AWS Well-Architected Framework

可持续性支柱

可持续性支柱: AWS Well-Architected Framework

Table of Contents

可持续性支柱 – AWS Well-Architected Framework

发布日期： 2023 年 10 月 3 日 (文档修订)

本白皮书重点介绍 Amazon Web Services（AWS）Well-Architected Framework 的可持续性支柱。它提供了设计原则、操作指南、最佳实践、潜在权衡和改进计划，可用于满足 AWS 工作负载的可持续性目标。

简介

AWS Well-Architected Framework 可帮助您认识到在 AWS 上构建工作负载时所做决策的优缺点。使用该框架有助于您了解在 AWS Cloud 中设计和运行安全、可靠、高效且经济实惠的可持续工作负载的架构最佳实践。该框架提供了一种方法，使您能够根据最佳实践持续衡量架构，并确定需要改进的方面。拥有架构良好的工作负载可以极大地提高您支持业务成果的能力。

该框架基于六大支柱：

- 卓越运营
- 安全性
- 可靠性
- 性能效率
- 成本优化
- 可持续性

本文档重点介绍可持续性支柱，在可持续性范围内，侧重于环境可持续性。本文档面向的是技术人员，例如首席技术官（CTO）、架构师、开发人员和运维团队成员。

阅读本文档后，您将了解可在设计注重可持续性的云架构时使用的 AWS 最新建议和策略。通过采用本白皮书中的实践，您可以构建能够最大限度地提高效率和减少浪费的架构。

云可持续性

可持续性学科解决您的业务活动对环境、经济和社会的长期影响。如示例所示，联合国世界环境与发展委员会 将可持续发展定义为“在不损害子孙后代满足其自身需求的能力的前提下，满足当前需求的发展”。 您的企业或组织可能会对环境产生负面影响，例如直接或间接的碳排放、不可回收的废弃物以及对清洁水等共享资源的破坏。

在构建云工作负载时，可持续性实践是了解所使用服务的影响，量化整个工作负载生命周期的影响，并应用设计原则和最佳实践来减少这些影响。本文档重点介绍环境影响，尤其是能源消耗和效率，因为它们是构架师在直接采取行动以减少资源使用时依据的重要杠杆。

在专注于环境影响时，您应了解这些影响通常是如何计算的，以及对您组织自身的排放量核算的后续影响。如示例所示， 温室气体核算协议 将碳排放归入以下范围，以及云提供商（例如 AWS）的每个范围内的相关排放示例：

- 范围 1： 来自组织活动或由组织控制的所有直接排放。例如，数据中心备用发电机的燃料燃烧。
- 范围 2： 购买供数据中心和其他设施使用的电力带来的间接排放。例如，来自商用发电站的排放。
- 范围 3： 并非由组织控制的来源的活动产生的所有其他间接排放。AWS 示例包括与数据中心建设相关的排放，以及数据中心内部署的 IT 硬件的制造和运输。

