 在 SQL Server 中使用[本机备份和还原](#)进行数据库级备份

对于[数据库级本机备份](#)，您具有以下存储选项：

- 使用 [Amazon EBS 卷](#)进行本地备份
- 使用亚马逊 ([FSx 适用于 Windows 文件服务器](#)) 或亚马逊 ([适用于 NetApp ONTAP](#)) [FSx 进行网络文件系统](#)系统备份
- 使用 [AWS Storage Gateway](#) 进行到 Amazon Simple Storage Service (Amazon S3) 的网络备份
- 直接备份到适用于 SQL Server 2022 的 Amazon S3

该部分会执行以下操作：

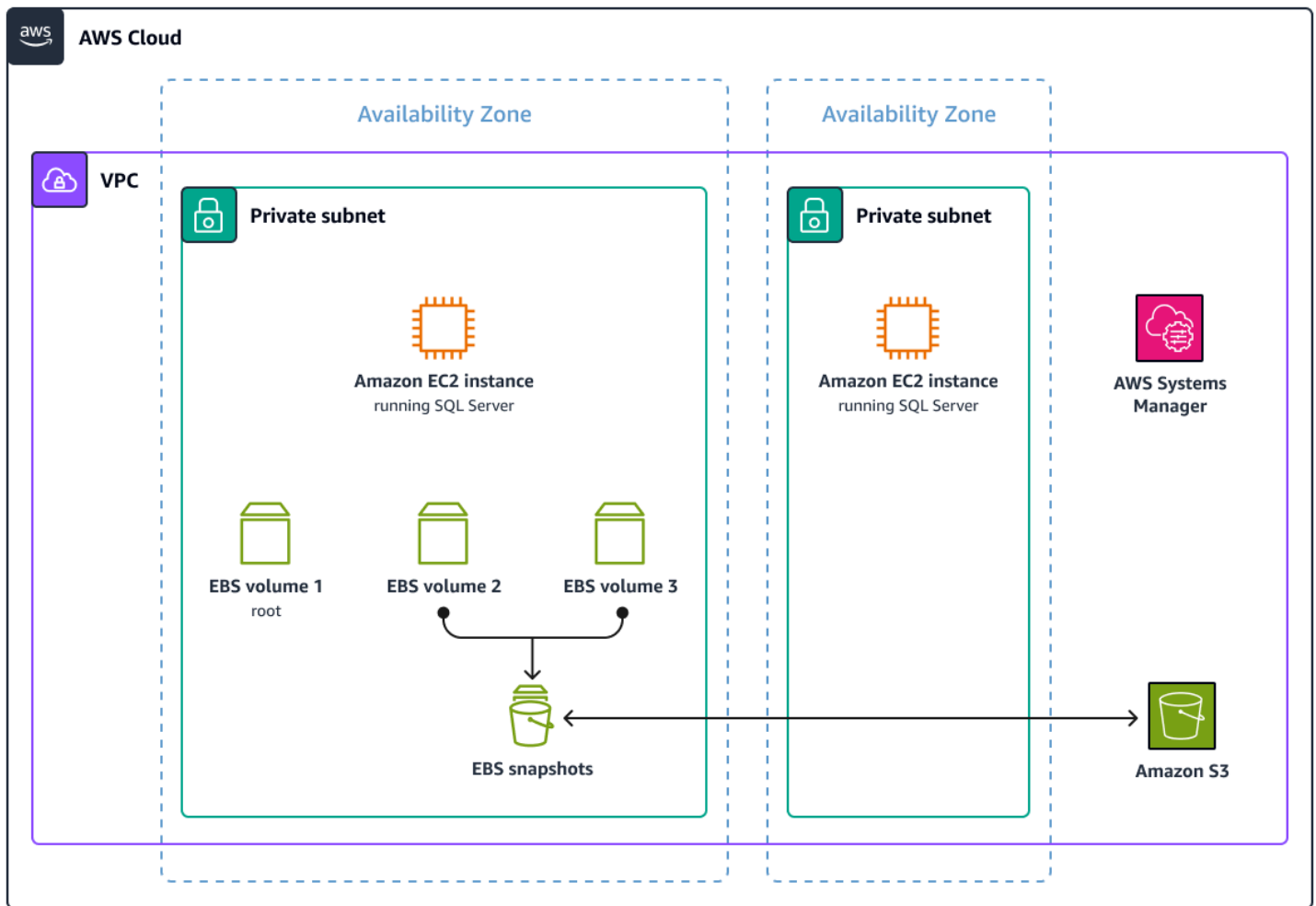
- 突出展示有助于您节省存储空间的功能
- 比较不同后端存储选项的成本
- 提供指向深入文档的链接，以帮助实施这些建议

使用启用 VSS 的快照进行服务器级备份

支持 VSS 的快照架构使用 AWS Systems Manager [运行命令在](#) SQL Server 实例上安装 VSS 代理。您还可以使用 Run Command 调用整个工作流程，包括将操作系统和应用程序缓冲区刷新到磁盘、暂停 I/O 操作、拍摄 EBS 卷的 point-in-time快照，然后恢复 I/O。

这个 Run Command 会为所有连接到目标实例的 EBS 卷创建自动化快照。您还可以选择排除根卷，因为用户数据库文件通常存储在其他卷上。如果您对多个 EBS 卷进行条带化来为 SQL Server 文件创建单个文件系统，Amazon EBS 还支持使用单个 API 命令的崩溃一致性多卷快照。有关[启用 VSS 的应用程序一致性 EBS 快照](#)的更多信息，请参阅 Amazon EC2 文档中的[创建 VSS 应用程序一致性快照](#)。

以下图表展示了使用启用 VSS 的快照进行的服务器级备份的架构。



请考虑使用启用 VSS 的快照的以下益处：

- 数据库实例的第一个快照包含完整数据库实例的数据。相同数据库实例的后续快照为[增量](#)快照，这意味着仅会保存在最新快照后更改的数据。
- EBS 快照提供 point-in-time 恢复功能。
- 您可以[从快照还原到新的 SQL Server EC2 实例](#)。
- 如果使用 Amazon EBS 对实例进行加密，或者使用 TDE 对实例中的数据库进行加密，则该实例或数据库将使用相同的加密方式自动还原。
- 您可以复制您的[自动跨区域备份](#)。
- 从快照中还原 EBS 卷时，应用程序可以立即对其进行访问。这意味着，在从快照中还原 SQL Server 的一个或多个底层 EBS 卷后，您可以立即将 SQL Server 置于在线状态。
- 默认情况下，还原后的卷会在应用程序首次尝试读取卷时从 Amazon S3 获取底层块。这意味着从快照还原 EBS 卷后，性能可能会出现延迟。卷最终捕获到标称性能。但是，您可以使用[快速快照还原 \(FSR\)](#) 快照避免出现该延迟。

- 您可以使用 [EBS 快照的生命周期管理](#)。

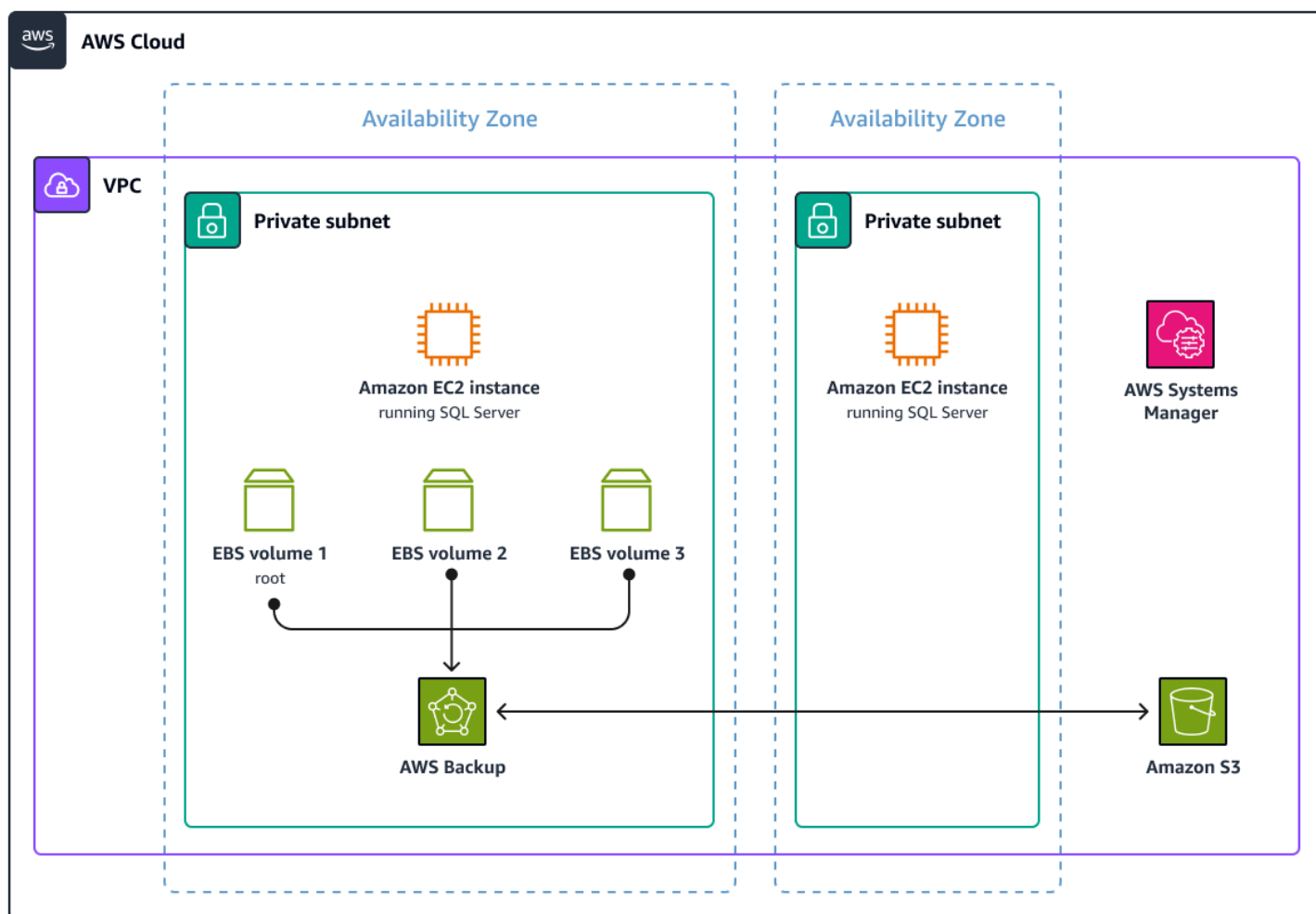
请考虑使用启用 VSS 的快照的以下限制：

- 您无法使用 SQL Server 实例的加密快照执行跨区域 point-in-time 恢复。
- 您无法创建未加密实例的加密快照。
- 您无法还原单个数据库，因为快照是在 EBS 卷级拍摄的。
- 您无法将实例还原到其自身位置。
- 数据库实例的快照必须使用与数据库实例相同的 AWS Key Management Service (AWS KMS) 密钥进行加密。
- I/O 在快照备份过程中，存储会暂停几分之一秒 (大约 10 毫秒)。

使用 SQL 服务器备份 AWS Backup

您可以使用 [AWS Backup](#) 来集中和自动执行数据 AWS 服务保护。AWS Backup 提供了一种经济实惠、完全托管、基于策略的解决方案，可大规模简化数据保护。AWS Backup 还可以帮助您履行监管合规义务并实现业务连续性目标。结合使用 AWS Organizations，AWS Backup 使您能够集中部署数据保护 (备份) 策略，以配置、管理和治理您 AWS 账户 组织和资源的备份活动。

下图显示了通过使用 AWS Backup 在 EC2 上构建 SQL Server 备份与还原解决方案的架构。



请考虑使用 AWS Backup 备份 SQL Server 的以下益处：

- 您可以自动备份计划、保留管理和生命周期管理。
- 您可以将备份策略集中到整个组织中，跨多个账户和。AWS 区域
- 您可以集中监控备份活动，并跨 AWS 服务发出提醒。
- 您可以为灾难恢复计划实施跨区域备份。
- 该解决方案支持跨账户备份。
- 您可以使用二级备份加密来执行安全备份。
- 所有备份都支持使用加密密钥进行 AWS KMS 加密。
- 该解决方案适用于 TDE。
- 您可以从 AWS Backup 控制台中还原到特定的恢复点。
- 您可以备份整个 SQL Server 实例，其中包括所有 SQL Server 数据库。

数据库级备份

这些方法使用 Microsoft SQL Server 的本机备份功能。您可以对 SQL Server 实例上的单个数据库进行备份，也可以恢复单个数据库。

这些用于本机 SQL Server 备份和恢复的选项还支持以下内容：

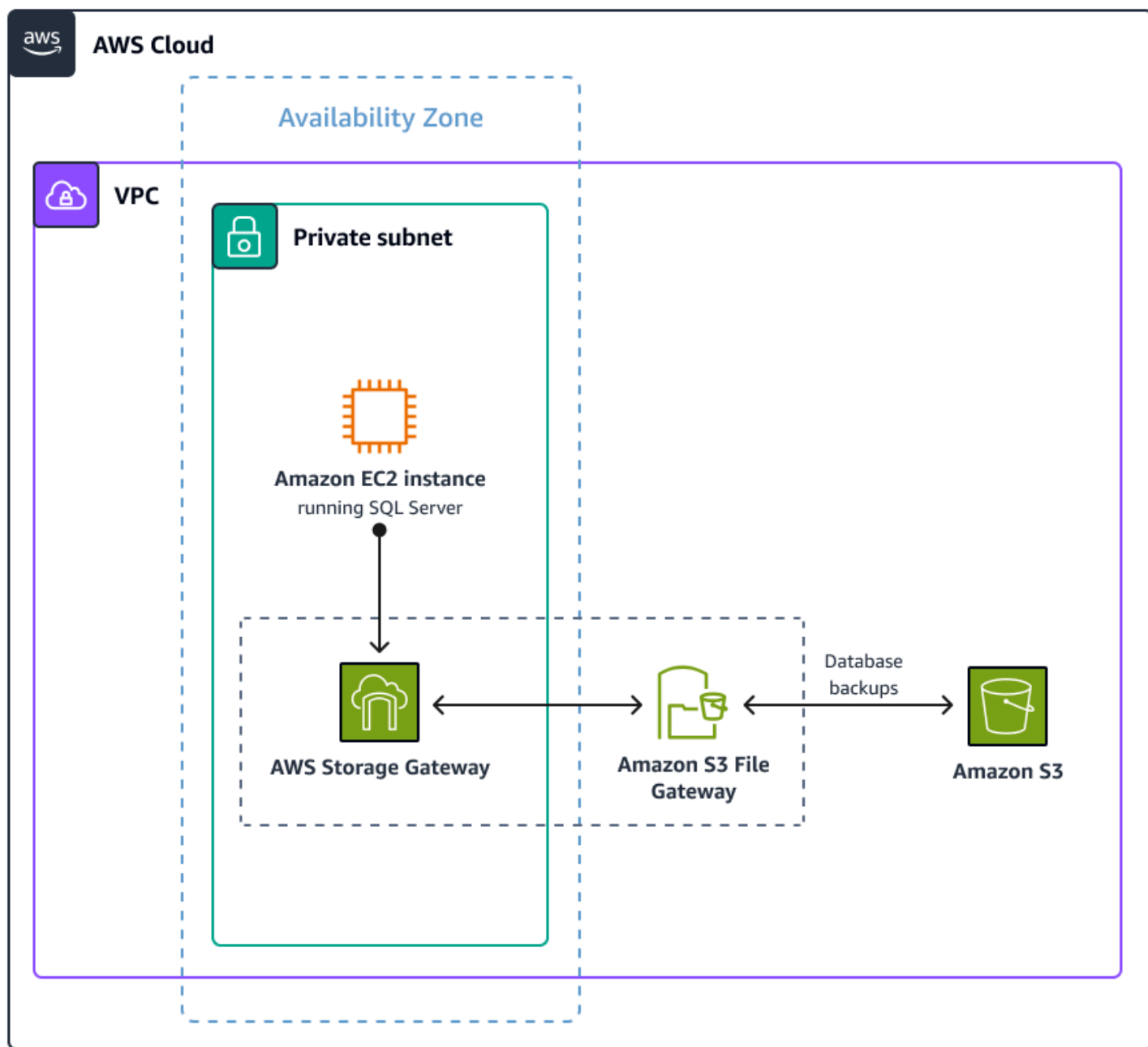
- 压缩备份和多文件备份
- 完整备份、差异备份和 T 日志备份
- TDE 加密的数据库

SQL Server 本机备份并还原到 Amazon S3

Amazon EC2 上的 SQL Server 支持 SQL Server 数据库的本机备份和恢复。您可以对 SQL Server 数据库进行备份，然后将备份文件还原到现有数据库、新的 SQL Server EC2 实例、Amazon RDS for SQL Server 或本地服务器。

Storage Gateway 是一项混合云存储服务，它为本地应用程序提供对几乎无限的云存储的访问权限。您可以使用存储网关将您的 Microsoft SQL Server 数据库直接备份到 Amazon S3，从而减少本地存储占用空间，并使用 Amazon S3 实现持久、可扩展且经济实惠的存储。

下图显示了使用存储网关和 Amazon S3 的本机备份和还原解决方案的架构。



请考虑使用 存储网关进行本机 SQL Server 备份的以下益处：

- 您可以将存储网关映射为 EC2 实例上的服务器消息块 (SMB) 文件共享，并将备份发送到 Amazon S3。
- 备份直接或通过存储网关文件缓存发送到 S3 存储桶。
- 支持多文件备份。

考虑使用存储网关进行本机备份的以下限制：

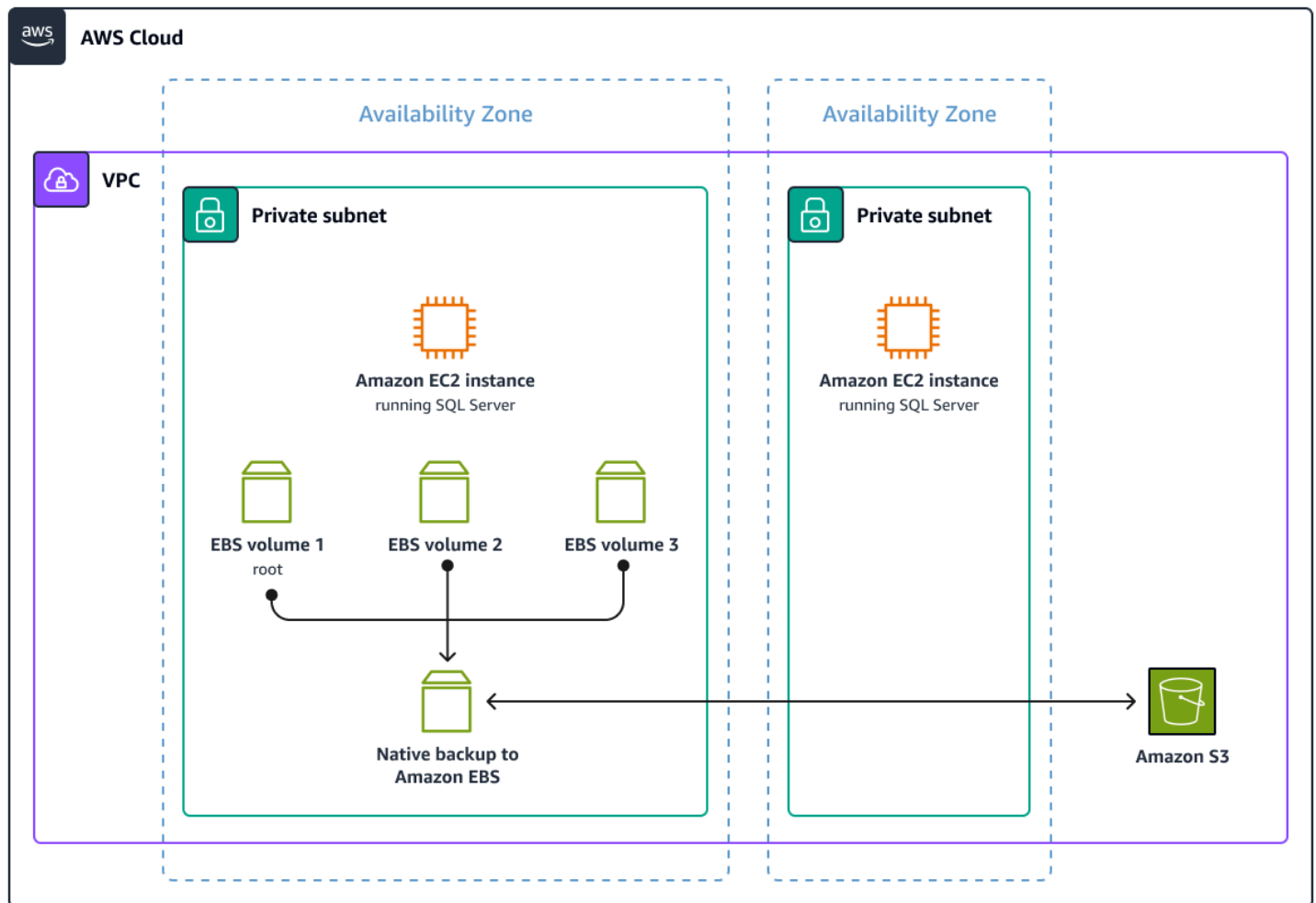
- 必须为每个单独的数据库设置备份和恢复。
- 您必须为备份文件管理 [Amazon S3 生命周期政策](#)。

有关如何设置存储网关的更多信息，请参阅 AWS 博客上的[使用 AWS Storage Gateway 将 SQL Server 备份存储到 Amazon S3 中](#)。

SQL Server 本机备份到 EBS 卷

您可以对 SQL Server 数据库进行本机备份，并将文件存储在 Amazon EBS 卷中。Amazon EBS 是一项高性能的数据块存储服务。EBS 卷具有弹性，支持加密功能。它们可以分离并连接到 EC2 实例。您可以将 EC2 实例上的 SQL Server 在相同或不同的 EBS 卷类型上进行备份。备份到不同的 EBS 卷的一个优点是可以节省成本。

下图显示了 EBS 卷的本机备份的架构。



考虑使用 SQL Server 本机备份到 EBS 卷的以下益处：

- 您可以对 SQL Server EC2 实例上的单个数据库进行备份，也可以恢复单个数据库，无需恢复整个实例。
- 支持多文件备份。
- 您可以使用 SQL Server Agent 和 SQL Server 作业引擎安排备份作业。
- 您可以通过选择硬件来实现更高的性能。例如，您可以使用 st1 存储卷来实现更高的吞吐量。

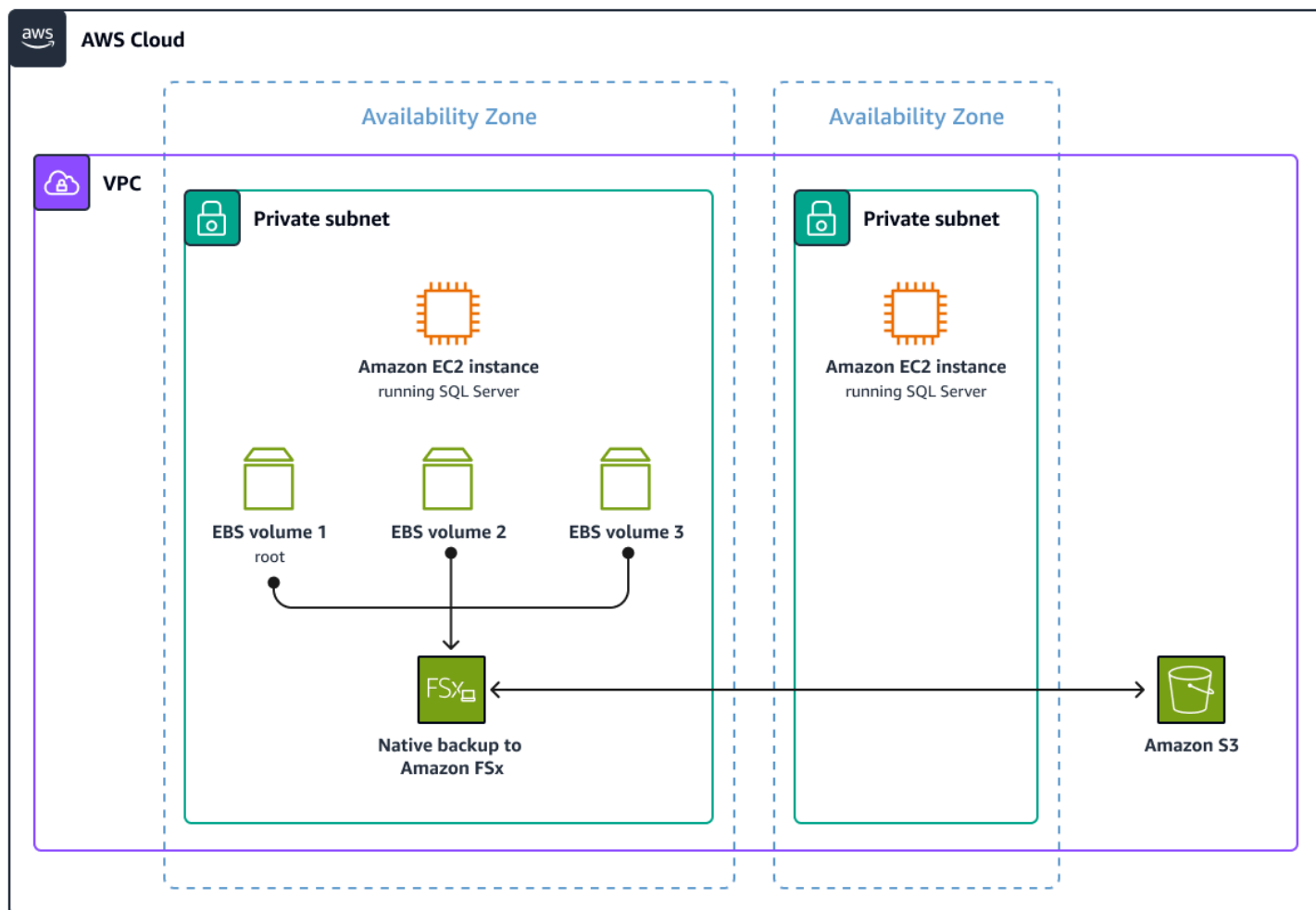
考虑使用到 EBS 卷的本机备份的以下限制：

- 您必须手动将备份从 EBS 卷移至 Amazon S3。
- 对于大型备份，您必须管理 Amazon EC2 上的磁盘空间。
- 在 EC2 实例上，Amazon EBS 吞吐量可能是一个瓶颈。
- 在 Amazon EBS 上存储备份需要额外的存储空间。

FSx 适用于 Windows 文件服务器的 SQL Server 本机备份到亚马逊

[Amazon FSx for Windows 文件服务器](#) 是一个完全托管的本机 Windows 文件系统，可提供高达 64 TB 的存储空间，旨在提供快速、可预测和稳定的性能。AWS 在 Windows 文件服务器上引入了[对多可用区文件系统部署 FSx 的本机支持](#)。本机支持具有跨多个可用区的高可用性和冗余性，使在 AWS 上部署 Windows 文件存储变得更加容易。AWS 还推出了对[SMB 连续可用的 \(CA\) 文件共享](#)的支持。您可以将 Windows 文件服务器 FSx 用作 SQL Server 数据库的备份存储。

下图显示了 Windows 文件服务器的本机 SQL Server 备份 FSx 的体系结构。



考虑使用本机 SQL Server 备份到 FSx Windows 文件服务器有以下好处：

- 您可以将 SQL Server 数据库备份到亚马逊 FSx 文件共享中。
- 您可以对 SQL Server 实例上的单个数据库进行备份，也可以恢复单个数据库，无需恢复整个实例。
- 支持多部分备份。
- 您可以使用 SQL Server Agent 和作业引擎安排备份作业。
- 与 Amazon EBS 相比，这些实例的网络带宽更高。

使用本机 SQL Server 备份到 FSx Windows 文件服务器时，请考虑以下限制：

- 您必须使用 AWS Backup 或手动将备份从亚马逊移至 Amazon FSx S3 AWS DataSync。
- 在 Amazon 上，大型备份可能需要额外的磁盘空间管理开销 FSx。
- EC2 实例网络吞吐量可能是一个瓶颈。
- 需要额外的存储空间才能存储 Windows 文件服务器 FSx 的备份。

FSx 适用于 NetApp ONTAP 的 SQL Server 备份到亚马逊

使用 f FSx or ONTAP 的快照始终保持崩溃一致性，但它们需要您暂停（或暂停）数据库才能创建应用程序一致 I/O 性的快照。您可以将（带有适用于特定应用程序 NetApp SnapCenter（包括 SQL Server）的插件的编排工具）与 FSx for ONTAP 一起使用，以创建应用程序一致的快照并保护、复制和克隆数据库，无需支付额外费用。

NetApp SnapCenter

NetApp SnapCenter 是一个用于实现应用程序一致性数据保护的统一平台。SnapCenter 将快照称为备份。本指南采用相同的命名惯例。SnapCenter 提供了用于管理应用程序一致性备份、恢复和克隆的单一管理面板。您可以为特定的数据库应用程序添加 SnapCenter 插件来创建应用程序一致性备份。SQL Server SnapCenter 插件提供以下功能，可简化您的数据保护工作流程。

- 适用于完整备份和日志备份的细粒度备份与还原选项
- 就地还原并恢复到其他位置

有关更多信息 SnapCenter，请参阅 AWS 存储博客上的“[使用 NetApp SnapCenter Amazon FSx for NetApp ONTAP 保护你的 SQL Server 工作负载](#)”一文。

备份的成本优化

以下选项可以帮助您降低在 AWS 上存储 SQL Server 备份的成本。

- 在创建备份文件期间启用 [SQL Server 压缩](#) 功能，并将尽可能小的文件发送到存储中。例如，3:1 的压缩比意味着您能节省约 66% 的磁盘空间。要对这些列进行查询，您可以使用以下 Transact-SQL 语句：SELECT backup_size/compressed_backup_size FROM msdb..backupset;。
- 对于传送到 S3 存储桶的备份，启用 [Amazon S3 Intelligent-Tiering](#) 存储类别将存储成本降低 30%。
- 对 FSx 于 Windows 文件服务器或 ONTAP FSx 的备份，使用单个可用区可节省 50% 的成本（与使用多个可用区相比）。有关定价信息，[FSx 请参阅亚马逊 Windows 文件服务器定价](#)和[亚马逊 FSx NetApp ONTAP 定价](#)。
- SQL Server 2022 最高效的备份选项是直接备份到 Amazon S3。您可以通过避免使用存储网关来节省额外的费用。

备份的基准测试结果

本部分依据本指南中所涵盖的备份解决方案的性能基准测试结果，从成本和性能的角度，针对一个 1TB 的数据库样本，对以下选项进行比较。

- EC2 实例规格：搭载 Windows Server 2019 和 SQL Server 2019 开发人员版的 r5d.8xlarge
- 数据库规格：禁用 TDE 时大小为 1TB

测试是以 r5d.8xlarge 实例和 1TB SQL Server 数据库作为源进行的。源系统是根据最佳实践配置的，源数据库包含四个数据文件（每个文件为 250GB）和一个日志文件（50GB），分布在不同的 gp3 卷中。SQL Server 本机 BACKUP 命令包括写入 10 个备份文件、使用压缩来优化备份性能，以及减少通过网络发送写入目标的数据量。在所有测试案例中，存储性能都是瓶颈所在。

这些类型的测试几乎有无穷无尽的可能配置。此测试侧重于从性能、成本、可扩展性以及实际使用案例等方面进行优化。以下表格列出了针对备份目标选项所获取的性能指标。

备份选项	级别	运行持续时间 (Appx)	备份速率	每月费用 (美元) *
本机备份到本地 EBS st1 HDD , 2TB	数据库	00:30:46 分钟	554.7 Mbps	92.16 美元
本机备份到本地 EBS SSD gp3 , 2TB	数据库	00:22:00 分钟	512 Mbps	193.84 美元
Windows 文件服务器硬盘 FSx 的本机备份 , 2 TB @512 Mbps 吞吐量	数据库	00:20:58 分钟	814.0Mbps	1,146 美元
Windows 文件服务器固态硬盘 FSx 的本机备份 , 2 TB @512 Mbps 吞吐量	数据库	00:20:00 分钟	814.0Mbps	1,326 美元
本机备份到具有 2TB gp3 的 S3 文件网关	数据库	00:23:20 分钟	731.5Mbps	470.42 美元

备份选项	级别	运行持续时间 (Appx)	备份速率	每月费用 (美元) *
m6i.4xlarge (16 vCPU , 64GB)				
EBS VSS 快照	EBS 卷	00:00:02 秒 00:00:53 秒	N/A 快照	51 美元
AWS Backup (AMI 备份)	AMI	00:00:04 秒 00:08:00 分钟	N/A 快照	75 美元
直接 SQL Server 本机备份到 Amazon S3 (SQL Server 2022)	数据库	00:12:00 分钟	731.5Mbps	前 50TB /月 , 每 GB 0.023 美元 , 每月 23.55 美元
ONTAP FSx 的本机备份 (使用 SnapCenter)	数据库	—	—	440.20 美元

上述表格进行如下假设：

- 数据传输和 Amazon S3 费用不包括在内。
- 存储价格包含在实例定价中。
- 费用以 us-east-1 区域为准。
- 当进行多次备份时，吞吐量和 IOPS 分别增长 10%。而整个月内的总体变化率为 10%。

测试结果显示，最快的选择是将本机 SQL Server 数据库备份到 FSx Windows 文件服务器。备份到存储网关以及本地连接的 EBS 卷是一种成本效益更高的选项，但其性能会相对较慢。对于服务器级备份 (AMI)，我们建议使用 AWS Backup 以获得最佳性能、成本和可管理性。

成本优化建议

了解在 Amazon EC2 上备份 SQL Server 的可能解决方案至关重要，它能够保障您的数据安全、确保您满足备份需求以及制定计划以从关键事件中恢复过来。本部分探讨的备份和恢复 SQL Server 实例及数据库的不同方法，能够帮助您制定出既能保护数据又能满足组织需求的备份和恢复策略。

本部分将介绍以下备份选项：

- 压缩
- Amazon S3 Intelligent-Tiering
- 单可用区
- 备份到 URL

针对这些选项所提供的指南都是较为宏观的。如果您希望在您的组织中实施上述任何一项建议，我们建议您联系您的客户团队。然后，该团队可以与 Microsoft 的专业服务代表进行沟通，由其主导对话过程。您也可以通过发送电子邮件至 optimize-microsoft@amazon.com 进行联系。

总之，我们建议执行下列操作：

- 如果您使用的是 SQL Server 2022，则备份到 Amazon S3 是最经济高效的选项。
- 如果您使用的是 SQL Server 2019 及更早版本的 SQL Server，则考虑备份至由 Amazon S3 支持的存储网关，因为该方法是最经济高效的选项。

压缩

压缩的目的是减少每次备份所占用的存储空间，这对于各种存储选项都是有益的。您必须在 [SQL Server 实例](#) 级别为 SQL Server 备份启用压缩。以下示例展示了如何为备份数据库添加压缩关键字：

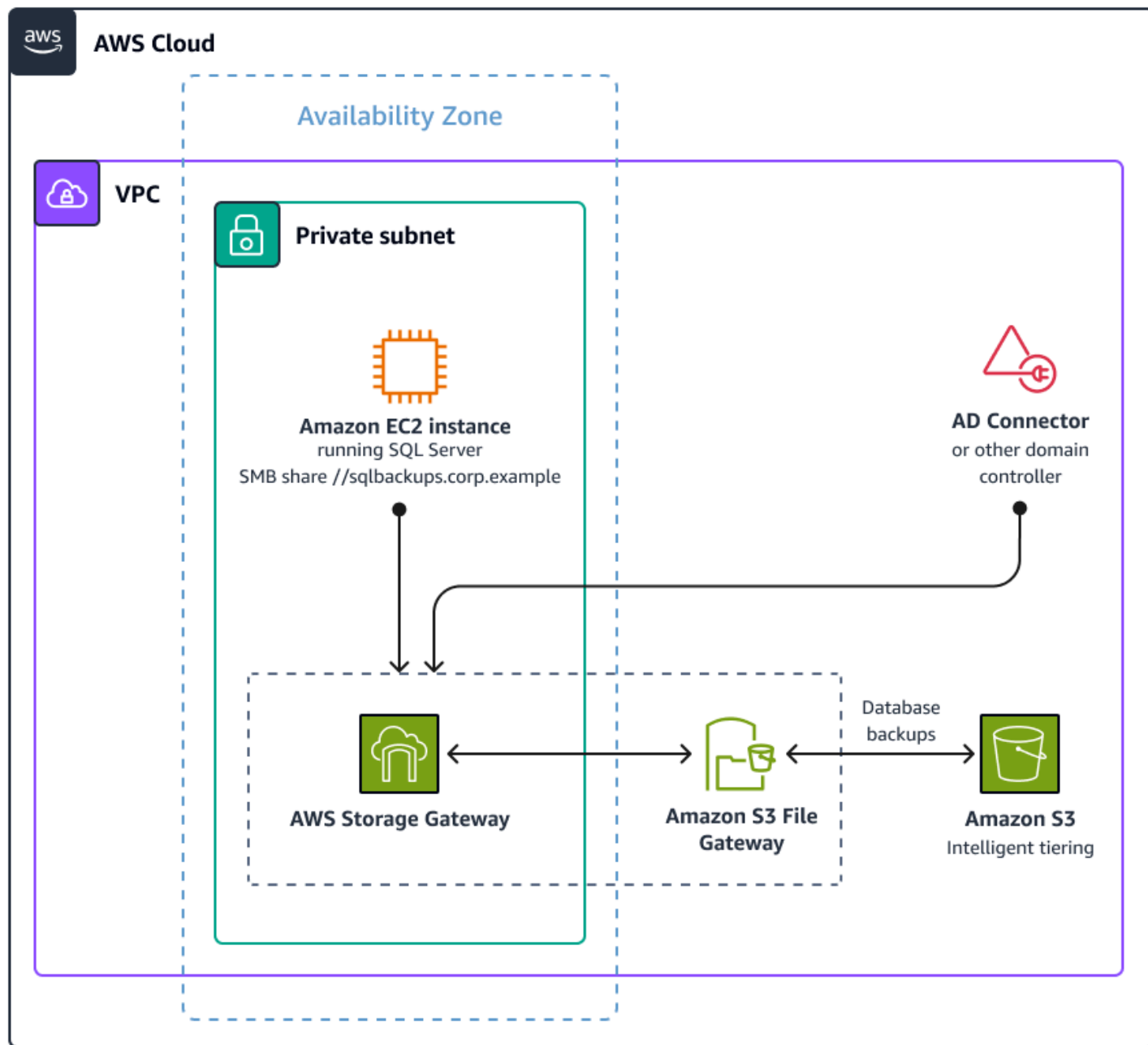
```
BACKUP DATABASE <database_name> TO DISK WITH COMPRESSION (ALGORITHM = QAT_DEFLATE)
```

Amazon S3 Intelligent-Tiering

对于传送到 Amazon S3 存储桶的备份，您可以启用 [Amazon S3 Intelligent-Tiering](#) 作为您的 Amazon S3 文件网关 [存储类别](#)。这会将存储成本最高降低 30%。然后，您可以通过使用与 [Active Directory 与集成的 SMB 文件共享](#) 来将 S3 文件网关挂载到您的 SQL 服务器上。这为您提供了对共享的访问控制，让您能够利用现有的服务账户，并能够通过一个以 Microsoft 为中心的通用文件协议访问 Amazon

S3。对于那些可能无法直接连接到域控制器的账户，您可以使用 [Active Directory Connector](#) 来实现与本地或云的 Active Directory 的通信。要在网关上配置 Active Directory 设置，必须 IP 为域控制器指定 Active Directory 连接器，以代理对活动目录的请求。

以下图表展示了基于 S3 Intelligent-Tiering 的解决方案的一个架构。



默认情况下，写入 S3 存储桶的备份文件使用的是标准层。要将备份文件从标准层转换为 S3 Intelligent-Tiering，必须[创建生命周期规则](#)。您还可以使用 [AWS 管理控制台](#) 启用 S3 Intelligent-Tiering。有关更多信息，请参阅 AWS 文档中的[开始使用 Amazon S3 Intelligent-Tiering](#)。

单可用区

要创建单可用区文件系统，请在[创建 FSx 适用于 Windows 文件服务器的文件系统](#)时选择单可用区选项。Amazon FSx 还使用 Windows 卷影复制服务每天对您的文件系统进行高度持久的备份（存储在 Amazon S3 中），并允许您随时进行其他备份。请记住使用单可用区时所存在的某些问题。例如，如果预调配文件系统的某个受影响可用区因故障而长时间（数小时）无法运行，则该 SMB 文件共享就会无法使用。如果您需要访问这些数据，必须从源区域中的某个可用区中的备份恢复数据。有关更多信息，请参阅本指南的[使用单可用区](#)部分。

备份到 URL

对于 SQL Server 2022，[备份到 URL](#) 功能支持直接备份到 Amazon S3。这是运行 SQL Server 2022 的理想备份方法，AWS 因为您可以在存储层获得 Amazon S3 的完整功能集，并消除了先前版本中实现此功能所需的 AWS Storage Gateway 设备成本。在实施此功能时，需要考虑两项主要费用：数据传输费用以及所选择的 S3 存储类别费用。如果您希望利用 Amazon S3 的原生灾难恢复功能，则必须考虑到[跨区域复制](#)会产生跨区域[数据流出费用](#)。要了解有关如何配置此选项的更多信息，请参阅 AWS 博客上有关 Microsoft 工作负载的[将 SQL Server 数据库备份到 Amazon S3](#)一文。

其他资源

- [亚马逊 EC2 上的 SQL Server 的备份和还原选项](#)（AWS 规范性指南）
- [Point-in-time 使用 AWS Backup（AWS 存储博客）对 Amazon RDS 进行恢复和持续备份](#)
- [使用 NetApp SnapCenter Amazon FSx for NetApp ONTAP 保护你的 SQL Server 工作负载](#)（AWS 存储博客）
- [开始使用 Amazon S3 智能分层](#)（AWS 入门资源中心）
- 适用于 [SQL Server 的 Amazon RDS 的备份和还原策略](#)（AWS 数据库博客）
- [将@@ 本地微软 SQL Server 数据库迁移到亚马逊 EC2](#)（AWS 规范性指南）
- [在亚马逊 EC2 上部署微软 SQL Server 的最佳实践](#)（AWS 白皮书）

现代化 SQL Server 数据库

概述

如果您正着手实现遗留数据库的现代化，以提升其可扩展性、性能和进行成本优化，那么您可能会遇到使用 SQL Server 这类商业数据库时所面临的种种难题。商业数据库价格昂贵，会将客户牢牢束缚在其中，并且其许可条款还颇具惩罚性。本部分对从 SQL Server 迁移与现代化至开源数据库的各种方案进行了概述，并介绍了如何根据您的工作负载选择最佳方案的相关信息。

您可以将 SQL Server 数据库重构为像 Amazon Aurora PostgreSQL 这样的开源数据库，以此来节省 Windows 和 SQL Server 的许可费用。像 Aurora 这样的云原生现代数据库将开源数据库的灵活性和低成本与商业数据库可靠的企业级功能相结合。如果您有不稳定的工作负载或者多租户工作负载，您也可以迁移到 [Aurora Serverless V2](#)。这能够将成本降低 90% 之多，具体取决于工作负载的特点。[此外，还 AWS 提供诸如适用于 Aurora PostgreSQL 的 Babelfish、\(\) 之类的工具以及诸如 AWS Schema Conversion Tool \(AWS SCT\) 之类的服务，用于简化 S QL Server 数据库的迁移和现代化。AWS Database Migration ServiceAWS DMS AWS](#)

数据库产品

从 Windows 上的 SQL Server 迁移到像 Amazon Aurora、Amazon RDS for MySQL 或 Amazon RDS for PostgreSQL 之类的开源数据库，可以在不影响性能或功能的情况下节省大量费用。请考虑以下事项：