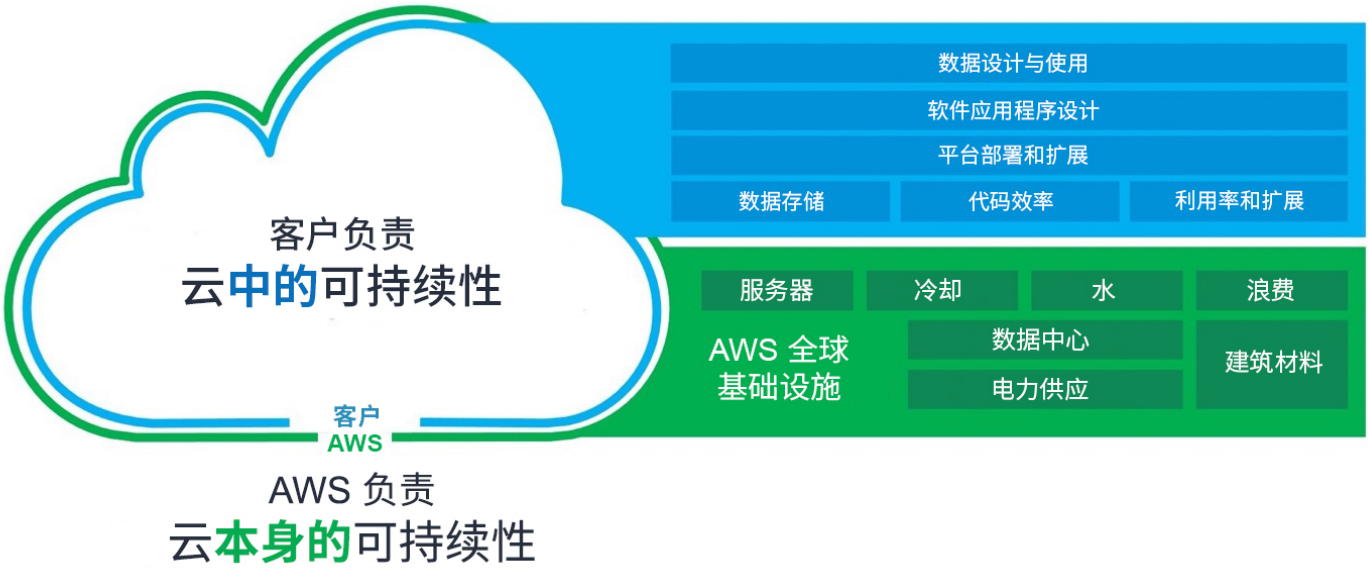
从 AWS 客户的角度来看，来自您在 AWS 上运行的工作负载的排放将被计为间接排放，并且属于范围 3 排放。部署的每个工作负载产生的排放均为来自上述每个范围的总 AWS 排放量的一小部分。实际数量因工作负载而异，具体取决于多个因素，包括使用的 AWS 服务、这些服务消耗的能源、为 AWS 数据中心供电的电网的碳排放强度以及 AWS 对可再生能源的采购。

本文档首先描述了环境可持续性的责任共担模式，然后提供了架构最佳实践，以便您能够通过减少在 AWS 数据中心运行工作负载所需的资源总量，最大程度地减小工作负载的影响。

责任共担模式

环境可持续性是客户和 AWS 共同承担的责任。

- AWS 负责优化云 本身 的可持续性 – 提供高效、共享的基础设施、水资源管理和采购可再生能源。
- 客户负责云 中 的可持续性 – 优化工作负载和资源利用，并最大限度地减少针对工作负载进行部署所需的资源总量。



责任共担模式

云本身的可持续性

与典型的本地替代方案相比，云提供商的碳足迹更低且更节能，因为他们投资于高效的供电和冷却技术，运行节能的服务器群组，并实现较高的服务器利用率。云工作负载通过利用共有资源（如联网、供电、冷却和物理设施等）来减小影响。您可以在更高效的技术出现时将您的云工作负载迁移到这些技术上，并使用基于云的服务对您的工作负载进行转型，以提高可持续性。

资源

- 迁移到 Amazon Web Services 所创造的碳减排机会
- AWS 支持可持续性解决方案

云中的可持续性

云中的可持续性是一项持续的工作，主要关注工作负载的所有组件的节能和效率，通过从预置的资源中获得最大收益，并最大限度地减少所需的总资源来达成此目标。这项工作范围很广，包括一开始就选择高效的编程语言、采用现代算法、使用高效的数据存储技术、部署到适当规模的高效计算基础设施，以及最大限度地减少对高功耗最终用户硬件的需求。

通过云实现的可持续性

除了最大限度地减少已部署工作负载所产生的影响之外，您还可以使用 AWS Cloud 运行工作负载来应对更广泛的可持续性挑战。这些挑战的示例包括减少碳排放、降低能源消耗、水的循环利用或减少企业或组织在其他方面的浪费。

通过云实现 的 可持续性可是指您使用 AWS 技术解决更广泛的可持续性挑战。例如，您可以使用机器学习服务（例如 Amazon Monitron ） 检测工业机械的异常行为。利用这些检测数据，您可以进行预测性维护，降低因意外设备故障引发环境事故的风险，并确保机械继续以最高效率运行。

云中的可持续性的设计原则

在构建云工作负载时应用这些设计原则，可以最大限度地提高可持续性，并将影响降至最低。

- 了解您的影响： 衡量您的云工作负载的影响并为您的工作负载的未来影响建模。包括所有影响来源，例如客户使用您的产品所产生的影响，以及产品最终淘汰和停用所产生的影响。通过查看每个工作单元所需的资源和排放量，将生产性输出与云工作负载的总体影响进行比较。使用这些数据来建立关键绩效指标（KPI），评估在降低影响的同时提高生产力的方法，并估计提议的更改随时间的推移所产生的影响。
- 设定可持续性目标： 对于每个云工作负载，建立长期可持续性目标，例如减少每个事务所需的计算和存储资源。针对现有工作负载的可持续性改进的投资回报进行建模，并为负责人提供投资于可持续性目标所需的资源。规划增长并构建您的工作负载，以便增长可降低影响强度（以适当的单位衡量，例如每用户或每事务）。目标可帮助您支持您的企业或组织更广泛的可持续发展目标、识别回归并确定潜在改进领域的优先级。
- 实现利用率最大化： 适当调整工作负载规模并实施高效设计，以确保高利用率并最大限度地提高底层硬件的能源效率。由于每台主机的基准功耗，两台以 30% 利用率运行的主机的效率低于一台以 60% 利用率运行的主机。同时，消除或尽可能减少空闲资源、处理和存储，以减少支持工作负载所需的总能源。
- 预测并采用更高效的新硬件和软件产品/服务： 支持您的合作伙伴和供应商进行上游改进，以帮助您减少云工作负载的影响。持续监控和评估更高效的新硬件和软件产品。设计灵活性以允许快速采用高效的新技术。
- 使用托管服务： 在庞大的客户群中共享服务有助于更充分地利用资源，从而减少支持云工作负载所需的基础设施数量。例如，客户可以通过将工作负载迁移到 AWS Cloud 并采用托管服务（例如，用于无服务器容器的 AWS Fargate，AWS 在其中大规模运行并负责其高效运行）来分散电力和网络等常见数据中心组件的影响。使用有助于将影响降至最低的托管服务，例如使用 Amazon S3 生命周期配置将不经常访问的数据自动移动到冷存储，或使用 Amazon EC2 Auto Scaling 来调整容量以满足需求。

- 减少云工作负载的下游影响： 减少使用您的服务所需的能源或资源量。减少或消除客户为了使用您的服务而升级其设备的需求。使用设备场进行测试以了解预期影响，并对客户进行测试以了解使用您服务的实际影响。

改进过程

架构改进过程包括了解您拥有什么以及您可以采取哪些改进措施，选择改进目标，测试改进，采用成功的改进，量化您取得的成功并分享您学到的经验，以便可以在其他地方复制，然后重复该过程。

您的改进目标可以是：

- 消除浪费、低利用率和闲置或未使用的资源
- 最大化您所使用资源的价值

> **Note**
>
> 使用您预置的所有资源，以尽可能少的资源完成相同的工作。

在优化的早期阶段，首先消除存在浪费或利用率低的领域，然后转向更有针对性的优化，以适应您的特定工作负载。

监控资源消耗随时间的变化。确定累积变化导致资源消耗效率低下或显著增加的地方。确定改进需求以解决消费变化并实施您优先考虑的改进。

以下各步骤设计为一个迭代过程，用于评估、划分优先级、测试和部署以可持续性为重点的云工作负载改进。

1. 确定改进目标： 根据本文档中确定的可持续性最佳实践审查您的工作负载，并确定改进目标。
2. 评估具体改进： 对可能改进的具体变更、预计成本和业务风险进行评估。
3. 优先事项和计划改进： 优先考虑以最低成本和风险提供最大改进的变更，并制定测试和实施计划。
4. 测试和验证改进：在测试环境中实施变更以验证其改进潜力。
5. 将变更部署到生产环境： 跨生产环境实施变更。
6. 衡量结果并复制成功： 寻找机会跨工作负载复制成功，并恢复具有不可接受结果的变更。

示例场景

本文档后面将引用以下示例场景来说明改进过程的每个步骤。

您的公司有一个工作负载，它在 Amazon EC2 实例上执行复杂的图像操作，并存储修改后的文件和原始文件以供用户访问。处理活动是 CPU 密集型的，而且输出文件非常大。

确定改进目标

了解可以帮助您实现可持续性目标的最佳实践。您可以在本文档后面找到这些 最佳实践 的详细描述和改进建议。

审查您的工作负载和使用的资源。识别 热点， 例如大型部署和常用资源。评估这些热点，寻找机会提高资源的有效利用率，并减少实现业务成果所需的资源总量。

根据最佳实践审查您的工作负载，并确定需要改进的地方。

将此步骤应用于 示例场景，您将以下最佳实践确定为可能的改进目标：

- 使用最少的硬件来满足您的需求
- 使用最能支持您的数据访问和存储模式的技术

资源

- 优化您的 AWS 基础设施以实现可持续性，第 I 部分：计算
- 优化您的 AWS 基础设施以实现可持续性，第 II 部分：存储
- 优化您的 AWS 基础设施以实现可持续性，第 III 部分：联网