- 从 Amazon EC2 上的 SQL Server 企业版切换到 Amazon RDS for PostgreSQL 或 Amazon RDS for MySQL 最多可以节省 80% 的费用。
- 从 Amazon EC2 上的 SQL Server 企业版切换到 Amazon Aurora PostgreSQL 兼容版或 Amazon Aurora MySQL 兼容版最多可以节省 70% 的费用。

对于传统的数据库工作负载，Amazon RDS for PostgreSQL 和 Amazon RDS for MySQL 能够满足需求，并为关系型数据库提供了一种经济高效的解决方案。Aurora 增加了众多以往仅由昂贵的商业供应商才能提供的可用性和性能方面的功能。Aurora 中的韧性功能需要额外付费。然而，与其他商业供应商提供的类似功能相比，Aurora 的韧性功能成本仍然低于商业软件针对同类功能所收取的费用。Aurora 架构经过了优化，与标准的 MySQL 和 PostgreSQL 部署相比，能够显著提升性能。

由于 Aurora 能够与开源的 PostgreSQL 和 MySQL 数据库兼容，所以还具有可移植性的额外优势。究竟最佳选项是 Amazon RDS for PostgreSQL、Amazon RDS for MySQL 还是 Aurora，这取决于对业务需求的了解以及将必要的功能与最佳选项进行匹配。

Amazon RDS 和 Aurora 的比较

下表总结 Amazon RDS 与 Amazon Aurora 之间的关键差异。

类别	Amazon RDS for PostgreSQL 或 Amazon RDS for MySQL	Aurora PostgreSQL 或 Aurora MySQL
性能	性能良好	3 倍或以上的性能

类别	Amazon RDS for PostgreSQL 或 Amazon RDS for MySQL	Aurora PostgreSQL 或 Aurora MySQL
失效转移	通常为 60–120 秒*	通常为 30 秒*
可扩展性	最多 5 个只读副本 延迟时间（以秒为单位）	最多 15 个只读副本 延迟时间（以毫秒为单位）
仓储服务	高达 64 TB	高达 128 TB
存储 HA	多可用区配置，其中包含一个或两个备用实例，每个备用实例都有数据库副本	默认情况下，在 3 个可用区中有 6 个数据副本
备份	每日快照和日志备份	持续、异步备份到 Amazon S3
Aurora 的创新	NA	100 GB 快速数据库克隆
	自动扩缩只读副本	
	查询计划管理	
	Aurora Serverless	
	跨区域副本与全局数据库	
	集群缓存管理**	
	并行查询	
	数据库活动流	

*大型事务会增加失效转移时间

**在 Aurora PostgreSQL 中可用

以下表格显示了本部分所涵盖的不同数据库服务的每月预估费用。

数据库服务	每月费用 (美元) *	AWS 定价计算器 (需要 AWS 账户)
Amazon RDS for SQL Server 企业版	3,750 美元	Estimate
Amazon RDS for SQL Server 标准版	2,318 美元	Estimate
Amazon EC2 上的 SQL Server 企业版	2,835 美元	Estimate
Amazon EC2 上的 SQL Server 标准版	1,345 美元	Estimate
Amazon RDS for PostgreSQL	742 美元	Estimate
Amazon RDS for MySQL	712 美元	Estimate
Aurora PostgreSQL	1,032 美元	Estimate
Aurora MySQL	1,031 美元	Estimate

*存储价格包含在实例定价中。费用以 us-east-1 区域为准。吞吐量和 IOPS 是假设值。这些计算针对的是 r6i.2xlarge 和 r6g.2xlarge 实例。

成本优化建议

异构数据库迁移通常需要将数据库的架构从源数据库引擎转换为目标数据库引擎，并将数据从源数据库迁移到目标数据库。迁移的第一步是评估并将 SQL Server 的架构和代码对象转换为目标数据库引擎。

您可以使用 [AWS Schema Conversion Tool \(AWS SCT \)](#) 来评估和评测该数据库是否与各种目标开源数据库选项兼容，例如 Amazon RDS for MySQL 或 Amazon RDS for PostgreSQL、Aurora MySQL 和 PostgreSQL。您也可以使用 Babelfish Compass 工具来评测与适用于 Aurora PostgreSQL 的 Babelfish 的兼容性。这使得 AWS SCT 和 Compass 成为了在决定迁移策略之前了解所涉及的前期工作的强大工具。如果您决定继续，AWS SCT 会自动对架构进行所需的更改。Babelfish Compass 背后的核心理念是让 SQL 数据库能够迁移到 Aurora，无需进行任何修改，或者只需进行极少的修改。Compass 将对现有的 SQL 数据库进行评估，以确定是否能够实现这一目标。这样一来，在将数据从 SQL Server 迁移到 Aurora 之前，结果就已经明确了。

AWS SCT 自动将数据库架构和代码转换和迁移到目标数据库引擎。您可以使用适用于 Aurora PostgreSQL 的 Babelfish 将数据库和应用程序从 SQL Server 迁移到 Aurora PostgreSQL，无需或只需进行少量的架构更改。这可以加快您的迁移速度。

架构迁移完成后，您可以使用迁 AWS DMS 移数据。AWS DMS 可以执行完整数据加载和复制更改，以便在最短的停机时间内执行迁移。

本部分将更详细地探讨以下工具：

- AWS Schema Conversion Tool
- 适用于 Aurora PostgreSQL 的 Babelfish
- Babelfish Compass
- AWS Database Migration Service

AWS Schema Conversion Tool

您可以使用 AWS SCT 评估现有的 SQL Server 数据库并评估与 Amazon RDS 或 Aurora 的兼容性。为了简化迁移过程，您还可以在异构数据库迁 AWS SCT 移中使用将架构从一个数据库引擎转换为另一个数据库引擎。您可以使用评估您的应用程序 AWS SCT，并转换编写 C#、C++、Java 和其他语言的应用程序的嵌入式应用程序代码。有关更多信息，请参阅 AWS SCT 文档中的[使用 AWS SCT 转换应用程序 SQL](#)。

AWS SCT 是一款支持许多数据库[源的](#)免费 AWS 工具。要使用 AWS SCT，请将其指向源数据库，然后进行评估。然后，[AWS SCT](#) 将评估架构并生成评测报告。评测报告包括一份执行摘要、复杂性及迁移所需的工作量、适用的目标数据库引擎以及转换建议。要下载 AWS SCT，请参阅 AWS SCT 文档 AWS SCT 中的[安装、验证和更新](#)。

下表显示了由 AWS SCT 生成的执行摘要示例，旨在说明将数据库转换至不同目标平台所涉及的复杂性。

目标平台	自动更改或最小更改			复杂操作		
	存储对象	代码对象	转换操作	存储对象		代码对象
Amazon RDS for MySQL	60 (98%)	8 (35%)	42	1 (2%)	1	15 (65%) 56

Amazon Aurora MySQL 兼容版	60 (98%)	8 (35%)	42	1 (2%)	1	15 (65%)	56
Amazon RDS for PostgreSQL	60 (98%)	12 (52%)	54	1 (2%)	1	11 (48%)	26
Amazon Aurora PostgreSQL L 兼容版	60 (98%)	12 (52%)	54	1 (2%)	1	11 (48%)	26
Amazon RDS for MariaDB	60 (98%)	7 (30%)	42	1 (2%)	1	16 (70%)	58
Amazon Redshift	61 (100%)	9 (39%)	124	0 (0%)	0	14 (61%)	25
AWS Glue	0 (0%)	17 (100%)	0	0 (0%)	0	0 (0%)	0
Babelfish	59 (97%)	10 (45%)	20	2 (3%)	2	12 (55%)	30

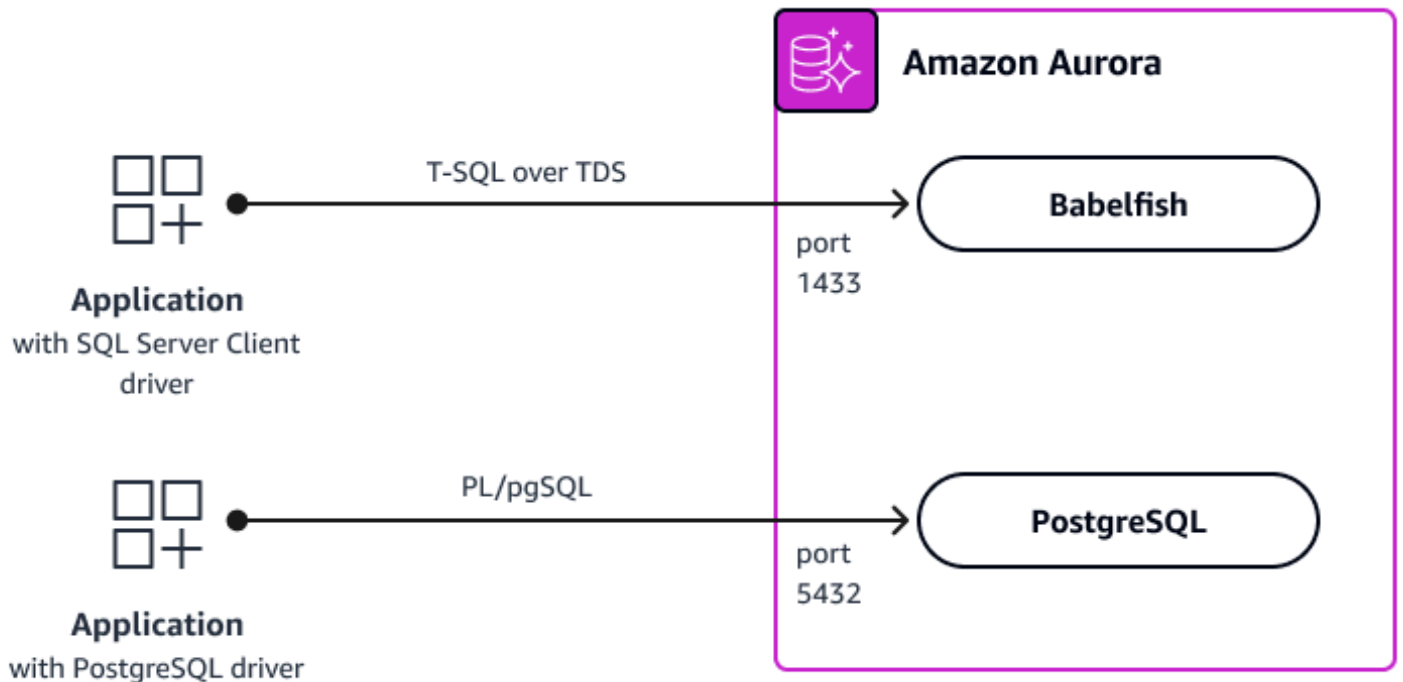
AWS SCT 报告还提供有关无法自动转换的架构元素的详细信息。您可以参考[AWS 迁移行动手册](#)来缩小 AWS SCT 转化差距并优化目标架构。有许多数据库迁移行动手册可以帮助进行异构迁移。

适用于 Aurora PostgreSQL 的 Babelfish

适用于 Aurora PostgreSQL 的 Babelfish 扩展了您的 Aurora PostgreSQL，以能够接受来自 SQL Server 客户端的数据库连接。使用 Babelfish，最初为 SQL Server 构建的应用程序可以直接与 Aurora PostgreSQL 协作，代码更改很少，而且无需更改数据库驱动程序。Babelfish 将 Aurora PostgreSQL 变成了双语，这样 Aurora PostgreSQL 就可以同时使用 T-SQL 和语言。PL/pgSQL Babelfish 最大程度减少了从 SQL Server 迁移到 Aurora PostgreSQL 的工作量。这能加快迁移进程，最大程度降低风

险，并大幅减少迁移成本。您可以继续在迁移后使用 T-SQL，但也可以[选择使用 PostgreSQL 原生工具](#)进行开发。

以下图表展示了使用 T-SQL 的应用程序如何连接到 SQL Server 中的默认端口 1433，并通过 Babelfish 翻译器与 Aurora PostgreSQL 数据库进行通信，而使用 PL/pgSQL 的应用程序则可以直接并同时通过 Aurora PostgreSQL 中的默认端口 5432 与 Aurora PostgreSQL 数据库进行连接。



Babelfish 不支持某些 SQL Server T-SQL 功能。因此，Amazon 提供了评估工具，用于 line-by-line 分析您的 SQL 语句并确定 Babelfish 是否不支持其中任何语句。

Babelfish 评估有两种选择。AWS SCT 可以评估你的 SQL Server 数据库与 Babelfish 的兼容性。另一个选项是 Babelfish Compass 工具，这是推荐的解决方案，因为 Compass 工具已根据新版本的适用于 Aurora PostgreSQL 的 Babelfish 进行了更新。

Babelfish Compass

[Babelfish Compass](#) 是一款可免费下载的工具，与最新版本的适用于 Aurora PostgreSQL 的 Babelfish 保持一致。相比之下，AWS SCT 将在一段时间后支持更新的 Babelfish 版本。[Babelfish Compass](#) 针对 SQL Server 数据库架构进行运行。您也可以使用 SQL Server Management Studio (SSMS) 等工具提取源 SQL Server 数据库架构。然后，您可以通过 Babelfish Compass 运行架构。这将生成报告，其中详细说明 SQL Server 架构与 Babelfish 的兼容性情况，以及在迁移之前是否需要进行任何更改。Babelfish Compass 工具还能自动完成其中的许多更改，从而最终加快您的迁移进程。

评测和更改完成后，您可以使用 SSMS 或 sqlcmd 等 SQL Server 原生工具将架构迁移到 Aurora PostgreSQL。有关说明，请参阅 AWS 数据库博客上的[使用 Babelfish 从 SQL Server 迁移到 Amazon Aurora](#) 一文。

AWS Database Migration Service

迁移架构后，您可以使用 AWS Database Migration Service (AWS DMS) 在停机时间最短的情况下将数据迁移到。AWS DMS 不仅可以加载完整数据，还可以在源系统启动并运行时将更改从源系统复制到目标。源数据库和目标数据库同步后，可以在应用程序指向完成迁移的目标数据库时进行直接转换活动。AWS DMS 目前仅使用 Babelfish 对 Aurora PostgreSQL 目标执行完整数据加载，不会复制更改。有关更多信息，请参阅文档 AWS Database Migration Service 中的“[使用 Babelfish 作为目标](#)”AWS DMS。

AWS DMS 可以进行同构（跨同一个数据库引擎）和异构（跨不同的数据库引擎）迁移。AWS DMS 支持许多源数据库和目标数据库引擎。有关更多信息，请参阅数据库博客中的[使用 AWS DMS 文章将您的 SQL Server 数据库迁移到 Amazon RDS for SQL Server](#)。AWS

其他资源

- [再见微软 SQL Server，你好 Babelfish](#)（AWS 新闻博客）
- [使用 AWS Schema Conversion Tool CLI（数据库博客）转换 AWS 数据库架构和应用程序 SQL](#)
- [使用最佳实践和现场 AWS 经验教训将 SQL Server 迁移到 Amazon Aurora PostgreSQL（数据库博客）](#)
- [在@@ 从微软 SQL Server 迁移到亚马逊 RDS for PostgreSQL 和亚马逊 Aurora PostgreSQL（数据库博客）AWS 后验证数据库对象](#)

优化 SQL Server 的存储

概述

本部分重点介绍针对 EC2 工作负载上的 SQL Server 的 Amazon Elastic Block Store（Amazon EBS）SSD 存储的成本优化。

您可以使用多种存储选项来部署和运行 SQL Server 工作负载 AWS。应根据用途、架构、耐久性、性能、容量和成本来选择合适的存储。AWS 运行 SQL Server 工作负载的客户通常会组合使用亚马逊 EBS、NVMe FSx、亚马逊和亚马逊简单存储服务 (Amazon S3) Service 存储。

Amazon EBS 是与 EC2 计算实例相连的网络附加存储，用于存储和处理操作系统、应用程序、数据库以及备份文件等。Amazon EBS 固态驱动器 (SSD) 存储包括：通用型 SSD (gp2 和 gp3) 和预调配 IOPS SSD (io1、io2 和 io2BX)。请考虑以下事项：

- 某些 EC2 实例 (例如 r5d) 在本地 NVMe SSDs 物理连接到主机实例。这些卷提供通常用于 SQL Server tempdb 或缓冲池扩展的块级存储。
- 适用于 Windows File Server 的 Amazon FSx 是一项完全托管的文件存储服务，而 NetApp 适用于 ONTAP 的 Amazon FSx 是 NetApp 基于广受欢迎的 ONTAP 文件系统构建的完全托管的共享存储。Amazon FSx 经常用于在高可用性的 SQL Server 失效转移群集实例 (FCI) 配置中运行 SQL Server 工作负载。该解决方案托管 SQL Server 数据和日志文件，从而降低 EC2 实例上的 EBS 性能要求。
- Amazon S3 是一项对象存储服务，可提供行业领先的可扩展性、数据可用性、安全性和性能。您可以在 Amazon S3 上存储 SQL Server 原生备份文件 AMLs、EBS 快照、应用程序日志等。

Amazon EBS 的 SSD 存储类型、性能和成本

Amazon EBS 的 SSD 存储成本通常会随着持久性和性能的提升而增加。存储目前有五种卷类型，每种都有[自己独特的性能指标](#)。有关支持 SSD 的卷的使用案例和特征的摘要，请参阅 Amazon EBS 文档的[固态驱动器 \(SSD \) 卷](#)部分的表格。

您可以使用 Amazon CloudWatch 监控固态硬盘性能，捕获趋势数据，并在达到特定阈值时设置警报。如果您正在运行 SQL Server 工作负载 AWS，请考虑启用[详细监控](#)并部署[CloudWatch 自定义指标](#)以捕获详细的卷性能指标，例如磁盘延迟、IOPS、吞吐量、磁盘队列长度、已用容量与可用容量等。您可以使用这些 CloudWatch 性能指标来识别配置不足和过度配置的存储，并提供历史数据点以准确定义存储需求。

Amazon EBS 的 SSD 存储成本也因分配的容量而异。下面的表格对不同的卷类型进行了比较。所有卷类型都有 1TB 的容量和类似的性能配置。

卷类型	最大 IOPS (16KiB I/O)	最大吞吐量 (128 KiB I/O)	每 TB 价格	节省成本百分比
gp2	3000	250	102.40 美元	
gp3	3000	250	86.92 美元	15%
io1	16000	500	1,168 美元	

卷类型	最大 IOPS (16KiB I/O)	最大吞吐量 (128 KiB I/O)	每 TB 价格	节省成本百分比
io2	16000	500	1,168 美元	
gp3	16000	500	146.92 美元	87%
io2bx	16000	4,000	1,168 美元	
gp3	16000	1000	181.92 美元	84%

Note

前面表格中的性能和成本指标基于 AWS 定价计算器中的[估算值](#)按每个卷进行计算。需要 AWS 账户 使用才能访问中的估算值 AWS 定价计算器。

Amazon EBS SSD gp3 卷以低成本提供卓越的性能。对于需要低于 16,000 IOPS 和 500 吞吐量的工作负载，如果您选择 gp3 卷而不是 io1 或 io2 卷，则最多可以节省 87%。MiBps

io2 Block Express (io2BX) 卷相比常规的 io2 卷具有更高的性能。在 16,000 IOPS 时，io1 或 io2 卷只能达到 500 个 MiBps 吞吐量，而 io2bx 卷最多可以配置为 4,000 个吞吐量。MiBps 与 io1 和 io2 卷相比，io2BX 卷能够以完全相同的价格提供四倍以上的吞吐量，介于 16000 至 64000 IOPS 之间。可以通过将常规的 io2 卷附加到 io2BX 支持的 EC2 实例，将其转换为 io2BX 卷。有关支持 io2BX 的 EC2 实例的列表，请参阅 Amazon EBS 文档中的[预调配 IOPS SSD 卷](#)。在部署新存储之前，您可以使用 [AWS 定价计算器](#) 来估算每月成本，并根据持久性、性能和容量之间的权衡来了解对成本的影响。

Amazon EBS 的通用 SSD 成本优化

我们建议您对所存储的内容进行评估，并确保您使用的是正确的存储类型和类别。例如，Amazon S3 提供了极具性价比的价格、内置的生命周期策略以及适合 SQL Server 备份的复制选项。SQL Server 2022 能够直接将数据备份到 Amazon S3，而之前版本的 SQL Server 则依赖于本机备份。如果您运行的是旧版本的 SQL Server，请考虑备份到 Amazon EBS HDD 卷，然后将备份复制到 Amazon S3。与使用 gp3 卷进行备份相比，此解决方案可以节省 53% 的费用。

下表显示了 Amazon EBS gp3、Amazon EBS HDD st1 和 Amazon S3 上 1TB 存储空间的价格差异。

存储类型	Capacity	价格 pm
EBS gp3 500 MiBps	1TB	96.92 美元
EBS st1 burst 500 MiBps		46.08 美元
S3 标准		23.55 美元
S3 Standard (不频繁访问)		12.80 美元
S3 Glacier Deep Archive		1.03 美元

Note

前面表格中的成本指标基于 AWS 定价计算器中的[估算值](#)进行计算。需要 AWS 账户 使用才能访问中的估算值 AWS 定价计算器。

我们建议您考虑以下几点：

- 启用详细监控并部署 CloudWatch 自定义指标，以准确捕获其存储性能需求。
- 将 Amazon EBS 存储从 gp2 升级到 gp3，以降低成本、增强灵活性并提升性能。
- 将 Amazon EBS 存储从 io1 升级到 io2，以提高持久性和性能灵活性。
- 尽可能使用 io2BX 而非 io1 或 io2，以提高持久性和性能。
- 在选择存储器时，请考虑一种有助于降低容量需求和高性能卷成本 mix-and-match 的方法。例如，您可以将低成本的 gp3 卷用于根卷（操作系统）、SQL Server 安装、系统数据库（不包括 tempdb）以及性能较低的用户数据库。这有助于降低 io2 卷的容量和成本，从而使其能够专门用于高性能用户数据库。
- 如果您在上托管 SQL Server 数据库 AWS，我们建议您在每个数据库中使用多个 SQL Server 数据文件。这允许将 read/write 工作负载分配到多个卷，从而降低每个卷的性能和容量需求，从而降低成本。
- 即便生产工作负载需要性能更高的存储，例如 io1 或 io2/io2BX 类型的存储，对于非生产工作负载而言，也应考虑使用 gp3 卷，以帮助降低成本。
- 跟踪并分析存储的使用情况随时间的变化趋势，以便轻松识别使用高峰和意外产生的费用。
- 将 [AWS Compute Optimizer](#) 用于根据实际使用情况对 EBS 卷进行纵向扩展或缩减的建议操作。

- 使用的弹性 AWS 来调整 Amazon EBS 固态硬盘卷的性能和容量需求。与本地环境不同的是，您无需为未来的工作负载过度预调配存储性能和容量。您可以将现有的 SQL Server 工作负载迁移到上 AWS 并根据需要调整性能或容量，同时保持数据库在线。

其他资源

- [Amazon EBS 卷类型](#) (Amazon EBS 文档)
- [Amazon Elastic Block Store \(Amazon EBS \)](#) (Amazon EBS 文档)
- [预调配 IOPS SSD 卷](#) (Amazon EBS 文档)
- [SSD 实例存储卷](#) (Amazon EC2 文档)
- [亚马逊 EBS 的亚马逊 CloudWatch 指标](#) (亚马逊 EBS 文档)
- [Amazon EC2 存储优化型实例规格](#) (Amazon EC2 文档)
- [使用 NetApp SnapCenter Amazon FSx for NetApp ONTAP 保护你的 SQL Server 工作负载](#) (AWS 存储博客)
- [亚马逊 EC2 常见问题解答](#) (AWS 产品页面)

使用 Compute Optimizer 优化 SQL Server 许可

有关如何使用优化 SQL Server 许可证的指南 AWS Compute Optimizer。

概述

[AWS Compute Optimizer](#) 可以推荐针对 Amazon Elastic Compute Cloud (Amazon EC2) 上的 Microsoft SQL Server 工作负载的许可优化机会。Compute Optimizer 可以提供自动化的建议，以降低许可费用。Compute Optimizer 提供的建议列在具有 Microsoft SQL Server 许可证的每个 EC2 实例旁边。提供的信息包括建议节省机会、EC2 实例按需价格和每小时自带许可 (BYOL) 价格。此信息将有助于您决定是否应降低您的许可证版本。

Compute Optimizer 会根据推断出的工作负载类型自动发现您在 Amazon EC2 上的 SQL Server 实例。要查看许可建议，您可以在 Compute Optimizer 中选择 SQL Server 实例，然后使用您的只读数据库凭证通过 [Amazon App CloudWatch lication Insigh ts](#) 进行身份验证。Compute Optimizer 会分析您是否在使用任何 SQL Server 企业版功能。如果未使用企业版的任何功能，Compute Optimizer 建议您降级到标准版以降低许可成本。

您还可以使用 Compute Optimizer 为运行 SQL Server 工作负载的 Amazon EC2 实例提出调整大小建议。有关更多信息，请参阅本指南中的[使用 Compute Optimizer 优化 SQL Server 的大小](#)。

成本优化建议

Compute Optimizer 中的许可证建议能够帮助您评估在 Microsoft SQL Server 中所使用的各项功能，并为您的工作负载选择最经济高效的版本。SQL Server 企业版的价格要比标准版贵得多。有关更多信息，请参阅本指南中的[比较 SQL Server 版本](#)和 Microsoft 网站上的[SQL Server 2022 定价](#)。花些时间来配置 Compute Optimizer，以评估您的 SQL Server 实例集并提供相关建议，这样可以显著降低您的许可费用。

在许可证详情页面上，提供以下信息：

- 请使用表格将您当前的许可证设置（例如版本、模型以及实例核心数量）与 Compute Optimizer 的建议进行对比。
- 使用利用率图表查看分析期间使用的企业版功能数量。

有关更多信息，请参阅 Compute Optimizer 文档中的[查看商业软件许可证建议的详细信息](#)。

配置 Compute Optimizer

Compute Optimizer 使用 `mssql_enterprise_features_used` 指标分析商业软件许可证。有关该指标的更多信息，请参阅[商业软件许可证的指标](#)。

1. 确保您拥有选择加入 Compute Optimizer 的适当权限。有关更多信息，请参阅下列内容：
 - [选择加入 Compute Optimizer 的政策](#)
 - [为独立 AWS 账户授予访问 Compute Optimizer 的权限的策略](#)
 - [授予组织管理账户访问 Compute Optimizer 访问权限的策略](#)
2. 附加 App CloudWatch Location Insights 所需的实例角色和策略。有关说明，请参阅[启用商业软件许可证建议的策略](#)。
3. 使用你的 Microsoft SQL Server 数据库凭据启用 CloudWatch 应用程序见解。有关说明，请参阅 CloudWatch 文档中的[设置应用程序以进行监控](#)。

Note

要生成商用软件许可证推荐，至少需要连续 30 小时的 CloudWatch 指标数据。有关更多信息，请参阅[CloudWatch 指标要求](#)。

4. 使用以下 SQL 查询配置 App CloudWatch Location Insights 的最低权限访问权限。

```
GRANT VIEW SERVER STATE TO [LOGIN];  
GRANT VIEW ANY DEFINITION TO [LOGIN];
```

这将启用一项新服务 PrometheusSqlExporter SQL。

5. 从目标 AWS 账户 或组织管理账户中，选择加入 Compute Optimizer。有关说明，请参阅[选择加入您的账户](#)。

 Note

选择加入后，最长可能需要 24 小时才能生成结果和优化建议。

6. 在 [Compute Optimizer 控制台](#) 中，选择导航窗格中的许可证。
7. 在调查发现列中，搜索具有指标不足调查发现的任何实例。如果 Compute Optimizer 检测到 CloudWatch 应用程序见解未启用或权限不足，则会返回此结果。有关更多信息，请参阅[调查发现原因](#)。请按照以下步骤来解决这些调查发现：
 - a. 选择实例。
 - b. 添加机密密钥。
 - c. 确认已附加实例角色和策略。
 - d. 选择启用许可证建议。
8. 在调查发现列中，搜索调查发现为未优化的任何实例。Compute Optimizer 在检测到 Amazon EC2 基础设施没有使用您购买的任何 Microsoft SQL Server 许可证功能时返回此调查发现。有关更多信息，请参阅[调查发现原因](#)。请按照以下步骤来解决这些调查发现：
 - a. 选择实例。
 - b. 将当前许可证版本与建议版本进行比较。
 - c. 查看当前的许可证使用图表。
 - d. 如果要降级许可证，请选择实施建议。
 - e. 查看要求并按照说明降级许可证。如果要自动化该过程，请参阅[使用 AWS Systems Manager 文档降级 SQL Server 企业版以降低成本](#)（AWS 博客）。

其他资源

- 通过 AWS Compute Optimizer（AWS 博客）[降低微软 SQL Server 许可成本](#)
- [什么是 AWS Compute Optimizer？](#)（AWS 文档）

- [查看商业软件许可证建议](#) (AWS 文档)
- [降级您的 Microsoft SQL Server 版本](#) (AWS 文档)
- [AWS上的 Microsoft SQL Server](#) (AWS)
- [AWS上的 Microsoft 许可](#) (AWS)
- [Microsoft SQL Server 2019 定价](#) (Microsoft)
- [Microsoft SQL Server 2022 定价](#) (Microsoft)

使用 Compute Optimizer 优化 SQL Server 的大小

概述

[AWS Compute Optimizer](#) 帮助数据库管理员 (DBA) 在 Amazon Elastic Compute Cloud (Amazon EC2) 上发现 Microsoft SQL Server 工作负载并适当调整 EC2 实例的大小，从而将许可证费用最多降低 25%。Compute Optimizer 中的[推断工作负载类型](#)功能使用机器学习 (ML)，并自动检测可能在您的资源上 AWS 运行的应用程序。Compute Optimizer 支持将 SQL Server 作为一种推断的工作负载类型。通过使用推断的工作负载类型功能，您可以根据您在 Amazon EC2 实例上运行的特定工作负载，精准找出节省成本的途径。

使用此功能，您可以按支持的推断工作负载类型 (例如 SQL Server) 对节省成本的机会进行分类。Compute Optimizer 可以自动发现过度预调配的 SQL Server EC2 实例。您可以切换到 EC2 控制台来缩小实例大小，这有助于降低许可和基础设施成本。

您也可以使用 Compute Optimizer 提出 SQL Server 许可建议。有关更多信息，请参阅本指南中的[使用 Compute Optimizer 优化 SQL Server 许可](#)。

配置 Compute Optimizer

有关将 Compute Optimizer 用于 SQL Server 推断的工作负载的说明，请参阅[优化性能和降低许可成本：利用 Amazon AWS Compute Optimizer EC2 SQL Server 实例](#) (AWS 博客)。您可以选择加入独立账户、属于某个组织成员的账户和组织的管理账户。对于独立账户和成员账户，选择加入仅会为该账户启用 Compute Optimizer。对于组织管理账户，您可以选择是仅在该账户中启用 Compute Optimizer，还是为该组织的所有成员账户启用 Compute Optimizer。

Compute Optimizer 选择加入流程会自动创建一个 AWS Identity and Access Management (IAM) 服务相关角色。有关更多信息，请参阅[为 AWS Compute Optimizer 使用服务相关角色](#)。

Compute Optimizer 根据亚马逊 CloudWatch 指标 (例如 CPU、I/O、网络和亚马逊弹性区块存储 (Amazon EBS) 的使用情况) 分析您的资源。要生成推荐，需要在过去 14 天内至少连续 30 小时的

CloudWatch 指标数据。如果您启用增强型基础设施指标功能，则该利用率指标将延长至 93 天。有关更多信息，请参阅 Compute Optimizer [文档中的 CloudWatch 指标要求和增强型基础架构指标](#)。

Compute Optimizer 根据 vCPU、内存、存储、网络、风险和迁移工作量提供选项以及与每个选项相关的节省金额。您可以使用 CloudWatch 指标仪表板来分析用于提出建议的数据。利用这些数据，您可以调整运行 SQL Server 工作负载的 EC2 实例的大小。有关如何更改实例类型的更多信息，请参阅 Amazon EC2 文档中的[更改实例类型](#)。

其他资源

- [AWS Compute Optimizer 识别和筛选微软 SQL Server 工作负载 \(AWS\)](#)
- [优化性能和降低许可成本：利用 AWS Compute Optimizer Amazon EC2 SQL Server 实例 \(AWS 博客\)](#)
- [什么是 AWS Compute Optimizer ? \(AWS 文档\)](#)
- [查看 EC2 实例建议 \(AWS 文档\)](#)

查看 SQL Server 工作负载 Trusted Advisor 的建议

概述

[AWS Trusted Advisor](#) 提供了一些建议，可帮助您遵循 AWS 最佳实践。通过分析您的使用情况、配置和支出，Trusted Advisor 提供切实可行的建议，以降低成本、提高系统可用性和性能，或帮助填补安全漏洞。本节重点介绍可帮助您降低在中操作 SQL Server 工作负载的成本的 Trusted Advisor 检查 AWS 云。

成本优化建议

Trusted Advisor 提供了一些建议，可帮助您在亚马逊弹性计算云 (Amazon EC2) 上优化 SQL Server 工作负载。这些检查可用于检查您的 SQL Server 工作负载并自动列出需要优化的实例。实施 Trusted Advisor 建议可以降低成本并改善组织的安全状况。

以下是以微软 SQL Server 为重点的 Trusted Advisor 检查：

- [为 Microsoft SQL Server 过度预调配的 Amazon EC2 实例](#) – 此检查会分析运行 SQL Server 的 Amazon EC2 实例，并在实例超出 SQL Server 软件 vCPU 限制时提醒您。例如，采用 SQL Server 标准版的实例最多可以使用 48 v CPUs。带有 SQL Server Web 的实例最多可以使用 32 v CPUs。

Edition	最少 vCPU	最多 vCPU
Web	4	32
标准	4	48
Enterprise	4	操作系统限制

- [Microsoft SQL Server 的 Amazon EC2 实例整合](#) – 此检查会分析您的 Amazon EC2 实例，并在实例的 SQL Server 许可证数量少于最低数量时提醒您。您可以整合较小的 SQL Server 实例以帮助降低成本。如果您有许多包含许可证的小型 SQL Server 实例，请考虑整合。根据 [Microsoft SQL Server 2019 许可指南](#)，SQL Server 的每个实例至少需要 4 个 vCPU 许可证。如果您整合这些数据库，则可以节省许可成本。您可以根据实例上的数据库数量、最大数据库大小和数据库总大小来做出决定。SQL Server 的 Web 版、标准版和企业版均支持整合。有关更多信息，请参阅[整合 SQL Server 数据库](#)（Microsoft 博客文章）。

AWS 不建议将大型生产数据库只放在一台服务器上。但是，您可以整合那些用于非生产环境（如开发、测试和暂存环境）的较小数据库。这取决于您当前的 SQL Server 使用情况；如果您的数据库使用量较低，则可以整合到一台服务器上。

配置 Trusted Advisor

执行以下操作来评估以 SQL Server 为重点的签入 Trusted Advisor。

1. 登录到 AWS 管理控制台。
2. 打开 [AWS Trusted Advisor 控制台](#)。
3. 在导航窗格中的建议下，选择成本优化。
4. 在成本优化检查清单中，查看 Microsoft SQL Server 的 Amazon EC2 实例整合和为 Microsoft SQL Server 过度预调配的 Amazon EC2 实例检查的状态。
 - 绿色勾号表示您的 Amazon EC2 实例已实现最佳配置。
 - 橙色提醒符号表示有改进的机会。
5. 选择一项检查以查看其详细信息和建议。
6. 按照检查提供的说明进行操作，以优化运行 SQL Server 工作负载的 Amazon EC2 实例。
7. 定期监控您的实例，并定期刷新检查。

其他资源

- [Trusted Advisor 查看参考资料](#) (AWS 文档)
- [AWS上的 Microsoft SQL Server](#) (AWS)
- [AWS上的 Microsoft 许可](#) (AWS)
- [SQL Server 2019 定价](#) (Microsoft)
- [AWS Launch Wizard 适用于 SQL 服务器](#) (AWS 文档)

容器

现代化是一个转型之旅，提供了许多选择，包括将单体分解为微服务、使用无服务器函数 (AWS Lambda) 将应用重新架构为事件驱动型，以及将数据库从 SQL Server 迁移到 Amazon Aurora 或专用托管式数据库。与其他现代化选项相比，将 .NET Framework 应用程序的平台更换为 Linux 和 Windows 容器的现代化途径所需的工作量更少。容器具有以下优点：

- 加快创新：迁移到容器可以更轻松地实现开发生命周期各个阶段的自动化，包括构建、测试及部署应用程序。通过实现这些流程的自动化，开发和运营团队可以有更多时间专注于创新。
- 降低总拥有成本 (TCO)：迁移到容器还可以减少对许可证管理和端点保护工具的依赖。由于容器是临时计算单元，因此您可以自动化和简化诸如修补、扩展以及备份和恢复之类的管理任务。这降低了管理和操作基于容器的工作负载的 TCO。最后，与虚拟机相比，容器效率更高，因为您可以使用容器通过提供更好的隔离来最大限度地部署应用程序。这可以提高应用程序基础设施资源的利用率。
- 提高资源利用率：与虚拟机相比，容器效率更高，因为您可以使用容器最大限度地部署应用程序。这通过提供更好的隔离性，来提高应用程序基础设施资源的利用率。
- 缩小技能差距 — AWS 提供沉浸式学习日，提高开发团队在容器技术和 DevOps 实践方面的技能。

本节将介绍以下主题：

- [将 Windows 应用程序移到容器](#)
- [优化 Amazon ECS 上 AWS Fargate 任务的成本](#)
- [了解您的 Amazon EKS 成本](#)
- [使用 App2Container 更换 Windows 应用程序平台](#)

有关许可信息，请参阅 [Amazon Web Services and Microsoft: Frequently Asked Questions](#) 的 Licensing 部分，或通过电子邮件将您的问题发送至 microsoft@amazon.com。

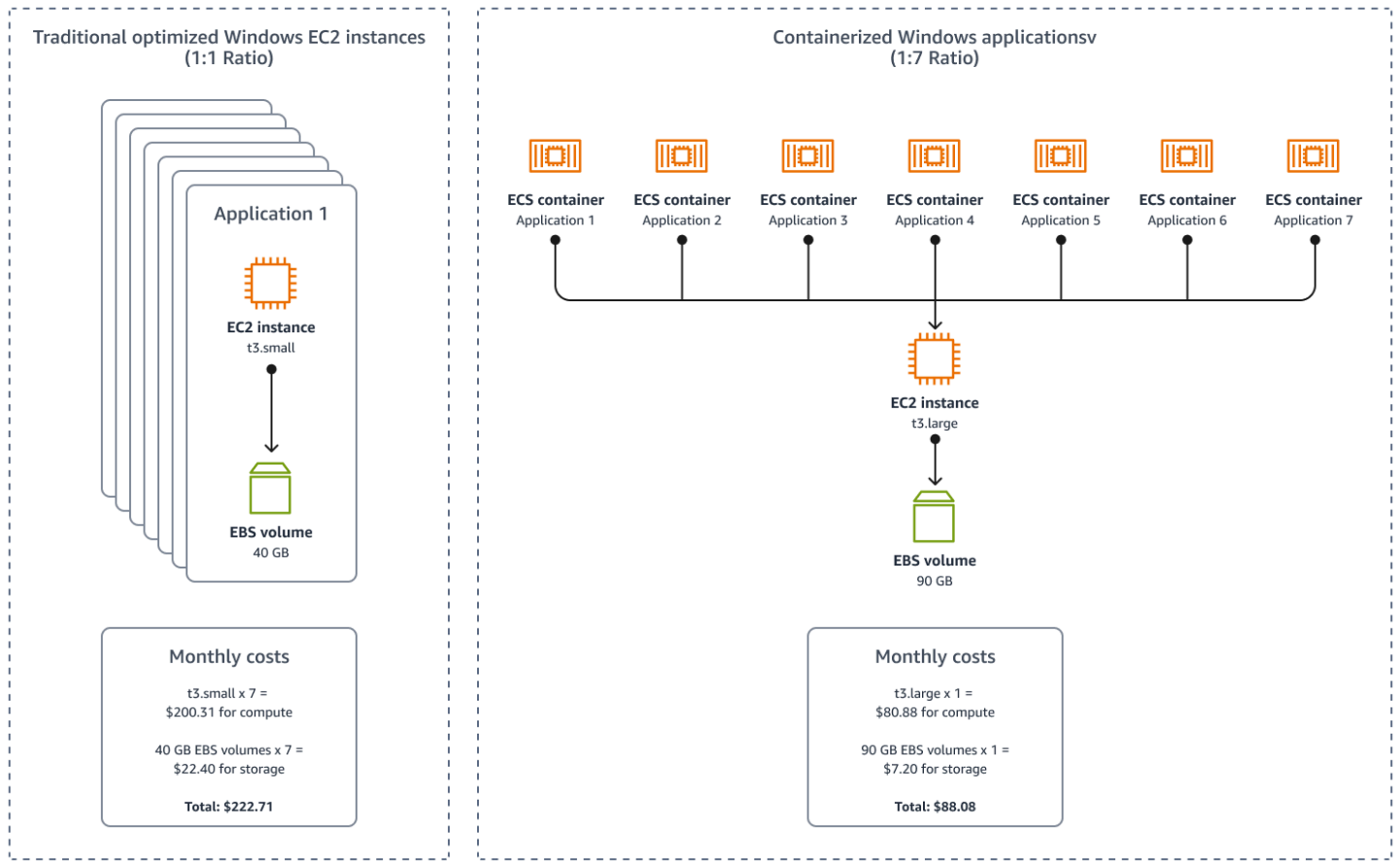
将 Windows 应用程序移到容器

概述

根据 [CNCF Annual Survey 2021](#)，96% 的组织正在使用或评估容器来实现基础设施的现代化。这是因为容器可以帮助您的组织降低风险，提高运营效率和速度，并实现敏捷性。您还可以使用容器来降低应用程序的运行成本。本节为跨 AWS 容器服务 (包括 [亚马逊弹性容器服务 \(Amazon ECS\)](#)、[Amazon Elastic Kubernetes Service \(Amazon EKS \)](#)) 和 [AWS Fargate](#)

成本优势

以下信息图显示了企业根据[优化和许可评估AWS \(OLA\) 建议](#)，将其 [ASP.NET Framework 应用程序整合到亚马逊弹性计算云 \(Amazon EC2\)AWS 实例](#)上可以节省的成本。以下信息图显示了通过将应用程序移到 Windows 容器可以节省的额外费用。



AWS OLA 建议企业进行提升，转向单个 t3.small 实例。如以下性能利用率分析所示，该企业可以通过在本地服务器上运行七个 ASP.NET 应用程序来实现这些节省。

			Not sharable		Nearing end of life		Approximately 15% overhead		The OS accounts for 1.2 GB per instance		Optimized for EC2	
Server name	Storage	Operating system	On-premises CPU AVG utilization	On-premises CPU peak utilization	On-premises RAM (GB)	On-premises RAM AVG utilization (GB)	On-premises RAM peak utilization (GB)	Instance size	vCPU	RAM (GB)		
1 AppServer01	60	Windows Server 2012	7.00%	17.00%	8	13.50%	17.10%	t3.small	2	2		
2 AppServer02	39	Windows Server 2012	20.07%	22.00%	16	7.50%	12.40%	t3.small	2	2		
3 AppServer03	39	Windows Server 2012	24.00%	25.50%	16	8.80%	11.90%	t3.small	2	2		
4 AppServer04	4	Windows Server 2012	21.40%	24.00%	16	7.80%	10.70%	t3.small	2	2		
5 AppServer05	40	Windows Server 2012	21.30%	23.00%	16	8.20%	12.00%	t3.small	2	2		
6 AppServer06	39	Windows Server 2012	21.50%	23.50%	16	7.90%	10.90%	t3.small	2	2		
7 AppServer07	39	Windows Server 2012	21.60%	22.90%	16	8.40%	11.50%	t3.small	2	2		

进一步分析表明，通过在容器上运行工作负载，该企业可以节省更多成本。容器可减少操作系统对系统资源（如 CPU、RAM 和磁盘使用量）的开销（在下一节中介绍）。在这种情况下，企业可以将所有七个应用程序整合到一个 t3.large 实例，但仍有 3 GB 的备用 RAM。通过使用容器代替 Amazon EC2，迁移到容器可以帮助企业在计算和存储方面平均节省 64% 的成本。

成本优化建议

以下部分提供了通过整合应用程序和使用容器来优化成本的建议。

减少 Windows on Amazon EC2 占用空间

Windows 容器允许您将更多的应用程序整合到更少的 EC2 实例上，从而减少您的 Windows on Amazon EC2 占用空间。例如，假定您有 500 个 ASP.NET 应用程序。如果每个 Windows on Amazon EC2 应用程序运行一个核心，则等于 500 个 Windows 实例 (t3.small)。如果假设使用 Windows 容器（采用 t3.large）的比例为 1:7（根据 EC2 实例类型/大小，该比例可能会大幅增加），那么您只需要大约 71 个 Windows 实例。这意味着您的 Windows on Amazon EC2 占用空间减少了 85.8%。

降低 Windows 许可成本

如果您许可 Windows 实例，则无需许可在该实例上运行的容器。因此，使用 Windows 容器整合 ASP.NET 应用程序可以显著降低 Windows 许可成本。

减少存储占用空间

每次启动新的 EC2 实例时，都需要创建一个新的 Amazon Elastic Block Store（Amazon EBS）卷并支付相应的费用来存放操作系统。随着规模的扩展，成本也会随之增加。如果您使用容器，则可以降低存储成本，因为所有容器都共享相同的基本操作系统。此外，容器使用层的概念，将容器映像的不可变部分重复用于所有基于该映像的正在运行的容器。在上述示例方案中，所有容器都运行 .NET Framework，因此所有容器都共享中间且不可变的 ASP.NET 框架层。

将 end-of-support 服务器迁移到容器

对 Windows Server 2012 和 Windows Server 2012 R2 的支持已于 2023 年 10 月 10 日结束。您可以迁移在 Windows Server 2012 或之前版本上运行的应用程序，方法是将其容器化为在新操作系统上运行。这样，您就可以避免在不合规的操作系统上运行应用程序，同时利用容器提供的成本效益、更低的风险、运营效率、速度及敏捷性。

使用这种方法需要考虑的一个警告是，您的应用程序是否需要与当前使用的操作系统版本 APIs 相关的特定内容（例如 COM Interop）。在这种情况下，您必须测试将应用程序迁移到较新的 Windows 版

本。Windows 容器会将基本容器映像（例如 Windows Server 2019）与容器主机的操作系统（例如 Windows Server 2019）保持一致。通过更改 Dockerfile 内的基础映像并部署到一组运行最新版本的 Windows 的新主机，测试和迁移到容器可以让将来更轻松地升级操作系统。

删除第三方管理工具和许可证

管理服务器机群需要使用多种第三方系统操作工具进行修补和配置管理。这些都会使基础设施管理变得复杂，而且您通常还会产生第三方许可成本。如果您在上使用容器 AWS，则无需在操作系统方面管理任何内容。容器运行时管理容器。这意味着底层主机是临时的，可以很容易地被替换。无需直接管理容器主机即可运行容器。此外，您还可以使用免费工具 AWS Systems Manager Session Manager，例如轻松访问主机和解决问题。

改进控制和可移植性

与对 EC2 实例相比，容器可让您更精细地控制服务器资源（如 CPU 和 RAM）。对于 EC2 实例，您可以通过选择实例系列、实例类型和 [CPU 选项](#) 来控制 CPU 和 RAM。但是，使用容器，您可以精确定义要分配给 ECS 任务定义中的容器或分配给 [Amazon EKS 中的容器组 \(pod\)](#) 的 CPU 或 RAM 数量。实际上，我们建议为 Windows 容器 [指定容器级 CPU 和内存](#)。此粒度级别带来了成本效益。请考虑以下示例代码：

```
json
{
  "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/demo-
service:1",
  "containerDefinitions": [
    {
      "name": "demo-service",
      "image": "mcr.microsoft.com/dotnet/framework/samples:aspnetapp-
windowsservercore-ltsc2019",
      "cpu": 512,
      "memory": 512,
      "links": [],
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 0,
          "protocol": "tcp"
        }
      ],
    }
  ],
}
```

加速创新

迁移到容器可以更轻松地实现开发生命周期各个阶段的自动化，包括构建、测试及部署应用程序。如果将这些流程自动化，就能让开发和运营团队有更多时间专注于创新。

降低 TCO

迁移到容器通常可以减少对许可证管理和端点保护工具的依赖。由于容器是临时计算单元，因此您可以自动化和简化诸如修补、扩展以及备份和恢复之类的管理任务。这可以降低管理和操作基于容器的工作负载的 TCO。与虚拟机相比，容器效率更高，因为它们可以最大限度地部署应用程序，从而提高应用程序基础设施资源的利用率。

缩小技能差距

AWS 提供项目和沉浸式学习日，以提高客户开发团队在容器和技术 DevOps 方面的技能。其中包括动手实践咨询和支持。

重构到 .NET 5+ 并使用 Linux 容器

虽然您可以通过将 .NET Framework 应用程序迁移到容器来降低成本，但如果在 AWS 上将遗留 .NET 应用程序重构为云原生替代方案，则可以进一步节省成本。

消除许可成本

将您的应用程序从 Windows 上的 .NET Framework 重构为 Linux 上的 .NET Core 可节省大约 45% 的成本。

访问最新的增强功能

将应用程序从 Windows 上的 .NET Framework 重构为 Linux 上的 .NET Core 可让您访问最新的增强功能，例如 Graviton2。Graviton2 的性能比同类实例性价比高出 40%。

提高了安全性和性能

将应用程序从 Windows 上的 .NET Framework 重构为 Linux 容器上的 .NET Core 可提高安全性和性能。这是因为您可以获得最新的安全补丁，受益于容器隔离，且可以访问新功能。

使用 Windows 容器，而不是在一个 IIS 的实例上运行多个应用程序

请考虑使用 Windows 容器而不是使用互联网信息服务 (IIS) 在一个 EC2 Windows 实例上运行多个应用程序具有以下优点：

- 安全性：容器提供了开箱即用的安全级别，这是通过 IIS 级别的隔离无法实现的。如果一个 IIS 网站或应用程序遭入侵，则所有其他托管式网站都将暴露且易受攻击。容器逃逸的情况很少见，且比通过 Web 漏洞控制服务器更难被利用。
- 灵活性：能够在进程隔离的情况下运行容器并拥有自己的实例，从而可以提供更精细的联网选项。容器还提供跨许多 EC2 实例的复杂分发方法。将应用程序整合到单个 IIS 实例上时，您将无法获得这些好处。
- 管理开销：服务器名称指示 (SNI) 会产生需要管理和自动化的开销。此外，您还必须努力处理典型的操作系统管理操作，例如修补、BSOD 问题排查 (如果没有自动扩缩)、端点保护等。根据[安全最佳实践](#)配置 IIS 站点是一项耗时且持续的工作。您甚至可能需要设置[信任级别](#)，这也会增加管理开销。容器的设计目标是无状态且不可变。最终，如果您改用 Windows 容器，您的部署将更快、更安全、更可重复。

后续步骤

投资现代基础架构来运行您的传统工作负载可为您的组织带来巨大好处。AWS 容器服务使您可以更轻松的管理您的底层基础架构，无论是在本地还是在云中，因此您可以专注于创新和业务需求。AWS 如今，云端所有容器中有将近80%在运行。AWS 为几乎所有用例提供了一组丰富的容器服务。要开始使用，请参阅 [AWS上的容器](#)。

其他资源

- [使用 ECS 容量提供商和 EC2 竞价型实例优化容器工作负载的成本](#) (AWS 博客)
- [Amazon ECS 和 AWS Fargate的成本优化核对清单](#) (AWS 博客)
- [AWS Graviton2 上的 Amazon EKS 现已上市：多架构应用程序的注意事项](#) (博客) AWS
- [Kubernetes 的成本优化](#) (博客) AWS
- 通过 [Karpenter 整合优化你的 Kubernetes 计算成本](#) (博客) AWS

优化 Amazon ECS 上 AWS Fargate 任务的成本

概述

正确调整 AWS Fargate 任务规模是成本优化的重要一步。通常，应用程序是针对 Fargate 任务使用任意大小构建的，且永远不会重访。这可能会导致 Fargate 任务过度预调配和产生不必要的支出。本节向您展示如何使用 [AWS Compute Optimizer](#) 提供可操作的建议，以便您可以针对 Fargate 上运行的 Amazon Elastic Container Service (Amazon ECS) 服务优化任务 CPU 和内存。Compute Optimizer

还量化了采用这些建议所产生的成本影响。这使您能够根据节省机会的大小确定优化工作的优先顺序。Compute Optimizer 建议提供容器级 CPU 和内存配置，以缩小任务规模。

成本优势

在 Fargate 上调整 Amazon ECS 任务可能会降低长时间运行任务的成本 30-70%。无需查看应用程序性能指标来调整任务规模，您就可以将在 EC2 计算实例上使用的相同思维方式应用于容器大小调整。这会导致 Fargate 任务过大，从而增加闲置资源的成本。您可以使用 Compute Optimizer 来快速发现调整机会。理想情况下，应用程序所有者会审查特定的应用程序性能指标，并消除操作系统开销，以确保指定正确的任务大小。有关更多信息，请参阅本指南的[将 Windows 应用程序移到容器](#)部分。

成本优化建议

本节提供了有关使用 Compute Optimizer 调整 Fargate 任务上的 Amazon ECS 的建议。

作为成本优化过程的一部分，建议执行以下操作：

- 启用 Compute Optimizer
- 使用 Compute Optimizer 结果
- 标记要调整的任务
- 启用成本分配标签以使用 AWS 计费工具
- 实施调整建议
- 在 Cost Explorer 成本管理服务中查看成本前后

启用 Compute Optimizer

您可以在 AWS Organizations 中，在组织级别或单个账户级别启用 [AWS Compute Optimizer](#)。组织范围的配置为所有成员账户的整个实例集中的新实例和现有实例提供持续报告。这样一来，正确调整大小就可以成为重复的活动，而不是一项 point-in-time 活动。

组织级别

对于大多数组织来说，使用 Compute Optimizer 的最有效方式是在组织层面。这为您的组织提供了多账户和多区域可见性，并将数据集中到一个来源供审查。要在组织层面启用此功能，请执行以下操作：

1. 使用具有[所需权限](#)的角色登录您的 [AWS Organizations 管理账户](#)，然后选择加入该组织内的所有账户。您的组织必须[已启用所有功能](#)。

2. 启用管理账户后，您可以登录该账户，查看所有其他成员账户，并浏览他们的推荐。

Note

最佳做法是为 Compute Optimizer 配置[委托管理员账户](#)。这使您能够行使最低权限原则，最大限度地减少对 AWS Organizations 管理账户的访问权限，同时仍提供对组织范围服务的访问权限。

单个账户级别

如果您的目标账户成本高昂，但又没有 AWS Organizations 的访问权限，您仍然可以为该账户和区域启用 Compute Optimizer。要了解选择加入流程，请参阅[入门](#)。AWS Compute Optimizer

Note

建议每天刷新，最多可能需要 12 小时才能生成。请记住，Compute Optimizer 需要过去 14 天内 24 小时的指标才能为 Fargate 上的 Amazon ECS 生成建议。有关更多信息，请参阅 Compute Optimizer 文档中的[Requirements for Amazon ECS services on Fargate](#)。

Compute Optimizer 会自动分析 Fargate 上 CloudWatch 亚马逊 ECS 服务的以下亚马逊和亚马逊 ECS 利用率指标：

- CPUUtilization：服务中使用的 CPU 容量的百分比。
- MemoryUtilization：服务中使用的内存的百分比。

使用 Compute Optimizer 结果

举一个例子，该示例侧重于在单个账户和单个区域内进行调整更改。在本示例中，Compute Optimizer 在组织级别为所有账户启用。请记住，调整属于破坏性过程，在大多数情况下，应用程序所有者会在几周的计划维护时段内精确地执行此过程。

如果您从组织的管理账户中导航到 Compute Optimizer（如以下步骤所示），则可以选择要调查的账户。在此示例中，一个任务正在 us-east-1 中过度预调配的单个账户中运行。重点是将大小调整为 Amazon ECS 服务的建议大小。

1. 打开 [Compute Optimizer 控制台](#)。

2. 在控制面板页面上，按调查发现=过度预调配筛选，以查看 Fargate 上的所有 Amazon ECS 服务。
3. 要查看 Fargate 上过度预调配的 ECS 服务的详细建议，请向下滚动，然后选择查看建议。
4. 选择导出，并保存文件以备将来使用。

Note

要保存建议以备将来查看，您必须有一个可用的 S3 存储桶供每个区域的 Compute Optimizer 写入数据。有关更多信息，请参阅 Compute Optimizer 文档中的 [Amazon S3 bucket policy for AWS Compute Optimizer](#)。

要查看 Compute Optimizer 提供的建议，请执行以下操作：

1. 在 [Compute Optimizer 控制台](#) 中，前往导出建议页面。
2. 对于 S3 存储桶目标，选择 S3 存储桶。
3. 在导出筛选条件部分的资源类型中，选择 Fargate 上的 ECS 服务。
4. 在 Fargate 上的 ECS 服务建议页面上，深入研究 Fargate 上的一项 ECS 服务，并查看 Compute Optimizer 提供的 CPU 和内存建议。例如，查看将当前设置与建议的任务大小进行比较和将当前设置与建议的容器大小进行比较部分中的建议。

要获取需要调整的 Fargate 的 ECS 服务列表，请执行以下操作：

1. 打开 [Amazon S3 控制台](#)。
2. 在导航窗格中，选择存储桶，然后选择在其中导出结果的存储桶。
3. 在对象选项卡上，选择您的对象并选择下载。
4. 在您下载的结果中，筛选结果列，以仅显示 Fargate 上的 OVER_PROVISIONED Amazon ECS 服务。这显示了您计划进行调整的 Amazon ECS 服务。
5. 将任务定义存储在文本编辑器中，以备后续使用。

调整标签任务

标记工作负载是组织 AWS 资源的强大工具。您可以使用标签来更精细地了解成本并启用退款。有许多方法和策略可以向 AWS 资源添加标签以处理退款和自动化。有关更多信息，请参阅 AWS 白皮书《为资源[添加标签 AWS 的最佳实践](#)》。以下示例使用 [AWS CloudShell](#) 标记目标账户和 AWS 区域内任何 Amazon ECS 服务的所有任务。

```
#!/bin/bash
# Set variables
TAG_KEY="rightsizing"
TAG_VALUE="enabled"
# Get a list of ECS Clusters
ClustersArns=$( aws ecs list-clusters -query 'clusterArns' -output text)
for ClustersArn in $ClustersArns; do
    ServiceArns=$( aws ecs list-services -cluster $ClustersArn -query 'serviceArns' -
output text)
    for ServiceArn in $ServiceArns; do
        TasksArns=$( aws ecs list-tasks -cluster $ClustersArn -service-name $ServiceArn -
query 'taskArns' -output text)
        for TasksArn in $TasksArns; do
            aws ecs tag-resource -resource-arn $TasksArn -tags key=$TAG_KEY,value=$TAG_VALUE
        done
    done
done
```

以下代码示例演示了如何为所有 Amazon ECS 服务启用[标签传播](#)。

```
#!/bin/bash
# Set variables
TAG_KEY="rightsizing"
TAG_VALUE="enabled"
# Get a list of ECS Clusters
ClustersArns=$(aws ecs list-clusters --query 'clusterArns' --output text)
for ClustersArn in $ClustersArns; do
    ServiceArns=$(aws ecs list-services --cluster $ClustersArn --query 'serviceArns' --
output text)
    for ServiceArn in $ServiceArns; do
        aws ecs update-service --cluster $ClustersArn --service $ServiceArn --propagate-tags
SERVICE &>/dev/null
        aws ecs tag-resource --resource-arn $ServiceArn --tags key=$TAG_KEY,value=$TAG_VALUE
    done
done
```

启用成本分配标签以使用 AWS 计费工具

建议激活用户定义的成本分配标签。这样，就可以在 AWS 计费工具（例如和）中识别和筛选 Right sizing 标签。AWS Cost Explorer AWS 成本和使用情况报告如果您不启用此选项，则标签筛选选项和数据将不可用。有关使用成本分配标签的更多信息，请参阅 AWS 账单与成本管理 文档中的 [Activating user-defined cost allocation tags](#)。

等待 24 小时后，您可以在 Cost Explorer 成本管理服务中看到标签，然后在下一节中实施调整建议。为此，请在 Cost Explorer 成本管理服务中搜索调整标签。

实施调整建议

Compute Optimizer 将提供任务或容器大小建议。要实施调整建议，请执行以下操作。

1. 打开 [Amazon ECS 控制台](#)。
2. 从导航栏中，选择包含您的任务定义的地区。
3. 在导航窗格中，选择 Task definitions (任务定义) 。
4. 在 Task definitions (任务定义) 页面上，选择任务，然后选择 Create new revision (创建新修订版) 。
5. 在 Create new task definition revision (创建新任务定义修订版) 页面上，进行更改。要更新容器大小建议，请在 [ECS 任务定义](#) 的 containerDefinitions 数据块下更新 cpu 和 memory。例如：

```
"containerDefinitions": [  
  {  
    "name": "your-container-name",  
    "image": "your-image",  
    "cpu": 1024,  
    "memory": 2048,  
  }  
],
```

6. 验证信息并选择 Create (创建) 。

要更新 Amazon ECS 服务，请执行以下操作：

1. 打开 [Amazon ECS 控制台](#)。
2. 在 Clusters (集群) 页面上，选择集群。
3. 在 Cluster overview (集群概述) 页面上，选择服务，然后选择 Update (更新) 。
4. 对于任务定义，选择要使用的任务定义族和修订。

对于高级操作员，您可以使用 CloudShell 更新 Amazon ECS 服务。例如：

```
bash  
#!/bin/bash
```

```
# Set variables
ClustersName="workshop-cluster"
ServiceName="lab7-fargate-service"
TaskDefinition="lab7-fargate-demo:3"
# update the service
aws ecs update-service --cluster $ClustersName --service $ServiceName --task-definition
$TaskDefinition
```

审查调整前后的成本

调整资源后，您可以使用 Cost Explorer 成本管理服务通过调整标签显示调整前后的成本。请记住，您可以使用[资源标签](#)来跟踪成本。通过使用多层标签，您可以精细地了解成本。在本指南中介绍的示例中，调整标签用于将通用标签应用于所有目标实例。然后，使用团队标签来进一步整理资源。下一步是引入应用程序标签，以进一步显示运行特定应用程序的成本影响。

举一个例子，说明在单个账户级别使用调整标签可以降低成本。在本示例中，运营成本从每天 30.26 美元降至每天 7.56 美元。假设每月 744 小时，调整前的年度成本为 11,044.9 美元。调整后，年度成本降为 2,759.4 美元。这意味着该账户的计算成本降低了 75%。想象一下这对大型组织的影响。

在开始调整旅程之前，考虑以下事项：

- AWS 提供了许多降低成本的选项。这包括 [AWS OLA](#)，AWS 它会在迁移到本地实例之前先检查您的本地实例 AWS。O AWS LA 还为您提供合适规模的建议和许可指南。
- 在购买[节省计划](#)之前，完成所有调整。这可以帮助您避免超额购买节省计划承诺。

后续步骤

我们建议您完成以下后续步骤：

1. 查看您的现有格局，并考虑将 Amazon EBS gp2 卷转换为 gp3 卷。
2. 查看[节省计划](#)。

其他资源

- [Compute Optimizer 入门](#) (文档) AWS
- [标记 AWS 资源的最佳实践](#) (AWS 白皮书)
- [Windows 容器已开启 AWS](#) (工作AWS 室工作室)

了解您的 Amazon EKS 成本

概述

要有效监控 Kubernetes 部署的成本，需要从整体角度进行考量。唯一固定且已知的成本是 Amazon Elastic Kubernetes Service (Amazon EKS) 控制面板的成本。其中包括构成部署的所有其他组件，从计算和存储到网络，其数量根据您的应用程序需求而定。

您可以使用 [Kubecost](#) 来分析 Kubernetes 基础设施的成本，从[命名空间](#)和[服务](#)一直到各个[容器组 \(pod \)](#)，然后在控制面板中显示数据。Kubecost显示了计算和存储等集群内成本，以及[亚马逊简单存储服务 \(Amazon S3\)](#) [存储桶](#)和[亚马逊关系数据库服务 \(Amazon RDS\)](#) 实例 out-of-cluster等成本。Kubecost 将根据此数据提出调整建议，并显示可能影响系统的关键提醒。Kubecost 可以与 [AWS 成本和使用情况报告 集成](#)，以显示通过[计算类节省计划](#)、[预留实例](#)及其他折扣计划实现的节省。

成本优势

Kubecost 提供报告和控制面板，用于可视化您的 Amazon EKS 部署成本。它使您能够从集群向下钻取到各个组件，例如控制器、服务、节点、容器组 (pod) 和卷。这使您可以全面了解在 Amazon EKS 环境中运行的应用程序。通过启用此可见性，您可以按照 Kubecost 建议采取行动，或者精细地查看每个应用程序的成本。调整 Amazon EKS 节点组可以像调整标准 EC2 实例一样节省成本。如果您可以调整容器和节点，则可以从运行容器所需的实例大小和自动扩缩组中所需的 EC2 实例数量中消除计算膨胀。

成本优化建议

要利用 KubeCost，我们建议您执行以下操作：

1. 将 Kubecost 部署到您的环境中
2. 获取 Windows 应用程序的详细成本明细
3. 调整集群节点
4. 调整容器请求
5. 管理未充分利用的节点
6. 补救放弃的工作负载
7. 根据建议采取行动
8. 更新自行管理的节点

将 Kubecost 部署到您的环境中

[亚马逊 EKS Finhack 研讨](#)会教你如何部署配置为在自有账户中使用 Kubecost 的 Amazon EKS 环境。AWS 这可让您获得有关技术方面的实践经验。如果您有兴趣在组织中举办此讲习会，请联系您的客户团队。

要使用 [Helm](#) 将 Kubecost 部署到你的 Amazon EKS 集群，请参阅博客上[AWS 和 Kubecost 作为 EKS 客户提供成本监控](#)的帖子。AWS 或者，您可以参阅 [Kubecost 官方文档](#)，了解有关安装和配置 Kubecost 的说明。有关 Kubecost 对 Windows 节点的支持的信息，请参阅 Kubecost 文档中的 [Windows Node Support](#)。

获取 Windows 应用程序的详细成本明细

虽然使用 [Amazon EC2 竞价型实例](#)可以节省大量成本，但您也可以从 Windows 工作负载往往是有状态这一事实中受益。竞价型实例的使用取决于应用程序，我们鼓励您验证它们是否适用于您的使用案例。

要获取 Windows 应用程序的详细成本明细，请[登录 Kubecost](#)。在导航页面中，选择节省。

调整集群节点

在 [Kubecost](#) 中，从导航栏中选择节省，然后选择调整集群节点。

举一个例子，Kubecost报告集群在 vCPU 和 RAM 方面都被过度预调配。下表显示了 Kubecost 的详细信息和建议。

	Current	建议：简单	建议：复杂
总数	每月 3462.57 美元	每月 137.24 美元	每月 303.68 美元
节点计数	4	5	4
CPU	74 VCPUs	10 VCPUs	8 VCPUs
RAM	152 GB	20GB	18 GB
实例明细	2 个 c5.xlarge + 外加 2 个	5 个 t3a.medium	2 个 c5.xlarge + 外加 1 个

正如 Kubecost 博客文章 [Find an optimal set of nodes for a Kubernetes cluster](#) 中所述，简单选项使用单节点组，而复杂选项使用多节点组方法。了解如何采用按钮可以一键调整集群大小。它需要安装 [Kubecost 集群控制器](#)。

如果您使用的是非 [eksctl](#) 创建的 [自行管理的 Windows 节点](#)，请参阅[更新现有的自行管理的节点组](#)。这些说明向您展示了如何在[自动扩缩组](#)使用的 Amazon EC2 启动模板中更改实例类型。

调整容器请求

在 [Kubecost](#) 中，从导航栏中选择节省，然后转到请求调整建议页面。本页面显示了容器组 (pod) 的[效率](#)、调整建议及预计的成本节省。您可以使用自定义按钮按集群、节点、命名空间\控制器等进行筛选。

举个例子，假设 Kubecost 计算出您的某些容器组 (pod) 在 CPU 和 RAM (内存) 方面被过度预调配。然后，Kubecost 建议您调整到新的 CPU 和 RAM 值，以实现其预计的每月节省。要更改 CPU 和 RAM 值，您必须更新[部署清单](#)文件。

管理未充分利用的节点

在 [Kubecost](#) 中，从导航栏中选择节省，然后选择管理未充分利用的节点。

举一个例子，页面显示集群中的一个节点在 CPU 和 RAM (内存) 方面未充分利用，因此可能会耗尽、终止或调整大小。选择未通过节点和容器组 (pod) 检查的节点可让您更详细了解为什么它们无法被耗尽。

补救放弃的工作负载

在 [Kubecost](#) 中，从导航栏中选择节省，然后选择放弃的工作负载页面。在此示例中，您按名为 windows 的命名空间进行筛选。此页面显示了尚未达到流量阈值且视为已放弃的容器组 (pod)。容器组 (pod) 需要在定义的时间段内发送或接收一定数量的网络流量。

在仔细确认一个或多个容器组 (pod) 已废弃后，您可以通过缩减副本数量、删除部署、调整其大小以消耗更少的资源或通知应用程序所有者您认为部署已放弃来节省成本。

根据建议采取行动

在调整集群节点部分中，Kubecost 分析了集群中 Worker 节点的使用情况，并就如何调整节点以降低成本提出了建议。以下两种类型的节点组可与 Amazon EKS 一起使用：[自行管理](#)和[托管式](#)。

更新自行管理的节点

有关更新自行管理节点的信息，请参阅 Amazon EKS 文档中的[自行管理的节点更新](#)。它指出，使用 `eksctl` 创建的节点组无法更新，且必须将其迁移到具有新配置的新节点组。

例如，假设您有一个名为 `ng-windows-m5-2xlarge` 的 Windows 节点组（使用 `m5.2xlarge` EC2 实例），并且您想将容器组（pod）迁移到名为 `ng-windows-t3-large` 的[新节点组](#)（由 `t3.large` EC2 实例支持，以节省成本）。

要在使用由 `eksctl` 部署的节点组时迁移到新的节点组，请执行以下操作：

1. 要查找容器组（pod）当前所在的节点，请运行 `kubectl describe pod <pod_name> -n <namespace>` 命令。
2. 运行 `kubectl describe node <node_name>` 命令。输出显示该节点正在一个 `m5.2xlarge` 实例上运行。它还与节点组名称（`ng-windows-m5-2xlarge`）匹配。
3. 要将部署更改为使用节点组 `ng-windows-t3-large`，请删除节点组 `ng-windows-m5-2xlarge` 并运行 `kubectl describe svc,deploy,pod -n windows`。由于节点组已删除，部署将立即开始重新部署。

Note

删除节点组后，服务将停机。

4. 在几分钟后再次运行 `kubectl describe svc,deploy,pod -n windows` 命令。输出显示容器组（pod）都再次处于正在运行状态。
5. 要显示容器组（pod）现在正在节点组 `ng-windows-t3-large` 上运行，请再次运行 `kubectl describe pod <pod_name> -n <namespace>` 和 `kubectl describe node <node_name>` 命令。

其他调整大小方法

此方法适用于自行管理或托管式节点组的任意组合。[Seamlessly migrate workloads from EKS self-managed node group to EKS-managed node groups](#) 博客文章提供了如何将工作负载从实例类型过大的一个节点组迁移到已调整的节点组，而无需任何停机的指引。

后续步骤

Kubecost 可轻松地可视化 Amazon EKS 环境的成本。Kubecost 与 Kubernetes 的深度集成 AWS APIs 可以帮助你找到潜在的成本节约。您可以在 Kubecost 的节省控制面板中查看这些建议。Kubecost 还可以通过[集群控制器功能](#)为您实施其中一些建议。

我们建议您在 AWS 容器博客中查看 step-by-step 部署情况 [AWS , Kubecost 合作为 EKS 客户提供成本监控](#)。

其他资源

- [Amazon EKS Workshop](#) (Amazon EKS 研讨会)
- [AWS 与 Kubecost 合作为 EKS 客户提供成本监控 \(AWS 博客 \)](#)
- [亚马逊 EKS Finhack AWS 研讨会](#) (工作室工作室)
- [Windows 容器已开启 AWS](#) (工作AWS 室工作室)

使用 App2Container 更换 Windows 应用程序平台

概述

[AWS App2Container](#) 是一款命令行工具，用于将 Java 和 .NET Web 应用程序迁移到容器中并实现现代化。App2Container 分析并构建在裸机、虚拟机、Amazon Elastic Compute Cloud (Amazon EC2) 实例或其他云提供商中运行的所有应用程序的清单。您可以选择要容器化的应用程序。App2Container 将应用程序构件和依赖关系打包到容器映像中，配置网络端口，并生成必要的 Amazon Elastic Container Service (Amazon ECS) 和 Amazon Elastic Kubernetes Service (Amazon EKS) 部署构件，这些构件是基础设施即代码 (IaC) 模板。App2Container 预调配将容器化应用程序部署到生产环境所需的云基础设施和 CI/CD 管线。有关更多信息，请参阅 App2Container 文档中的 [How App2Container works](#)。

借助 App2Container，您可以将应用程序迁移到容器 AWS 并对其进行现代化改造，同时还可以标准化应用程序的部署和操作。您可以使用 App2Container 来帮助快速构建概念验证 (PoC) 或加快在容器中部署生产工作负载。

使用 Windows 应用程序时，需要记住以下几点。App2Container 支持将部署在 Microsoft Internet Information Services (IIS) 上的 ASP.NET 应用程序容器化，包括在 Windows Server 2016、Windows Server 2019 或 Windows Server Core 2004 上运行的 IIS 托管的 Windows Communication Foundation (WCF) 应用程序。有关更多信息，请参阅 App2Container 文档中的

[Supported applications for Windows](#)。App2Container 使用 Windows Server Core 作为容器构件的基础映像，使 Windows Server Core 容器版本与您运行容器化命令的服务器的操作系统（OS）版本相匹配。此方法将应用程序与底层操作系统解耦，以便您可以升级操作系统，而无需执行传统迁移。

如果您使用 Worker 计算机容器化应用程序，则容器基础映像 [例如 Windows Server 2019 长期服务频道（LTSC）] 与您的 Worker 计算机操作系统（例如 Windows Server 2019）相匹配。如果您直接在应用程序服务器上运行容器化，则版本与您的应用程序服务器操作系统相匹配。如果您的应用程序在 Windows Server 2008 或 2012 R2 上运行，您仍然可以通过为容器化和部署步骤设置 Worker 计算机来使用 App2Container。App2Container 不支持在 Windows 客户端操作系统（例如 Windows 7 或 Windows 10）上运行的应用程序。App2Container 支持 Java 进程的 Tomcat、ToMee 和 JBoss（独立模式）框架。有关更多信息，请参阅 [App2Container 兼容性](#)。

成本优势

与 one-application-to-one 服务器部署设计模式相比，容器化和整合应用程序可以[节省高达 60% 的计算成本](#)。App2Container 有助于加快应用程序容器化过程。以下是使用 App2Container 满足现代化需求的一些优势：

- App2Container 不收取额外费用。
- App2Container 在一个容器映像中支持多个应用程序。
- 使用 App2Container 将您的遗留 .NET 应用程序迁移到容器，以应对即将终止支持的操作系统问题。您可以迁移到更新的操作系统，无需为扩展支持付费，并降低安全风险。
- 容器是打包 .NET 应用程序的一种高效且经济实惠的方法。在 [MACO 建议：迁移到容器](#) 中查看容器的好处。
- 应用程序整合和容器化通过更有效地使用计算资源，来帮助减少计算、存储和许可占用空间。
- 迁移到容器可以降低运营开销和基础设施成本，并提高开发可移植性和部署灵活性。

成本优化建议

有关如何使用 App2Container 的说明，请参阅 [Getting started with AWS App2Container](#)。有关 App2Container 命令的更多信息，请参阅 [App2Container command reference](#)。

后续步骤

App2Container 可以加快容器化应用程序及部署到 Amazon EKS 或 Amazon ECS 的过程。将应用程序部署到容器可降低计算、网络 and 存储成本，并降低应用程序操作者的运营开销。

如需亲身体验 App2Container，请参阅[使用 AWS App2Container 现代化研讨会](#)。如果您想获得深入的学习体验，请让您的 AWS 客户团队安排一个 App2Container 沉浸式体验日。

其他资源

- [使用容器化复杂的多层 Windows 应用程序 AWS App2Container](#) (AWS 博客文章)
- [使用容器化旧版 ASP.NET 应用程序 AWS App2Container](#) (AWS 博客文章)
- [App2Container 支持的应用程序](#) (文档) AWS
- 利用@@ [AWS App2Container 研讨会实现现代化](#) (AWS 工作室工作室)
- [AWS App2Container FAQs](#) (AWS 网站)

仓储服务

为您的 Microsoft 工作负载选择合适的存储方案是一项至关重要的架构决策。作为决策过程的一部分，我们建议您制定存储计划，并确定您的应用程序和服务的功能需求。本章将概述以下几种可能会影响您的规划决策的存储选项。

部分：

- [Amazon EBS](#)
- [Amazon FSx](#)
- [AWS Storage Gateway](#)

Amazon EBS

Amazon Elastic Block Store (Amazon EBS) 是一项完全托管的块存储服务，它能让您创建持久的块级存储卷，这些存储卷可与 Amazon Elastic Compute Cloud (Amazon EC2) 实例一起使用。您可以利用 Amazon EBS 中的多种功能，以有效的方式管理和优化您在云中运行的 Windows 工作负载的存储资源。例如，您可以使用 Amazon EBS 来为您的工作负载预调配所需的精确 IOPS 和吞吐量，从一系列卷类型中进行选择以满足您的工作负载需求，并使用相关工具来识别并消除不必要的存储资源浪费。这种对存储性能和使用情况的精细控制，能帮助您优化存储资源，同时避免不必要的成本。

本节将介绍以下主题：

- [将 Amazon EBS 卷从 gp2 迁移到 gp3](#)
- [修改 Amazon EBS 快照](#)
- [删除未附加的 Amazon EBS 卷](#)

将 Amazon EBS 卷从 gp2 迁移到 gp3

概述

固态硬盘 (SSD) 是生产和高性能工作负载的标准存储选项。Amazon EBS 为中高性能工作负载提供[通用型 SSD 卷](#)。许多通用固态硬盘 AWS 服务 卷 (包括 Amazon EC2) 的标准是 [gp2](#)，这是第二代通用固态硬盘卷。名为 [gp3 的第三代通用 SSDs 型于 2020 年 12 月发布](#)。

gp3 版本在性能自定义方面相较于前一代有了显著的提升。对于 Amazon EBS gp2 卷，其性能与卷的大小密切相关。每增加 1GB 的存储容量，gp2 卷获取了 3 IOPS 的性能。也就是说，一个 2000GB 的

gp2 卷能够实现 6000 次 IOPS。对于 gp3 卷，其性能可独立于存储容量进行自定义。这使得即使是小容量的卷也能实现高达 80,000 IOPS 和 2,000 MB/s 吞吐量的性能。

gp3 卷的另一个主要变化在于基准 IOPS 性能。gp3 卷的起始值为 3000 IOPS。相比之下，gp2 卷的大小必须达到 1TiB 才能实现相同的性能。对于 C: 盘通常小于 1TiB 的 Windows Server 来说，从 gp2 升级到 gp3 是一项显著的性能提升。

最后，gp3 卷的价格相比 gp2 卷有了最大的降幅。gp3 卷提供了增强的性能，但成本却比 gp2 卷低 20%。

成本影响

由于能够独立于容量来调整性能，因此了解额外增加 IOPS 和吞吐量的定价方式就显得尤为重要。对于 gp2 卷，定价基于每月每 GiB 0.10 美元的预调配容量。对于 gp3 卷，定价与高性能[预调配 IOPS SSD 卷](#)类似，这类存储卷的定价包含容量费用以及额外 IOPS 和吞吐量的单独费用。

如下表所示，gp3 卷的容量价格为每月 GiB 0.08 美元（比 gp2 低 20%），吞吐量超过 3,000 的 IOPS 费用分别为每预配置 IOPS 每月 0.005 美元，吞吐量超过 125 美元，每个预配置月 0.04 美元。MiBs

	gp3	gp2
卷大小	1 GiB — 64 TiB	1 GiB : 16 TiB
基准 IOPS	3000	3 IOPS/GiB (至少 100 IOPS) 至最大 16,000 IOPS 小于 1TiB 的卷最高可爆发至 3000 IOPS
每个卷的最大 IOPS	80,000	16000
基准吞吐量	125 MiBs	吞吐量限制介于 128 MiBs -250 之间 MiBs，具体取决于卷大小
每个卷的最大吞吐量	2000 MiBs	250 MiBs
价格	每月 0.08 美元/Gib	0.10 USD/GiB-月

	gp3	gp2
	前 3000 IOPS 免费，超过 3000 时每个预调配 IOPS 每月 0.005 美元	
	125 美元 MiBs 免费，每月预配置 0.04 MiBs 美元，超过 125 美元 MiBs	

 Important

尽管 gp3 卷的容量和性能有单独的成本，但如果配置的性能水平相同，gp3 卷始终比 gp2 卷便宜。

以下表格显示了在不同容量和性能配置下，将 gp2 卷转换为 gp3 卷所能实现的成本节省示例。

gp2 配置的示例

卷大小 (GiB)	最大 IOPS	吞吐量 (MiBs)	成本 (美元/月)
30	3000	128	3.00 美元
100	3000	128	10.00 美元
500	3000	250	50.00 美元
1000	3000	250	100.00 美元
2000	6000	250	200.00 美元
6000	16000	250	600.00 美元

gp3 (基准) 配置的示例

最大 IOPS	吞吐量 (MiBs)	成本 (美元/月)	成本降低 (与 gp2 相比)
3000	125	2.40 美元	20%
3000	125	8.00 美元	20%
3000	125	40.00 美元	20%
3000	125	80.00 美元	20%
3000	125	160.00 美元	20%
3000	125	480.00 美元	20%

gp3 (gp2 匹配) 配置的示例

最大 IOPS	吞吐量 (MiBs)	成本 (美元/月)	成本降低 (与 gp2 相比)
3000	128	2.52 美元	16%
3000	128	8.12 美元	19%
3000	250	45.00 美元	10%
3000	250	85.00 美元	15%
6000	250	180.00 美元	10%
16000	250	550.00 美元	8%

有关成本分析，请参阅 [Amazon EBS 资源](#) 中的 EBS gp2 到 gp3 的迁移成本节省计算器部分。您可以下载该计算器，并利用它来计算将 gp2 卷迁移至 gp3 后，您能够节省多少费用。

成本优化建议

有关如何完成迁移过程的说明，请参阅存储博客上的 [“将您的 Amazon EBS 卷从 gp2 迁移到 gp3 并节省高达 20% 的成本”](#) 一文。AWS

其他资源

- [将您的 Amazon EBS 卷从 gp2 迁移到 gp3，最多可节省 20% 的成本](#) (存储博客) AWS
- [制定 AWS Config 自定义规则以优化 Amazon EBS 卷类型](#) (AWS 云运营和迁移博客)
- [通过删除未使用的 Amazon EBS 卷来控制 AWS 成本](#) (AWS 云运营和迁移博客)
- [亚马逊 EBS 迁移实用程序](#) () GitHub
- [从 2020 re:Invent 公告找到节省金额](#) (AWS 云财务管理)
- [成本优化研讨会](#) (Well-Archit AWS ected Labs)
- [gp2 到 gp3 迁移的成本节省计算器](#) (下载)

修改 Amazon EBS 快照

概述

删除 EBS 卷并管理快照的保留与归档是控制成本的首要关键环节。您可以通过拍摄快照将 EBS 卷上的数据备份到亚马逊简单存储服务 (Amazon S3)。point-in-time 快照属于增量备份，因此它只会保存设备上在最新快照之后更改的那些数据块。由于无需复制数据，这将最大限度缩短创建快照所需的时间和增加存储成本节省。每个快照都包含了将您的数据 (从该快照创建之时起) 恢复到新的 EBS 卷所需的所有信息。

EBS 快照的费用按每月每 GB 计算。将按快照的大小以及您保留该快照的时间长短计费。定价因存储层不同而不同。对于[标准层](#)，您只需为存储的已更改块付费。对于存档层，您需要为存储的所有快照块付费。您还需要为从[存档层](#)中检索快照付费。下面是每个存储层的示例方案：

- 标准层：您有一个可存储 100GB 数据的卷。您需要为第一个快照 (快照 A) 中全部 100GB 数据付费。在下次快照 (快照 B) 生成之时，您拥有的数据量为 105GB。然后，您只需为增量快照 B 所增加的 5GB 存储付费即可。
- 存档层：存档快照 B。然后，将该快照移至存档层，您需要为全部 105GB 快照块付费。

您可以使用 [Amazon Data Lifecycle Manager](#) 来帮助您设置生命周期，以便按计划保留和管理快照。

成本影响

EBS 卷和快照的费用分别进行管理。EBS 快照的计费费率低于活动的 EBS 卷。当实例终止时，每个连接的 EBS 卷的[DeleteOnTermination 属性](#)值将决定是保留还是删除该卷。默认情况下，根卷的

DeleteOnTermination 属性将设置为 True。对于所有其他卷类型，该值设置为 False。这会导致出现这样的情况：操作人员本打算删除 EC2 实例，但却留下了除了根卷之外添加到该实例中的其他卷。有关检查不再需要的卷（及其相关快照）的说明，请参阅 Amazon EBS 文档中的[查看有关 Amazon EBS 卷的信息](#)。

默认情况下，快照在创建时将存储在 Amazon EBS 快照标准层中（标准层）。存储在标准层中的快照可递增。这意味着只在卷上保存在最新快照之后发生更改的数据块。[Amazon EBS 快照归档](#)是一个新存储层，可用于长期低成本地存储很少访问且不需要频繁或快速检索的快照。标准层和存档层的价格差异很大，因此在设置快照策略时，这一点应当成为重要的考量因素。Amazon EBS 快照归档为您存储极少需要访问的快照达 90 天或更长时间，同时存储成本降低 75%。

Amazon EBS 快照存储	成本
标准	0.05 美元/GB/月
档案	0.0125 美元/GB/月

在较小的环境中，成本节省可能不会十分显著。如果有多个账户和成千上万个 EC2 实例，即使删除 EBS 卷也有 EBS 快照，那么在大规模情况下，节省的费用就更显著了。TBs

下表比较了每月标准层和归档层的使用情况（在使用量仅为 50TB 时）。即便是在这种较小的规模下，每年仍能节省数千美元。

Amazon EBS 快照存储	每月费用	每年费用
标准 50TB	312.50 美元	3,750 美元
存档 50TB	78.13 美元	937.60 美元
每年节省费用		2,812.40 美元

成本优化建议

删除快照可能不会降低组织的数据存储成本。其他快照可引用已删除快照的数据，已引用的数据总是会被保留。例如，当您为具有 10GiB 数据的卷创建首个快照时，快照大小也为 10GiB。由于快照是增量式的，您为同一个卷创建的第二个快照仅包含在第一个快照创建后发生变更的数据块。第二个快照还会引用第一个快照中的数据。如果您在更改 4GiB 数据后创建了第二个快照，第二个快照的大小为

4GiB。此外，第二个快照将引用第一个快照中没有发生变更的 6GiB 数据。有关更多信息，请参阅[删除我的 EBS 卷快照然后删除卷本身，为什么我的存储成本没有降低？](#)在 AWS 知识中心中。

请考虑以下事项：

- 您无需为他人 AWS 账户 拥有并与您的账户共享的快照付费。只有在您将共享快照复制到您的账户时，才会向您收费。您还需要为通过共享快照创建的 EBS 卷付费。
- 如果快照（快照 A）被另一个快照（快照 B）引用，则删除快照 B 可能不会降低存储成本。删除快照时，仅会移除该快照特有的数据。被其他快照引用的数据仍然存在，您需要为此引用的数据付费。要删除增量快照，请参阅 Amazon EBS 文档中的[增量快照删除](#)。