评估具体改进

了解您的工作负载为完成一个工作单元而预置的资源。评估潜在的改进，并估计其潜在影响、实施成本和相关风险。

要衡量一段时间内的改进情况，首先要了解您在 AWS 中预置了哪些资源，以及这些资源的使用情况。

从全面了解您的 AWS 使用情况开始，并使用 AWS 成本和使用情况报告来帮助识别热点。使用此 AWS 示例代码 在 Amazon Athena 的帮助下查看和分析您的报告。

代理指标

在评估具体变更时，您还必须评估哪些指标最能量化该变更对关联资源的影响。这些指标称为 代理指标。选择最能反映您正在评估的改进类型的代理指标，以及改进所针对的资源。这些指标可能会随着时间的推移而变化。

为支持您的工作负载而预置的资源包括计算、存储和网络资源。使用代理指标评估预置的资源，以了解这些资源的使用情况。

使用代理指标来衡量为实现业务成果而预置的资源。

资源	代理指标示例	改进目标
计算	vCPU 分钟数	最大限度地利用预置资源
存储	预置容量（GB）	减少预置总量
网络	传输的流量（GB）或传输的数据包数	减少传输总量和传输距离

业务指标

选择业务指标来量化业务成果的实现情况。业务指标应反映工作负载提供的价值，例如，同时活跃的用户数、处理的 API 调用数量或完成的事务数。这些指标可能会随着时间的推移而变化。在评估基于财务的业务指标时要谨慎，因为交易价值的不一致会使比较无效。

关键性能指标

使用以下公式，将预置资源除以实现的业务成果，以确定每个工作单元的预置资源。

$$按工作单元预置的资源 = \frac{预置资源的代理指标}{成果的业务指标}$$

KPI 公式

将每个工作单元的资源用作您的 KPI。基于预置的资源建立基线，作为比较的基础。

资源	KPI 示例	改进目标
计算	每个事务的 vCPU 分钟数	最大限度地利用预置资源
存储	每个事务的 GB	减少预置总量
网络	每个事务传输的数据量（GB）或每个事务传输的数据包数	减少传输总量和传输距离

估算改进

估算改进，即减少的预置资源数量（如代理指标所示）和相对于每个工作单元的预置资源基线的变化百分比。

资源	KPI 示例	改进目标
计算	每个事务的 vCPU 分钟数减少百分比	实现利用率最大化
存储	每个事务数据量（GB）减少百分比	减少预置总量
网络	每个事务传输的数据量（GB）减少百分比或每个事务传输的数据包数减少百分比	减少传输总量和传输距离

评估改进

根据预期的净收益评估潜在的改进。评估实施和维护的时间、成本和工作量级别，以及意外影响等业务风险。

有针对性的改进通常意味着在所消耗资源类型之间进行权衡。例如，为了减少计算消耗，您可以存储结果，或者为了限制传输的数据，您可以在将结果发送到客户端之前处理数据。稍后将详细讨论这些 权衡 。

在评估工作负载的风险时包括非功能性要求，包括安全性、可靠性、性能效率、成本优化以及改进对工作负载运行能力的影响。

将此步骤应用于 示例场景，您使用以下结果评估目标改进：

最佳实践	有针对性的改进	潜在	成本	风险
使用最少的硬件来满足您的需求	实施预测性扩展以减少低利用率时段	中	低	低

最佳实践	有针对性的改进	潜在	成本	风险
使用最能支持您的数据访问和存储模式的技术	实施更有效的压缩机制以减少总存储量和实现时间	高	低	低

实施预测性调度可减少未充分利用或未使用的实例所消耗的 vCPU 小时数，相比现有扩展机制可提供适度的效益，估计可减少 11% 的资源消耗。所涉及的成本很低，包括云资源的配置和 Amazon EC2 Auto Scaling 的预测性扩展操作。当响应超出预测的需求而被动地执行横向扩展时，风险是性能会受限。

实施更有效的压缩将产生重大影响，大幅减少所有原始图像和经过处理的图像的文件大小，估计生产中的存储需求减少 25%。实施新算法是一种低工作量的替换方案，涉及的风险很小。