在 AWS 中运行工作负载时，快照清理是标准操作实践。随着时间的推移，这些快照可能会累积产生高额的数据费用，而这些数据其实是您并不需要的。

其他资源

- [通过删除未使用的 Amazon EBS 卷来控制 AWS 成本](#)（AWS 云运营和迁移博客）
- [删除 Amazon EBS 快照](#)（Amazon EBS 文档）
- [成本优化研讨会](#)（Well-Archit AWS ected Labs）
- [使用 Amazon Data Lifecycle Manager 自动存档 Amazon EBS 快照](#)（AWS 存储博客）

删除未附加的 Amazon EBS 卷

概述

未连接（孤立）的 EBS 卷可能会给您的环境带来不必要的存储成本。AWS 作为 AWS 环境卫生的一部分，必须定期审查和删除未使用和未使用的 EBS 卷。最佳做法是建立一个持续审查 EBS 卷使用情况的流程。您可以使用 [AWS Compute Optimizer](#) 来查看未充分利用的实例。本部分可帮助您识别、管理和删除未附加或未充分利用的 EBS 卷。

Amazon EBS

[Amazon Elastic Block Store（Amazon EBS）](#) 是为 Amazon Elastic Compute Cloud（Amazon EC2）实例提供存储卷的块级设备。EBS 提供持久性存储服务，同时具备附加和从 EC2 实例分离的灵活性。这意味着，即使 EC2 实例终止，EBS 卷的生命周期仍然会持续存在。该 [DeleteOnTermination](#) 属性是一项功能，用于控制在实例终止时是保留还是删除附加的 EBS 卷。默认情况下，根卷的属性被设置为 True，从而导致删除。对于其他卷，它被设置为 False，从而导致保留。

成本影响

未附加的 EBS 卷 (也被称为未使用或孤立的卷) 与已附加的卷所产生的费用相同，具体取决于预调配的存储大小和存储类型。尽管 Amazon EBS 的平均费用看似很低，仅为每 GB 每月 0.10 美元，但必须认识到，未使用的 EBS 卷的累积会随着时间的推移产生巨大的费用。

例如，假设保留 50 个未使用的 EBS 卷 (每个卷预调配了 100GB 的存储大小) ，其影响情况如下表所示。

存储卷的数量	卷类型	Size	月度总费用
50 个卷	gp2 (0.10 美元)	100 GB	100 GB × 50 个 EBS 卷 × 0.10 美元/GB-月 = 500.00 美元

根据上表所示的情况，每月可节省约 500 美元，每年则可节省约 6000 美元。这是降低成本的有效举措。请务必将删除未连接的 EBS 卷作为 AWS 环境卫生的常规做法。

成本优化建议

您可以使用轻松 AWS 地自动删除未连接的 EBS 卷。例如，您可以使用 AWS Lambda、AWS Config CloudWatch、Amazon 和，根据年限、标签和其他规格 AWS Systems Manager 来定义删除独立卷的标准。您还可以使用它们 AWS 服务 来大规模自动执行清理过程。

为避免产生意外后果，我们建议您在删除未附加的 EBS 卷之前先进行充分的调查。

管理未附加的 EBS 卷

我们建议您考虑以下最佳实践：

- 满足合规性要求：确认删除未附加的 EBS 卷的行为符合您所在组织的治理和合规性要求。
- 设置数据备份和保留策略：在删除未附加的 EBS 卷之前，请将所有重要数据备份到另一个存储库 (例如 [Amazon S3](#)) 。在数据留存方面，[Amazon EBS 快照](#) 是一种比 EBS 卷更经济实惠的数据保留方式，并且如果日后需要，还可以通过它们来恢复该卷。有关有效管理快照的更多信息，请参阅本指南的[修改 Amazon EBS 快照](#)部分。
- 检查依赖关系：检查未附加的 EBS 卷与其他 AWS 资源之间是否存在任何依赖关系。您可以使用 [AWS 管理控制台 或 API](#) 来收集有关您的 EBS 卷的描述性信息，例如大小、状态以及相关资源。这是防止意外删除任何暂时未附加资源的重要步骤。

- **创建保留策略：**为未附加的 EBS 卷设定保留期限。这可以帮助您确定删除未连接卷的合适时间，从而确保您的 AWS 环境保持优化。例如，您可以创建 A [amazon EventBridge](#) 规则来按计划启动 Lambda 函数。Lambda 函数可以使用 AWS SDK 主动识别任何未连接的 EBS 卷，应用标记机制以便于跟踪，并在未连接的 EBS 卷达到或超过定义的阈值时发出通知。
- **标记未附加的 EBS 卷：**[标记](#) EBS 卷是一项有用的实践，它有助于根据诸如环境、应用程序或所有者等属性对卷进行分类和标识。在决定删除哪些未附加的卷时，这一点尤其有用，因为它能让您根据标签快速识别出不再需要的卷。
- **确保安全删除：**查看 EBS 卷上一次被附加的时间，有助于您确定删除该卷是否安全。有关更多信息，请参阅[如何使用 AWS CLI 命令列出特定 Amazon EBS 卷的附件或分离历史记录？](#)在 AWS 知识中心中。
- **识别未充分利用的 EBS 卷** — 为了降低存储成本和维护优化的环境，强烈建议识别和删除未充分利用的 EBS 卷。AWS AWS Trusted Advisor 并[AWS Compute Optimizer](#)可以帮助您识别未充分利用的 EBS 卷，并提供降低成本和提高效率的建议。例如，请参阅使用 () [设置自动优化 EBS 卷](#)、[建立 Trusted Advisor 组织 AWS Trusted Advisor\(TAOGitHub\) 控制面板 \(Workshop AWS p Studio \)](#) 和[使用 AWS Compute Optimizer \(AWS 存储博客 \) 优化 Amazon EBS 卷的成本](#)。

自动清理未附加的 EBS 卷

我们建议您考虑使用以下工具来自动清理未附加的 EBS 卷：

- [AWS APIs \(DescribeVolumes\)](#) — 您可以使用 AWS SDKs 或 AWS Command Line Interface (AWS CLI) 筛选和查找未连接的 EBS 卷。通过使用脚本或按计划运行的 [Lambda 函数](#)来实现此流程自动化，您可以节省时间和精力。中的[示例脚本](#) GitHub 演示了其工作原理。该脚本使用 Lambda 来分析 AWS CloudTrail 日志并识别未连接的 EBS 卷。
- [AWS Systems Manager 自动化](#) — 这使您能够自动执行基础架构中的日常维护和补救任务。首先，[请创建自动化运行手册](#)，该手册会定义一系列按特定顺序执行的步骤。例如，您可以创建一个运行手册，其步骤首先是创建一个未附加 EBS 卷的快照，然后删除该卷本身。这可以帮助您将那些原本需要耗费大量时间且手动操作时容易出错的任务实现自动化。
- [AWS Config](#) — 这使您能够评估、审计和跟踪 AWS 资源随时间推移而发生的变化。通过捕获配置更改，您可以使用 AWS Config 来评估环境中的合规性、治理和资源利用率。例如，AWS Config 可以识别[未使用的 EBS 卷](#)。此外，您可以将 AWS Systems Manager Automation 与关联 AWS Config 以自动修复未使用 EBS 卷的删除。

其他资源

- 使用 AWS Systems Manager 和 [AWS \(规范性指南\) 删除未使用的亚马逊 Elastic Block Store \(Amazon EB AWS Config S\) 卷](#)
- [通过删除未使用的 Amazon EBS 卷来控制 AWS 成本](#) (AWS 云运营和迁移博客)
- [AWSConfigRemediation-DeleteUnusedEBSVolume](#) (AWS Systems Manager 自动化运行手册参考)

Amazon FSx

Amazon FSx for Windows 文件服务器是一项完全托管的文件存储服务，已针对 Windows 工作负载进行了优化。它为您提供了一种简单且可扩展的解决方案，让您能够运行基于 Windows 的应用程序和工作负载，而无需进行复杂的存储基础设施管理。您可以使用 FSx Windows File Server 轻松配置和访问原生支持 Windows 应用程序的共享文件存储，包括微软 SQL Server SharePoint、微软和自定义 .NET 应用程序。此外，FSx for Windows File Server 通过提供灵活的定价选项（例如 pay-as-you-go 和存储配额）和自动重复数据删除来减少存储占用空间并优化性能和成本，从而帮助您管理成本。

本节将介绍以下主题：

- [选择合适的 SMB 文件存储](#)
- [在 Amazon 中启用重复数据删除 FSx](#)
- [了解 Windows 文件服务器中的 FSx 数据分片](#)
- [了解 Amazon 中的硬盘容量使用情况 FSx](#)
- [使用单可用区](#)

选择合适的 SMB 文件存储

概述

AWS 提供各种完全托管的存储服务，为您提供业界领先的文件服务的丰富功能，同时将最新的 AWS 基础架构创新与安全性相结合。您可以将 AWS 服务整合到基础设施即代码 (IaC) 工作流程中，并将其与 AWS 计算、监控和数据保护服务集成。对于 Windows 工作负载，您可以从两种完全托管的文件服务中进行选择，以满足您的应用程序需求：适用 FSx 于 Windows 文件服务器和 Amazon FSx for NetApp ONTAP。

FSx 适用于 Windows 文件服务器

Amazon FSx for Windows File Server 提供基于 Windows Server 的完全托管的共享存储，并提供各种数据访问、数据管理和安全管理功能。FSx for Windows File Server 可轻松与 Windows 环境集成，因为它是一项 Windows 原生服务。我们建议将 Windows 文件服务器 FSx 用于用户和组共享，对于 SQL Server、Windows 应用程序和虚拟桌面基础架构 (VDI) 使用始终开启故障转移群集实例。FSx 适用于 Windows 的文件服务器还可以很好地与亚马逊 FSx 文件网关、亚马逊 Kendra、亚马逊 S3 的审计日志和亚马逊 Data Firehose 集成。

FSx 适用于 ONTAP

FSx 因为 ONTAP 基于 NetApp 的专有 ONTAP 文件系统。这需要一定程度的技能提升，主要推荐给现有的本地 NetApp 用户。典型的用例包括用户和组共享、SQL Server 的始终开启故障转移群集实例以及 Windows 应用程序。FSx for ONTAP 支持多种协议、大于 64 TB 的文件系统（没有 DFS 命名空间服务器的 PB 规模）、克隆、复制、快照、压缩（存储效率）和数据的智能分层。

成本影响

FSx 适用于 Windows 文件服务器

FSx Windows 文件服务器是第一个用于为 SQL Server 部署故障转移群集实例的共享存储解决方案。AWS 对 FSx 于 Windows 文件服务器，您可以使用 SQL 标准版许可启动故障转移群集实例。然而，这却使得您无法使用 Always On 可用性组，因为此类组需要安装 SQL Server 企业版许可证。通过从 SQL Server 企业标准版切换到 SQL Server 标准版，您可以节省 65-75% 的 [SQL Server 许可](#) 费用。

您可以使用适用于故障转移群集实例的 Windows File Server for File Server 来卸载典型的 EBS I/O 存储中的存储。通过卸载 I/O 到 FSx for Windows File Server，您可以在不影响存储吞吐量的情况下缩减依赖高亚马逊 EBS 吞吐量和 IOPS 的 EC2 实例。

FSx 适用于 ONTAP

您可以使用 FSx for ONTAP 在块协议 iSCSI 上运行 Microsoft 故障转移群集，并从 SQL Server 即时文件初始化、跨区域复制使用 SnapMirror、防病毒支持和克隆中受益。如果您为测试创建了多个数据库副本，则克隆功能将会在存储空间占用以及数据库副本的创建速度方面带来显著的差异。此外，您可以使用 f FSx or ONTAP NetApp SnapCenter 来管理适用于 SQL Server 的 EC2 实例的备份、还原和克隆功能。FSx for ONTAP 还提供从 SSD 到低成本容量池存储的自动分层，以实现性能和成本效益的结合。

FSx for ONTAP 支持 NetApp 的文件系统 (ONTAP)，这与支持 Windows 原生 NTFS 文件系统的 Windows 文件服务器不同 FSx。ONTAP 的最小大小 FSx 为 1024 GB，而 FSx Windows 文件服务器的最小起始大小可以低至 32 GB。

与 Microsoft 分布式文件系统集成

FSx 适用于 Windows 文件服务器和 ONTAP FSx 可与微软的[分布式文件系统 \(DFS\)](#) 集成，以实现与现有部署的无缝集成。在规划架构时，请记住以下事项：

- FSx 适用于 Windows 文件服务器和 ONTAP FSx 的两种部署类型 ([多个可用区和单个可用区](#)) 都支持 [DFS 命名空间 \(DFS-N\)](#)。
- 仅 FSx 适用于 Windows 文件服务器支持 [DFS 复制 \(DFS-R\)](#)，并且仅在使用单个可用区时才支持。

成本优化建议

Windows 文件服务器和 ONTAP FSx 的性能在很大程度上取决于配置，它们的定价也是如此。FSx 适用于 Windows File Server 的定价主要取决于存储容量和存储类型、吞吐容量、备份和传输的数据。FSx 对于 ONTAP，您需要为固态硬盘存储、固态硬盘 IOPS、容量池使用量、吞吐容量和备份付费。

文件服务	5TB 存储空间的成本	配置	Region
FSx 适用于 Windows 文件服务器	982.78 美元	单可用区 SSD (15000 IOPS) 32 MBps 5TB 的备份 (没有通过重复数据删除获得的节省)	美国东部 (弗吉尼亚州北部)
FSx 适用于 ONTAP	979.28 美元	单可用区 100% SSD 15000 读写容量层 15000 SSD IOPS 128 MBps	美国东部 (弗吉尼亚州北部)

文件服务	5TB 存储空间的成本	配置	Region
		5TB 的备份 (没有通过重复数据删除获得的节省)	

记住以下内容：

- 数据去重和压缩能够通过缩小数据大小来使您能够在物理设备上存储更多的数据，但您需要为所预调配的固态硬盘 (SSD) 或硬盘驱动器 (HDD) 存储空间付费。
- 您可以使用 ONTAP FSx 对数据进行分层。只有极少数情况下，所有数据会被定期访问，并且需要使用 SSD 进行存储。您可以将那些不常访问的冷数据迁移到容量层，以节省成本。
- 此处提到的价格基于 SSD 层上 100% 的数据以及 SSD 层上 15000 次 IOPS 计算得出。

备份

默认情况下，ONTAP 和 FSx Windows 文件服务器都 FSx 将其完全托管的备份存储在 Amazon S3 上。但是，FSx 对于 ONTAP，还有一个额外的备份选项 SnapVault，它可以将备份配置为驻留在容量层中。使用备份 SnapVault 是一种自我管理的机制，它比默认的完全托管备份选项更具成本效益。完全托管的备份选项为每月每 GB 0.05 美元。ONTAP (10:1 固态硬盘到容量池存储) 的 SnapVault 备份费用为 0.03221 美元 (0.9x0.0219+0.1x0.125)。FSx

记住以下内容：

- AWS 托管备份的间隔时间为一小时。[SnapVault](#)可以让你低至五分钟。
- 您可以使用 NetApp的工具 (例如 CLI 和 API) 来配置 SnapVault 关系和快照复制。
- 在卷上启用all分层策略以使用容 SnapVault 量层作为备份数据的存储。
- SnapVault 目的地可以位于同一个区域 AWS 区域、跨区域或本地。这通常是指单个可用区或多个可用区的文件系统备份目的地。相比之下，AWS Backup 它得益于 Amazon S3 的区域弹性。

合理调整大小

您还可以通过合理调整大小并避免过度预调配来节省成本，并充分利用您的文件系统。

为合理调整大小，请执行以下操作：

1. 根据数据确定您当前的需求。对于典型的 Windows 工作负载，您可以使用[性能监视器](#)等内置操作系统工具。
2. 在性能监视器中，使用以下计数器来衡量您当前的性能需求。将捕获间隔设定为一秒，最大日志大小为 1000MB，并启用覆盖功能。

```
Logman.exe create counter PerfLog-Short -o "c:\perflogs\PerfLog-Long.blg" -f bincirc  
-v mmddhhmm -max 1024 -c "\LogicalDisk(*)\*" "\Memory\*" "\.NET CLR Memory(*)\*"  
"\Cache\*" "\Network Interface(*)\*" "\Paging File(*)\*" "\PhysicalDisk(*)\*"  
"\Processor(*)\*" "\Processor Information(*)\*" "\Process(*)\*" "\Thread(*)\*"  
"\Redirector\*" "\Server\*" "\System\*" "\Server Work Queues(*)\*" "\Terminal  
Services\*" -si 00:00:01
```

3. 要开始捕获日志，请运行 `logman start PerfLog-Short` 命令。要停止捕获日志，请运行 `logman stop PerfLog-Short` 命令。

Note

您可以在运行捕获操作的服务器上的 `c:\perflogs` 中找到性能日志文件。有关更多信息，请参阅 Microsoft 文档中的 [Windows 性能监视器概述](#)。

4. 确定正确的配置后，使用诸如 Microsoft [DISKSPD](#) 之类的磁盘压力工具，在亚马逊 FSx 文件系统中测试您的估计是否正确。
5. 如果您对性能感到满意，请切换到文件共享。

我们建议采取保守的存储容量规划方法，因为存储容量只能纵向扩展。吞吐能力可以根据需要纵向扩展和缩减。

其他资源

- [FSx 适用于 NetApp ONTAP 的亚马逊 FAQs](#) (AWS 网站)
- [使用新指标优化 Amazon FSx 的 Windows 文件服务器性能](#) (AWS 存储博客)

在 Amazon 中启用重复数据删除 FSx

概述

重复数据删除是一项功能，它能让您更高效地存储数据，并且降低存储容量的需求。它涉及在不损害数据准确性或完整性的情况下，找出并消除数据中的重复内容。重复数据删除采用子文件的可变大小分块

以及压缩技术，这使得一般文件服务器的优化比率为 2:1，而虚拟化数据的优化比率最高达 20:1。重复数据删除的效果要比 NTFS 压缩好得多。重复数据删除架构的固有特点是能够在硬件出现故障时保持系统的韧性：对数据和元数据进行完整的校验和验证，包括为元数据和最常访问的数据块提供冗余。

FSx 适用于 Windows 文件服务器完全支持重复数据删除。使用它能够使通用型文件共享平均节省 50% 至 60% 的费用。在这些共享中，用户文档的节省范围介于 30% 至 50% 之间，而软件开发数据集的节省范围最高达 70% 至 80%。需要明白的是，通过重复数据删除所能实现的存储空间节省效果取决于您的数据集的性质，包括文件之间存在的重复数据量的多少。如果所存储的数据具有动态性，那么重复数据删除功能就不是一个好的选择。

成本影响

为了应对企业中数据存储量的增长，管理员们会对服务器进行整合，并将容量扩展和数据优化列为关键目标。重复数据删除的默认设置能够立即节省成本，或者管理员也可以对设置进行微调以获取更多收益。例如，您可以将其配置为仅在特定文件类型上运行重复数据删除，也可以创建自定义作业计划。

总体上看，重复数据删除有三种类型的作业：优化、垃圾回收和清理。请注意，只有在完成优化操作后运行垃圾回收作业，空间才会被释放。您可以安排作业，也可以手动运行它。计划重复数据删除作业时可用的所有设置，在您手动启动作业时也同样适用（除了那些与特定于调度的设置之外）。

即使重复数据删除仅有效节省了 25%，但 Windows 文件服务器还是可以节省大量成本。FSx 这些预计的节省金额基于 AWS 定价计算器中的[估算值](#)。

成本优化建议

默认情况下，Window FSx s 文件服务器文件系统未启用重复数据删除。要使用[远程管理](#)启用重复数据删除 PowerShell，必须运行 Enable-FSxDedup 命令，然后使用该 Set-FSxDedupConfiguration 命令来设置配置。有关更多信息，请参阅 Windows [文件服务器文档中的 FSx 管理文件系统](#)。

要启用重复数据删除，请运行以下命令：

```
PS C:\Users\Admin> Invoke-Command -ComputerName amznfsxxxxxxx.corp.example.com -  
ConfigurationName FSxRemoteAdmin -ScriptBlock {Enable-FsxDedup }
```

要验证重复数据删除配置，请运行以下命令。

```
Invoke-Command -ComputerName amznfsxxxxxxx.corp.example.com -ConfigurationName  
FSxRemoteAdmin -ScriptBlock {  
Set-FSxDedupSchedule -Name "CustomOptimization" -Type Optimization -Days  
Mon,Tues,Wed,Sat -Start 09:00 -DurationHours 7
```

```
}
```

通过运行 PowerShell Measure-DedupFileMetadata cmdlet，您可以确定如果删除一组文件夹、一个文件夹或单个文件，然后运行垃圾收集作业，则可以回收卷上多少潜在的磁盘空间。具体而言，DedupDistinctSize 值告诉您在删除这些文件后可以回收多少空间。文件中通常包含与其他文件夹共享的数据块，因此，重复数据删除引擎会计算出哪些是在垃圾回收作业后将被删除的唯一数据块。

默认的[重复数据删除作业计划](#)旨在很好地适用于推荐的工作负载，并尽可能减少对系统的影响（不包括针对备份使用类型启用的优先级优化作业）。如果工作负载对资源的需求量较大，我们建议您仅将作业安排在空闲时段运行，或者减少或增加重复数据删除作业所能使用的系统资源量。

默认情况下，重复数据删除会占用可用内存的 25%。但是，可以通过使用 -memory switch 来增加此值。对于优化作业，我们建议您将范围设置为 15 到 50。对于计划作业，您可以使用更高的内存消耗量。例如，对于垃圾回收和清理作业（通常安排在工作时间进行），您可以设置更高的内存消耗量（例如 50）。

有关重复数据删除设置的更多信息，请参阅适用于 Windows File Server 的文档中的[使用重复数据删除降低存储成本](#)。FSx

其他资源

- [了解重复数据删除](#)（Microsoft 文档）
- 通过@@ [重复数据删除降低存储成本](#)（FSx 适用于 Windows 文件服务器文档）

了解 Windows 文件服务器中的 FSx 数据分片

概述

FSx 对于 Windows 文件服务器，性能取决于配置。它主要基于存储类型、存储容量和吞吐量配置。您选择的吞吐容量决定了文件服务器可用的性能资源，包括网络 I/O 限制、CPU 和内存以及文件服务器施加的磁盘 I/O 限制。您选择的存储容量和存储类型决定了存储卷的可用性能资源，即存储磁盘施加的磁盘 I/O 限制。除了性能外，配置选择还会影响成本。FSx 适用于 Windows File Server 的定价主要取决于存储容量和存储类型、吞吐容量、备份和传输的数据。

如果您有相对较大的文件存储和性能需求，则可以从数据分片中获益。数据分片设计到[将文件数据划分为较小的数据集（分片）](#)，然后将其存储在不同的文件系统中。从多个实例访问您的数据的应用程序可以通过并行读取和写入这些分片来实现高水平性能。同时，您仍然可以在公共命名空间下向应用程序

呈现统一视图。此外，它还可以帮助大型文件数据集扩展文件数据存储，使其超出每个文件系统所支持的容量（64 TB），最高可达数百 PB。

成本影响

对于大型数据集，通常更有效的做法是为 Windows File Server 文件系统部署多个小型 FSx 数据集，而不是部署一个大型 SSD 共享以实现相同的性能级别。将 FSx 适用于 Windows File Server 的 HDD 和 SSD 存储类型组合使用可以更好地节省成本，并使您能够将工作负载与最佳底层磁盘子系统相匹配。在以下表格中，您可以看到单个 17TB 文件系统之间的区别，并将其与添加到相同容量的多个较小文件系统进行比较。

具有多个工作负载的大型 SSD 文件系统

服务器名称	成本	配置	Region
FSx 适用于 Windows 文件服务器的亚马逊	5716 美元	17TB SSD	美国东部（弗吉尼亚州北部）
		30% 重复数据删除	
		256 Mbps	
		17TB 备份	

已使用 DFSN 对工作负载进行分区

服务器名称	成本	配置	Region	共享
FSx 适用于 Windows 文件服务器的亚马逊	1024 美元	2TB SSD	美国东部（弗吉尼亚州北部）	共享 1
		20% 重复数据删除		
		128 Mbps		
		2TB 备份		
		多可用区		

服务器名称	成本	配置	Region	共享
FSx 适用于 Windows 文件服务器的亚马逊	2132 美元	5TB SSD 30% 重复数据删除 256 Mbps 5TB 备份 多可用区	美国东部（弗吉尼亚州北部）	共享 2
FSx 适用于 Windows 文件服务器的亚马逊	1036 美元	10 TB 硬盘 40% 重复数据删除 128 Mbps 10TB 备份 多可用区	美国东部（弗吉尼亚州北部）	共享 3
DFSN Windows EC2 实例	27 美元	t3a.medium 2 v CPUs 4GiB 内存	美国东部（弗吉尼亚州北部）	DFSN 实例

大型 SSD 文件系统的年成本为 68592 美元。分区工作负载的年成本为 50640 美元。在此示例中，通过将工作负载与合适的后端存储相匹配，可以实现 26% 的成本节省。有关定价估算的更多信息，请参阅 [AWS 定价计算器](#) 估算。

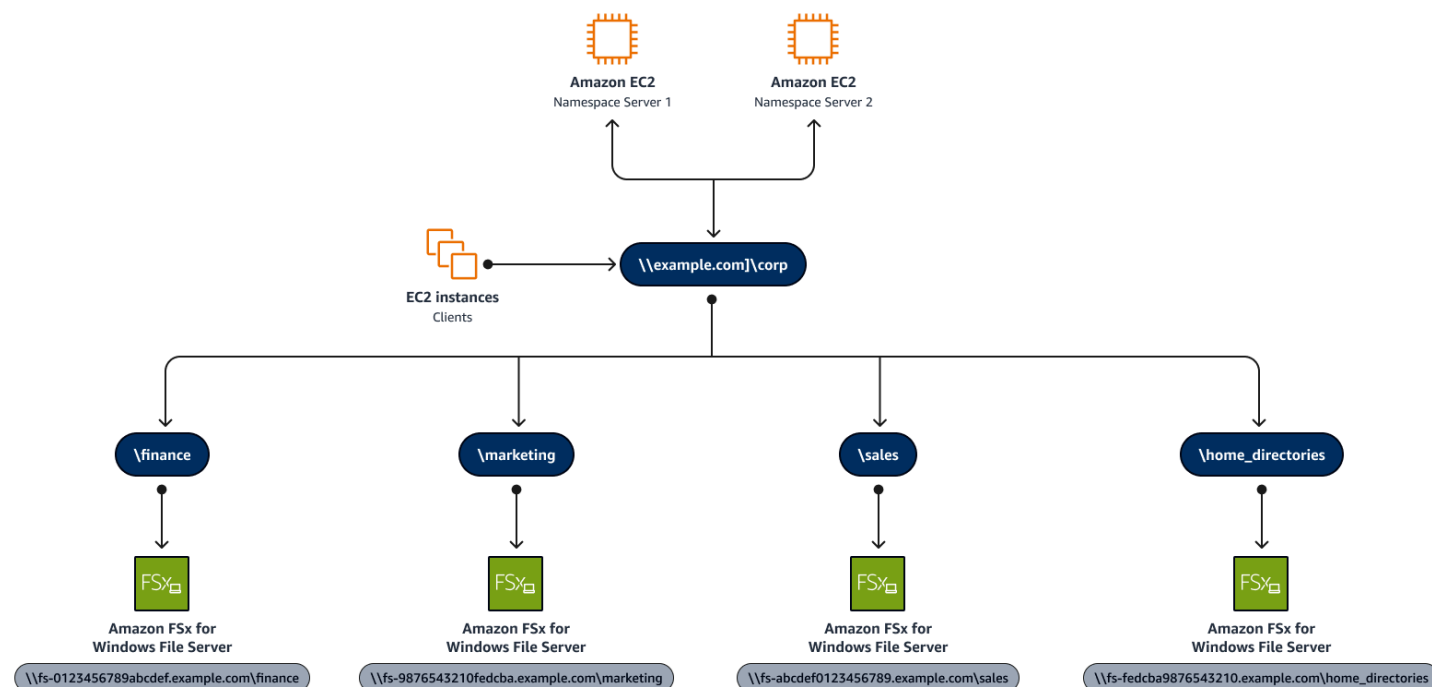
成本优化建议

要部署重复数据删除解决方案，必须根据数据类型、I/O 大小和 I/O 访问模式设置 [Microsoft DFS 命名空间](#)。每个命名空间总共支持最多 5 万个文件共享和数百 PB 的存储容量。

选择一种分片约定，在计划使用的所有文件系统中均匀分布 I/O，这样的效果最为显著。监控您的工作负载将有助于进一步优化或降低成本。如果您在衡量亚马逊 FSx 文件系统的性能信息时需要帮助，[FSx 请参阅 Windows 文件服务器文档中的 Windows FSx 文件服务器性能](#)。

选择分片策略后，您可以利用 DFS 命名空间对文件系统进行分组，以便更轻松地了解您的共享。这使得用户能够看到一个同质文件系统，但实际上他们所访问的却是具有专用使用案例的各种不同的文件系统。重要的是要按照适当的命名规则来创建共享，这样您的最终用户就能轻松理解这些共享是用于何种工作负载的。此外，还需要对生产共享和非生产共享进行标注，以免最终用户因误操作而将文件放置在错误的文件系统中。

下图显示了如何使用单个 DFS 命名空间作为多个 Amazon FSx 文件系统的访问点。



记住以下内容：

- 您可以将现有 FSx 的 Windows 文件服务器共享添加到 DFS 树中。
- FSx 无法将 Amazon 添加到 DFS 共享路径的根目录。您只有一个子文件夹。
- 您必须部署一个 EC2 实例来提供 DFS 命名空间配置。

有关 DFS-N 配置的更多信息，请参阅 Microsoft 文档中的 [DFS 命名空间概述](#)。有关使用 DFS 命名空间的更多信息，请观看上的“在 [Amazon FSx for Windows 文件服务器上使用 DFS 命名空间](#)”视频。
YouTube

其他资源

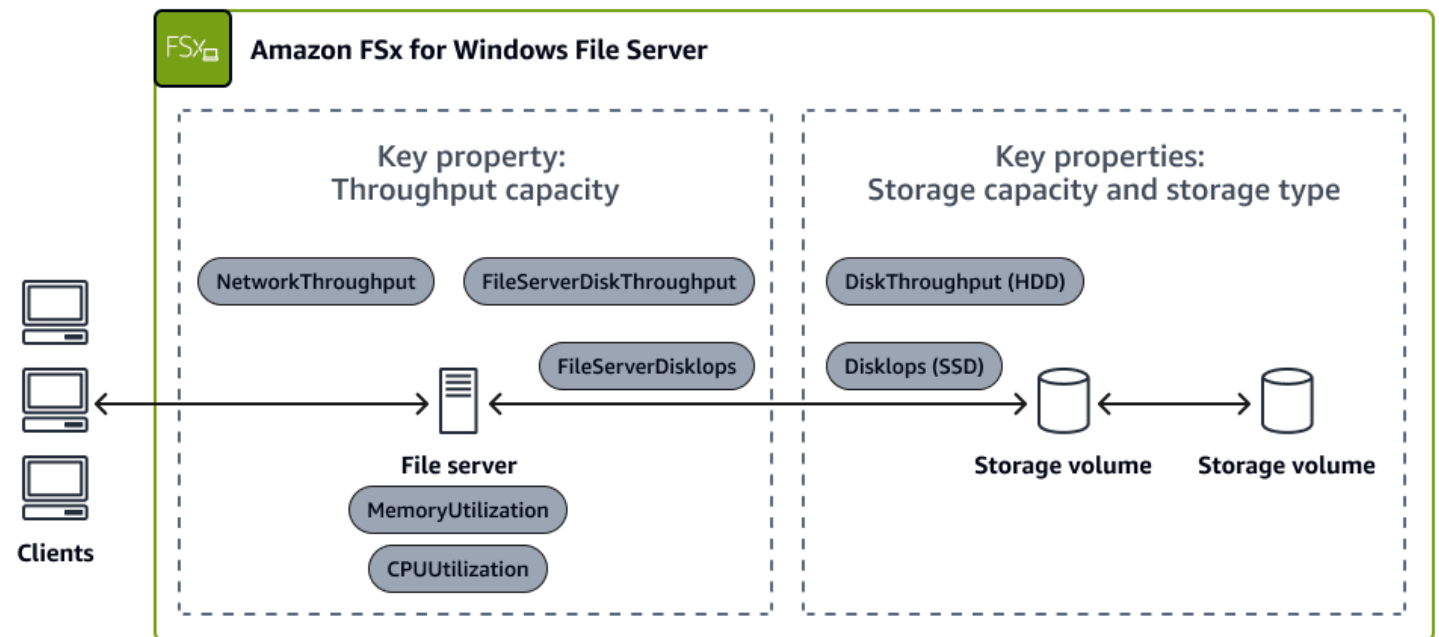
- [使用 DFS 命名空间对多个文件系统进行分组 \(Amazon 文档 \)](#) FSx
- [演练 6：使用分片扩展性能 \(Ama FSx zon 文档 \)](#)
- 在 [Amazon 上使用 DFS 命名空间 FSx 适用于 Windows 文件服务器 \(实验室 \)](#) AWS

了解 Amazon 中的硬盘容量使用情况 FSx

概述

Amazon FSx for Windows File Server 提供了独立于文件系统容量选择吞吐量的灵活性。有两种容量设置可用：硬盘驱动器 (HDD) 和固态硬盘 (SSD)。

以下图表显示吞吐量和存储设置之间的关系。



使用基于 HDD 的存储，您将获得 12 IOPS 基准，突发磁盘 IOPS 80 (每 IOPS TiB 存储)，吞吐量为 12 Megabytes/second，突发量为 80 (Megabytes/second 每 TiB 存储)。例如，如果您的共享大小为 50TB，则您将获得 $50 * 12 = 600$ 作为吞吐量和 IOPS 的基准值。

Amazon FSx for Windows 文件服务器提供 80 次突发 IOPS。当您的使用率低于基准速率时，爆发服务抵扣金会自动补充；而当您的使用率高于基准速率时，突发服务抵扣金则会被自动消耗掉。例如，如果您的工作负载在接下来的一个小时内仅使用 10 IOPS/TB (比基线高 2 IOPS/TB below your baseline rate), you can then utilize 14 IOPS/TB (2 IOPS/TB)，然后又用完了突发积分。

对于文件操作，Amazon FSx for Windows File Server 为固态硬盘存储提供稳定的亚毫秒延迟，为硬盘存储提供个位数毫秒的延迟。对于所有文件系统，包括带有硬盘存储空间的文件系统，Amazon FSx for Windows File Server 在文件服务器上提供了快速（内存中）缓存，因此无论存储类型如何，您都可以为主动访问的数据获得高性能和亚毫秒级的延迟。

在适当情况下，使用 HDD 存储可以帮助降低您的整体存储容量的成本，并为您的需求提供一个可靠的存储平台。

成本影响

Amazon FSx for Windows 文件服务器的性能取决于三个因素：存储容量、存储类型和吞吐量。网络 I/O 性能和内存中缓存大小完全由吞吐容量决定，而磁盘 I/O 性能则由吞吐容量、存储类型和存储容量的组合决定。

虽然建议将固态硬盘用于 I/O 密集型工作负载，但硬盘性能规格可以满足各种工作负载的需求。HDD 存储专为各种工作负载而设计，包括主目录、用户和部门共享以及内容管理系统。例如，如果您的用户仅需要对支持当前项目的数据进行低延迟访问，那么您所存储的大部分数据都很少被访问。

您可以使用 [AWS 定价计算器](#) 将 20TB SSD 与 us-east-1 中的 HDD 文件系统进行比较。如下表所示，即便不考虑重复数据删除带来的节省，将 HDD 文件系统与 SSD 文件系统进行比较时，成本差异也相当显著。

Amazon FSx 文件系统配置	月度成本
20TB 多可用区 SSD (us-east-1)	4,699.30 美元
20TB 多可用区 HDD (us-east-1)	542.88 美元
预计月度节省	4,156.42 美元

 Note

FSx 有关 Windows 文件服务器节省的其他费用，请参阅本指南的“[在 Amazon 中启用重复数据删除 FSx](#)”部分。

通过准确识别性能需求，您可以为您的工作负载选择合适的存储，并降低相关成本。

成本优化建议

如果您决定使用 HDD 存储，请测试您的文件系统以确保它能够满足您的性能要求。与 SSD 存储相比，HDD 存储的成本更低，但单位存储磁盘吞吐量和磁盘 IOPS 也更低。它可能适用于 I/O 要求较低的通用用户共享和主目录、不经常检索数据的大型内容管理系统或包含少量大文件的数据集。

无法更改现有文件系统的存储类型。要转换 Amazon FSx for Windows 文件服务器文件系统的存储类型，您必须备份现有文件系统并将其恢复到具有所需存储类型的新文件系统。如果您想将现有的 SSD 文件系统转换为 HDD 文件系统，请注意，HDD 的最小容量要求要高得多，为 2TB。

要使用其他存储类型恢复备份，请执行以下操作：

1. [备份您的现有文件系统](#)。
2. 使用硬盘存储类型@@ [创建一个新 FSx 的 Amazon 文件系统](#)。
3. 将备份还原到具有所需存储类型的新文件系统中。
4. 确认新的文件系统具有正确的存储类型，并且您的数据完好无损。

在将更改转移到生产环境之前，我们建议您分析您的 Amazon FSx 文件系统的性能并确认更改是否可以接受。有关更多指导，请参阅 AWS 存储博客上的“[使用新指标优化 Amazon FSx for Windows 文件服务器性能](#)”一文。

其他资源

- 利用@@ [亚马逊优化成本 FSx](#) (亚马逊 FSx 文档)

使用单可用区

概述

本节说明何时使用[亚马逊 FSx for Windows 文件服务器](#)的单一可用区实现更有利。它涵盖了迁移到单个可用区可以降低成本，同时仍允许您使用 FSx 适用于 Windows 的 Amazon File Server 作为托管文件存储服务的场景。我们建议您为生产工作负载实施一个适用于 Amazon FSx 的单一可用区。这有助于确保您拥有多个可用区的冗余配置。

成本影响

单个可用区的文件系统相较于多可用区的实施方式，能节省约 40% 的成本。采用多可用区文件系统时，使用 SSD 的费用为每 GB 月 0.230 美元，使用 HDD 的费用为每 GB 月 0.025 美元，而单个可用

区文件系统中使用 SSD 的费用为每 GB 月 0.130 美元，使用 HDD 的费用为每 GB 月 0.013 美元。您可以使用 [AWS 定价计算器](#) 查看成本比较并创建自己的估算值。

对于 10TB 的文件系统，采用多个可用区的方式每月的费用大约为 1200 美元，而使用单个可用区的方式则每月费用为 680 美元。此[示例](#)使用带固态硬盘 FSx 的 Windows 文件服务器文件系统的 10 TB。重复数据删除功能预估可节省 50% 的成本。总体而言，单个可用区的进入成本较低，但同时也有一些需要注意的事项，这些将在下一部分中进行说明。

成本优化建议

单一可用区部署

为确保单个可用区恰到好处，请考虑您自己的内部可用区，SLAs 以便存储 Windows File Server 的数据。FSx 这需要了解您是否必须 SLAs 向客户（内部和外部）提供商品，以及亚马逊 FSx 单一可用区的三个九个可用性是否仍然允许您满足这些 SLAs 需求。FSx 对于具有单个可用区的 Windows 文件服务器，其正常运行时间仍为 99.9%。亚马逊 FSx 针对多个可用区的服务等级协议高于 99.99%。对于任务关键型工作负载，我们建议您使用多个可用区而非单个可用区，即便这样做会产生额外成本。

单个可用区部署非常适合用于诸如 SQL Server 数据库备份这类的工作负载。它们能够通过 HDD 层提供低成本的存储服务，同时还能确保您拥有一致的运行时间。如果您需要为生产工作负载提供更高的可用性，例如高可用性的 SQL 服务器或生产应用程序访问服务，则单个可用区将无法您的工作负载需求。对于备份、非生产测试和开发环境，实施 Amazon FSx 单一可用区可以降低您的运营成本。

Amazon FSx 单可用区文件系统运行良好的一个用例是在生产环境中，使用多个 Amazon FSx 单可用区文件系统，作为使用 Always On 可用性组的高可用性 SQL Server 集群中的每台服务器存储。有关更多信息，请参阅 AWS 存储博客[上 AWS 发布的“优化高可用性 SQL Server 部署的成本”](#)。

多区域复制

如果您想利用 Amazon FSx 的多区域复制，那么使用单个可用区文件系统（只有一个可用区文件系统可以工作的文件系统）来降低成本的一个潜在选择是。您可以部署支持与本机 Microsoft DFS-R 协同使用的[单可用区文件系统](#)。DFS-R 具备自动在不同区域及多个站点之间复制数据的能力。有关使用亚马逊配置 DFS-R 的更多信息 FSx，请参阅亚马逊文档中的使用[微软分布式文件系统复制](#)。FSx

另一种节省多区域成本的替代方案是使用 AWS Storage Gateway。这使您能够在另一个区域实施[亚马逊 FSx 文件网关](#)，以实现亚马逊 FSx 的多区域访问。有关更多信息，请参见本指南的[AWS Storage Gateway](#) 部分。

如果您跨区域工作，则必须考虑跨区域数据流量所产生的数据传输费用。在不同区域之间传输流量需支付每 GB 0.02 美元的费用。因此，如果数据频繁出现大量变动，这将会增加您的总体成本。[例如](#)，1TB 的数据传输等于大约 20.48 美元。

维护窗口

如果您在 Amazon 上使用单一可用区，则维护时段是一个关键考虑因素 FSx。在维护时段内，由于对底层 Windows 服务器进行例行软件修补，Amazon FSx 文件系统将在大约 20 分钟内不可用。如果您使用文件系统进行隔夜备份，请相应地调整 Amazon FSx 维护时段，以免备份期间出现中断。您可以在创建 Amazon FSx 文件系统后调整[维护](#)时段。

其他资源

- [可用性和耐久性：单可用区和多可用区文件系统](#) (Ama FSx zon 文档)
- [亚马逊 Window FSx s 文件服务器定价](#) (AWS 网站)

AWS Storage Gateway

AWS Storage Gateway 是一种混合云存储服务，可将本地环境与 AWS 云存储连接起来。它允许您将现有的本地基础设施与无缝集成 AWS，使您能够从云中存储和检索数据，并在混合环境中运行应用程序。对于 Windows 工作负载，您可以使用存储网关通过 SMB 和 NFS 等原生 Windows 协议来存储和访问数据。您可以使用 Storage Gateway AWS 通过使用本地硬件和软件作为通往云端的桥梁，来降低与运行 Windows 工作负载相关的成本。这使您无需对现有基础架构进行重大更改 AWS 即可利用的可扩展性和成本效益。

在 Storage Gateway 的保护下，你可以获得 Amazon S3 FSx 文件网关、亚马逊文件网关、磁带网关和卷网关。S3 文件网关和 FSx 文件网关最常用于 Microsoft 工作负载。

Amazon S3 文件网关

借助 [Amazon S3 文件网关](#)，您可以将文件存储在 Amazon S3 中，同时使用传统的 SMB 共享向用户提供访问权限。这提供了熟悉的用户界面，并且通过将您的数据存储在 Amazon S3 中并利用 Amazon S3 的各种存储层，帮助降低成本。您可以使用 S3 智能分级功能来实施存储网关，这将帮助您自动将生命周期文件移动至成本最低的存储层，从而进一步降低您的成本。我们推荐使用 S3 文件网关来实现横向扩展、只读访问、快速重复读取（从缓存中）以及数据库转储功能。一般不建议将其用于高性能或高可用性的写入操作、文件编辑或部门共享等场景。

Amazon FSx 文件网关

使用@@ [亚马逊 FSx Windows FSx 文件系统时，Amazon File Gateway](#) 还可以节省成本。您可以搭建 FSx 文件网关以提供对另一个地区的 Amazon FSx 文件系统的本地化访问权限，从而避免拥有两个独立文件系统的成本。如果您有多台本地文件服务器，并且希望将这些服务器整合起来以避免为多个硬件设备付费，那么这样做也会很有帮助。

成本影响

Amazon S3 文件网关

设置 S3 文件网关非常简单，因为您可以使用存储网关的启动向导来进行操作。通过在您的 AWS 环境中使用 EC2 实例，您可以在几分钟内部署网关。设置好网关后，您可以将存储网关共享配置为可通过 SMB 和 NFS 协议进行访问。对于典型的 Windows 工作负载，您还可以采用此设置来利用 Active Directory 环境，并为您的文件共享设置权限。您可以将存储网关有效地集成到您的日常使用之中，因为它可以充当典型的 Windows 文件共享来使用。文件和文件夹存储为对象，NTFS 访问控制列表 (ACLs) 存储为元数据。

以下表格对比了 10TB 存储容量的费用以及三种可供选择的存储选项：

- FSx 适用于 Windows 文件服务器
- Amazon S3 文件网关
- Amazon Elastic Block Store (Amazon EBS)

如果您使用 Amazon S3 来存储 10TB 的数据，其成本会大大降低，因为您可以将数据分区到不同的使用层级中。在定价估算中，将因其定价灵活性而使用 S3 智能分层。这包括 80% 的 S3 标准版、10% 的不常访问层和 10% 的 Amazon Glacier。尽管您可以使用 Amazon Glacier，但设置适当的生命周期规则非常重要，这样可以确保任何被迁移到 Amazon Glacier 的文件无需立即被访问。Amazon Glacier 专用于存档用途，而非常规访问用途。

存储系统	10TB 存储空间的成本	Region
FSx 适用于 Windows 文件服务器 (假设重复数据删除可节省 50%)	683.20 美元 SSD	美国东部 (弗吉尼亚州北部)
Amazon S3 文件网关	449.51 美元智能分层	美国东部 (弗吉尼亚州北部)
Amazon EBS	1,335.69 美元 GP3	美国东部 (弗吉尼亚州北部)

请考虑以下事项：

- 在 Amazon Glacier 中，除非您使用 [RestoreObject](#) API 将对象恢复到 Amazon S3，否则您会收到一般 I/O 错误。我们建议您使用 Amazon CloudWatch Events 来通知此 I/O 错误。这样一来，您的运

营团队就能够针对用户在可能需要访问的文件上出现此错误的情况做出反应。有关这些错误的更多信息，请参阅 Amazon S3 文件网关文档 InaccessibleStorageClass中的[错误](#)：

- 除了 Amazon Glacier 的访问限制外，Storage Gateway 上[每个对象/文件夹只 ACLs 允许 10 个](#)。在决定使用存储网关之前，请确保您不需要超过 10 个 ACL 条目。

Amazon FSx 文件网关

与 Amazon S3 文件网关类似，FSx 文件网关提供对可长期保留数据的文件系统的访问。在 Amazon S3 文件网关中，数据驻留在 Amazon S3 中。对于 FSx File Gateway，您的数据驻 FSx 留在 Windows 文件服务器上。尽管多可用区选项可用 FSx 于 Windows 文件服务器，但没有多区域选项。如果您拥有跨国公司或远程办公场所，那么您可能需要提供一个地理位置更靠近最终用户的共享存储平台，以避免网络延迟问题。如果您要部署另一个 Amazon FSx 文件系统，则会增加全新 Amazon FSx for Windows 文件服务器文件系统的成本和必要的存储空间。为避免创建全新的文件系统和重复成本，您可以在辅助区域部署 FSx File Gateway。这为用户提供了本地化的文件访问方式，同时也有助于降低您的总体成本。

存储系统	10TB 存储空间的成本	Region
FSx 适用于 Windows 文件服务器的亚马逊	683.20 美元 SSD	美国东部（弗吉尼亚州北部）
Amazon FSx 文件网关	503.70 美元/单网关	美国东部（弗吉尼亚州北部）

 Note

上述表格中的价格基于[存储网关定价](#)。

记住以下内容：

- FSx 文件网关可以帮助您每月为多区域工作负载节省大约 180 美元（或每年 2100 美元）。
- FSx File Gateway 的数据传输费用要低得多，因为它只需要缓存定期访问的文件，而不是完整的辅助副本。
- 尽管你可以在不同的区域部署两个 Windows File Server，并使用 AWS Backup 或更新它们 AWS DataSync，但这两个选项都不是实时的。FSx

成本优化建议

Amazon S3 文件网关

S3 文件网关为存储文件提供了一种成本较低的选择，但在其如何实施及使用文件系统方面存在一些需要考虑的问题。例如，S3 文件网关需要使用虚拟机才能运行存储网关软件。在中 AWS，Storage Gateway 默认使用 m5.xlarge 实例部署在 Amazon EC2 中。如果您想降低本地存储成本，可以将 Storage Gateway 作为虚拟设备部署在虚拟化平台（例如 VMware 和 Hyper-V）上。

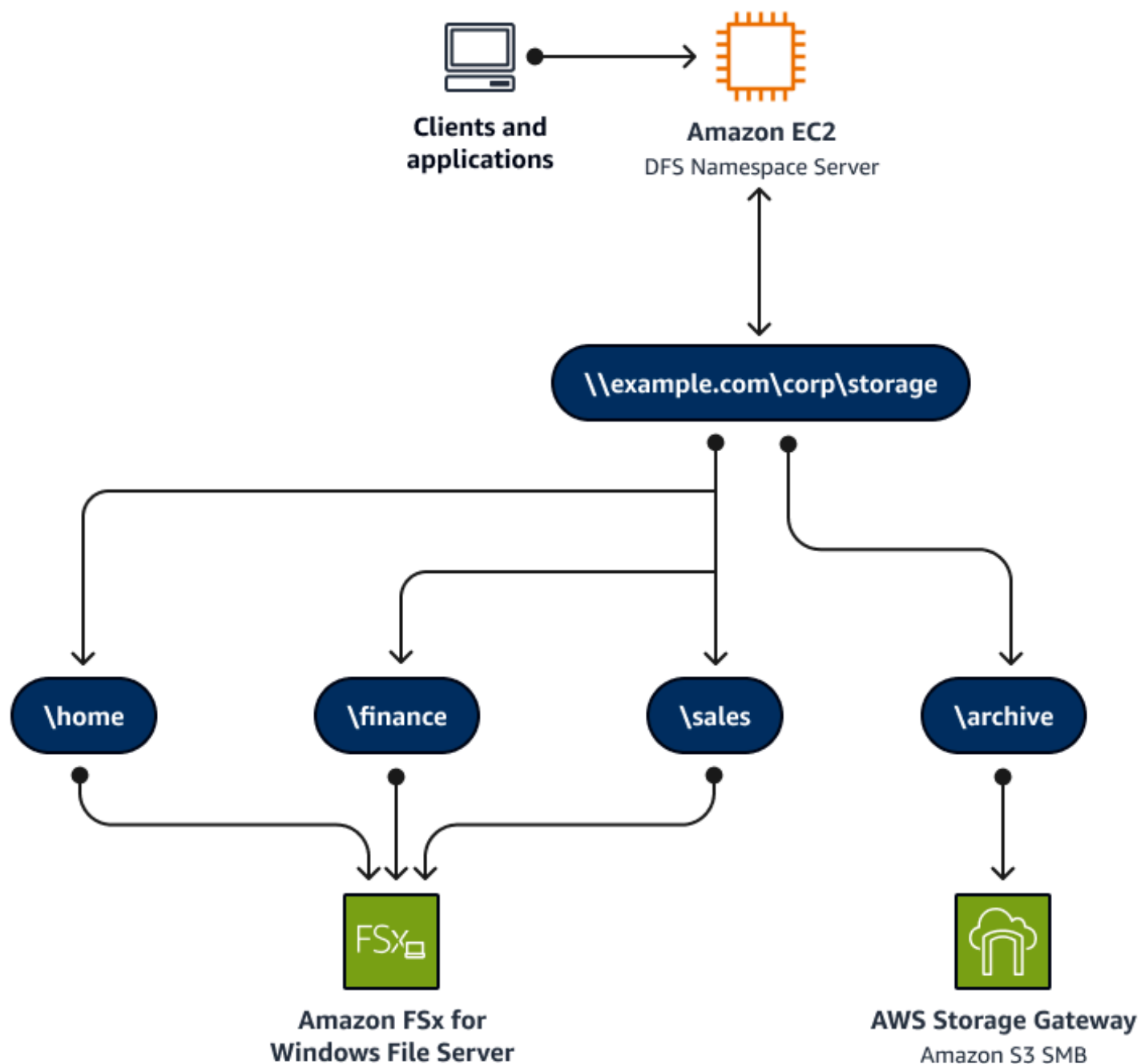
高可用性注意事项

运行中的存储网关是访问文件的单点故障。为避免不必要的停机时间，我们建议您实施严格的访问控制，明确哪些用户能够对存储网关实例进行修改、或停止和启动操作。此外，对于上的部署 AWS，使用 Amazon Data Lifecycle Manager 创建路由快照以快速恢复您的 Storage Gateway 实施也是有益的。如果您使用在本地运行 Storage Gateway VMware，则可以对其进行配置以实现[高可用性](#)。

运行多个文件系统

将日常使用的文件工作负载与存档工作负载分开，有助于您避免不必要的存储费用。Storage Gateway 能够与 FSx 适用于 Windows 的文件服务器文件系统一起部署。通过使用 [DFS 命名空间](#)，您可以呈现在 Windows 文件服务器上 FSx 运行的主要日常使用存储空间和在 Amazon S3 中运行的存储（可通过 Storage Gateway 访问）。

以下图表展示了如何将一个单一的 DFS 命名空间用作不同后端存储选项的前端接入点。



客户端会被定向到文件夹结构，例如 `\\example.com\storage`。这个主目录包含子目录。FSx 适用于 Windows 的文件服务器文件系统包含正常访问的文件共享。您可以使用在存储网关上创建的文件共享来存储存档数据。用户可以手动将项目存入存档文件夹，或者您也可以构建一个流程来自动将某些文件从常规文件共享迁移到存档文件夹。

请考虑以下事项：

- 查看您的存储需求并为[缓存提供足够的存储空间](#)。
- 将您的网关添加到 Active Directory 配置中，然后使用[标准 Windows ACLs 来访问文件](#)。

FSx 文件网关

FSx 文件网关的部署与 S3 文件网关的部署类似，但如果使用启动向导，则会更容易。有关详细说明，请参阅 [Amazon FSx File Gateway 文档中的步骤 3：创建并激活](#) 亚马逊 FSx 文件网关。在您的环境中部署 FSx File Gateway 后，您可以将其与现有 Amazon FSx 文件系统关联并访问您的文件。

部署 FSx 文件网关时，存储是首要考虑因素。默认存储空间提供 150GB，这对于缓存文件来说是一个相当充足的存储容量。创建针对磁盘可用空间不足的监控提醒，以帮助实现存储资源的合理调整，避免过度分配。

其他资源

- [AWS Storage Gateway 资源](#) (AWS 文档)

Active Directory

运行 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2) 是一个安全、可靠且高性能的环境，可用于部署基于 Windows 的应用程序和工作负载。您可以快速预调配实例，并根据需要纵向扩展或缩减，同时仅按实际用量付费。在 Windows Server 环境中，Active Directory 服务用作身份管理的主要来源。

本节将介绍以下主题：

- [Amazon EC2 上的自行管理的 Active Directory](#)
- [AWS Managed Microsoft AD](#)
- [AD Connector](#)

Amazon EC2 上的自行管理的 Active Directory

概述

本节提供降低在 Amazon Elastic Compute Cloud (Amazon EC2) 上运行 Active Directory 成本的建议。主要重点是确保您可以适当地调整 Active Directory 域控制器的大小，并利用的 AWS 云 灵活性根据环境的需要进行调整。AWS 可以帮助您轻松停止实例并调整其大小以满足不断变化的需求，或者在扩展速度过快时缩小实例的大小。选择正确的实例大小和类型可以节省大量资金。

成本影响

下表显示了选择可爆发实例系列实例而不是通用实例之间的区别。此选择每月可为您节省大量资金。对您的实例进行适当的规划和规模调整可以帮助您管理成本。

实例类型	实例的数量	vCPU	内存	成本
t3a.medium	2	2	8	大约每月 81.76 美元
m5a.large	2	2	8	大约每月 259.88 美元

有关成本的更多信息，请参阅 AWS 定价计算器 [估算值](#)。

每月节省 178.12 美元，您的域控制器每年可节省超过 2,000 美元。请记住，这仅适用于一个账户中仅有两个域控制器的小型规模场景。当进行规模化部署，涉及多个账户和额外的域控制器时，此类节省可不断累积而显著降低成本。

成本优化建议

当您部署 Active Directory 环境时，Microsoft 会提供[容量规划建议](#)。我们建议您在规划或扩展 Active Directory 环境时考虑以下主要组成部分：

- 内存
- Network
- 仓储服务
- 处理器

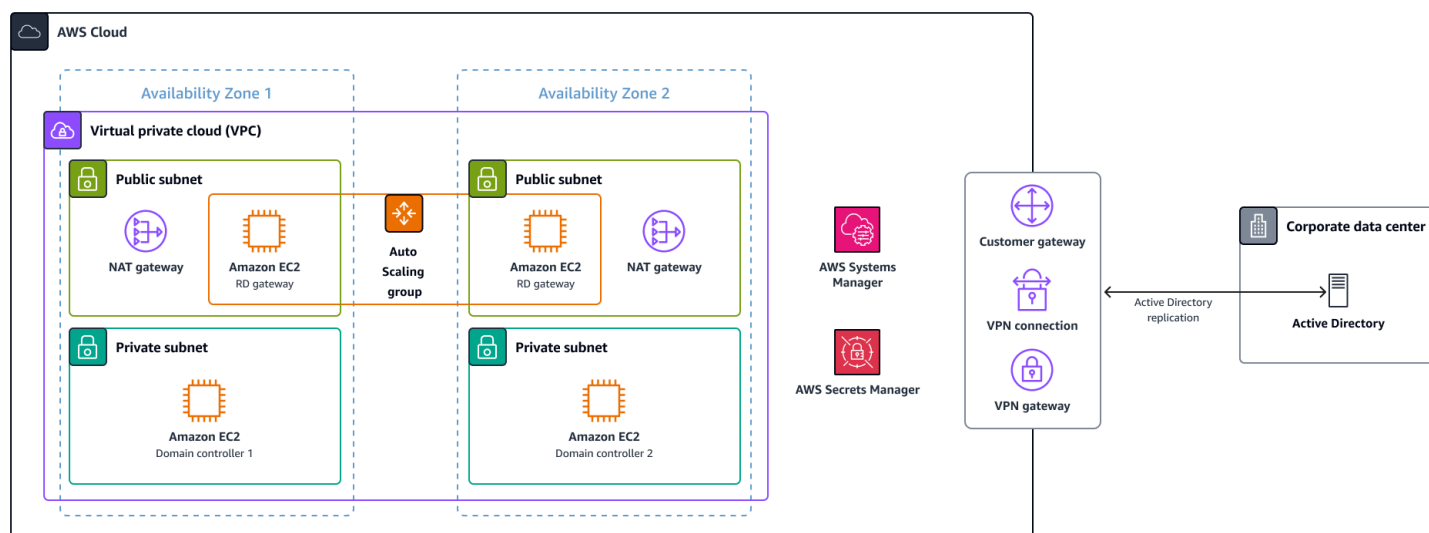
牢记这些主要组成部分，您就可以选择适合您在 AWS 上的 Active Directory 环境的实例类型。本节介绍 AWS 部署场景的几个活动目录示例。这些场景清楚地表明，如果您不打算像在本地环境中 AWS 那样处理相同数量的用户和计算机，则没有必要在中复制本地环境。

下表重点介绍了与 vCPU、内存和磁盘有关的重要组件，以占用您的 AWS 占用空间。

组件	估算
存储/数据库大小	每位用户 40-60 KB
RAM	数据库大小 基本操作系统建议 第三方应用程序
Network	1 GB
CPU	每个核心 1000 个并发用户

混合部署场景

下图显示了 Active Directory 混合部署的示例架构。



如图所示，您通常会有一个本地环境，然后将其扩展到 AWS 云。在迁移的初始阶段中，您通常不会将所有用户和服务器都部署在 AWS 中。因此，为了节省迁移工作的费用，最初部署较小规模的环境非常重要。

如果您打算保留本地环境（服务器和用户在本地进行身份验证），那么在 AWS 中就不需要为域控制器部署相同的环境。通过遵循 Active Directory 最佳实践，您可以实施适当的 [Active Directory 站点和服务](#)，对本地环境的用户和计算机进行身份验证，同时仅向 AWS 中的域控制器验证您的 AWS 环境。这使您能够 AWS 通过仅使用 AWS 资源而不是所有本地基础架构来避免 Active Directory 占用空间过大。有关设计混合设置的指引，请参阅 Microsoft 文档中的 [Proper placement of domain controllers and site considerations](#)。

通过调整大小来优化 AWS 迁移

如果您要为用户部署新的 Active Directory 实例，或者计划完全迁移到 AWS Active Directory 基础架构，我们建议您根据上表中实例选择的 microsoft 对 vCPU、内存和磁盘空间的建议来规划规模。

如果这是一个新的环境，您可以从小规模开始，并利用可以轻松[更改实例类型](#)的功能，随着 AWS 环境的发展调整其大小。本指南的 [Windows on Amazon EC2](#) 部分向您展示了如何在 AWS 上监控和查看您的 CPU 和内存使用率。这样，您就能知道何时增加 EC2 实例的大小。

如果您要将本地 Active Directory 环境完全迁移到 AWS，则可以实施相同的规模调整计划以确保适当的性能。在复制本地内容之前 AWS，我们建议您完成对 Active Directory 环境的全面审查。这可以帮助您防止过度预调配。务必使用性能监视器来收集有关现有域控制器的流量和利用率的信息。这可以让您了解总体使用情况，以便您可以合理调整大小并最终降低成本。

优化活动目录 AWS

如果您在上运行 Active Directory AWS，则还必须持续监控利用率并根据需要更改实例大小以减少开支。您可以使用 AWS Compute Optimizer 来获取有关您正在运行的资源的信息 AWS。有关使用 Compute Optimizer 合理调整您的 Windows 工作负载大小的信息，请参阅本指南的 [Windows on Amazon EC2](#) 部分。要进行更全面的深入研究，您可以使用性能监视器来监控 Active Directory 域控制器的利用率，评估性能，然后相应地调整大小。

您还可以 CloudWatch 使用监控域控制器的性能。要优化您的域控制器（向上或向下扩展），您可以使用中提供的指标 CloudWatch 来帮助您做出正确的决策。您可以使用 CloudWatch代理配置要发送以收集数据的自定义性能监控指标。有关说明，请参阅[如何使用 CloudWatch 代理在 Windows 服务器上查看性能监控器的指标？](#) 在 AWS 知识中心。

部署 CloudWatch 代理后，可以在代理配置文件中的下配置以下指标metrics_collected：

指标类别	指标名称
数据库到实例（NTDSA）	数据库缓存命中率
I/O 数据库读取平均延迟	
I/O database reads/sec	
I/O 日志写入平均延迟	
DirectoryServices (NTDS)	LDAP 绑定时间
DRA 挂起的复制操作	
DRA 挂起的复制同步	
DNS	递归查询/秒
递归查询失败/秒	
TCP 接收查询/秒	
总接收查询/秒	
总发送响应/秒	

指标类别	指标名称
UDP 接收查询/秒	
LogicalDisk	平均磁盘队列长度
可用空间百分比	
内存	已提交字节数使用百分比
长期平均备用缓存生命周期 (秒)	
网络接口	发送的字节数/秒
Bytes Received/sec	
当前带宽	
NTDS	ATQ 估计队列延迟
ATQ 请求延迟	
DS 目录读取/秒	
DS 目录搜索/秒	
DS 目录写入/秒	
LDAP 客户端会话	
LDAP 搜索/秒	
LDAP 成功绑定/秒	
处理器	处理器时间百分比
安全系统范围的统计数据	Kerberos 身份验证
NTLM 身份验证	

其他资源

- [Active Directory 域名服务 AWS：合作伙伴解决方案部署指南](#) (AWS 文档)
- [Capacity planning for Active Directory Domain Services](#) (Microsoft 文档)
- 在 [EC2 实例上运行 Active Directory 的设计注意事项](#) (AWS 白皮书)

AWS Managed Microsoft AD

概述

AWS Directory Service for Microsoft Active Directory，也称为 AWS Managed Microsoft AD，由 Windows 服务器 Active Directory 提供支持并由管理 AWS。您可以使用将各种支持 A AWS Managed Microsoft AD ctive Directory 的应用程序迁移到。AWS 云 AWS Managed Microsoft AD 适用于各种本机 Active Directory 应用程序和服务。它还支持 [AWS 托管应用程序和服务](#)。尽管 AWS Managed Microsoft AD 由于该服务及其计费机制，成本优化杠杆不多，但有一些设计原则可以帮助您将成本降至最低。

成本影响

由于 AWS Managed Microsoft AD 是基于当前的托管服务 SKUs，因此规模调整是一个相对简单的过程。目前有两种尺寸 SKUs 可供选择：标准版和企业版。其他 SKUs 包括目录共享、添加其他域控制器（包括其他区域）和跨区域数据传输。

成本优化建议

AWS Managed Microsoft AD 标准版和 AWS Managed Microsoft AD 企业版有区别。企业版最多支持 50 万个 Active Directory 对象、500 个账户共享（软限制），且支持多区域。标准版最多支持 3 万个 Active Directory 对象、5 个账户共享（软限制为最大约 25 个），且不支持多区域。

Note

Active Directory 对象的上限是近似值。根据对象大小及应用程序的行为和性能需求，您的目录支持的对象数可能更多，也可能更少。

在选择目录类型之前要考虑的问题是：

- 是否需要多区域支持？

- 该目录是否会与超过 25 个账户共享？
- Active Directory 对象计数是否会超过 3 万？

如果以上任何一个问题的答案为“是”，则需要使用企业版。如果所有问题的答案都为“否”，我们建议您先从标准版开始。

Note

您可以将目录从标准版升级到企业版，但无法降级目录。部署标准版并非一条单行道。如果您想将目录升级到企业版，请联系 AWS。

在 AWS Managed Microsoft AD 企业版中共享目录时，每次共享都会产生费用。这比在每个账户中部署目录的成本更低，但请记住，如果不加以检查，共享成本可能会逐渐增加。我们建议您仅与包含亚马逊关系数据库服务 (Amazon RDS) 和亚马逊版 Windows 文件服务器 FSx 的账户共享目录，因为只有这些服务支持此功能。请记住，你可以选择将 Windows 文件服务器与自我管理 FSx 的 Active Directory 集成，包括。AWS Managed Microsoft AD 如果其他账户只需要亚马逊 FSx，则 AWS Managed Microsoft AD 无需共享目录即可针对该账户进行自我管理的 Amazon FSx 部署。

在决定何时部署其他域控制器时，请记住，在同一 VPC 的不同可用区中 AWS Managed Microsoft AD 仅支持两个子网。添加其他域控制器不允许您添加其他子网。要确定是否由于性能问题而必须添加其他域控制器，请查看[中的域控制器性能指标 CloudWatch](#)。这会告诉您是否一个或所有域控制器不堪重负。如果您确定只有一个域控制器不堪重负，那么添加额外的域控制器并不能减轻负载，您需要更深入地研究没有在当前可用的域控制器之间进行负载均衡的应用程序。如果所有域控制器都处于高强度使用状态，则添加额外的域控制器可以减少现有域控制器的负载。有关如何自动扩展的说明，请参阅 AWS 安全博客中的[如何根据利用率指标自动 AWS Managed Microsoft AD 扩展](#)。

如果您将目录扩展到多个区域，我们建议您不要使用 NETLOGON 或 SYSVOL 目录共享进行文件存储。所有域控制器都会复制这些共享的内容。不使用共享文件存储可将数据传输成本降至最低。

您还可以选择通过注册企业协议 AWS。企业协议让您可以选择定制最适合您需求的协议。有关更多信息，请参阅[企业客户](#)。

其他资源

- [AWS Managed Microsoft AD 配额](#) (AWS Directory Service 文档)
- [AWS Directory Service 定价](#) (AWS 网站)
- [AWS 上的 Active Directory 域服务](#) (AWS 白皮书)

AD Connector

概述

[AD Connector](#) 是一项代理服务，它提供了一种简单的方法，可以将你现有的本地 Microsoft Active Directory 连接到兼容的[AWS 应用程序](#)，例如亚马逊 WorkSpaces、Amazon QuickSight，以及亚马逊弹性计算云 (Amazon EC2) 实例的无缝域加入，而无需在云中缓存任何信息。您可以使用 AD Connector 将一个服务账户添加到 Active Directory。AD Connector 可以避免目录同步的需求，也避免了托管联合身份基础设施的成本和复杂性。由于 AD Connector 服务及其账单机制的性质，其成本优化方法并不多，但您可以遵循本节中的设计建议，将成本降至最低。

成本影响

AD Connector 是一项基于预设的托管服务 SKUs。这使得调整大小成为一个简单的过程。有两种尺寸 SKUs 可供选择：小号和大号。您可以使用[AWS 定价计算器](#)来估算涉及 AD Connector 的成本。

成本优化建议

除了后端计算资源之外，小号和大号连接器大小没有区别。

在选择目录类型之前要考虑的问题是：

- 是否有大量（10,000 多个）活跃用户在使用与 AD Connector 集成的 AWS 应用程序？
- 用户是否属于多个、深层或循环嵌套组的成员？

如果所有问题的答案都为“否”，我们建议您从小号开始。如果您对上述任一问题的回答为“是”，那么大号可能值得考虑。您可以从小号 AD Connector 开始，如果目录因性能而受损，则可以请求将该目录升级到大号。

Note

您可以将 AD Connector 从小号升级到大号，但是 AD Connector 无法降级。

大多数性能问题与 AD Connector 无关，而是由于许多用户属于许多嵌套很深或循环嵌套的组，导致本地 Active Directory 域控制器不堪重负。

您还可以选择通过注册企业协议 AWS。企业协议让您可以选择定制最适合您需求的协议。有关更多信息，请参阅[企业客户](#)。

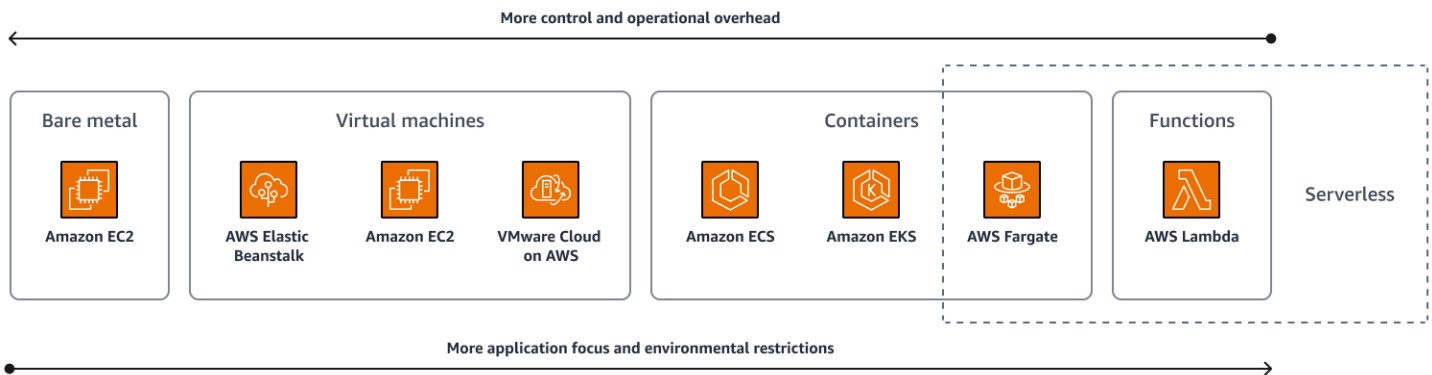
其他资源

- [AD Connector 配额](#) (AWS Directory Service 文档)
- [其他目录类型定价](#) (AWS 网站)
- [AWS上的 Active Directory 域服务](#) (AWS 白皮书)

.NET

开发和部署 .NET 应用程序是帮助您实现云计算所提供规模和敏捷性的关键所在。对于许多传统的 .NET 应用程序，最适合在中运行应用程序的计算选择 AWS 是通过 AWS Elastic Beanstalk 或亚马逊弹性计算云 (Amazon EC2) 使用虚拟机。此外，还可以在 Windows 和 Linux 容器中运行 .NET 应用程序。

引入 .NET Core 使您能够设计出利用云所有优势的现代化 .NET 应用程序。现代应用程序可以使用传统的计算选择集，也可以针对各种类型的无服务器环境，包括 AWS Fargate 或 AWS Lambda。 .NET 6+ 版本现在能够为 ARM64 EC2 实例（如 Graviton2 系列的 EC2 实例）上的工作负载提供高效托管。这样一来，便能够使用 Amazon EC2 上提供的最新一代处理器。这意味着，您的应用程序可以托管在专门针对您的工作负载类型（如视频编码、Web 服务器和高性能计算（HPC））的计算设备上。



本部分提供了相关建议，旨在帮助您对 .NET 应用程序进行调整，以充分利用云技术带来的优势，并着重强调成本效益。

本节将介绍以下主题：

- [重构为现代的 .NET，并迁移到 Linux 上](#)
- [容器化 .NET 应用程序](#)
- [使用 Graviton 实例和容器](#)
- [支持对静态 .NET Framework 应用程序进行动态扩展](#)
- [利用缓存来降低数据库需求](#)
- [考虑使用无服务器 .NET](#)
- [考虑使用专用数据库](#)

重构为现代的 .NET，并迁移到 Linux 上

概述

实现旧版 .NET Framework 应用程序的现代化，能够帮助您提升安全性、性能和可扩展性。实现 .NET Framework 应用程序现代化的有效方法是将其迁移到现代的 .NET 版本（6+）。以下是将这些应用程序迁移到开源 .NET 所带来的部分主要优势：

- 通过在 Linux 操作系统上运行它们来降低 Windows 许可成本
- 充分利用现代语言的可用性
- 获得经过优化可在 Linux 上运行的性能

许多组织仍在运行较旧版本的 .NET Framework。这可能会带来安全风险，因为 Microsoft 不再解决旧版本中的漏洞。Microsoft 已停止对 .NET Framework 4.5.2、4.6 和 4.6.1 这些新版本的支持。评估继续运行该框架旧版本所带来的风险与收益非常重要。为了降低风险并减少成本，值得投入时间和精力重构为现代的 .NET 版本。

成本影响

考虑通用型 EC2 实例类型（m5），它能提供计算、内存和网络资源的均衡。这些实例适用于各种应用程序，例如 Web 服务器、中型数据库和源代码存储库。

例如，美国东部（弗吉尼亚北部）的 Windows Server（含许可证）上的 4 v CPUs 和 16 GB 内存的按需 m5.xlarge 实例每月的费用为 274.48 美元。在 Linux 服务器上使用同样的资源，每月费用为 140.16 美元。在此示例中，如果将您的应用程序从 .NET Framework 迁移到现代版本的 .NET 并在 Linux 服务器上运行应用程序，成本将降低 49%。您的成本可能会因您在选择 [EC2 实例](#) 时所选的选项（例如实例类型、操作系统、存储等）的不同而有所差异。您可以使用 [节省计划](#) 或 [预留实例](#) 进一步优化成本。有关更多详细信息，请使用 [AWS 定价计算器](#) 进行成本估算。对于包含 Windows 的实例，无论定价模式如何，许可费用均为 [每小时每个 vCPU 0.046 美元](#)。

将这些基于 .NET Framework 应用程序移植到现代 .NET 需要开发人员付出努力。您必须评估您的应用程序及其依赖关系，以查看它们是否与目标平台版本兼容。[AWS Porting Assistant for .NET](#) 是一款辅助工具，它能够扫描 .NET Framework 应用程序，并生成 .NET 兼容性评测报告，以帮助您更快地移植应用程序以与 Linux 兼容。Porting Assistant for .NET 能够识别与 .NET 的不兼容之处，找出已知的替换方案，并生成详细的兼容性评测报告。在移植您的解决方案之后，您必须手动更改代码，以便您的项目能够使用依赖关系成功编译。这减少了将您的应用程序现代化迁移至 Linux 所需的人工工作量。如果您的应用程序支持 ARM 处理器，迁移到 Linux 便能解锁使用 Graviton 实例的能力。这将有助于您

再节省 20% 的成本。有关更多信息，请参阅《计算》博客中的[“使用 AWS Graviton2 为 .NET 5 提供支持：基准测试”](#) AWS。

还有一些其他工具，例如[适用于 .NET 重构的 AWS Toolkit](#) 和 [.NET 升级助手](#)，它们能够帮助您将遗留的 .NET Framework 应用程序移植到现代的 .NET。

成本优化建议

要迁移 .NET Framework 应用程序，请执行以下操作：

1. 先决条件 – 要使用 Porting Assistant for .NET，您必须在计划分析应用程序源代码的计算机上安装 .NET 5+ 版本。计算机上的资源必须至少具有 1.8 的 GHz 处理速度、4 GB 的内存和 5 Gb 的存储空间。有关更多信息，请参阅 Porting Assistant for .NET 文档中的[先决条件](#)。
2. 评测 – 下载 Porting Assistant for .NET 作为[可执行](#)（下载）文件。您可以下载并将该工具安装到您的计算机上，以开始对您的应用程序进行评测。评估页面包含移植的项目、软件包 APIs，它们与现代 .NET 不兼容。因此，在评测完成后，您在解决方案中会遇到构建错误。您可以查看评测调查发现或将其下载到 CSV 文件中。有关更多信息，请参阅 Porting Assistant for .NET 文档中的[移植解决方案](#)。
3. 重构 – 评测应用程序后，您可以将项目移植到目标框架版本。移植解决方案时，您的项目文件以及部分代码将会由 Porting Assistant 进行修改。您可以检查日志，以查看对您源代码所做的更改。在大多数情况下，要完成代码迁移和测试工作以使其达到可投入生产状态，还需要付出额外的努力。根据应用程序的不同，这些更改可能包括实体框架、身份和身份验证等。有关更多信息，请参阅 Porting Assistant for .NET 文档中的[移植解决方案](#)。

这是将您的应用程序现代化到容器的第一步。将您的 .NET Framework 应用程序现代化到 Linux 容器，可能有多种商业和技术方面的推动因素。其中一个重要的推动因素是通过从 Windows 操作系统转向 Linux 系统来降低总拥有成本。这在将您的应用程序迁移到跨平台的 .NET 版本以及迁移到容器中以优化资源利用时，能够降低许可费用。

将您的应用程序移植到 Linux 后，您可以使用 [AWS App2Container](#) 对应用程序进行容器化。App2Container 使用 Amazon ECS 或 Amazon EKS 作为您可以直接部署到的端点服务。App2Container 提供了所有必要的基础设施即代码（IaC）部署构件，以帮助您反复将应用程序容器化。

其他注意事项和资源

- 如果你有在 VB.NET (2002 年的传统框架) 上构建的应用程序并想将其移植到 .NET 6 , 请参阅博客上的 Microsoft Workloads [博客上的“使用适用于 .NET 的移植助手将传统 VB.NET 应用程序移植到 .NET 6.0”](#)。AWS
- 如果您在 Windows Communication Foundation (WCF) 上运行的是旧版应用程序 , 并且想在现代 .NET 上运行它们 , 您可以采用 CoreWCF。有关更多信息 , 请参阅 Microsoft [Workloads 博客上的“使用适用于 .NET 的移植助手将传统 WCF 应用程序现代化为 CoreWCF”](#) 的文章。AWS
- 您可以将移植助手作为扩展程序添加到您的 Visual Studio IDE 中。这使得您能够执行代码转换所需的所有任务 , 而无需在您的 IDE 和 Porting Assistant for .NET 工具之间来回切换。有关更多信息 , 请参阅微软工作负载博客上的 [“使用适用于 .NET 的移植助手 Visual Studio IDE 加速 .NET 应用程序现代化”](#) 扩展 AWS 文章。
- [AWS Porting Assistant for .NET 现在是一款开源工具](#) , 其中包含源代码以及评测的兼容性分析组件。这能够鼓励您的开发人员去使用和共享 .NET 移植知识和最佳实践。
- 您可以使用 .NET 重构 AWS 工具包将 .NET 框架应用程序移植到 Linux 上的现代 .NET 中。有关更多信息 , 请参阅 Microsoft [Workloads 博客上的“使用适用于 .NET 重构的 AWS 工具包加速 .NET 现代化”](#) AWS 一文。
- 您可以[加快 ASP.NET Core 应用程序的容器化以及向使用的迁移](#)。AWS AWS App2Container

容器化 .NET 应用程序

概述

容器是一种轻量且高效的方案 , 能够以一致且可重复的方式对应用程序进行打包和部署。本部分说明如何使用无服务器容器服务 AWS Fargate 来降低 .NET 应用程序的成本 , 同时提供可扩展且可靠的基础设施。

成本影响

影响使用容器节省成本的有效性的一些因素包括应用程序的规模和复杂程度、需要部署的应用程序数量 , 以及应用程序所承受的流量和需求水平。对于小型或简单的应用程序而言 , 与传统的 инфраструктура 方法相比 , 容器可能并不会带来显著的成本节省 , 因为管理容器及其相关服务所产生的开销实际上可能会增加成本。然而 , 对于规模较大或更为复杂的应用程序而言 , 使用容器能够通过提高资源利用率并减少所需实例的数量来实现成本节省。

我们建议您在容器节省成本时 , 务必牢记以下几点 :

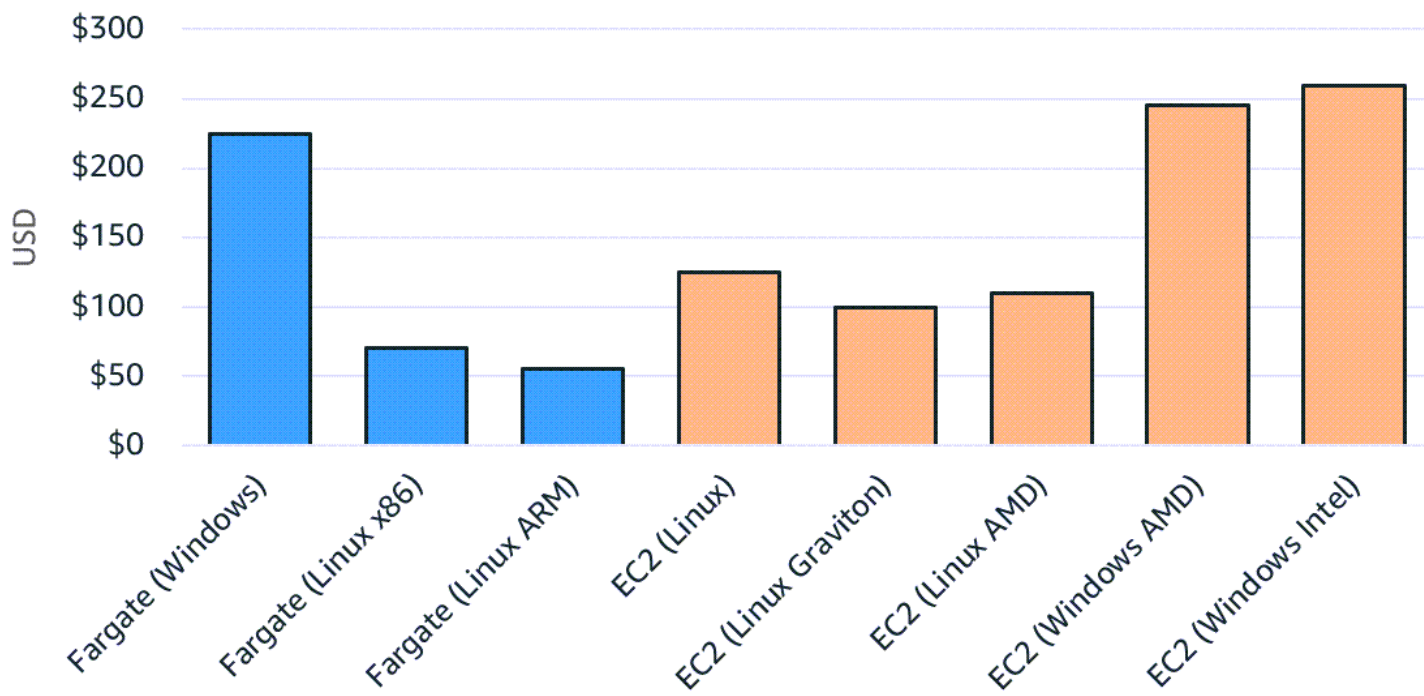
- 应用程序规模和复杂性 – 规模更大、更复杂的应用程序更适合容器化，因为这类应用程序通常需要更多的资源，并且能从资源利用效率的提升中获得更大的益处。
- 应用程序数量 – 您所在组织必须部署的应用程序越多，通过容器化实现的成本节省就可能越高。
- 流量和需求 – 那些流量大、需求高的应用程序能够从容器所提供的可扩展性和弹性中获益。这样可以节省成本。

不同的架构和操作系统会影响容器的成本。如果您使用的是 Windows 容器，那么由于许可方面的因素，成本可能不会降低。使用 Linux 容器时，许可费用更低或者根本就没有费用。下图使用了美国东部（俄亥俄州）地区的基本配置，其设置如下：AWS Fargate 在分配了 4 v CPUs 和 8 GB 内存的情况下，每月运行 30 个任务，每项任务持续 12 小时。

您可以选择两个主要的计算平台来运行您的容器 AWS：[基于 EC2 的容器主机和无服务器或](#)。[AWS Fargate](#)如果您使用 Amazon Elastic Container Service（Amazon ECS）而非 Fargate，则必须保持运行中的计算（实例），以便让放置引擎在需要时能够实例化容器。如果您改用 Fargate，则仅会预调配所需的计算容量。

下面的图表显示了等效容器使用 Fargate 与 Amazon EC2 产生的区别。由于 Fargate 具有灵活性，应用程序的任务每天可以运行 12 个小时，且在非工作时间不会占用资源。但是，对于 Amazon ECS，您必须使用 EC2 实例的[自动扩缩组](#)来控制计算容量。这可能会导致容量全天 24 小时运行，从而最终增加成本。

Monthly costs of Fargate and Amazon EC2



成本优化建议

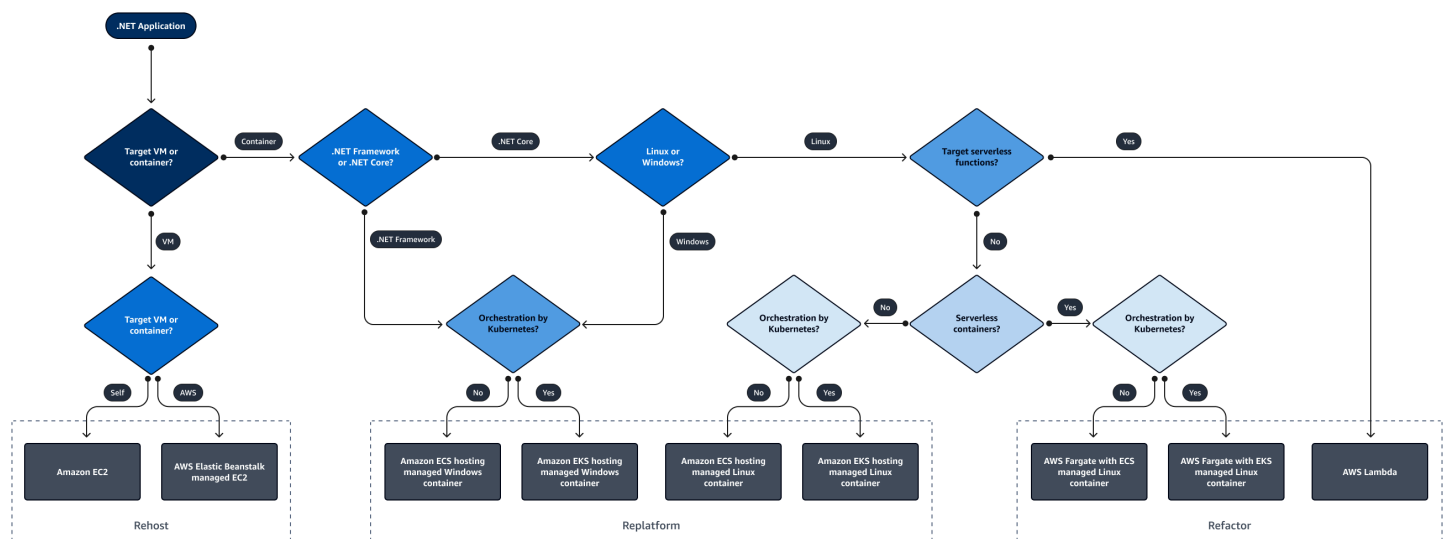
使用 Linux 容器而非 Windows

如果您使用 Linux 容器而非 Windows 容器，就能实现显著的节省。例如，如果您在 EC2 Linux 上运行 .NET Core 而非在 EC2 Windows 系统上运行 .NET Framework，那么在计算成本方面就能节省约 45% 的费用。如果您使用 ARM 架构 (AWS Graviton) 而不是 x86，则可以额外节省 40%。