优先事项和计划改进

基于最大的预期影响、最低的成本和可接受的风险，优先考虑已确定的改进。

决定最初的重点放在哪些改进上，并将它们包含在资源规划和开发路线图中。

将此步骤应用于 示例场景，您可以优先考虑目标改进，如下所示：

优先级	提升	潜在	成本	风险
1	实施更有效的压缩机制	高	低	低
2	实施预测性扩展	中	低	低

更新文件压缩方法的潜在影响很高，而成本和低风险较低，使其成为公司的高价值目标，并且优先于实施预测性扩展。您确定，在完成更换文件压缩方法后，实施具有中等潜在影响、低成本和低风险的预测性扩展应该是优先改进事项。

您指派一名团队成员来实施改进的文件压缩，并将预测性扩展添加到您的待办事项列表中。

测试和验证改进

以最少的投资执行小型测试，以降低大规模工作的风险。

在测试环境中实施工作负载的代表性副本，以限制执行测试和验证的成本和风险。执行一组预定义的测试事务，测量预置的资源，并确定每个工作单元使用的资源以建立测试基线。

在测试环境中实施您的目标改进，并在相同条件下使用相同的方法重复测试。然后，在落实了改进的情况下，衡量预置的资源和每个工作单元使用的资源。

针对为每个工作单元预置的资源基线，计算相对的变化百分比，并确定生产环境中预置的资源的预期减少数量。将这些值与预期值进行比较。确定结果是否处于可接受的改进水平。评估对额外资源的消耗是否有任何权衡，使得改进的净收益不如意。

确定改进是否成功，以及是否应该投入资源在生产中实施变更。如果此时变更被评估为不成功，请重定向资源以测试和验证下一个目标，并继续改进周期。

每个工作单元的预置资源减少百分比	预置资源的减少数量	操作
达到预期	达到预期	继续改进
未达到预期	达到预期	继续改进
达到预期	未达到预期	寻求替代改进
未达到预期	未达到预期	寻求替代改进

将此步骤应用于 示例场景，您执行测试以验证是否成功。

在对改进的压缩算法执行测试后，每个工作单元预置的资源（原始图像和修改后的图像所需的存储）减少百分比达到了预期，预置的存储空间平均减少了 30%，计算负载的增加可以忽略不计。

您确定，对生产中的现有文件应用改进的压缩算法时，所需的额外计算资源与所实现的存储减少相比微不足道。您确认在所需资源（TB 存储量）的数量减少方面取得了成功，并且已批准改进用于生产部署。

将变更部署到生产环境

在生产环境中实施经过测试、验证和批准的改进。使用有限的部署进行实施，确认工作负载的功能，测试在有限的部署中预置的资源和每个工作单元所消耗资源的实际减少情况，并检查变更的意外结果。成功测试后继续进行完整部署。

如果测试失败或者您的变更遭遇不可接受的意外结果，请还原变更。

将此步骤应用于 示例场景，您采取以下操作。

您通过蓝绿部署方法使用有限的部署在生产环境中实施变更。针对新部署的实例的功能测试是成功的。您可以看到，原始图像文件和经过处理的图像文件的预置存储平均减少了 26%。您没有看到压缩新文件时计算负载增加的任何迹象。

您注意到压缩图像文件所用时间出乎意料地减少，并将此归因于新压缩算法的高度优化代码。

您继续进行新版本的全面部署。

衡量结果并复制成功

通过以下方式衡量结果并复制成功：

- 衡量每个工作单元的预置资源的初始改进和预置资源的减少数量。
- 将初始估计值和测试结果与生产测量值进行比较。确定可能导致差异的因素，并酌情更新您的估计和测试方法。
- 确定成功和成功程度，并与利益相关者分享结果。
- 如果由于测试失败或变更带来的意外负面结果而不得不还原变更，请确定造成这种情况的因素。在可行时进行迭代，或者评估新的方法以实现变更的目标。
- 利用您所学到的知识，建立标准，并将成功的改进应用到其他可以同样受益的系统。跨团队和组织捕获和共享您的方法、相关构件和净收益，以便其他人可以采用您的标准并复制您的成功。
- 监控每个工作单元的预置资源，并随着时间的推移跟踪变更和总体影响。对工作负载的变更或客户使用工作负载的方式，都可能会对改进效果产生影响。如果您注意到改进的效果在短期内显著下降，或者随着时间的推移而累积下降，请重新评估改进机会。
- 量化一段时间内您的改进带来的净收益（包括可能会有的其他团队应用您的改进获得的收益），以展示您的改进活动的投资回报。