如果您计划为现有的 .NET Framework 应用程序运行基于 Linux 的容器，则您必须将这些应用程序移植到现代的、跨平台的 .NET 版本 (例如 [.NET 6.0](#)) 中，这样才能使用 Linux 容器。一个重要的考虑因素是：权衡重构的成本与通过降低 Linux 容器的成本所获得的节省成本之间的关系。有关将应用程序移植到现代 .NET 的更多信息，请参阅 AWS 文档中的 [Porting Assistant for .NET](#)。

迁移到现代 .NET (即从 .NET Framework 中迁出) 的另一个好处是，可以获得更多的现代化机会。例如，您可以考虑对您的应用程序进行重新架构，使其采用基于微服务的架构，这样会更具可扩展性、敏捷性和成本效益。

以下图表展示了探索现代化机遇的决策流程。



利用节省计划

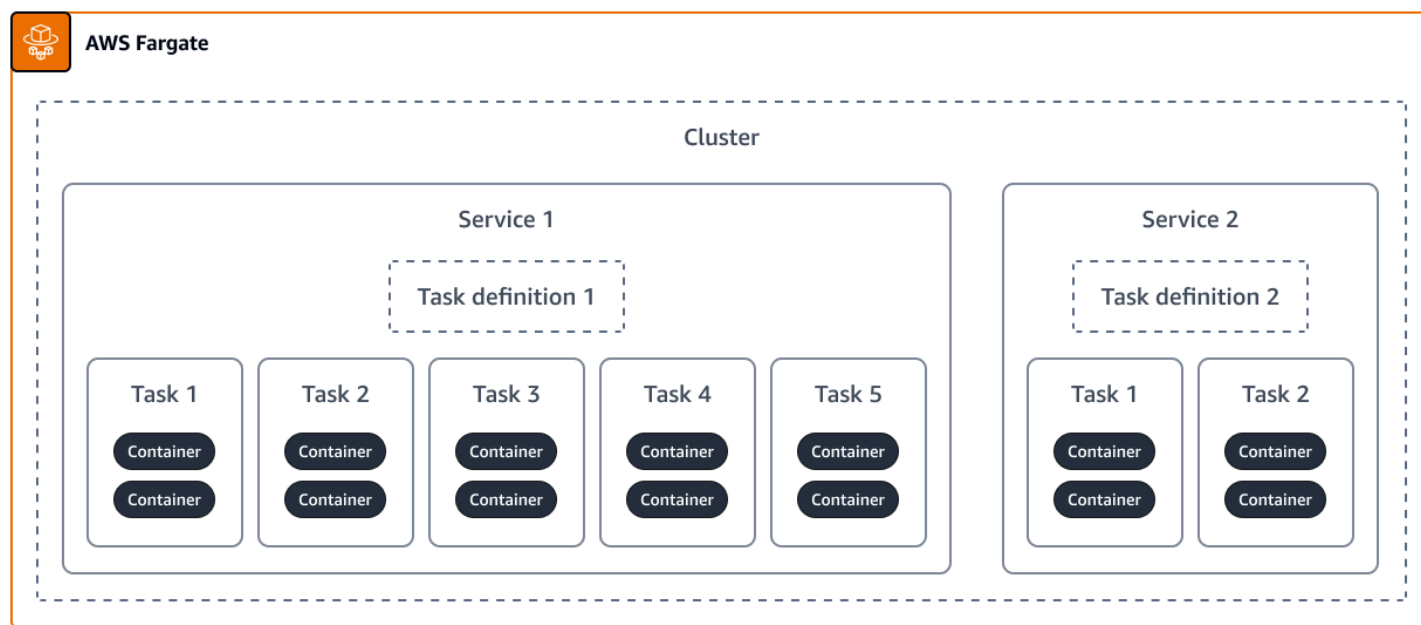
容器可以帮助您利用[计算类节省计划](#)来降低 Fargate 成本。这种灵活的折扣模式所提供的折扣与可转换预留实例的折扣相同。Fargate 定价是根据从您开始下载容器映像之时起直至 Amazon ECS 任务终止期间所使用的 vCPU 和内存资源计算得出 (向上取整到最接近的一秒)。 [Fargate 的节省计划](#) 可为用户节省最多 50% 的 Fargate 使用费用，条件是用户需承诺在一年或三年的时间内使用特定数量的计算资源 (以每小时美元计价)。您可以使用 [AWS Cost Explorer](#) 来帮助您选择节省计划。

需要明白的是，计算类节省计划会优先应用于能为您带来最高节省的用量。例如，如果您在 us-east-2 中运行一个 t3.medium Linux 实例以及一个相同的 Windows t3.medium 实例，那么 Linux 实例会首先享受到节省计划的益处。这是因为 Linux 实例的节省潜力为 50%，而相同的 Windows 实例的节省潜力则为 35%。如果您有其他符合储蓄计划条件的资源 AWS 账户，例如亚马逊 EC2 或 Lambda，则无需先将您的储蓄计划应用于 Fargate。有关更多信息，请参阅本指南的[储蓄计划文档中的了解储蓄计划如何适用于您的 AWS 使用量](#)，以及本指南的“[优化 Windows 在 Amazon EC2 上的支出](#)”部分。

调整 Fargate 任务的大小

重要的是确保 Fargate 任务的大小正确，以实现最大程度的成本优化。通常情况下，在最初为应用程序中使用的 Fargate 任务确定配置时，开发人员并不具备所有必要的使用信息。这可能会导致任务过度预调配，进而造成不必要的开支。为避免这种情况，我们建议您对 Fargate 上运行的应用程序进行负载测试，以了解特定任务配置在不同使用场景下的表现情况。您可以利用负载测试结果、任务的 vCPU 和内存分配情况以及自动扩缩策略，来找到性能与成本之间的恰当平衡。

以下图表显示了 Compute Optimizer 如何为最佳任务和容器大小生成建议。



一种方法是使用负载测试工具（如[上 AWS 的“分布式负载测试”](#)中所述的工具）来建立 vCPU 和内存利用率基准。在您运行负载测试以模拟典型的应用程序负载之后，您就可以对任务的 vCPU 和内存配置进行微调，以满足基准利用率要求。

其他资源

- [Amazon ECS 和 AWS Fargate 的成本优化检查清单](#) (AWS 容器博客文章)
- [Amazon ECS 的理论成本优化启动类型 : Fargate vs EC2](#) (AWS 容器博客文章)
- [.NET 移植助手](#) (AWS 文档)
- [分布式负载测试 AWS](#) (AWS 解决方案库)
- [AWS Compute Optimizer 在 AWS Fargate \(AWS 云财务管理博客文章 \) 上推出对 Amazon ECS 服务的支持](#)

使用 Graviton 实例和容器

概述

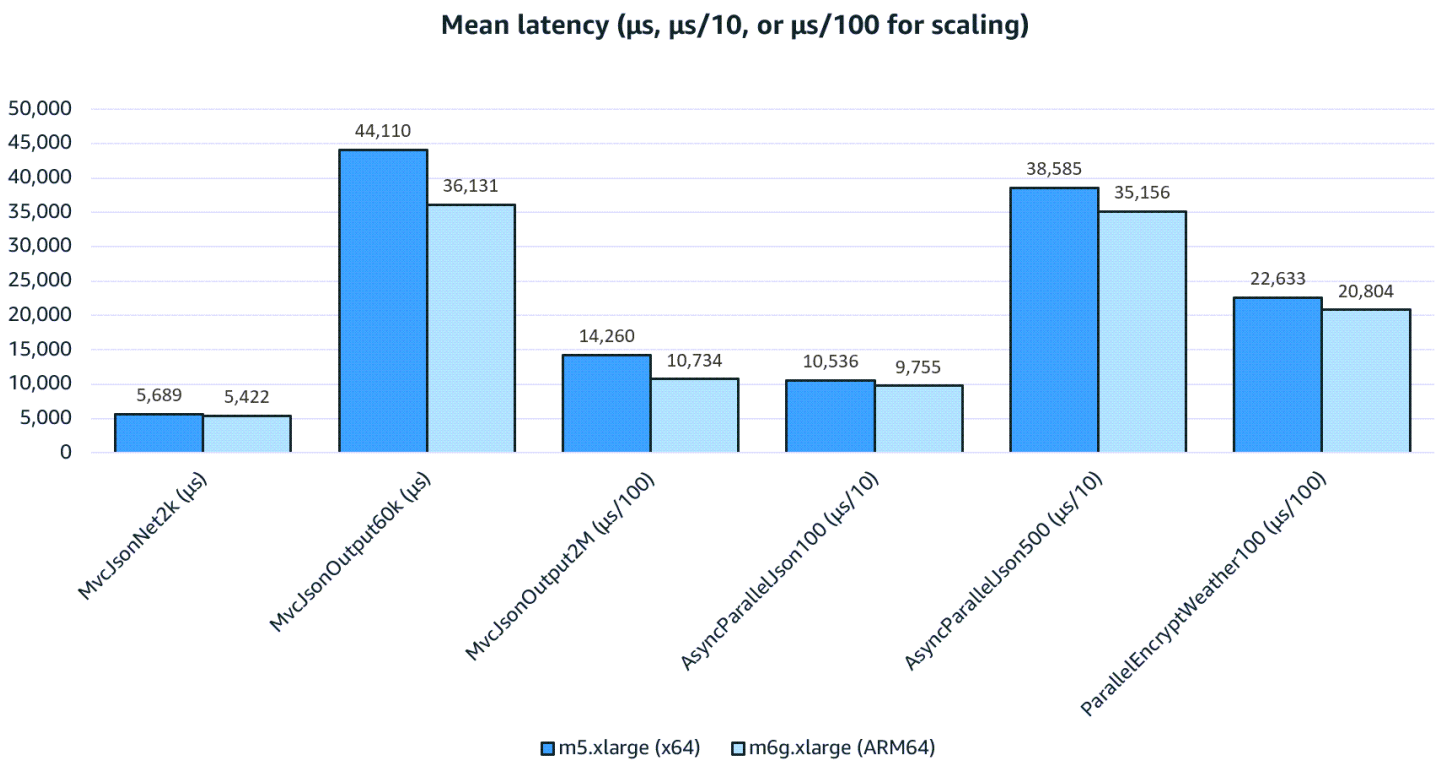
AWS Graviton 实例由 ARM 处理器提供支持，旨在为在亚马逊弹性计算云 (Amazon EC2) 中运行的云工作负载 (包括在其中运行的容器) 提供最佳的性价比。AWS 目前，Amazon EC2 上有三代 Graviton 可供使用。本指南重点介绍如何将 Graviton 2 和 3 与 .NET 应用程序配合使用，因为使用 Graviton 的最新版本能够显著节省成本。请记住，Graviton 实例仅运行 Linux 操作系统。因此，Graviton 实例对于在 Linux 上运行的 .NET 应用程序来说是一个非常出色的服务，但对于 Windows 操作系统或遗留的 .NET Framework 应用程序来说则并非可行之选。

Graviton 3 的效率比同类的 EC2 实例高出 60%，性能则提升了最多 40%。本指南重点介绍了使用 Graviton 所带来的成本效益，但需要指出的是，Graviton 还具有提升性能和增强环境可持续性额外优势。

成本影响

当您切换为使用 Graviton 时，最多可以节省 45% 的成本。在将任何遗留的 .NET Framework 应用程序重构到现代的 .NET 版本之后，您便能够使解锁使用 Graviton 实例的能力。迁移到 Graviton 是一种适用于 .NET 开发者的高效成本优化手段。

下表中的示例展示了通过迁移到 Graviton 实例所能实现的性能改进潜力。



有关上图中用于创建结果的基准测试方法的完整细分和解释，请参阅 Compute 博客中的“使用 [AWS Graviton2 为.NET 5 提供支持：基准](#)”。AWS

效率提升的原因之一在于 x86 和 Graviton 中的 vCPU 意义存在差异。在 x86 架构中，vCPU 是通过超线程技术实现的一个逻辑内核。在 Graviton 中，vCPU 等同于一个物理内核，这使得 vCPU 能够完全投入到工作负载中。

使用 Graviton2 的结果是，其性价比比同类的 x86/x64 实例高出 40%。Graviton3 相较于 Graviton2 具有以下优势：

- 性能得到提升，提高达 25% 以上
- 最高比浮点性能提高两倍
- 最高比加密工作负载性能快两倍
- 最高比机器学习性能提高三倍

此外，Graviton3是云中第一个以内存为特色的实例。DDR5

下表显示了基于 Graviton 的实例与同等的基于 x86 的实例在成本节省方面的差异。

此表显示 Graviton 节省了 19.20% 的成本。

实例类型	架构	vCPU	内存 (GB)	每小时成本 (按需)
t4g.xlarge	ARM	4	16	0.1344 美元
t3.xlarge	x86	4	16	0.1664 美元

此表显示 Graviton 节省了 14.99% 的成本。

实例类型	架构	vCPU	内存 (GB)	每小时成本 (按需)
c7g.4xlarge	ARM	16	32	0.5781 美元
c6i.4xlarge	x86	16	32	0.6800 美元

在考虑 Graviton 时，测试您的应用程序的性能表现非常重要。Graviton 不能取代扎实的软件开发流程。您可以通过测试来验证是否充分利用了底层计算资源。

成本优化建议

有几种方法可以利用 Graviton 处理器/实例。本部分将指导您了解从使用 x86 架构的计算机转换到 Graviton (ARM) 实例所需的各项更改。

更改 Lambda 中的运行时设置

我们建议您在中切换运行时设置 AWS Lambda。有关更多信息，请参阅 Lambda 文档中的[修改运行时环境](#)。由于 .NET 是一种编译语言，因此您必须遵循一套构建流程才能使其正常运行。有关如何执行此操作的示例，请参阅。 [.NET 在 Graviton 上线](#) GitHub

容器

对于容器化工作负载，请创建多架构容器映像。您可以通过在 Docker 构建命令中指定多个架构来实现这一点。例如：

```
docker buildx build -t "myImageName:latest" --platform linux/amd64,linux/arm64 --push .
```

您也可以使用诸如之类的工具 AWS Cloud Development Kit (AWS CDK) 来帮助[编排构建](#)。有关 Docker 的示例，请参阅 Docker 文档中的[使用 Docker 桌面为 Arm 和 x86 构建多架构映像](#)。

Amazon EC2

要从 x86/x64 迁移到 ARM，请在编译步骤中以 ARM 架构为目标。在 Visual Studio 中，您可以创建一个 ARM64 CPU。有关说明，请参阅 Microsoft 文档中的[将项目配置为以 Arm64 和其他平台为目标](#)。

如果您使用的是 .NET CLI，那么在 ARM 计算机上运行该版本会生成与 Graviton 兼容的版本。要观看演示，请在 Gr [AWS aviton2 上开启 Arm64 的情况下观看“加速.NET 6 性能”](#)。YouTube 依赖项问题会导致编译时间错误，而这些错误可以逐一进行解决。只要存在适用于任何依赖项的 ARM 库，那么转换过程应该会相对简单顺利。

其他资源

- [如何在 Amazon ECS 上为 ARM 构建容器并使用 Graviton 和 Spot 实例进行保存 \(博客 \)](#) AWS
- [AWS Lambda 由 AWS Graviton2 处理器提供支持的功能 — 在 Arm 上运行您的功能，性价比最高可提高 34% \(博客 \)](#) AWS
- [将 AWS Lambda 函数迁移到基于 ARM 的 AWS Graviton2 处理器 \(博客 \)](#) AWS
- [使用 \(博客 \) 在 ARM 支持的 AWS Graviton 2 Amazon ECS 集群 AWS CDK 上构建和部署 .NET Web 应用程序](#) AWS
- [Graviton Fast Start — 一项帮助将工作负载转移到 AWS Graviton 的新计划 \(博客 \)](#) AWS
- [使用 AWS Graviton2 为 .NET 5 提供动力：基准测试 \(博客 \)](#) AWS

支持对静态 .NET Framework 应用程序进行动态扩展

概述

将云用于应用程序的一大关键优势在于其弹性，即可以根据需求灵活地横向扩展或缩减计算资源的使用量。这使得您只需为实际所需的计算容量付费，而无需为峰值使用量进行预调配。“网络星期一”是一个很好的例子，说明了弹性经济的概念。在这个特殊日子里，线上零售商能够获得比平时高出数倍甚至更多的流量（例如，[短短几分钟内达到数千个百分点的增幅](#)）。

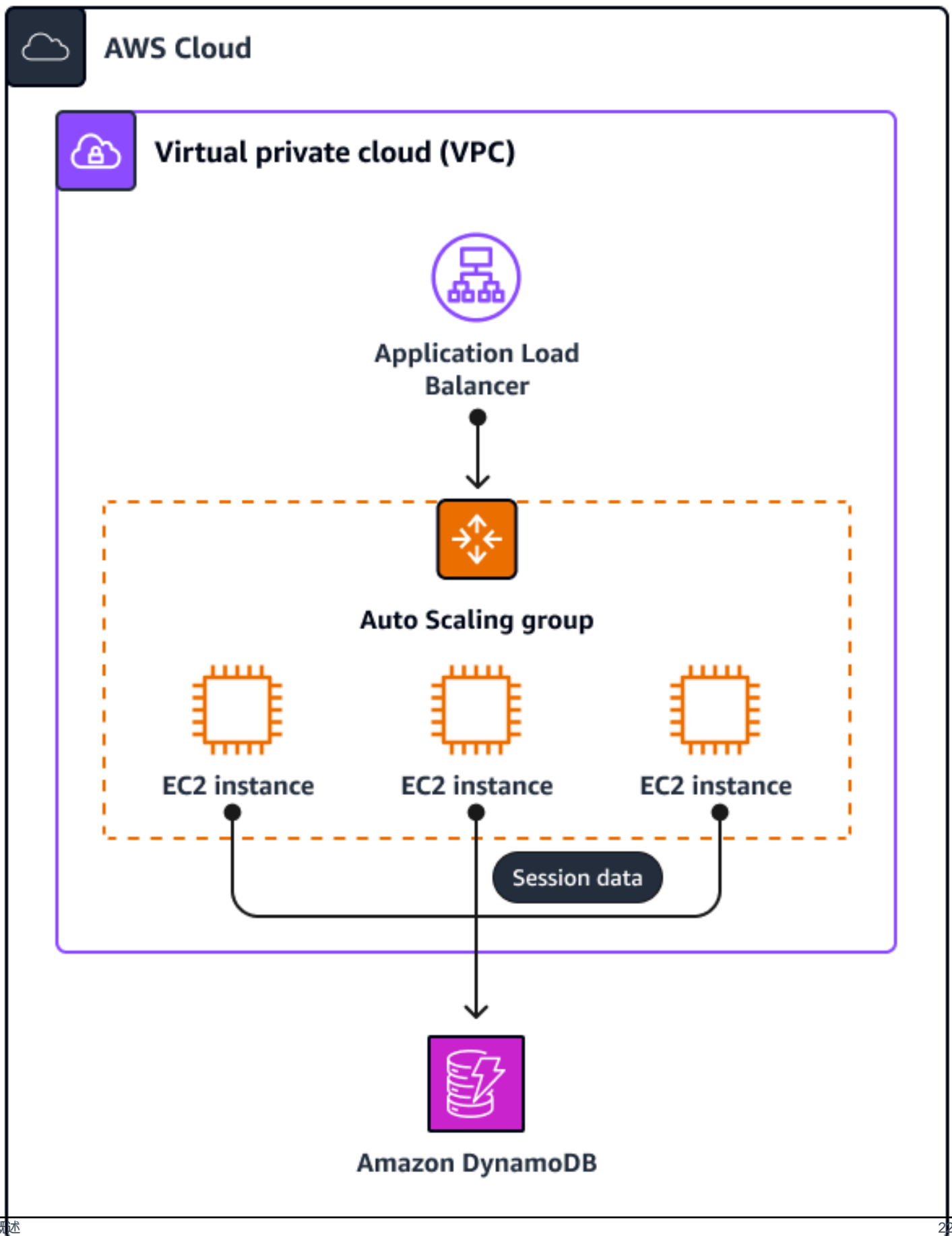
如果您要将遗留的 .NET Web 应用程序迁移到云中（例如，在 IIS 上运行的 ASP.NET Framework 应用程序），由于应用程序的有状态特性，可能难以或无法快速扩展负载均衡的服务器场。用户会话数据存储在应用程序的内存中，通常与保留必须持久保存的跨请求数据的 [ASP.NET 会话状态](#) 或静态变量一起存储。用户会话亲和性通常通过负载均衡器的粘性会话来保持。

事实证明，这在操作上具有挑战性。当需要增加容量时，您必须有意预调配并添加服务器。这可能是一个缓慢的过程。在进行补丁安装或出现意外故障时，若要将节点停用，可能会对最终用户体验造成影响，从而使所有与受影响节点相关联的用户都将丢失状态。最理想的情况是，这只需要用户再次登录即可。

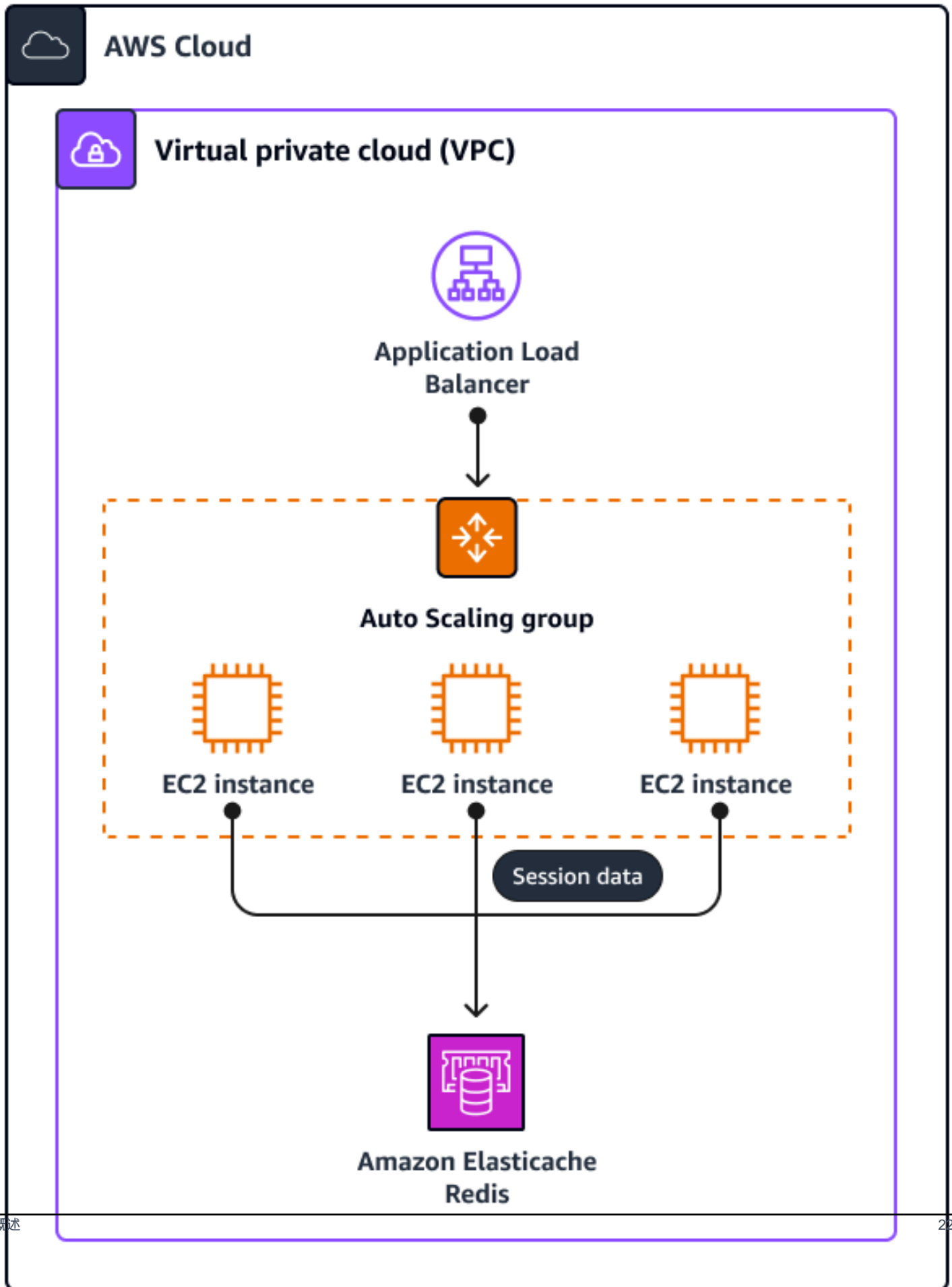
通过为 ASP.NET 应用程序集中管理会话状态，并对遗留的 ASP.NET 应用程序应用自动扩展规则，您可以充分利用云的弹性，并在运行应用程序时可能节省成本。例如，通过计算资源可扩展性，您可以降低成本，同时您还可以选择不同的定价模式，比如减少[预留实例的使用量](#)并采用 [Amazon 竞价型实例定价](#)。

两种常见的技术包括使用[亚马逊 DynamoDB 作为会话状态提供](#)程序和使用 [ElastiCache 亚马逊 \(Redis OSS \) 作为 ASP.NET 会话存储](#)。

下图显示了将 DynamoDB 用作会话状态提供者的架构。



下图显示了使用 ElastiCache (Redis OSS) 作为会话状态提供程序的架构。

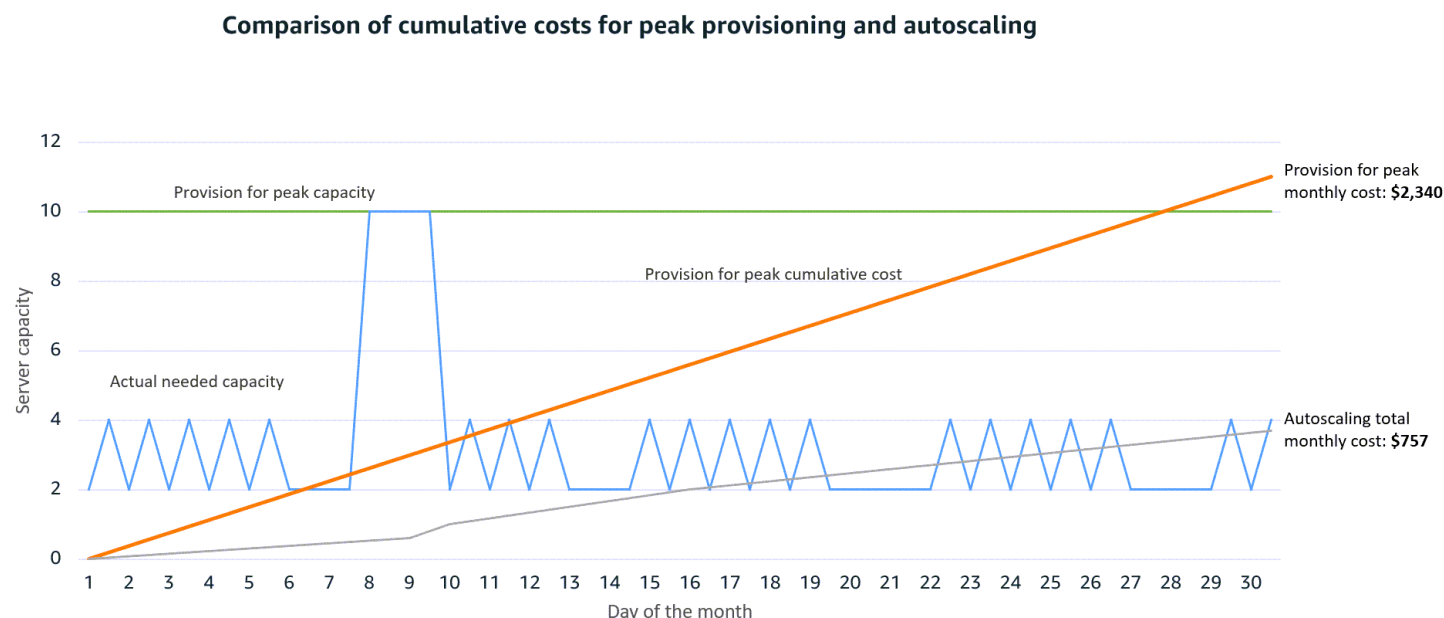


成本影响

要确定扩展为生产应用程序带来的益处，我们建议您对实际需求进行建模。本部分基于以下假设来构建一个示例应用程序模型：

- 添加和从旋转操作中移除的实例是完全相同的，并且不会带来实例大小的变化。
- 服务器的利用率始终不会低于两台活动服务器，以维持应用程序的高可用性。
- 服务器的数量会与流量呈线性关系扩展（也就是说，流量增加一倍，所需的计算能力也会增加一倍）。
- 以六小时为增量对一个月的流量进行建模，其中包含日内差异以及一天中出现的一次异常的流量高峰（例如，促销活动期间的流量会达到平时的 10 倍）。基于基础利用率对周末流量进行建模。
- 夜间流量按照基础利用率进行建模，而工作日的流量则按照 4 倍的利用率进行建模。
- 预留实例定价采用为期一年的无预付定价。普通的日间定价采用按需定价模式，而爆发需求则采用竞价型实例定价模式。

下图说明了此模型如何在 .NET 应用程序中充分利用弹性特性，而非为峰值使用量进行预调配。这样可以节省大约 68% 的成本。



如果您使用 DynamoDB 作为会话状态存储机制，请使用以下参数：

Storage: 20GB
Session Reads: 40 million
Session Writes: 20 million

```
Pricing Model: On demand
```

这项服务的每月预计费用约为 35.00 美元。

如果您使用 ElastiCache (Redis OSS) 作为会话状态存储机制，请使用以下参数：

```
Number of Nodes: 3
Node size: cache.t4g.medium
Pricing Model: 1y reserved
```

这项服务的每月预计费用约为 91.00 美元。

成本优化建议

第一步是在遗留的 .NET 应用程序中实施会话状态管理。如果您使用 ElastiCache 作为状态存储机制，请按照 AWS 开发者工具博客中[ElastiCache 作为 ASP.NET 会话存储](#)中的指导进行操作。如果您使用的是 DynamoDB，请按照文档中的[内容中的](#)指导进行操作 适用于 .NET 的 AWS SDK。适用于 .NET 的 SDK

如果应用程序一开始就使用 InProc 会话，请确保计划存储在会话中的所有对象都可以序列化。为执行此操作，请使用 `SerializableAttribute` 属性来修饰那些实例将被存储在会话中的类。例如：

```
[Serializable()]
public class TestSimpleObject {
    public string SessionProperty {get;set;}
}
```

此外，所有正在使用的服务器上的 .NET MachineKey 必须保持一致。通常情况下，如果实例是基于通用的亚马逊机器映像 (AMI) 创建的，就会出现这种情况。例如：

```
<machineKey
validationKey="some long hashed value"
decryptionKey="another long hashed value"
validation="SHA1"/>
```

然而，重要的是确保在更改基础映像时，其配置应与相同的 .NET 机器映像保持一致（这种配置可以在 IIS 或服务器级别进行设置）。有关更多信息，请参阅[SystemWebSectionGroup. MachineKey 微软文档中的属性](#)。

最后，您必须确定在响应扩展事件时向自动扩缩组添加服务器的机制。我们可以通过多种方式来完成此操作。我们建议使用以下方法将 .NET Framework 应用程序无缝部署到自动扩缩组中的 EC2 实例中：

- 使用 [EC2 Image Builder](#) 配置包含完全配置的服务器和应用程序的 AMI。然后，您可以使用此 AMI 来配置 [自动扩缩组的启动模板](#)。
- [AWS CodeDeploy](#) 用于部署您的应用程序。CodeDeploy 允许直接与 [Amazon EC2 Auto Scaling](#) 集成。这为针对应用程序的每个版本创建新 AMI 提供了另一种选择。

其他资源

- [使用 EC2 Image Builder 创建映像](#) (EC2 Image Builder 文档)
- [使用 AWS CodeDeploy Visual Studio 团队服务部署 .NET Web 应用程序](#) (AWS 开发者工具博客)

利用缓存来降低数据库需求

概述

您可以将缓存作为一种有效的策略来用于降低您的 .NET 应用程序的成本。许多应用程序在需要频繁访问数据时会使用后端数据库，比如 SQL Server。维护这些后端服务以满足需求的成本可能会很高，但您可以采用有效的缓存策略，通过降低其规模和扩展需要来减轻后端数据库的负载。这能够帮助您降低成本并提升应用程序的性能。

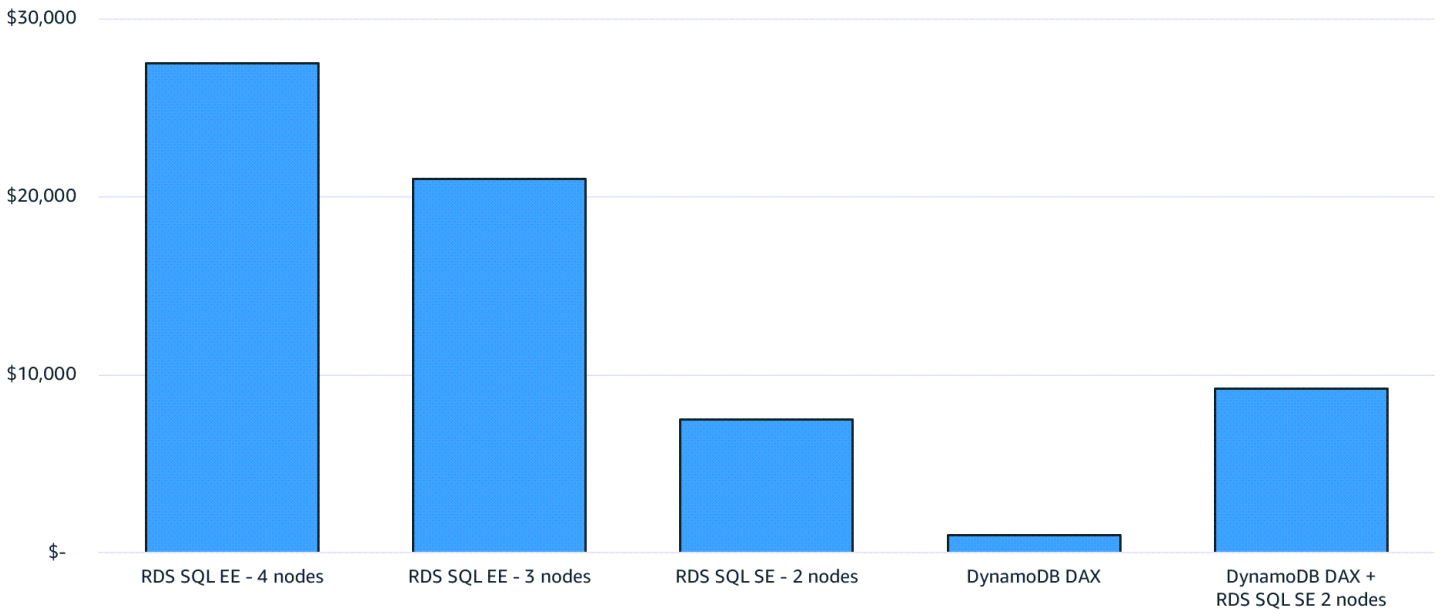
缓存是一种非常有效的技术，能够节省因执行读取密集型工作负载 (这类工作负载会消耗更多昂贵的资源，比如 SQL Server) 而产生的相关成本。为您的工作负载采用正确的方法很重要。例如，本地缓存不具备可扩展性，而且需要您为每个应用程序实例维护一个本地缓存。您应当权衡其对性能影响与潜在成本之间的关系，这样底层数据来源的较低成本就能抵消与缓存机制相关的任何额外成本。

成本影响

SQL Server 要求您在确定数据库大小时将读取请求的因素考虑在内。这可能会对成本产生影响，因为您可能需要引入只读副本以承受负载。如果您正在使用只读副本，请务必清楚，这种功能仅在 SQL Server 企业版中可用。此版本所需的许可证费用比 SQL Server 标准版更高。

以下图表旨在帮助您了解缓存效果。它展示了 Amazon RDS for SQL Server，其中包含四个运行 SQL Server 企业版的 db.m4.2xlarge 节点。它采用了具有一个只读副本的多可用区配置。专属读取流量 (例如，SELECT 查询) 会被定向到只读副本。相比之下，Amazon DynamoDB 使用的是一个 r4.2xlarge 双节点 DynamoDB Accelerator (DAX) 集群。

下图显示了取消使用专门负责处理高读取流量的只读副本这一需求后的结果。



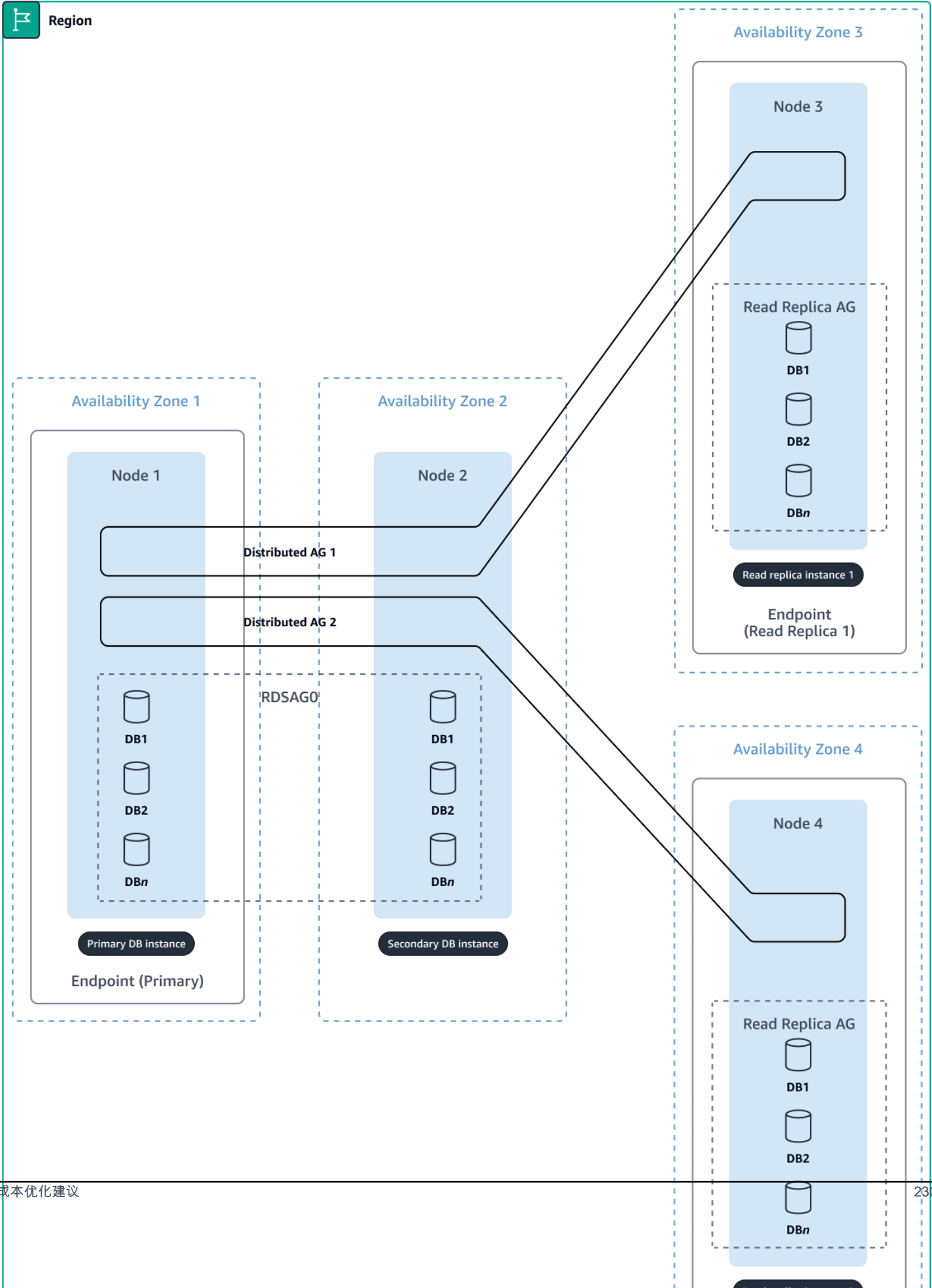
通过使用本地缓存（无需只读副本）或者在 Amazon RDS 上将 DAX 与 SQL Server 并行配置为缓存层，您可以实现显著的成本节省。该层可将任务从 SQL Server 上移除，并减小运行数据库所需的 SQL Server 的规模。

成本优化建议

本地缓存

本地缓存是用于为部署在本地环境或部署在云中的应用程序缓存内容的最常用方式之一。这是因为它的实施相对简单且直观。本地缓存是指从数据库或其他来源获取内容，然后将其在内存或磁盘上进行本地缓存，以便于更快地访问。这种方法虽然易于实施，但对于某些使用案例而言却并非理想选择。例如，这包括那些需要使缓存内容长期保持不变的使用案例，比如保存应用程序状态或用户状态。另一种使用案例是：当需要从其他应用程序实例中访问缓存内容时。

下面的图表展示了一个由四个节点和两个只读副本组成的高可用性 SQL Server 集群。



使用本地缓存时，您可能需要将流量在多个 EC2 实例之间进行负载均衡。每个实例都必须维护自己的本地缓存。如果缓存中存储的是有状态信息，则需要定期将数据提交到数据库中，并且用户在每个后续请求中可能都需要被转发到同一个实例（即粘性会话）。这在尝试扩展应用程序时就带来了一种挑战，因为有些实例可能会被过度使用，而有些则由于流量分布不均而未得到充分利用。

您可以为 .NET 应用程序使用本地缓存，无论是内存缓存还是使用本地存储。为此，您可以添加功能，以便将对象存储在磁盘上并在需要时进行检索；或从数据库中查询数据并将其持久保存在内存中。例如，要在 C# 中对来自 SQL Server 的数据进行本地内存缓存和本地存储操作，您可以使用 MemoryCache 和 LiteDB 这两个库的组合。MemoryCache 提供内存缓存功能，而 LiteDB 则是一个嵌入式的基于磁盘的 NoSQL 数据库，速度快且轻量级。

要执行内存缓存，请使用 .NET 库 System.Runtime.MemoryCache。以下代码示例演示了如何使用 System.Runtime.Caching.MemoryCache 类将数据缓存在内存中。该类提供了一种在应用程序内存中临时存储数据的方法。这有助于提高应用程序的性能，因为它减少了从更昂贵的资源（如数据库或 API）获取数据的需求。

以下是该代码的工作原理：

1. 创建一个名为 `_memoryCache` 的 MemoryCache 私有静态实例。为缓存赋予一个名称（`dataCache`）来标识它。然后，缓存存储和检索数据。
2. `GetData` 方法是一个通用方法，它有两个参数：一个 `string` 键和一个名为 `getData` 的 `Func<T>` 委派。键用于标识缓存的数据，而 `getData` 委派则代表了在数据未存在于缓存中时将执行的数据检索逻辑。
3. 该方法首先通过 `_memoryCache.Contains(key)` 方法来检查数据是否存在于缓存中。如果数据在缓存中，该方法便会通过使用 `_memoryCache.Get(key)` 来获取数据，并将其转换为预期的类型 `T`。
4. 如果数据不在缓存中，则该方法会调用 `getData` 委派来获取数据。然后，它使用 `_memoryCache.Add(key, data, DateTimeOffset.Now.AddMinutes(10))` 将数据添加到缓存中。此调用规定，缓存条目应在 10 分钟后过期，届时数据将自动从缓存中移除。
5. `ClearCache` 方法将 `string` 键作为参数，并使用 `_memoryCache.Remove(key)` 从缓存中移除与该键关联的数据。

```
using System;
using System.Runtime.Caching;

public class InMemoryCache
{
    private static MemoryCache _memoryCache = new MemoryCache("dataCache");
```

```
public static T GetData<T>(string key, Func<T> getData)
{
    if (_memoryCache.Contains(key))
    {
        return (T)_memoryCache.Get(key);
    }

    T data = getData();
    _memoryCache.Add(key, data, DateTimeOffset.Now.AddMinutes(10));

    return data;
}

public static void ClearCache(string key)
{
    _memoryCache.Remove(key);
}
}
```

您可以使用以下代码：

```
public class Program
{
    public static void Main()
    {
        string cacheKey = "sample_data";

        Func<string> getSampleData = () =>
        {
            // Replace this with your data retrieval logic
            return "Sample data";
        };

        string data = InMemoryCache.GetData(cacheKey, getSampleData);
        Console.WriteLine("Data: " + data);
    }
}
```

以下示例向您展示如何使用 [LiteDB](#) 在本地存储中缓存数据。您可以使用 LiteDB 作为内存缓存的替代或补充。以下代码演示了如何使用 LiteDB 库在本地存储中缓存数据。LocalStorageCache 类包含用于管理缓存的主要函数。

```
using System;
using LiteDB;

public class LocalStorageCache
{
    private static string _liteDbPath = @"Filename=LocalCache.db";

    public static T GetData<T>(string key, Func<T> getData)
    {
        using (var db = new LiteDatabase(_liteDbPath))
        {
            var collection = db.GetCollection<T>("cache");
            var item = collection.FindOne(Query.EQ("_id", key));

            if (item != null)
            {
                return item;
            }
        }

        T data = getData();

        using (var db = new LiteDatabase(_liteDbPath))
        {
            var collection = db.GetCollection<T>("cache");
            collection.Upsert(new BsonValue(key), data);
        }

        return data;
    }

    public static void ClearCache(string key)
    {
        using (var db = new LiteDatabase(_liteDbPath))
        {
            var collection = db.GetCollection("cache");
            collection.Delete(key);
        }
    }
}

public class Program
{
```

```
public static void Main()
{
    string cacheKey = "sample_data";

    Func<string> getSampleData = () =>
    {
        // Replace this with your data retrieval logic
        return "Sample data";
    };

    string data = LocalStorageCache.GetData(cacheKey, getSampleData);
    Console.WriteLine("Data: " + data);
}
}
```

如果您有不会频繁变化的静态缓存或静态文件，也可以将这些文件存储在 Amazon Simple Storage Service (Amazon S3) 的对象存储中。该应用程序可以在启动时获取静态缓存文件，并在本地使用该文件。有关如何使用 .NET 从 Amazon S3 检索文件的更多详细信息，请参阅 Amazon S3 文档中的[下载对象](#)。

使用 DAX 进行缓存

您可以使用一个可跨所有应用程序实例共享的缓存层。[DynamoDB Accelerator \(DAX \)](#) 是一款完全托管、高度可用的内存缓存服务，专为 DynamoDB 设计，能够实现十倍的性能提升。您可以使用 DAX 来降低成本，方法是减少在 DynamoDB 表中过度预调配读取容量单位的需求。这对于读取密集型且需要针对每个键进行重复读取的工作负载来说尤其有用。

DynamoDB 的定价方式为按需定价或按预调配容量计费，因此每月的读写次数会直接影响费用。如果您有读取密集型工作负载，DAX 集群可以通过减少 DynamoDB 表的读取次数，从而帮助降低成本。有关如何设置 DAX 的说明，请参阅 DynamoDB 文档中的[使用 DynamoDB Accelerator \(DAX \) 进行内存加速](#)。有关 .NET 应用程序集成的信息，请观看[将亚马逊 DynamoDB DAX 集成到您的 ASP.NET 应用程序中](#)。YouTube

其他资源

- [利用 DynamoDB Accelerator \(DAX \) 进行内存加速 - Amazon DynamoDB](#) (DynamoDB 文档)
- [将亚马逊 DynamoDB DAX 集成到你的 ASP.NET 应用程序中](#) () YouTube
- [下载对象](#) (Amazon S3 文档)

考虑使用无服务器 .NET

概述

无服务器计算已成为构建和部署应用程序的一种常用方式。这主要是因为无服务器方法在构建现代架构时所提供的可扩展性和敏捷性。但是，在某些情况下，考虑无服务器计算的成本影响也是很重要的。

Lambda 是一个无服务器计算平台，它能让开发者无需使用专用服务器即可运行代码。对于那些希望降低基础设施成本的 .NET 开发人员来说，Lambda 是一个极具吸引力的选择。借助 Lambda，.NET 开发人员能够开发并部署具有高度可扩展性且可能具有成本效益的应用程序。通过采用无服务器方法，开发人员不再需要为处理应用程序请求而预调配服务器。相反，开发人员可以创建那些按需执行的函数。这使得无服务器架构在可扩展性、可管理性以及潜在的成本效益方面都优于运行、管理和扩展虚拟机的方式。因此，您只需为应用程序所使用的资源付费，而无需担心资源未充分利用的情况或服务器维护费用。

开发人员可以使用现代的跨平台 .NET 版本来构建运行速度快、效率高且成本效益好的无服务器应用程序。.NET Core 及其更新版本是一个免费且开源的框架，相较于之前的 .NET Framework 版本，它更适合在无服务器平台上运行。这使得开发人员能够缩短开发时间并提高应用程序的性能。现代 .NET 还支持一系列编程语言，包括 C# 和 F#。因此，对于那些希望在云中构建现代化架构的开发者来说，这是一个颇具吸引力的选择。

本部分介绍如何通过将 Lambda 作为无服务器选项来实现成本节省。您可以通过微调 Lambda 函数的执行配置文件、合理调整 Lambda 函数的内存分配、采用[本机 AOT](#)并迁移到基于 Graviton 的函数等方式进一步优化成本。

成本影响

您可以降低多少成本取决于多个因素，包括您的无服务器函数将执行多少次以及分配的内存量和每个函数的持续时间。AWS Lambda 提供免费套餐，包括每月 100 万个免费请求和每月 400,000 GB 秒的计算时间。您可以显著降低超出或接近这些免费套餐限制范围的工作负载所产生的每月费用。

使用以 Lambda 函数为目标的负载均衡器时，可能还会产生额外费用。这是根据负载均衡器针对[Lambda 目标](#)处理的数据量计算得出的。

成本优化建议

调整您的 Lambda 函数的大小

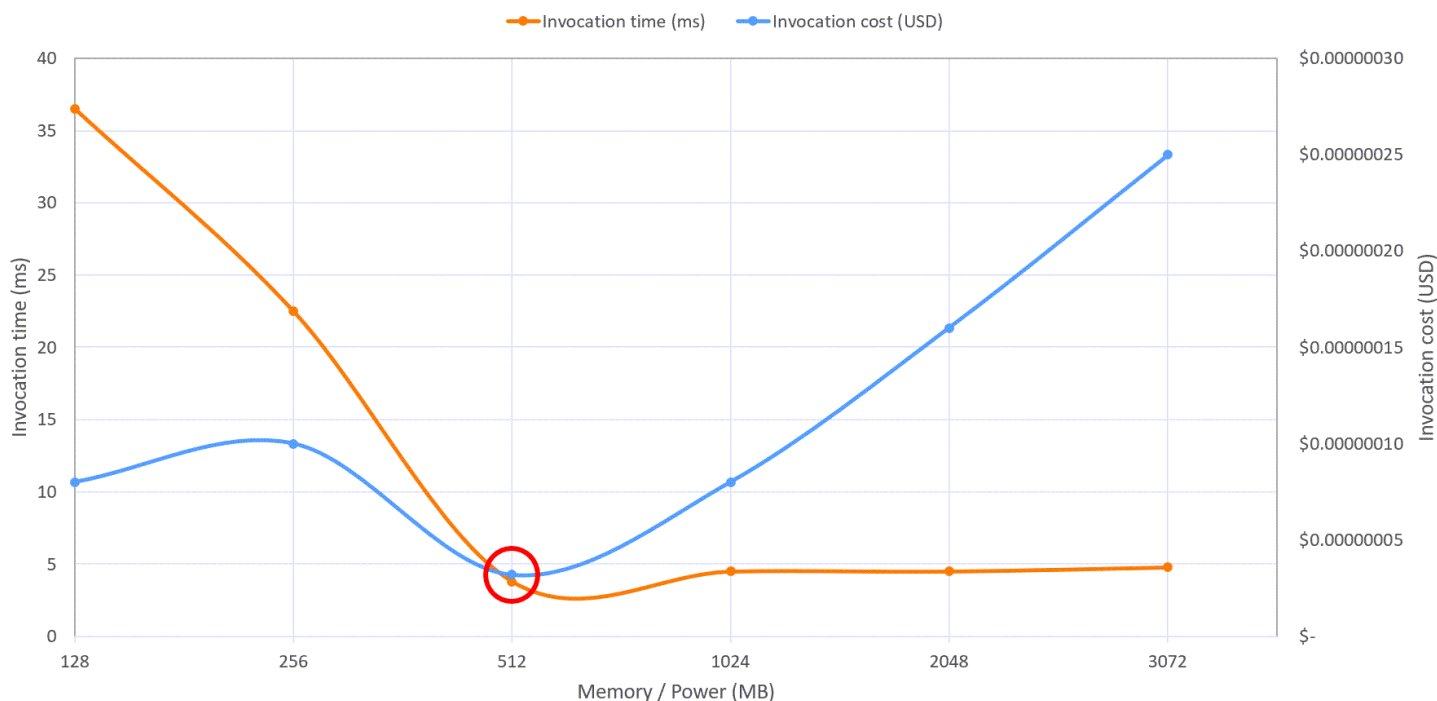
合理调整大小是实现基于 .NET 的 Lambda 函数成本优化的一项关键措施。这一过程包括确定既能保证性能又能兼顾成本效益的最优内存配置，且无需对代码进行任何更改。

通过为 Lambda 函数配置内存（范围从 128MB 到 10240MB 不等），您还可以调整调用时可用的 vCPU 数量。这使得内存或 CPU 密集型应用程序在执行过程中能够获取更多资源，从而有可能缩短调用持续时间并降低总体成本。

但是，确定适用于您基于 .NET 的 Lambda 函数的最佳配置可能是一个需要人工操作且耗时的过程，尤其是在更改频繁的情况下。[AWS Lambda Power Tuning 工具](#)能够通过将一组内存配置与示例有效载荷进行对比分析，帮助您确定合适的配置。

例如，为基于 .NET 的 Lambda 函数增加内存可以缩短总调用时间并降低成本，同时不会影响性能。函数的最佳内存配置可能有所不同。AWS Lambda Power Tuning 工具可以帮助确定每项功能的最具成本效益的配置。

在下面的示例图表中，对于此 Lambda 函数而言，总调用时间随着内存的增加而缩短。这可以降低总执行成本，同时又不影响该函数的原有性能。对于此函数，其最佳内存配置为 512MB，因为在此配置下，每次调用的资源利用率在总成本方面最为高效。这因函数而异，而通过在您的 Lambda 函数中使用该工具，可以确定这些函数受益于大小调整。



我们建议您定期完成此项操作，将其作为发布新更新时进行集成测试的一部分。如果更新频率较低，请定期进行此项练习，以确保函数得到优化和调整。确定了 Lambda 函数的合适内存设置之后，就可以对您的进程添加大小调整。AWS Lambda Power Tuning 工具会生成编程输出，在新代码发布期间，CI/CD 工作流程可使用这些输出。这使您能够自动进行内存配置。

您可以免费下载 [AWS Lambda Power Tuning 工具](#)。有关如何使用该工具的说明，请参阅中的[如何执行状态机](#) GitHub。

Lambda 还支持原生 AOT，它支持对 .NET 应用程序进行预先编译。这可以通过缩短 .NET 函数的执行时间来降低成本。有关创建本机 AOT 函数的更多信息，请参阅 Lambda 文档中的[使用本机 AOT 编译的 .NET 函数](#)。

避免闲置等待时间

Lambda 函数持续时间是用于计算账单的一个维度。当函数代码发出阻止调用时，将根据它等待接收响应的时间向您计费。当 Lambda 函数链接在一起时，或者函数充当其他函数的编排工具时，此等待时间可能会增加。如果您有诸如批处理操作或订单交付系统之类的工作流，将会增加管理开销。此外，可能无法在 Lambda 的最大超时时间 15 分钟内完成所有工作流逻辑和错误处理。

与其在函数代码中处理此逻辑，我们建议您重新构建解决方案，以将 [AWS Step Functions](#) 用作工作流的编排工具。使用标准工作流时，将为工作流中的每个[状态](#)转换向您计费，而不是工作流的总持续时间。此外，您可以将对重试的支持、等待条件、错误工作流和[回调](#)移至状态条件中，以便您的 Lambda 函数专注于业务逻辑。有关更多信息，请参阅 AWS Compute 博客中的[优化 AWS Lambda 成本 — 第 2 部分](#)。

迁移到基于 Graviton 的函数

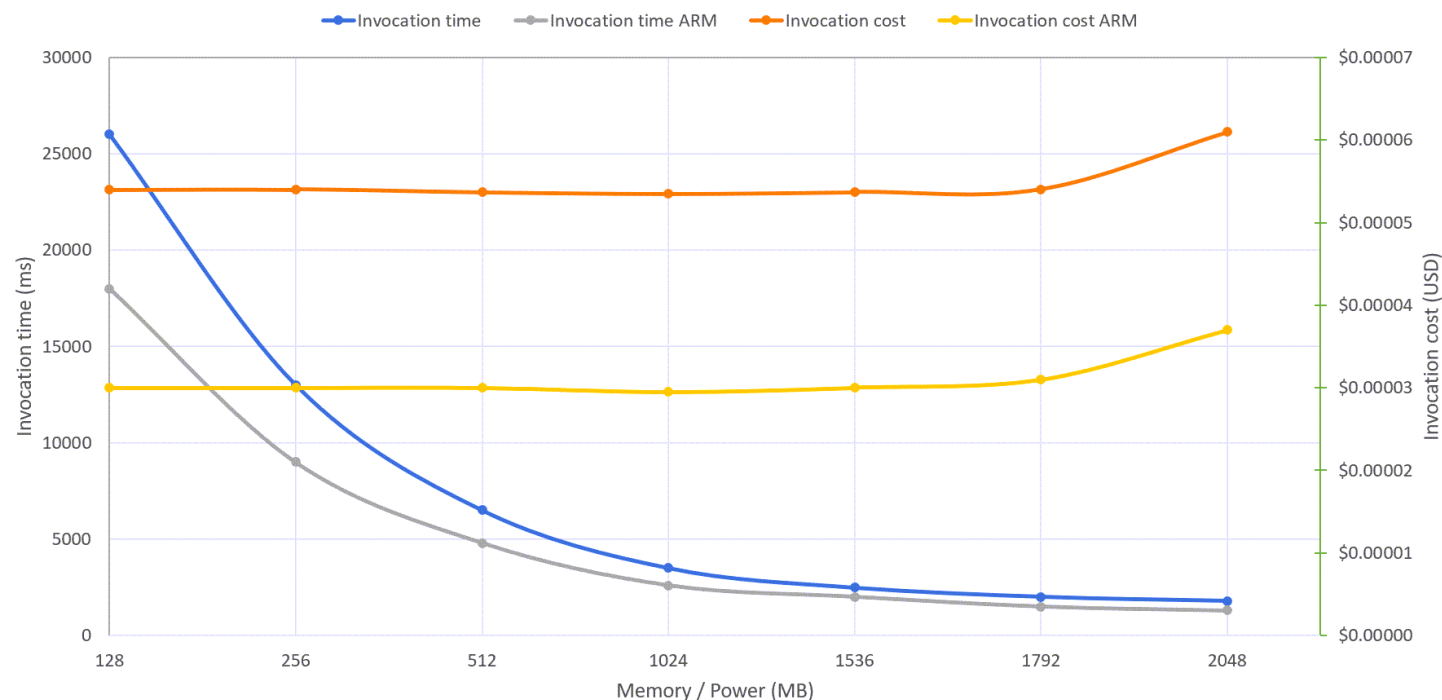
由下一代 Graviton2 处理器提供支持的 Lambda 函数现已全面推出。Graviton2 函数采用基于 ARM 的处理器架构，其设计旨在为各种无服务器工作负载提供最高 19% 的性能提升，同时将成本降低 20%。凭借更低的延迟和更出色的性能，由 Graviton2 处理器提供支持的函数非常适合为任务关键型无服务器应用程序提供支持。

对于希望优化 Lambda 成本的 .NET 开发人员而言，迁移到基于 Graviton 的 Lambda 函数可能是一种经济高效的选择。基于 Graviton 的函数使用基于 ARM 的处理器而不是传统的 x86 处理器。这能够带来显著的成本节省，同时又不会影响性能。

虽然迁移到基于 Graviton 的函数可以带来诸多好处，但同时也存在一些挑战和需要考虑的因素，我们建议您予以重视。例如，基于 Graviton 的函数需要使用 Amazon Linux 2，而该系统可能不与所有 .NET 应用程序兼容。此外，还可能存在着与第三方库或依赖关系的兼容性问题，这些库或依赖关系与基于 ARM 的处理器不兼容。

如果您正在运行 .NET Framework 应用程序，并希望利用 Lambda 实现无服务器，那么您可以考虑使用 [Porting Assistant for .NET](#) 将这些应用程序迁移到现代的 .NET 环境中。这将有助于您将遗留的 .NET 应用程序快速移植到现代的 .NET 环境中，从而使该应用程序能够在 Linux 上运行。

以下图表对比了计算质数的函数的 x86 和 ARM/Graviton2 架构结果。



该函数使用的是单线程。当内存配置为 1.8GB 时，两种架构的最短运行时间均被报告出来。除此之外，Lambda 函数可以使用超过 1 个 vCPU，但在这种情况下，该函数无法利用这些额外的计算能力。出于同样的原因，在内存最高达到 1.8GB 时，成本保持稳定。随着内存容量的增加，成本也会随之上升，因为此工作负载没有任何额外的性能益处。Graviton2 处理器显然在这一计算密集型函数方面提供了更高的性能和更低的成本。

要将您的函数配置为将基于 ARM 的处理器与 Graviton 结合使用，请执行以下操作：

1. 登录 AWS 管理控制台 并打开 [Lambda 控制台](#)。
2. 选择创建函数。
3. 对于函数名称，输入一个名称。
4. 对于 Runtime，请选择 .NET 6 (C#/ PowerShell)。
5. 对于架构，选择 arm64。
6. 进行您需要的任何其他配置，然后选择创建函数。

其他资源

- [Lambda 函数作为目标 \(文档 \)](#) AWS
- [使用 AWS Compute Optimizer \(AWS 计算博客 \)](#) 优化 AWS Lambda 成本和性能

- [优化 AWS Lambda 成本 — 第 1 部分](#) (AWS 计算博客)
- [优化 AWS Lambda 成本 — 第 2 部分](#) (AWS 计算博客)
- [AWS Lambda 使用 .NET 7 构建无服务器 .NET 应用程序](#) (AWS 计算博客)

考虑使用专用数据库

概述

运行基于 Microsoft 的工作负载所涉及的成本中，最大的一项支出来自于商业数据库（如 SQL Server）的许可费用。企业通常会将 SQL Server 作为首选的数据库平台加以标准化，并且这一选择会深深扎根于该组织的开发文化之中。无论使用案例如何，开发人员通常会选择基于关系型 SQL Server 的模型。原因包括：

- 该企业已经有可用的 SQL Server 实例 and/or 许可证。
- 通过使用共享库和业务逻辑，团队已经习惯了 SQL 编程模型。ORMs
- 管理层并不知晓其他的选择方案。
- 开发人员并不知晓其他的选择方案。

专用数据库能够适应您的使用案例的数据访问模式。随着企业采用更为现代化的架构（如微服务架构）以及单个应用程序的范围逐渐缩小，这些数据库正越来越受到企业的青睐。

专门构建的数据库并不意味着必须采用关系模型，也不一定需要使用 NoSQL（非关系型）模型。事实上，当根据工作负载的特定需求进行选择时，关系数据库会被视为是专门构建的。使用专用数据库能够帮助团队降低与 .NET 应用程序相关的数据库成本，同时还能享受到诸如可扩展性、韧性以及减少不必要的繁重工作等标准的云优势。

下表显示了提供的专门构建的数据库。AWS

数据库	Type	特性
Amazon Aurora PostgreSQL 或 Amazon Aurora MySQL	关系	数据具有固定结构的使用案例 关系数据库通过 ACID 事务来自然地保持数据一致性。
Amazon DynamoDB	键值对	使用哈希表数据结构存储数据的 NoSQL 数据库

数据库	Type	特性
		非结构化数据的高性能存储和检索 使用案例包括用户配置文件、会话状态和购物车数据
Amazon ElastiCache	内存中	将非结构化数据存储于内存中并实现亚毫秒级访问速度的高性能 NoSQL 数据库 用于频繁访问的、短暂存储的数据（例如用户会话），以及作为位于其他较慢的数据存储之前的缓存层 包括对 ElastiCache（Redis OSS）和 ElastiCache（Memcached）的支持
Amazon MemoryDB	持久内存	与 Redis 兼容的专用数据库，具有持久存储功能
Amazon Timestream	时间序列	专为按时间顺序进行高吞吐量数据摄取而设计的数据库 使用案例包括物联网（IoT）应用和存储指标或遥测数据
Amazon DocumentDB	文档	NoSQL 数据库，其存储的数据没有固定的结构，也不存在与其他数据之间的强制关联关系 通常用于读取密集型的工作负载，例如产品目录等

数据库	Type	特性
Amazon Neptune	图表	既保存数据又表示数据项之间关联关系的 NoSQL 数据库 使用案例包括欺诈检测、推荐引擎和社交应用程序
Amazon Keyspaces	宽列	基于 Apache Cassandra 的高性能分布式数据库 使用案例包括 IoT 应用程序、事件处理和游戏应用程序

采用专用数据库的一个重要驱动因素是由于不再需要支付商业许可费用。但是，[DynamoDB](#)（包括[按需模式](#)）、[Aurora](#)、[Amazon Neptune](#) 和 [Amazon Keyspaces](#) 等数据库的自动扩缩功能使您能够为一般情况而非峰值使用量来预调配容量。专用数据库（例如 Timestream）是无服务器的，无需任何预调配即可自动扩展以满足需求。

AWS 如果您想使用专门构建的、与开源兼容的关系数据库，但又无法或不愿对应用程序进行重大代码更改，则提供 [Babelfish for Aurora PostgreSQL](#)。在某些情况下，Babelfish 可让您使用现有的 SQL Server 访问代码，几乎无需进行任何修改。

在为应用程序选择专用的关系数据库时，务必保留您的应用程序所需的一致（或功能等效的）特征。此建议指出，专用数据库应作为应用程序的主要数据存储。特定应用程序（如缓存）在其他建议中进行了介绍。

成本影响

为 .NET 工作负载采用专门构建的数据库虽然不太可能 consumption/cost 直接影响计算，但会直接影响 .NET 应用程序使用的数据库服务成本。实际上，与敏捷性、可扩展性、韧性以及数据持久性所带来的诸多益处相比，节省成本可能只是次要目标。

本指南无法详尽阐述为应用程序选择专用数据库以及重新架构数据策略以有效利用它们的完整流程。有关更多信息，请参阅 AWS 教程目录中的 [专用数据库](#)。

以下表格展示了将 SQL Server 替换为专用数据库后，会如何改变应用程序成本的几个示例。请注意，这些只是大致的估算值。需要对实际工作负载进行基准测试和优化，以便计算出确切的生产成本。

以下是一些常用的专用数据库估算方案，其中包括按需计算功能以及 100GB 的 SSD，以及 us-east-1 中的单实例数据库。许可费用包括 SQL Server 许可证以及软件保障费用。

下表显示了商业数据库示例的估算成本。

数据库引擎	许可模式	实例类型/规格	AWS 计算 + 存储成本	许可证成本	月度总费用
Amazon EC2 上的 SQL Server 标准版	随附许可证	r6i.2xlarge (8 CPU/64 GB RAM)	1,345.36 美元	0.00	1,345.36 美元
Amazon EC2 上的 SQL Server 企业版	随附许可证	r6i.2xlarge (8 CPU/64 GB RAM)	2,834.56 美元	0.00	2,834.56 美元
Amazon EC2 上的 SQL Server 标准版	BYOL	r6i.2xlarge (8 CPU/64 GB RAM)	644.56 美元	456.00 美元	1,100.56 美元
Amazon EC2 上的 SQL Server 企业版	BYOL	r6i.2xlarge (8 CPU/64 GB RAM)	644.56 美元	1,750.00 美元	2,394.56 美元
Amazon RDS 上的 SQL Server 标准版		r6i.2xlarge (8 CPU/64 GB RAM)	2,318.30 美元	0.00	2,318.30 美元
Amazon RDS 上的 SQL Server 企业版		r6i.2xlarge (8 CPU/64 GB RAM)	3,750.56 美元	0.00	3,750.56 美元

下表显示了专用示例的预估成本。

数据库引擎	实例类型/规格	AWS 计算 + 存储成本	许可证成本	月度总费用
Amazon Aurora PostgreSQL	r6g.2xlarge (8 CPU/64 GB RAM)	855.87 美元	0.00	855.87 美元
DynamoDB	预调配基础 100 WCU/400 RCU	72.00 美元		72.00 美元
Amazon DocumentDB	r6i.2xlarge (8 CPU/64 GB RAM)	778.60 美元		778.60 美元

Important

该表格基于包含软件保障服务的 SQL Server 在购买后的前三年内的预计许可费用。对于 SQL Server 标准版：4100 美元，2 个核心包，3 年。对于 SQL Server 企业版：15700 美元，2 个核心包，3 年。

在采用专用数据库之前，我们建议您考虑成本影响。例如，将应用程序更新为使用专用数据库所需的费用与应用程序的复杂程度以及源数据库有关。在规划此次架构切换时，务必将总拥有成本纳入考虑范围。这包括重构您的应用程序、提升员工在新科技方面的技能，以及仔细规划每个工作负载预期的性能和消耗情况。从那里，您可以判断这项投资是否值得节省下来的成本。在大多数情况下，维护 end-of-support 产品会带来安全和合规风险，修复产品的成本值得付出努力和初始投资。

成本优化建议

对于正在访问 SQL Server 的 .NET 应用程序，专用关系数据库有相应的替代库可供使用。您可以在您的应用程序中实施这些库来替代类似的 SQL Server 应用程序功能。

以下表格重点介绍了在多种常见场景中均可使用的部分库。

图书馆	数据库	替代	框架兼容性
Npgsql Entity Framework Core 提供程序	Amazon Aurora PostgreSQL	Entity Framework Core SQL Server 提供程序	现代 .NET
Npgsql Entity Framework 6 提供程序	Amazon Aurora PostgreSQL	Entity Framework 6.0 SQL Server 提供程序	NET Framework。
Npgsql (兼容 ADO.NET 的 PostgreSQL 库)	Amazon Aurora PostgreSQL	ADO.NET	.NET Framework/ Modern T
MySQL Entity Framework Core 提供程序	Amazon Aurora MySQL	Entity Framework Core SQL Server 提供程序	现代 .NET
Pomelo。EntityFrameworkCore。MySQL	Amazon Aurora MySQL	Entity Framework Core SQL Server 提供程序	现代 .NET

使用 [Babelfish 连接到 Amazon Aurora PostgreSQL](#) 不需要任何特殊编码即可连接。但是，所有代码在使用前都必须经过严格测试。

其他专用数据库还设有用于访问与 .NET 兼容的库的接口，这使得您能够访问这些专用数据库。例子包括：

- [使用亚马逊 DynamoDB NoSQL 数据库](#) (文档) 适用于 .NET 的 AWS SDK
- [MongoDB C# 驱动程序](#) (MongoDB 文档)
- [.NET](#) (Timestream 文档)
- [使用 Cassandra .NET Core 客户端驱动程序以编程方式访问 Amazon Keyspaces](#) (Amazon Keyspaces 文档)
- [使用 .NET 连接到 Neptune 数据库实例](#) (Neptune 文档)

如果您迁移到专门构建的数据库，则可以使用以下工具 AWS 来帮助完成迁移过程：

- [AWS Schema Conversion Tool \(AWS SCT \)](#) 可以帮助您将 SQL Server 架构转换为 Amazon Aurora 和 Amazon DynamoDB。
- [AWS Database Migration Service \(AWS DMS \)](#) 能够帮助您将数据从 SQL Server 迁移到 Aurora 或 DynamoDB，无论是一次性迁移，还是持续性迁移。
- [Babelfish Compass](#) 可以帮助您检查 SQL Server 数据库与适用于 Aurora PostgreSQL 的 Babelfish 的兼容性。

其他资源

- [将 SQL Server 迁移到 Amazon Aurora PostgreSQL 的指南](#) (AWS 数据库博客)
- [Babelfish APP 现代化沉浸式体验日](#) (AWS 工作室工作室)
- [.NET 沉浸日](#) (工作室工作AWS 室)
- [开始使用带有.NET 的 Amazon Timestream](#) () GitHub
- [专为现代.NET 应用程序构建的数据库 AWS \(演示文稿 \)](#) AWS

后续步骤

完成查看本指南后，我们建议您采取以下后续步骤来实施 MACO：

1. 联系 MACO 专家。MACO 专家可以帮助您解答疑问并解决您的顾虑。如果您已经在与 AWS 客户团队合作，请联系该团队并向 MACO 专家请求帮助。如果您还没有客户团队，则请联系 optimize-microsoft@amazon.com。
2. 应用建议。请将您在本指南中以及与 MACO 专家交流时所学到的建议、最佳实践和策略付诸应用。
3. 跟踪成本变化。标记您的工作负载并使用 AWS Cost Explorer 和 AWS Budgets 之类的服务进行详细的成本跟踪、监测和控制。

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
SQL Server 更新	我们更新了“ 优化 SQL Server 工作负载的 CPU ”部分，添加了有关 Amazon EC2 实例优化 CPU 功能的更多信息。	2025 年 10 月 22 日
SQL Server 更新	我们更新了 优化 SQL Server 工作负载的 CPU 部分。	2024 年 10 月 25 日
SQL Server 和容器更新	我们添加了“ 使用 Compute Optimizer 优化 SQL Server 大小 ”、“ SQL Server 工作负载的查看 Trusted Advisor 建议 ”和“使用 App2Container 重新构建 Windows 应用程序 ”部分。	2024 年 6 月 29 日
SQL Server 许可优化	我们添加了 使用 Compute Optimizer 优化 SQL Server 许可 部分。	2024 年 5 月 22 日
初次发布	—	2023 年 12 月 21 日

AWS 规范性指导词汇表

以下是 AWS 规范性指导提供的策略、指南和模式中的常用术语。若要推荐词条，请使用术语表末尾的提供反馈链接。

数字

7 R

将应用程序迁移到云中的 7 种常见迁移策略。这些策略以 Gartner 于 2011 年确定的 5 R 为基础，包括以下内容：

- **重构/重新架构**：充分利用云原生功能来提高敏捷性、性能和可扩展性，以迁移应用程序并修改其架构。这通常涉及到移植操作系统和数据库。示例：将本地 Oracle 数据库迁移到 Amazon Aurora PostgreSQL 兼容版。
- **更换平台**：将应用程序迁移到云中，并进行一定程度的优化，以利用云功能。示例：将本地 Oracle 数据库迁移到 AWS 云中的 Amazon Relational Database Service (Amazon RDS) for Oracle。
- **重新购买**：转换到其他产品，通常是从传统许可转向 SaaS 模式。示例：将客户关系管理 (CRM) 系统迁移到 Salesforce.com。
- **重新托管 (直接迁移)**：将应用程序迁移到云中，无需进行任何更改即可利用云功能。示例：将本地 Oracle 数据库迁移到 AWS 云中 EC2 实例上的 Oracle。
- **重新放置 (虚拟机监控器级直接迁移)**：将基础设施迁移到云中，无需购买新硬件、重写应用程序或修改现有操作。您将服务器从本地平台迁移到同一平台的云服务中。示例：将 Microsoft Hyper-V 应用程序迁移到 AWS。
- **保留 (重访)**：将应用程序保留在源环境中。其中可能包括需要进行重大重构的应用程序，并且您希望将工作推迟到以后，以及您希望保留的遗留应用程序，因为迁移它们没有商业上的理由。
- **停用**：停用或删除源环境中不再需要的应用程序。

A

ABAC

请参阅[基于属性的访问控制](#)。

抽象服务

请参阅[托管服务](#)。

ACID

请参阅[原子性、一致性、隔离性、持久性](#)。

主动-主动迁移

一种数据库迁移方法，在这种方法中，源数据库和目标数据库保持同步（通过使用双向复制工具或双写操作），两个数据库都在迁移期间处理来自连接应用程序的事务。这种方法支持小批量、可控的迁移，而不需要一次性割接。它比[主动-被动迁移](#)更灵活，但工作量更大。

主动-被动迁移

一种数据库迁移方法，在这种方法中，源数据库和目标数据库保持同步，但在将数据复制到目标数据库时，只有源数据库处理来自连接应用程序的事务。目标数据库在迁移期间不接受任何事务。

聚合函数

一种 SQL 函数，它对一组行进行操作并计算该组的单个返回值。聚合函数的示例包括 SUM 和 MAX。

AI

请参阅[人工智能](#)。

AIOps

请参阅[人工智能运营](#)。

匿名化

永久删除数据集中个人信息的过程。匿名化可以帮助保护个人隐私。匿名化数据不再被视为个人数据。

反模式

一种用于解决反复出现的问题的常用解决方案，而在这类问题中，此解决方案适得其反、无效或不如替代方案有效。

应用程序控制

一种安全方法，仅允许使用经批准的应用程序，以帮助保护系统免受恶意软件的侵害。

应用程序组合

有关组织使用的每个应用程序的详细信息的集合，包括构建和维护该应用程序的成本及其业务价值。这些信息是[产品组合发现和分析过程](#)的关键，有助于识别需要进行迁移、现代化和优化的应用程序并确定其优先级。

人工智能 (AI)

计算机科学领域致力于使用计算技术执行通常与人类相关的认知功能，例如学习、解决问题和识别模式。有关更多信息，请参阅[什么是人工智能？](#)

人工智能操作 (AIOps)

使用机器学习技术解决运营问题、减少运营事故和人为干预以及提高服务质量的过程。有关如何在 AIOps AWS 迁移策略中使用的更多信息，请参阅[操作集成指南](#)。

非对称加密

一种加密算法，使用一对密钥，一个公钥用于加密，一个私钥用于解密。您可以共享公钥，因为它不用于解密，但对私钥的访问应受到严格限制。

原子性、一致性、隔离性、持久性 (ACID)

一组软件属性，即使在出现错误、电源故障或其他问题的情况下，也能保证数据库的数据有效性和操作可靠性。

基于属性的访问权限控制 (ABAC)

根据用户属性（如部门、工作角色和团队名称）创建精细访问权限的做法。有关更多信息，请参阅 AWS Identity and Access Management (IAM) [文档](#) [AWS 中的 ABAC](#)。

权威数据来源

存储主要数据版本的位置，被认为是最可靠的信息源。您可以将数据从权威数据来源复制到其他位置，以便处理或修改数据，例如对数据进行匿名化、编辑或假名化。

可用区

中的一个不同位置 AWS 区域，不受其他可用区域故障的影响，并向同一区域中的其他可用区提供低成本、低延迟的网络连接。

AWS 云采用框架 (AWS CAF)

该框架包含指导方针和最佳实践 AWS，可帮助组织制定高效且有效的计划，以成功迁移到云端。AWS CAF 将指导分为六个重点领域，称为视角：业务、人员、治理、平台、安全和运营。业务、人员和治理角度侧重于业务技能和流程；平台、安全和运营角度侧重于技术技能和流程。例如，人

员角度针对的是负责人力资源 (HR)、人员配置职能和人员管理的利益相关者。从这个角度来看，AWS CAF 为人员发展、培训和沟通提供了指导，以帮助组织为成功采用云做好准备。有关更多信息，请参阅 [AWS CAF 网站](#) 和 [AWS CAF 白皮书](#)。

AWS 工作负载资格框架 (AWS WQF)

一种评估数据库迁移工作负载、推荐迁移策略和提供工作估算的工具。AWS WQF 包含在 AWS Schema Conversion Tool (AWS SCT) 中。它用来分析数据库架构和代码对象、应用程序代码、依赖关系和性能特征，并提供评测报告。

B

恶意机器人

一种旨在扰乱或伤害个人或组织的[机器人](#)。

BCP

请参阅[业务连续性计划](#)。

行为图

一段时间内资源行为和交互的统一交互式视图。您可以使用 Amazon Detective 的行为图来检查失败的登录尝试、可疑的 API 调用和类似的操作。有关更多信息，请参阅 Detective 文档中的[行为图中的数据](#)。

大端序系统

一个先存储最高有效字节的系统。另请参阅[字节顺序](#)。

二进制分类

一种预测二进制结果 (两个可能的类别之一) 的过程。例如，您的 ML 模型可能需要预测诸如“该电子邮件是否为垃圾邮件？”或“这个产品是书还是汽车？”之类的问题

bloom 筛选条件

一种概率性、内存高效的数据结构，用于测试元素是否为集合的成员。

蓝/绿部署

一种部署策略，您可以创建两个独立但完全相同的环境。在一个环境中运行当前应用程序版本 (蓝色)，在另一个环境中运行新应用程序版本 (绿色)。此策略可帮助您在影响最小的情况下快速回滚。

自动程序

一种通过互联网运行自动任务并模拟人类活动或交互的软件应用程序。有些机器人是有用或有益的，例如在互联网上索引信息的 Web 爬网程序。还有一些被称为恶意机器人的机器人，其目的是扰乱或伤害个人或组织。

僵尸网络

被[恶意软件](#)感染并受单方（称为僵尸网络控制者或僵尸网络操作者）控制的[僵尸](#)网络。僵尸网络是最著名的扩展机器人及其影响力的机制。

分支

代码存储库的一个包含区域。在存储库中创建的第一个分支是主分支。您可以从现有分支创建新分支，然后在新分支中开发功能或修复错误。为构建功能而创建的分支通常称为功能分支。当功能可以发布时，将功能分支合并回主分支。有关更多信息，请参阅[关于分支](#)（GitHub 文档）。

紧急（break-glass）访问

在特殊情况下，通过批准的流程，用户 AWS 账户 可以快速访问他们通常没有访问权限的内容。有关更多信息，请参阅 AWS Well-Architected Guidance 中的 [Implement break-glass procedures](#) 指示器。

棕地策略

您环境中的现有基础设施。在为系统架构采用棕地策略时，您需要围绕当前系统和基础设施的限制来设计架构。如果您正在扩展现有基础设施，则可以将棕地策略和[全新](#)策略混合。

缓冲区缓存

存储最常访问的数据的内存区域。

业务能力

企业如何创造价值（例如，销售、客户服务或营销）。微服务架构和开发决策可以由业务能力驱动。有关更多信息，请参阅[在 AWS 上运行容器化微服务](#)白皮书中的[围绕业务能力进行组织](#)部分。

业务连续性计划（BCP）

一项计划，旨在应对大规模迁移等破坏性事件对运营的潜在影响，并使企业能够快速恢复运营。

C

CAF

请参阅 [AWS 云采用框架](#)。

金丝雀部署

缓慢而渐进地向最终用户发布版本。当您确信无误后，即可部署新版本，并完全替换当前版本。

CCoE

请参阅[云卓越中心](#)。

CDC

请参阅[更改数据捕获](#)。

更改数据捕获 (CDC)

跟踪数据来源 (如数据库表) 的更改并记录有关更改的元数据的过程。您可以将 CDC 用于各种目的，例如审计或复制目标系统中的更改以保持同步。

混沌工程

故意引入故障或破坏性事件来测试系统的韧性。您可以使用 [AWS Fault Injection Service \(AWS FIS\)](#) 来执行实验，对您的 AWS 工作负载施加压力并评估其响应。

CI/CD

请参阅[持续集成和持续交付](#)。

分类

一种有助于生成预测的分类流程。分类问题的 ML 模型预测离散值。离散值始终彼此不同。例如，一个模型可能需要评估图像中是否有汽车。

客户端加密

在目标 AWS 服务 收到数据之前，对数据进行本地加密。

云卓越中心 (CCoE)

一个多学科团队，负责推动整个组织的云采用工作，包括开发云最佳实践、调动资源、制定迁移时间表、领导组织完成大规模转型。有关更多信息，请参阅 AWS 云 企业战略博客上的 [CCoE 帖子](#)。

云计算

通常用于远程数据存储和 IoT 设备管理的云技术。云计算通常连接到[边缘计算](#)技术。

云运营模型

在 IT 组织中，一种用于构建、完善和优化一个或多个云环境的运营模型。有关更多信息，请参阅[构建您的云运营模型](#)。

云采用阶段

组织迁移到 AWS 云中时通常会经历四个阶段：

- 项目 - 出于概念验证和学习目的，开展一些与云相关的项目
- 基础 — 进行基础投资以扩大云采用率（例如，创建着陆区、定义 CCo E、建立运营模型）
- 迁移 - 迁移单个应用程序
- 重塑 - 优化产品和服务，在云中创新

Stephen Orban 在 AWS 云企业战略博客的博客文章 [《云优先之旅和采用阶段》](#) 中定义了这些阶段。有关它们与 AWS 迁移策略的关系的信息，请参阅 [迁移准备指南](#)。

CMDB

请参阅 [配置管理数据库](#)。

代码存储库

通过版本控制过程存储和更新源代码和其他资产（如文档、示例和脚本）的位置。常见的云存储库包括 GitHub 或 Bitbucket Cloud。每个版本的代码都称为一个分支。在微服务结构中，每个存储库都专门用于一个功能。单个 CI/CD 管线可以使用多个存储库。

冷缓存

一种空的、填充不足或包含过时或不相关数据的缓冲区缓存。这会影响性能，因为数据库实例必须从主内存或磁盘读取，这比从缓冲区缓存读取要慢。

冷数据

很少访问的数据，且通常是历史数据。查询此类数据时，通常可以接受慢速查询。将这些数据转移到性能较低且成本更低的存储层或类别可以降低成本。

计算机视觉 (CV)

一种 [AI](#) 领域，它使用机器学习来分析和提取数字图像和视频等视觉格式中的信息。例如，Amazon SageMaker AI 为 CV 提供了图像处理算法。

配置偏移

对于工作负载而言，一种偏离预期状态的配置更改。这可能会导致工作负载变得不合规，且通常是渐进的，不是故意的。

配置管理数据库 (CMDB)

一种存储库，用于存储和管理有关数据库及其 IT 环境的信息，包括硬件和软件组件及其配置。您通常在迁移的产品组合发现和分析阶段使用来自 CMDB 的数据。

合规性包

一系列 AWS Config 规则和补救措施，您可以汇编这些规则和补救措施，以自定义您的合规性和安全性检查。您可以使用 YAML 模板将一致性包作为单个实体部署在 AWS 账户 和区域或整个组织中。有关更多信息，请参阅 AWS Config 文档中的 [一致性包](#)。

持续集成和持续交付 (CI/CD)

自动执行软件发布过程的源代码、构建、测试、暂存和生产阶段的过程。CI/CD 通常被描述为管道。CI/CD 可以帮助您实现流程自动化、提高生产力、提高代码质量和更快地交付。有关更多信息，请参阅[持续交付的优势](#)。CD 也可以表示持续部署。有关更多信息，请参阅[持续交付与持续部署](#)。

CV

请参阅[计算机视觉](#)。

D

静态数据

网络中静止的数据，例如存储中的数据。

数据分类

根据网络中数据的关键性和敏感性对其进行识别和分类的过程。它是任何网络安全风险管理策略的关键组成部分，因为它可以帮助您确定对数据的适当保护和保留控制。数据分类是 Well-Architected AWS d Framework 中安全支柱的一个组成部分。有关详细信息，请参阅[数据分类](#)。