将此步骤应用于 示例场景，您衡量以下结果。

在部署新的压缩算法并将其应用于现有图像文件后，您的工作负载显示存储需求减少 23% 的初步改进。

测量值与初始估计值（25%）基本一致，与测试值（30%）相比，造成显著差异的原因可以确定是测试中使用的图像文件未能充分代表生产环境中存在的图像文件。您可以修改测试图像集，以更恰当地反映生产中的图像。

这种改进被认为是完全成功的。预置存储的总减少量比估计的 25% 低 2%，但 23% 仍然是对可持续性影响的巨大改进，并伴随着同等的成本节约。

变更带来的唯一意想不到的结果是，执行压缩所用时间的有益减少，以及 vCPU 消耗的等效减少。这些改进归功于高度优化的代码。

您建立一个内部开源项目，在其中共享您的代码、相关构件、有关如何实施变更的指南以及实施结果。利用该内部开源项目，您的团队可以轻松地为所有持久文件存储使用案例采用相关代码。您的团队采用改进作为标准。内部开源项目的第二个好处是，每个采用该解决方案的人员都会从解决方案的改进中受益，任何人都可以为项目的改进做出贡献。

您发布成功结果，并在整个组织中共享开源项目。采用该解决方案的每个团队都可以最少的投资再现收益，增加从您的投资中获得的净收益。您将这些数据作为一个持续的成功案例发布。

随着时间的推移，您将继续监控改进的影响，并将根据需要对内部开源项目进行更改。

可持续性作为非功能性要求

在您的业务要求列表中添加可持续性，可以带来更具成本效益的解决方案。专注于从您使用的资源中获得更多的价值，减少资源使用可直接转化为 AWS 上的成本节约，因为您只需为使用的资源付费。

实现可持续性目标可能不需要在一个或多个其他传统指标（如正常运行时间、可用性或响应时间）中进行等效权衡。您可以在可持续性方面得到明显收益，而不会对服务水平产生可测量的影响。如果需要进行较小的权衡，通过这些权衡所获得的可持续性改进可能会超过服务质量的变化。

鼓励您的团队成员在开发功能需求时不断尝试可持续性改进。团队还应该在设定目标时嵌入代理指标，以确保他们在开发工作负载时评估资源密度。

以下是可以减少云资源消耗的权衡示例：

调整结果质量： 通过近似计算，您可以用结果质量（QoR，Quality of Results）来换取工作负载强度的降低。近似计算的做法可以寻找机会，利用客户需求与实际得出结果之间的差异。例如，如果您将数据放在 固定 数据结构中，您可以在 SQL 中删除 ORDER BY 运算符，以删除不必要的处理，节省资源，同时仍然提供可接受的结果。

调整响应时间： 通过在最大程度上降低共同开销，响应时间较长的应答可以减少碳排放。处理专设、临时的任务可能产生启动开销。分批对任务进行分组和处理，而不是每次在任务出现时支付处理开销。批处理以延长响应时间换得启动实例、下载源代码和运行该流程所需的共同开销的减少。

调整可用性： 利用 AWS，您只需单击几下即可添加冗余并满足高可用性目标。您可以通过诸如静态稳定性之类的技术来增加冗余，方法是预置总是导致利用率降低的空闲资源。在设定目标时评估业务需求。可用性方面的相对较小的权衡可能会带来利用率的大幅提高。例如，静态稳定性架构模式涉及到预置空闲失效转移容量，以便在组件发生故障后立即承担负载。放宽可用性要求可以消除对闲置在线容量的需求，允许花时间来自动部署替换资源。按需增加失效转移容量可以提高整体利用率，而不会在正常运营期间对业务产生影响，并且具有第二个好处，也就是降低成本。

云中可持续性的最佳实践

优化工作负载放置，针对需求、软件、数据、硬件和流程优化您的架构，以提高能源效率。这些领域中的每一个都代表着采用最佳实践的机会，以便通过最大限度地提高利用率，尽可能减少浪费以及为支持您的工作负载而部署和使用的总资源数，来减少云工作负载的可持续性影响。