数据漂移

生产数据与用来训练机器学习模型的数据之间的有意义差异，或者输入数据随时间推移的有意义变化。数据漂移可能降低机器学习模型预测的整体质量、准确性和公平性。

传输中数据

在网络中主动移动的数据，例如在网络资源之间移动的数据。

数据网格

一种架构框架，可提供分布式、去中心化的数据所有权以及集中式管理和治理。

数据最少化

仅收集并处理绝对必要数据的原则。在中进行数据最小化 AWS 云 可以降低隐私风险、成本和分析碳足迹。

数据边界

AWS 环境中的一组预防性防护措施，可帮助确保只有可信身份才能访问来自预期网络的可信资源。有关更多信息，请参阅在[上构建数据边界](#)。AWS

数据预处理

将原始数据转换为 ML 模型易于解析的格式。预处理数据可能意味着删除某些列或行，并处理缺失、不一致或重复的值。

数据溯源

在数据的整个生命周期跟踪其来源和历史的过程，例如数据如何生成、传输和存储。

数据主体

正在收集和处理其数据的人。

数据仓库

一种支持商业智能（例如分析）的数据管理系统。数据仓库通常包含大量历史数据，通常用于查询和分析。

数据库定义语言（DDL）

在数据库中创建或修改表和对对象结构的语句或命令。

数据库操作语言（DML）

在数据库中修改（插入、更新和删除）信息的语句或命令。

DDL

请参阅[数据库定义语言](#)。

深度融合

组合多个深度学习模型进行预测。您可以使用深度融合来获得更准确的预测或估算预测中的不确定性。

深度学习

一个 ML 子字段使用多层人工神经网络来识别输入数据和感兴趣的目标变量之间的映射。

defense-in-depth

一种信息安全方法，经过深思熟虑，在整个计算机网络中分层实施一系列安全机制和控制措施，以保护网络及其中数据的机密性、完整性和可用性。当你采用这种策略时 AWS，你会在 AWS

Organizations 结构的不同层面添加多个控件来帮助保护资源。例如，一种 defense-in-depth 方法可以结合多因素身份验证、网络分段和加密。

委派管理员

在中 AWS Organizations，兼容的服务可以注册 AWS 成员帐户来管理组织的帐户并管理该服务的权限。此账户被称为该服务的委托管理员。有关更多信息和兼容服务列表，请参阅 AWS Organizations 文档中[使用 AWS Organizations 的服务](#)。

部署

使应用程序、新功能或代码修复在目标环境中可用的过程。部署涉及在代码库中实现更改，然后在应用程序的环境中构建和运行该代码库。

开发环境

请参阅[环境](#)。

侦测性控制

一种安全控制，在事件发生后进行检测、记录日志和发出提醒。这些控制是第二道防线，提醒您注意绕过现有预防性控制的安全事件。有关更多信息，请参阅在 AWS 上实施安全控制中的[侦测性控制](#)。

开发价值流映射 (DVSM)

用于识别对软件开发生命周期中的速度和质量产生不利影响的限制因素并确定其优先级的流程。DVSM 扩展了最初为精益生产实践设计的价值流映射流程。其重点关注在软件开发过程中创造和转移价值所需的步骤和团队。

数字孪生

真实世界系统的虚拟再现，如建筑物、工厂、工业设备或生产线。数字孪生支持预测性维护、远程监控和生产优化。

维度表

[星型架构](#)中的一种较小的表，其中包含事实表中定量数据的数据属性。维度表属性通常是文本字段或行为类似于文本的离散数字。这些属性通常用于查询约束、筛选和结果集标注。

灾难

阻止工作负载或系统在其主要部署位置实现其业务目标的事件。这些事件可能是自然灾害、技术故障或人为操作的结果，例如无意的配置错误或恶意软件攻击。

灾难恢复 (DR)

您用来最大程度地减少由[灾难](#)造成的停机时间和数据丢失的策略和流程。有关更多信息，请参阅 Well-Architected Framework AWS work 中的“[工作负载灾难恢复：云端 AWS 恢复](#)”。

DML

请参阅[数据库操作语言](#)。

领域驱动设计

一种开发复杂软件系统的方法，通过将其组件连接到每个组件所服务的不断发展的领域或核心业务目标。Eric Evans 在其著作领域驱动设计：软件核心复杂性应对之道 (Boston: Addison-Wesley Professional, 2003) 中介绍了这一概念。有关如何将领域驱动设计与 strangler fig 模式结合使用的信息，请参阅[使用容器和 Amazon API Gateway 逐步将原有的 Microsoft ASP.NET \(ASMX \) Web 服务现代化](#)。

DR

请参阅[灾难恢复](#)。

偏差检测

跟踪与基准配置的偏差。例如，您可以使用 AWS CloudFormation 来[检测系统资源中的偏差](#)，也可以使用 AWS Control Tower 来[检测着陆区中可能影响监管要求合规性的变化](#)。

DVSM

请参阅[开发价值流映射](#)。

E

EDA

请参阅[探索性数据分析](#)。

EDI

请参阅[电子数据交换](#)。

边缘计算

该技术可提高位于 IoT 网络边缘的智能设备的计算能力。与[云计算](#)比较时，边缘计算可以减少通信延迟并缩短响应时间。

电子数据交换 (EDI)

组织之间业务文件的自动交换。有关更多信息，请参阅[什么是电子数据交换](#)。

加密

一种将人类可读的纯文本数据转换为加密文字的计算流程。

加密密钥

由加密算法生成的随机位的加密字符串。密钥的长度可能有所不同，而且每个密钥都设计为不可预测且唯一。

字节顺序

字节在计算机内存中的存储顺序。大端序系统先存储最高有效字节。小端序系统先存储最低有效字节。

端点

请参阅[服务端点](#)。

端点服务

一种可以在虚拟私有云 (VPC) 中托管，与其他用户共享的服务。您可以使用其他 AWS 账户 或 AWS Identity and Access Management (IAM) 委托人创建终端节点服务，AWS PrivateLink 并向其授予权限。这些账户或主体可通过创建接口 VPC 端点来私密地连接到您的端点服务。有关更多信息，请参阅 Amazon Virtual Private Cloud (Amazon VPC) 文档中的[创建端点服务](#)。

企业资源规划 (ERP)

一种自动化和管理企业关键业务流程 (例如会计、[MES](#) 和项目管理) 的系统。

信封加密

用另一个加密密钥对加密密钥进行加密的过程。有关更多信息，请参阅 AWS Key Management Service (AWS KMS) 文档中的[信封加密](#)。

环境

正在运行的应用程序的实例。以下是云计算中常见的环境类型：

- 开发环境 — 正在运行的应用程序的实例，只有负责维护应用程序的核心团队才能使用。开发环境用于测试更改，然后再将其提升到上层环境。这类环境有时称为测试环境。
- 下层环境 — 应用程序的所有开发环境，比如用于初始构建和测试的环境。

- 生产环境 — 最终用户可以访问的正在运行的应用程序的实例。在 CI/CD 管道中，生产环境是最后一个部署环境。
- 上层环境 — 除核心开发团队以外的用户可以访问的所有环境。这可能包括生产环境、预生产环境和用户验收测试环境。

epic

在敏捷方法学中，有助于组织工作和确定优先级的功能类别。epics 提供了对需求和实施任务的总体描述。例如，AWS CAF 安全史诗包括身份和访问管理、侦探控制、基础设施安全、数据保护和事件响应。有关 AWS 迁移策略中 epics 的更多信息，请参阅[计划实施指南](#)。

ERP

请参阅[企业资源规划](#)。

探索性数据分析 (EDA)

分析数据集以了解其主要特征的过程。您收集或汇总数据，并进行初步调查，以发现模式、检测异常并检查假定情况。EDA 通过计算汇总统计数据和创建数据可视化得以执行。

F

事实表

[星型架构](#)中的中心表。它存储有关业务运营的定量数据。通常，事实表包含两种类型的列：包含度量的列和包含维度表外键的列。

快速失效机制

一种使用频繁且增量式的测试来缩短开发生命周期的理念。这是敏捷方法的关键部分。

故障隔离边界

在中 AWS 云，诸如可用区 AWS 区域、控制平面或数据平面之类的边界，它限制了故障的影响并有助于提高工作负载的弹性。有关更多信息，请参阅[AWS 故障隔离边界](#)。

功能分支

请参阅[分支](#)。

特征

您用来进行预测的输入数据。例如，在制造环境中，特征可能是定期从生产线捕获的图像。

特征重要性

特征对于模型预测的重要性。这通常表示为数值分数，可以通过各种技术进行计算，例如 Shapley 加法解释 (SHAP) 和积分梯度。有关更多信息，请参阅使用[机器学习模型的可解释性 AWS](#)。

功能转换

为 ML 流程优化数据，包括使用其他来源丰富数据、扩展值或从单个数据字段中提取多组信息。这使得 ML 模型能从数据中获益。例如，如果您将“2021-05-27 00:15:37”日期分解为“2021”、“五月”、“星期四”和“15”，则可以帮助学习与不同数据成分相关的算法学习精细模式。

少样本提示

在要求 [LLM](#) 执行类似任务之前，先向其提供少量示例，以演示任务和预期输出。此技术是上下文内学习的一种应用，其中模型可以从提示中嵌入的示例 (样本) 中学习。对于需要特定格式、推理或领域知识的任务，少样本提示可能非常有效。另请参阅[零样本提示](#)。

FGAC

请参阅[精细访问控制](#)。

精细访问控制 (FGAC)

使用多个条件允许或拒绝访问请求。

快闪迁移

一种数据库迁移方法，通过[更改数据捕获](#)使用连续数据复制，在极短的时间内迁移数据，而非使用分阶段方法。目标是将停机时间降至最低。

FM

请参阅[基础模型](#)。

基础模型 (FM)

一个大型深度学习神经网络，一直在广义和未标记数据的大量数据集上进行训练。FMs 能够执行各种各样的一般任务，例如理解语言、生成文本和图像以及用自然语言进行对话。有关更多信息，请参阅[什么是基础模型](#)。

G

生成式人工智能

[AI](#) 模型的一个子集，这些模型已经过大量数据训练，可以使用简单的文本提示来创建新的内容和构件，例如图像、视频、文本和音频。有关更多信息，请参阅[什么是生成式人工智能](#)。

地理阻止

请参阅[地理限制](#)。

地理限制 (地理阻止)

在 Amazon 中 CloudFront，一种阻止特定国家/地区的用户访问内容分发的选项。您可以使用允许列表或阻止列表来指定已批准和已禁止的国家/地区。有关更多信息，请参阅 CloudFront 文档[中的限制内容的地理分布](#)。

GitFlow 工作流程

一种方法，在这种方法中，下层和上层环境在源代码存储库中使用不同的分支。Gitflow 工作流程被认为是传统的工作流程，而[基于中继的工作流程](#)则是现代的、首选的方法。

黄金映像

系统或软件的快照，用作部署该系统或软件的新实例的模板。例如，在制造业中，黄金映像可用于在多个设备上预调配软件，并有助于提高设备制造操作的速度、可扩展性和生产效率。

全新策略

在新环境中缺少现有基础设施。在对系统架构采用全新策略时，您可以选择所有新技术，而不受对现有基础设施（也称为[棕地](#)）兼容性的限制。如果您正在扩展现有基础设施，则可以将棕地策略和全新策略混合。

防护机制

帮助管理各组织单位的资源、策略和合规性的高级规则 (OUs)。预防性防护机制会执行策略以确保符合合规性标准。它们是使用服务控制策略和 IAM 权限边界实现的。侦测性护栏会检测策略违规和合规性问题，并生成提醒以进行修复。它们通过使用 AWS Config、Amazon、AWS Security Hub CSPM GuardDuty AWS Trusted Advisor、Amazon Inspector 和自定义 AWS Lambda 支票来实现。

H

HA

请参阅[高可用性](#)。

异构数据库迁移

将源数据库迁移到使用不同数据库引擎的目标数据库（例如，从 Oracle 迁移到 Amazon Aurora）。异构迁移通常是重新架构工作的一部分，而转换架构可能是一项复杂的任务。[AWS 提供了 AWS SCT](#) 来帮助实现架构转换。

高可用性 (HA)

在遇到挑战或灾难时，工作负载无需干预即可连续运行的能力。HA 系统旨在自动进行故障转移、持续提供良好性能，并以最小的性能影响处理不同负载和故障。

历史数据库现代化

一种用于实现运营技术 (OT) 系统现代化和升级以更好满足制造业需求的方法。历史数据库是一种用于收集和存储工厂中各种来源数据的数据库。

保留数据

从用于训练[机器学习](#)模型的数据集中保留的一部分标注的历史数据。通过将模型预测与保留数据进行比较，您可以使用保留数据来评估模型性能。

同构数据库迁移

将源数据库迁移到共享同一数据库引擎的目标数据库 (例如，从 Microsoft SQL Server 迁移到 Amazon RDS for SQL Server)。同构迁移通常是更换主机或更换平台工作的一部分。您可以使用本机数据库实用程序来迁移架构。

热数据

经常访问的数据，例如实时数据或近期的转化数据。这些数据通常需要高性能存储层或存储类别才能提供快速的查询响应。

修补程序

针对生产环境中关键问题的紧急修复。由于其紧迫性，修补程序通常是在典型的 DevOps 发布工作流程之外进行的。

hypercure 周期

割接之后，迁移团队立即管理和监控云中迁移的应用程序以解决任何问题的时间段。通常，这个周期持续 1-4 天。在 hypercure 周期结束时，迁移团队通常会将应用程序的责任移交给云运营团队。

我

laC

请参阅[基础设施即代码](#)。

基于身份的策略

附加到一个或多个 IAM 委托人的策略，用于定义他们在 AWS 云环境中的权限。

空闲应用程序

90 天内平均 CPU 和内存使用率在 5% 到 20% 之间的应用程序。在迁移项目中，通常会停用这些应用程序或将其保留在本地。

IIoT

请参阅[工业物联网](#)。

不可变基础设施

一种模型，可为生产工作负载部署新的基础设施，而不是更新、修补或修改现有基础设施。不可变基础设施本质上比[可变基础设施](#)更一致、更可靠、更可预测。有关更多信息，请参阅 AWS Well-Architected Framework 中的[使用不可变基础设施进行部署](#)最佳实践。

入站 (入口) VPC

在 AWS 多账户架构中，一种接受、检查和路由来自应用程序外部的网络连接的 VPC。[AWS 安全参考架构](#)建议设置您的网络帐户，包括入站、出站和检查，VPCs 以保护您的应用程序与更广泛的互联网之间的双向接口。

增量迁移

一种割接策略，在这种策略中，您可以将应用程序分成小部分进行迁移，而不是一次性完整割接。例如，您最初可能只将几个微服务或用户迁移到新系统。在确认一切正常后，您可以逐步迁移其他微服务或用户，直到停用遗留系统。这种策略降低了大规模迁移带来的风险。

工业 4.0

该术语由 [Klaus Schwab](#) 在 2016 年提出，指的是通过连接、实时数据、自动化、分析和 AI/ML 的进步来实现制造流程的现代化。

基础设施

应用程序环境中包含的所有资源和资产。

基础设施即代码 (IaC)

通过一组配置文件预调配和管理应用程序基础设施的过程。IaC 旨在帮助您集中管理基础设施、实现资源标准化和快速扩展，使新环境具有可重复性、可靠性和一致性。

工业物联网 (IIoT)

在工业领域使用联网的传感器和设备，例如制造业、能源、汽车、医疗保健、生命科学和农业。有关更多信息，请参阅[制定工业物联网 \(IIoT\) 数字化转型战略](#)。

检查 VPC

在 AWS 多账户架构中，一种集中式 VPC，用于管理对 VPCs（相同或不同 AWS 区域）、互联网和本地网络之间的网络流量的检查。[AWS 安全参考架构](#)建议设置您的网络帐户，包括入站、出站和检查，VPCs 以保护您的应用程序与更广泛的互联网之间的双向接口。

物联网 (IoT)

由带有嵌入式传感器或处理器的连接物理对象组成的网络，这些传感器或处理器通过互联网或本地通信网络与其他设备和系统进行通信。有关更多信息，请参阅[什么是 IoT ?](#)

可解释性

它是机器学习模型的一种特征，描述了人类可以理解模型的预测如何取决于其输入的程度。有关更多信息，请参阅使用[机器学习模型的可解释性 AWS](#)。

物联网

请参阅[物联网](#)。

IT 信息库 (ITIL)

提供 IT 服务并使这些服务符合业务要求的一套最佳实践。ITIL 是 ITSM 的基础。

IT 服务管理 (ITSM)

为组织设计、实施、管理和支持 IT 服务的相关活动。有关将云运营与 ITSM 工具集成的信息，请参阅[运营集成指南](#)。

ITIL

请参阅[IT 信息库](#)。

ITSM

请参阅[IT 服务管理](#)。

L

基于标签的访问控制 (LBAC)

强制访问控制 (MAC) 的一种实施方式，其中明确为用户和数据本身分配了安全标签值。用户安全标签和数据安全标签之间的交集决定了用户可以看到哪些行和列。

登录区

landing zone 是一个架构精良的多账户 AWS 环境，具有可扩展性和安全性。这是一个起点，您的组织可以从这里放心地在安全和基础设施环境中快速启动和部署工作负载和应用程序。有关登录区的更多信息，请参阅[设置安全且可扩展的多账户 AWS 环境](#)。

大语言模型 (LLM)

一种基于大量数据进行预训练的深度学习 [AI](#) 模型。LLM 可以执行多项任务，例如回答问题、总结文档、将文本翻译成其他语言以及完成句子。有关更多信息，请参阅[什么是 LLMs](#)。

大规模迁移

迁移 300 台或更多服务器。

LBAC

请参阅[基于标签的访问控制](#)。

最低权限

授予执行任务所需的最低权限的最佳安全实践。有关更多信息，请参阅 IAM 文档中的[应用最低权限许可](#)。

直接迁移

请参阅 [7 R](#)。

小端序系统

一个先存储最低有效字节的系统。另请参阅[字节顺序](#)。

LLM

请参阅[大型语言模型](#)。

下层环境

请参阅[环境](#)。

M

机器学习 (ML)

一种使用算法和技术进行模式识别和学习的人工智能。ML 对记录的数据 (例如物联网 (IoT) 数据) 进行分析和学习，以生成基于模式的统计模型。有关更多信息，请参阅[机器学习](#)。

主分支

请参阅[分支](#)。

恶意软件

旨在危害计算机安全或隐私的软件。恶意软件可能会破坏计算机系统、泄露敏感信息或获得未经授权的访问权限。恶意软件的示例包括病毒、蠕虫、勒索软件、木马、间谍软件和键盘记录器。

托管式服务

AWS 服务 它 AWS 运行基础设施层、操作系统和平台，您可以访问端点来存储和检索数据。Amazon Simple Storage Service (Amazon S3) 和 Amazon DynamoDB 就是托管服务的示例。这些服务也称为抽象服务。

制造执行系统 (MES)

一种软件系统，用于跟踪、监控、记录和控制将原材料转化为成品的生产过程。

MAP

请参阅[迁移加速计划](#)。

机制

一个完整的过程，您可以在其中创建工具，推动工具的采用，然后检查结果以进行调整。机制是一种在运作过程中自我强化和改善的循环。有关更多信息，请参阅在 Well-Architect AWS ed 框架中[构建机制](#)。

成员账户

AWS 账户 除属于组织中的管理账户之外的所有账户 AWS Organizations。一个账户一次只能是一个组织的成员。

MES

请参阅[制造执行系统](#)。

消息队列遥测传输 (MQTT)

[一种基于发布/订阅模式的轻量级 machine-to-machine \(M2M\) 通信协议，适用于资源受限的物联网设备。](#)

微服务

一种小型的独立服务，通过明确的定义进行通信 APIs，通常由小型的独立团队拥有。例如，保险系统可能包括映射到业务能力（如销售或营销）或子域（如购买、理赔或分析）的微服务。微服务

的好处包括敏捷、灵活扩展、易于部署、可重复使用的代码和恢复能力。有关更多信息，请参阅[使用 AWS 无服务器服务集成微服务](#)。

微服务架构

一种使用独立组件构建应用程序的方法，这些组件将每个应用程序进程作为微服务运行。这些微服务使用轻量级通过定义明确的接口进行通信。APIs 该架构中的每个微服务都可以更新、部署和扩展，以满足对应用程序特定功能的需求。有关更多信息，请参阅[在上实现微服务](#)。AWS

迁移加速计划 (MAP)

AWS 该计划提供咨询支持、培训和服务，以帮助组织为迁移到云奠定坚实的运营基础，并帮助抵消迁移的初始成本。MAP 提供了一种以系统的方式执行遗留迁移的迁移方法，以及一套用于自动执行和加速常见迁移场景的工具。

大规模迁移

将大部分应用程序组合分波迁移到云中的过程，在每一波中以更快的速度迁移更多应用程序。本阶段使用从早期阶段获得的最佳实践和经验教训，实施由团队、工具和流程组成的迁移工厂，通过自动化和敏捷交付简化工作负载的迁移。这是 [AWS 迁移策略](#) 的第三阶段。

迁移工厂

跨职能团队，通过自动化、敏捷的方法简化工作负载迁移。迁移工厂团队通常包括运营、业务分析师和所有者、迁移工程师、开发人员和冲刺 DevOps 领域的专业人员。20% 到 50% 的企业应用程序组合由可通过工厂方法优化的重复模式组成。有关更多信息，请参阅本内容集中[有关迁移工厂的讨论](#)和[云迁移工厂](#)指南。

迁移元数据

有关完成迁移所需的应用程序和服务器信息。每种迁移模式都需要一套不同的迁移元数据。迁移元数据的示例包括目标子网、安全组和 AWS 账户。

迁移模式

一种可重复的迁移任务，详细列出了迁移策略、迁移目标以及所使用的迁移应用程序或服务。示例：使用 AWS 应用程序迁移服务重新托管向 Amazon EC2 的迁移。

迁移组合评测 (MPA)

一种在线工具，提供了用于验证迁移到 AWS 云的业务案例的信息。MPA 提供了详细的组合评测（服务器规模调整、定价、TCO 比较、迁移成本分析）以及迁移计划（应用程序数据分析和数据收集、应用程序分组、迁移优先级排序和波次规划）。所有 AWS 顾问和 APN 合作伙伴顾问均可免费使用 [MPA 工具](#)（需要登录）。

迁移准备情况评测 (MRA)

使用 AWS CAF 深入了解组织的云就绪状态、确定优势和劣势以及制定行动计划以缩小已发现差距的过程。有关更多信息，请参阅[迁移准备指南](#)。MRA 是 [AWS 迁移策略](#)的第一阶段。

迁移策略

将工作负载迁移到 AWS 云的方法。有关更多信息，请参见术语表中的 [7 R](#) 词条，以及[动员您的组织以加快大规模迁移](#)。

ML

请参阅[机器学习](#)。

现代化

将过时的（原有的或单体）应用程序及其基础设施转变为云中敏捷、弹性和高度可用的系统，以降低成本、提高效率和利用创新。有关更多信息，请参阅[在 AWS 云中实现应用程序现代化的策略](#)。

现代化准备情况评估

一种评估方式，有助于确定组织应用程序的现代化准备情况；确定收益、风险和依赖关系；确定组织能够在多大程度上支持这些应用程序的未来状态。评估结果是目标架构的蓝图、详细说明现代化进程发展阶段和里程碑的路线图以及解决已发现差距的行动计划。有关更多信息，请参阅[在 AWS 云中评估应用程序的现代化准备情况](#)。

单体应用程序（单体式）

作为具有紧密耦合进程的单个服务运行的应用程序。单体应用程序有几个缺点。如果某个应用程序功能的需求激增，则必须扩展整个架构。随着代码库的增长，添加或改进单体应用程序的功能也会变得更加复杂。若要解决这些问题，可以使用微服务架构。有关更多信息，请参阅[将单体分解为微服务](#)。

MPA

请参阅[迁移组合评测](#)。

MQTT

请参阅[消息队列遥测传输](#)。

多分类器

一种帮助为多个类别生成预测（预测两个以上结果之一）的过程。例如，ML 模型可能会询问“这个产品是书、汽车还是手机？”或“此客户最感兴趣什么类别的产品？”

可变基础设施

一种用于更新和修改生产工作负载的现有基础设施的模型。为了提高一致性、可靠性和可预测性，Well-Architect AWS ed Framework 建议使用[不可变基础设施](#)作为最佳实践。

O

OAC

请参阅[来源访问控制](#)。

OAI

请参阅[来源访问身份](#)。

OCM

请参阅[组织变革管理](#)。

离线迁移

一种迁移方法，在这种方法中，源工作负载会在迁移过程中停止运行。这种方法会延长停机时间，通常用于小型非关键工作负载。

OI

请参阅[运营集成](#)。

OLA

请参阅[运营级别协议](#)。

在线迁移

一种迁移方法，在这种方法中，源工作负载无需离线即可复制到目标系统。在迁移过程中，连接工作负载的应用程序可以继续运行。这种方法的停机时间为零或最短，通常用于关键生产工作负载。

OPC-UA

请参阅[开放流程通信 – 统一架构](#)。

开放流程通信 – 统一架构 (OPC-UA)

一种用于工业自动化的 machine-to-machine (M2M) 通信协议。OPC-UA 提供了一个包含数据加密、身份验证和授权方案的互操作性标准。

运营级别协议 (OLA)

一项协议，阐明了 IT 职能部门承诺相互交付的内容，以支持服务水平协议 (SLA)。

运营准备情况审查 (ORR)

一份问题核对清单和关联的最佳实践，可帮助您了解、评估、预防或缩小事件和可能的故障的范围。有关更多信息，请参阅 [AWS Well-Architected Framework 中的运营准备情况审查 \(ORR \)](#)。

运营技术 (OT)

与物理环境配合使用以控制工业运营、设备和基础设施的硬件和软件系统。在制造业中，OT 和信息技术 (IT) 系统的集成是[工业 4.0](#) 转型的关键重点。

运营整合 (OI)

在云中实现运营现代化的过程，包括就绪计划、自动化和集成。有关更多信息，请参阅[运营整合指南](#)。

组织跟踪

由 AWS CloudTrail 此创建的跟踪记录组织 AWS 账户 中所有人的所有事件 AWS Organizations。该跟踪是在每个 AWS 账户 中创建的，属于组织的一部分，并跟踪每个账户的活动。有关更多信息，请参阅 CloudTrail 文档中的[为组织创建跟踪](#)。

组织变革管理 (OCM)

一个从人员、文化和领导力角度管理重大、颠覆性业务转型的框架。OCM 通过加快变革采用、解决过渡问题以及推动文化和组织变革，帮助组织为新系统和战略做好准备和过渡。在 AWS 迁移策略中，该框架被称为人员加速，因为云采用项目需要变更的速度。有关更多信息，请参阅[OCM 指南](#)。

来源访问控制 (OAC)

在中 CloudFront，一个增强的选项，用于限制访问以保护您的亚马逊简单存储服务 (Amazon S3) 内容。OAC 全部支持所有 S3 存储桶 AWS 区域、使用 AWS KMS (SSE-KMS) 进行服务器端加密，以及对 S3 存储桶的动态PUT和DELETE请求。

来源访问身份 (OAI)

在中 CloudFront，一个用于限制访问权限以保护您的 Amazon S3 内容的选项。当您使用 OAI 时，CloudFront 会创建一个 Amazon S3 可以对其进行身份验证的委托人。经过身份验证的委托人只能通过特定 CloudFront 分配访问 S3 存储桶中的内容。另请参阅[OAC](#)，其中提供了更精细和增强的访问控制。

ORR

请参阅[运营准备情况审查](#)。

OT

请参阅[运营技术](#)。

出站 (出口) VPC

在 AWS 多账户架构中，一种处理从应用程序内部启动的网络连接的 VPC。[AWS 安全参考架构](#)建议设置您的网络帐户，包括入站、出站和检查，VPCs 以保护您的应用程序与更广泛的互联网之间的双向接口。

P

权限边界

附加到 IAM 主体的 IAM 管理策略，用于设置用户或角色可以拥有的最大权限。有关更多信息，请参阅 IAM 文档中的[权限边界](#)。

个人身份信息 (PII)

直接查看其他相关数据或与之配对时可用于合理推断个人身份的信息。PII 的示例包括姓名、地址和联系信息。

PII

请参阅[个人身份信息](#)。

playbook

一套预定义的步骤，用于捕获与迁移相关的工作，例如在云中交付核心运营功能。playbook 可以采用脚本、自动化运行手册的形式，也可以是操作现代化环境所需的流程或步骤的摘要。

PLC

请参阅[可编程逻辑控制器](#)。

PLM

请参阅[产品生命周期管理](#)。

policy

一个对象，可以定义权限（请参阅[基于身份的策略](#)）、指定访问条件（请参阅[基于资源的策略](#)）或定义 AWS Organizations 的组织中所有账户的最大权限（请参阅[服务控制策略](#)）。

多语言持久性

根据数据访问模式和其他要求，独立选择微服务的数据存储技术。如果您的微服务采用相同的数据存储技术，它们可能会遇到实现难题或性能不佳。如果微服务使用最适合其需求的数据存储，则可以更轻松地实现微服务，并获得更好的性能和可扩展性。

组合评测

一个发现、分析和确定应用程序组合优先级以规划迁移的过程。有关更多信息，请参阅[评估迁移准备情况](#)。

谓词

返回 true 或 false 的查询条件，通常位于 WHERE 子句中。

谓词下推

一种数据库查询优化技术，可在传输之前筛选查询中的数据。这将减少从关系数据库检索和处理的数据量，并提高查询性能。

预防性控制

一种安全控制，旨在防止事件发生。这些控制是第一道防线，帮助防止未经授权的访问或对网络的意外更改。有关更多信息，请参阅在 AWS 上实施安全控制中的[预防性控制](#)。

主体

中 AWS 可以执行操作和访问资源的实体。此实体通常是 IAM 角色的根用户或用户。AWS 账户有关更多信息，请参阅 IAM 文档中[角色术语和概念](#)中的主体。

隐私设计

一种在整个开发过程中都考虑隐私的系统工程方法。

私有托管区

一个容器，其中包含有关您希望 Amazon Route 53 如何响应针对一个或多个 VPCs 域名及其子域名的 DNS 查询的信息。有关更多信息，请参阅 Route 53 文档中的[私有托管区的使用](#)。

主动控制

一种[安全控制](#)，旨在防止部署不合规资源。这些控制会在资源预置之前对其进行扫描。如果资源与控制不兼容，则不会预置它。有关更多信息，请参阅 AWS Control Tower 文档中的[控制参考指南](#)，并参见在上实施安全[控制中的主动](#)控制 AWS。

产品生命周期管理 (PLM)

对产品在其整个生命周期内的数据和流程的管理，从设计、开发和发布，到增长和成熟，再到衰退和淘汰。

生产环境

请参阅[环境](#)。

可编程逻辑控制器 (PLC)

在制造业中，一种高度可靠、适应性强的计算机，用于监控机器并实现制造过程自动化。

提示串接

使用一个 [LLM](#) 提示的输出作为下一个提示的输入，以生成更好的响应。该技术用于将复杂的任务分解为子任务，或者迭代地完善或扩展初步响应。它有助于提高模型响应的准确性和相关性，并允许获得更精细的个性化结果。

假名化

用占位符值替换数据集中个人标识符的过程。假名化可以帮助保护个人隐私。假名化数据仍被视为个人数据。

publish/subscribe (pub/sub)

一种支持微服务间异步通信的模式，可提高可扩展性和响应能力。例如，在基于微服务的 [MES](#) 中，微服务可以将事件消息发布到其他微服务可以订阅的频道。系统可以在不更改发布服务的情况下添加新的微服务。

Q

查询计划

一系列用于访问 SQL 关系数据库系统中的数据的步骤，类似于指令。

查询计划回归

当数据库服务优化程序选择的最佳计划不如数据库环境发生特定变化之前时。这可能是由统计数据、约束、环境设置、查询参数绑定更改和数据库引擎更新造成的。

R

RACI 矩阵

请参阅[责任、问责、咨询和知情 \(RACI \)](#)。

RAG

请参阅[检索增强生成](#)。

勒索软件

一种恶意软件，旨在阻止对计算机系统或数据的访问，直到付款为止。

RASCI 矩阵

请参阅[责任、问责、咨询和知情 \(RACI \)](#)。

RCAC

请参阅[行列访问控制](#)。

只读副本

用于只读目的的数据库副本。您可以将查询路由到只读副本，以减轻主数据库的负载。

重新架构

请参阅 [7 R](#)。

恢复点目标 (RPO)

自上一个数据恢复点以来可接受的最长时间。这决定了从上一个恢复点到服务中断之间可接受的数据丢失情况。

恢复时间目标 (RTO)

服务中断和服务恢复之间可接受的最大延迟。

重构

请参阅 [7 R](#)。

Region

地理区域内的 AWS 资源集合。每一个 AWS 区域 都相互隔离，相互独立，以提供容错、稳定性和弹性。有关更多信息，请参阅[指定您的账户可以使用的 AWS 区域](#)。

回归

一种预测数值的 ML 技术。例如，要解决“这套房子的售价是多少？”的问题 ML 模型可以使用线性回归模型，根据房屋的已知事实（如建筑面积）来预测房屋的销售价格。

重新托管

请参阅 [7 R](#)。

版本

在部署过程中，推动生产环境变更的行为。

重新放置

请参阅 [7 R](#)。

更换平台

请参阅 [7 R](#)。

重新购买

请参阅 [7 R](#)。

韧性

应用程序抵御中断或从中断中恢复的能力。在 AWS 云中规划韧性时，[高可用性](#)和[灾难恢复](#)是常见的考虑因素。有关更多信息，请参阅 [AWS 云 韧性](#)。

基于资源的策略

一种附加到资源的策略，例如 AmazonS3 存储桶、端点或加密密钥。此类策略指定了允许哪些主体访问、支持的操作以及必须满足的任何其他条件。

责任、问责、咨询和知情 (RACI) 矩阵

定义参与迁移活动和云运营的所有各方的角色和责任的矩阵。矩阵名称源自矩阵中定义的责任类型：负责 (R)、问责 (A)、咨询 (C) 和知情 (I)。支持 (S) 类型是可选的。如果包括支持，则该矩阵称为 RASCI 矩阵，如果将其排除在外，则称为 RACI 矩阵。

响应性控制

一种安全控制，旨在推动对不良事件或偏离安全基线的情况进行修复。有关更多信息，请参阅在 AWS 上实施安全控制中的[响应性控制](#)。

保留

请参阅 [7 R](#)。

停用

请参阅 [7 R](#)。

检索增强生成 (RAG)

一种[生成式人工智能](#)技术，其中 [LLM](#) 在生成响应之前引用其训练数据来源之外的权威数据来源。例如，RAG 模型可以对组织的知识库或自定义数据执行语义搜索。有关更多信息，请参阅[什么是 RAG](#)。

轮换

定期更新[密钥](#)以使攻击者更难访问凭证的过程。

行列访问控制 (RCAC)

使用已定义访问规则的基本、灵活的 SQL 表达式。RCAC 由行权限和列掩码组成。

RPO

请参阅[恢复点目标](#)。

RTO

请参阅[恢复时间目标](#)。

运行手册

执行特定任务所需的一套手动或自动程序。它们通常是为了简化重复性操作或高错误率的程序而设计的。

S

SAML 2.0

许多身份提供商 (IdPs) 使用的开放标准。此功能支持联合单点登录 (SSO)，因此用户无需在 IAM 中为组织中的所有人创建用户即可登录 AWS 管理控制台 或调用 AWS API 操作。有关基于 SAML 2.0 的联合身份验证的更多信息，请参阅 IAM 文档中的[关于基于 SAML 2.0 的联合身份验证](#)。

SCADA

请参阅[监督控制和数据采集](#)。

SCP

请参阅[服务控制策略](#)。

机密密钥

在中 AWS Secrets Manager，您以加密形式存储的机密或受限信息，例如密码或用户凭证。它由密钥值及其元数据组成。密钥值可以是二进制、单个字符串或多个字符串。有关更多信息，请参阅 Secrets Manager 文档中的[什么是 Amazon Secrets Manager 密钥？](#)。

安全设计

一种在整个开发过程中都考虑安全的系统工程方法。

安全控制

一种技术或管理防护机制，可防止、检测或降低威胁行为体利用安全漏洞的能力。安全控制有以下四种类型：[预防性](#)、[检测性](#)、[响应性](#)和[主动性](#)。

安全固化

缩小攻击面，使其更能抵御攻击的过程。这可能包括删除不再需要的资源、实施授予最低权限的最佳安全实践或停用配置文件中不必要的功能等操作。

安全信息和事件管理 (SIEM) 系统

结合了安全信息管理 (SIM) 和安全事件管理 (SEM) 系统的工具和服务。SIEM 系统会收集、监控和分析来自服务器、网络、设备和其他来源的数据，以检测威胁和安全漏洞，并生成警报。

安全响应自动化

一种预定义的程序化操作，旨在自动响应或修复安全事件。这些自动化可作为[侦探或响应式](#)安全控制措施，帮助您实施 AWS 安全最佳实践。自动响应操作的示例包括修改 VPC 安全组、修补 Amazon EC2 实例或轮换凭证。

服务器端加密

由接收数据的人在目的地对数据 AWS 服务 进行加密。

服务控制策略 (SCP)

一种策略，用于集中控制组织中所有账户的权限 AWS Organizations。SCPs 定义防护措施或限制管理员可以委托给用户或角色的操作。您可以使用 SCPs 允许列表或拒绝列表来指定允许或禁止哪些服务或操作。有关更多信息，请参阅 AWS Organizations 文档中的[服务控制策略](#)。

服务端点

的入口点的 URL AWS 服务。您可以使用端点，通过编程方式连接到目标服务。有关更多信息，请参阅 AWS 一般参考 中的[AWS 服务 端点](#)。

服务水平协议 (SLA)

一份协议，阐明了 IT 团队承诺向客户交付的内容，比如服务正常运行时间和性能。

服务水平指示器 (SLI)

对服务性能方面的衡量，例如错误率、可用性或吞吐量。

服务水平目标 (SLO)

代表服务运行状况的目标指标，由[服务水平指示器](#)衡量。

责任共担模式

描述您在云安全与合规方面共同承担 AWS 的责任的模型。AWS 负责云的安全，而您则负责云中的安全。有关更多信息，请参阅[责任共担模式](#)。

SIEM

请参阅[安全信息和事件管理系统](#)。

单点故障 (SPOF)

应用程序的单个关键组件出现故障，可能会中断系统。

SLA

请参阅[服务水平协议](#)。

SLI

请参阅[服务水平指示器](#)。

SLO

请参阅[服务水平目标](#)。

split-and-seed 模型

一种扩展和加速现代化项目的模式。随着新功能和产品发布的定义，核心团队会拆分以创建新的产品团队。这有助于扩展组织的能力和服务，提高开发人员的工作效率，支持快速创新。有关更多信息，请参阅[在 AWS 云中实现应用程序现代化的分阶段方法](#)。

SPOF

请参阅[单点故障](#)。

星型架构

一种数据库组织结构，它使用一个大型事实表来存储事务数据或测量数据，并使用一个或多个较小的维度表来存储数据属性。此结构专为在[数据仓库](#)中使用或用于商业智能目的而设计。

strangler fig 模式

一种通过逐步重写和替换系统功能直至可以停用原有的系统来实现单体系统现代化的方法。这种模式用无花果藤作为类比，这种藤蔓成长为一棵树，最终战胜并取代了宿主。该模式是由 [Martin Fowler](#) 提出的，作为重写单体系统时管理风险的一种方法。有关如何应用此模式的示例，请参阅[使用容器和 Amazon API Gateway 逐步将原有的 Microsoft ASP.NET \(ASMX \) Web 服务现代化](#)。

子网

您的 VPC 内的一个 IP 地址范围。子网必须位于单个可用区中。

监督控制和数据采集 (SCADA)

在制造业中，一种使用硬件和软件来监控实物资产和生产操作的系统。

对称加密

一种加密算法，它使用相同的密钥来加密和解密数据。

综合测试

以模拟用户交互的方式测试系统，以检测潜在问题或监控性能。您可以使用 [Amazon S CloudWatch ynthetic](#) 来创建这些测试。

系统提示

一种为 [LLM](#) 提供上下文、说明或准则以指导其行为的技术。系统提示有助于设置上下文并制定与用户交互的规则。

T

标签

键值对，用作组织资源的元数据。AWS 标签有助于您管理、识别、组织、搜索和筛选 资源。有关更多信息，请参阅[标记您的 AWS 资源](#)。

目标变量

您在监督式 ML 中尝试预测的值。这也被称为结果变量。例如，在制造环境中，目标变量可能是产品缺陷。

任务列表

一种通过运行手册用于跟踪进度的工具。任务列表包含运行手册的概述和要完成的常规任务列表。对于每项常规任务，它包括预计所需时间、所有者和进度。

测试环境

请参阅[环境](#)。

训练

为您的 ML 模型提供学习数据。训练数据必须包含正确答案。学习算法在训练数据中查找将输入数据属性映射到目标（您希望预测的答案）的模式。然后输出捕获这些模式的 ML 模型。然后，您可以使用 ML 模型对不知道目标的新数据进行预测。

中转网关

一个网络传输中心，可用于将您的网络 VPCs 和本地网络互连。有关更多信息，请参阅 AWS Transit Gateway 文档中的[什么是公交网关](#)。

基于中继的工作流程

一种方法，开发人员在功能分支中本地构建和测试功能，然后将这些更改合并到主分支中。然后，按顺序将主分支构建到开发、预生产和生产环境。

可信访问权限

向您指定的服务授予权限，该服务可代表您在其账户中执行任务。AWS Organizations 当需要服务相关的角色时，受信任的服务会在每个账户中创建一个角色，为您执行管理任务。有关更多信息，请参阅 AWS Organizations 文档中的[AWS Organizations 与其他 AWS 服务一起使用](#)。

优化

更改训练过程的各个方面，以提高 ML 模型的准确性。例如，您可以通过生成标签集、添加标签，并在不同的设置下多次重复这些步骤来优化模型，从而训练 ML 模型。

双披萨团队

一个小 DevOps 团队，你可以用两个披萨来喂食。双披萨团队的规模可确保在软件开发过程中充分协作。

U

不确定性

这一概念指的是不精确、不完整或未知的信息，这些信息可能会破坏预测式 ML 模型的可靠性。不确定性有两种类型：认知不确定性是由有限的、不完整的数据造成的，而偶然不确定性是由数据中固有的噪声和随机性导致的。有关更多信息，请参阅[量化深度学习系统中的不确定性](#)指南。

无差别任务

也称为繁重工作，即创建和运行应用程序所必需的工作，但不能为最终用户提供直接价值或竞争优势。无差别任务的示例包括采购、维护和容量规划。

上层环境

请参阅[环境](#)。

V

vacuum 操作

一种数据库维护操作，包括在增量更新后进行清理，以回收存储空间并提高性能。

版本控制

跟踪更改的过程和工具，例如存储库中源代码的更改。

VPC 对等连接

两者之间的连接 VPCs，允许您使用私有 IP 地址路由流量。有关更多信息，请参阅 Amazon VPC 文档中的[什么是 VPC 对等连接](#)。

漏洞

损害系统安全的软件缺陷或硬件缺陷。

W

热缓存

一种包含经常访问的当前相关数据的缓冲区缓存。数据库实例可以从缓冲区缓存读取，这比从主内存或磁盘读取要快。

暖数据

不常访问的数据。查询此类数据时，通常可以接受中速查询。

窗口函数

一种对与当前记录有某种关联的一组行执行计算的 SQL 函数。窗口函数对于处理任务很有用，例如计算移动平均值或根据当前行的相对位置访问行的值。

工作负载

一系列资源和代码，它们可以提供商业价值，如面向客户的应用程序或后端过程。

工作流

迁移项目中负责一组特定任务的职能小组。每个工作流都是独立的，但支持项目中的其他工作流。

例如，组合工作流负责确定应用程序的优先级、波次规划和收集迁移元数据。组合工作流将这些资产交付给迁移工作流，然后迁移服务器和应用程序。

WORM

请参阅[一次写入多次读取](#)。

WQF

请参阅[AWS 工作负载资格鉴定框架](#)。

一次写入多次读取 (WORM)

一种存储模型，可一次写入数据并防止数据被删除或修改。授权用户可以根据需要多次读取数据，但无法对其进行更改。此数据存储基础设施被认为[不可变](#)。

Z

零日漏洞利用

一种利用[零日漏洞](#)的攻击，通常为恶意软件。

零日漏洞

生产系统中不可避免的缺陷或漏洞。威胁主体可能利用这种类型的漏洞攻击系统。开发人员经常因攻击而意识到该漏洞。

零样本提示

为[LLM](#)提供执行任务的说明，但没有可以帮助指导的示例（样本）。LLM 必须使用预先训练的知识来处理任务。零样本提示的有效性取决于任务的复杂性和提示的质量。另请参阅[少样本提示](#)。

僵尸应用程序

平均 CPU 和内存使用率低于 5% 的应用程序。在迁移项目中，通常会停用这些应用程序。

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。