主题

区域选择

为工作负载选择区域会显著影响其 KPI，包括性能、成本和碳足迹。为了有效提高这些 KPI，您应该根据业务需求和可持续发展目标为工作负载选择区域。

最佳实践

SUS01-BP01 根据业务需求和可持续发展目标选择区域

根据您的业务需求和可持续发展目标为您的工作负载选择一个区域，以优化其 KPI，包括性能、成本和碳足迹。

常见反模式：

- 您可以根据自己所在的位置选择工作负载的区域。
- 您可以将所有工作负载资源整合到一个地理位置中。

建立此最佳实践的好处：将工作负载放置在 Amazon 可再生能源项目或已发布碳强度较低的区域附近，有助于降低云工作负载的碳足迹。

在未建立这种最佳实践的情况下暴露的风险等级：中等

实施指导

AWS Cloud 是一个不断扩展的区域和入网点（PoP）网络，其全球网络基础设施将它们连接在一起。为工作负载选择区域会显著影响其 KPI，包括性能、成本和碳足迹。为了有效提高这些 KPI，您应该根据业务需求和可持续发展目标为工作负载选择区域。

实施步骤

- 按照以下步骤进行操作，根据您的业务需求（包括法规遵从性、可用功能、成本和延迟）评估工作负载的潜在区域并列入候选名单：
  - 根据您所需的当地法规，确认这些区域合规。
  - 使用 AWS 区域性服务列表，检查区域是否具有运行工作负载所需的服务和功能。
  - 使用 AWS Pricing Calculator 计算每个区域的工作负载成本。
  - 测试最终用户位置与每个 AWS 区域之间的网络延迟。
- 选择亚马逊可再生能源项目附近的区域和其电网公布的碳强度低于其他位置（或区域）的区域。
  - 确定您的相关可持续发展准则，以根据温室气体核算协议（基于市场和基于位置的方法）跟踪和比较逐年碳排放量。
  - 根据用于跟踪碳排放的方法选择区域。有关根据可持续性准则选择区域的更多详细信息，请参阅如何根据可持续性目标为工作负载选择区域。

资源

相关文档：

- 了解碳排放估算
- Amazon 遍布全球
- 可再生能源方法
- 为工作负载选择区域时应考虑的事项

相关视频：

- 可持续地构建并减少 AWS 碳足迹

符合需求

用户和应用程序使用您的工作负载及其他资源的方式可以帮助您找出改进措施，以实现可持续性目标。扩展基础设施以持续匹配需求，并验证您是否仅使用了支持用户所需的最少资源。使服务水平与客户需求保持一致。定位资源以限制用户和应用程序使用这些资源所需的网络。删除未使用的资产。为您的团队成员提供满足其需求的设备，并尽可能降低他们的可持续发展影响。

最佳实践

- SUS02-BP01 动态扩缩工作负载基础设施
- SUS02-BP02 使 SLA 与可持续性目标保持一致
- SUS02-BP03 停止创建和维护未使用的资产
- SUS02-BP04 根据其联网要求优化工作负载的地理位置
- SUS02-BP05 针对执行的活动优化团队成员资源
- SUS02-BP06 实施缓冲和节流以展平需求曲线

SUS02-BP01 动态扩缩工作负载基础设施

利用云的弹性并动态扩缩基础设施，以使云资源的供应与需求相匹配，避免在工作负载中过度调配容量。

常见反模式：

- 您没有扩缩基础设施以匹配用户负载。
- 您一直在手动扩缩基础设施。
- 在扩展事件之后，您将保留增加的容量，而不是缩减容量。

建立此最佳实践的好处：配置和测试工作负载弹性有助于有效地将云资源的供应与需求相匹配，并避免过度调配容量。您可以利用云中的弹性，在需求高峰期间和之后自动扩缩容量，以确保您只使用满足业务需求所需的适当数量的资源。

在未建立这种最佳实践的情况下暴露的风险等级：中等

实施指导

云让您能够通过各种机制灵活地动态扩展或缩减资源，以便满足不断变化的需求。供应与需求的最佳匹配提供了最低的工作负载环境影响。

需求可以是固定的，也可以是变化的，需要指标和自动化来确保管理不会变成沉重负担。应用程序可以通过修改实例大小来纵向扩展或缩减，通过修改实例数量来横向扩展或缩减，或者组合使用这两种方式。

您可以使用大量不同方法来实现资源的供需匹配。

- 目标跟踪方法：监控您的扩缩指标，并根据需要自动增加或减少容量。
- 预测性扩缩：根据每日和每周的趋势进行扩缩。
- 基于计划的方法：根据可预测的负载变化设置自己的扩缩计划。
- 服务扩缩：选择按设计可以原生扩缩或者将自动扩缩作为一项功能提供的服务（如无服务器）。

确定利用率低或无利用率的时段，缩减资源以消除过剩容量并提高效率。

实施步骤

- 弹性可根据对您拥有的资源的需求来提供这些资源。实例、容器和函数提供了弹性机制，可以与自动扩缩结合使用，也可以作为服务的一项功能。AWS 提供了一系列自动扩缩机制，以确保工作负载可以在低用户负载期间快速轻松地缩减。以下是自动扩缩机制的一些示例：

Auto scaling mechanism	Where to use
Amazon EC2 Auto Scaling	用于验证您拥有适量的 Amazon EC2 实例，可处理您应用程序的用户负载。
Application Auto Scaling	用于自动扩缩 Amazon EC2 以外的各项 AWS 服务的资源，比如 Lambda 函数或 Amazon Elastic Container Service (Amazon ECS) 服务。
Kubernetes Cluster Autoscaler	用于自动扩缩 AWS 上的 Kubernetes 集群。

- 扩缩通常与计算服务（如 Amazon EC2 实例或 AWS Lambda 函数）相关。考虑使用非计算服务配置（如 Amazon DynamoDB 读写容量单元或 Amazon Kinesis Data Streams 分片）来满足需求。
- 验证衡量扩展或缩减的指标已根据所部署的工作负载类型进行了验证。如果您正在部署一个视频转码应用程序，CPU 利用率预计为 100%，并且不应将此作为您的主要指标。如果需要，可以对扩缩策略使用自定义指标（如内存利用率）。要选择正确的指标，请考虑以下关于 Amazon EC2 的指导：
  - 该指标应该是有效的利用率指标，并描述实例的繁忙程度。

  - 该指标值必须随 Auto Scaling 组中的实例数量成比例地增加或减少。
- 对 Auto Scaling 组使用动态扩缩而不是手动扩缩。我们还建议您在动态扩缩中使用目标跟踪扩缩策略。
- 确认工作负载部署可以处理扩展事件和缩减事件。为缩减事件创建测试场景，以确认工作负载的行为符合预期，并且不会影响用户体验（如丢失粘滞会话）。您可以使用活动历史记录验证 Auto Scaling 组的扩缩活动。
- 评估您的工作负载以获得可预测的模式，并在您预期需求会发生预测和计划的变化时主动扩缩。借助预测性扩缩，您无需过度调配容量。有关详细信息，请参阅使用 Amazon EC2 Auto Scaling 进行预测性扩缩。

资源

相关文档：

- 开始使用 Amazon EC2 Auto Scaling
- 由机器学习提供支持的 EC2 预测式扩缩
- 使用 Amazon OpenSearch Service、Amazon Data Firehose 和 Kibana 分析用户行为
- 什么是 Amazon CloudWatch？
- 在 Amazon RDS 上使用性能详情监控数据库负载
- 介绍对 Amazon EC2 Auto Scaling 预测式扩缩的原生支持
- 介绍 Karpenter - 高性能开源 Kubernetes Cluster Autoscaler
- 深入探讨 Amazon ECS 集群 Auto Scaling

相关视频：

- 构建成本、能源和资源高效的计算环境
- 更好、更快、更便宜的计算：成本优化Amazon EC2（CMP202-R1）

相关示例：

- 实验室：Amazon EC2 Auto Scaling 组示例
- 实验室：使用 Karpenter 实施自动扩缩

SUS02-BP02 使 SLA 与可持续性目标保持一致

根据您的可持续发展目标审查和优化工作负载服务水平协议（SLA），以便在继续满足业务需求的同时，尽量减少支持您的工作负载所需的资源。

常见反模式：

- 工作负载 SLA 未知或模棱两可。
- 只针对可用性和性能定义您的 SLA。
- 对所有工作负载使用相同设计模式（如多可用区架构）。

建立此最佳实践的好处：使 SLA 与可持续发展目标一致，在满足业务需求的同时实现最佳资源使用率。

在未建立这种最佳实践的情况下暴露的风险等级：低

实施指导

SLA 定义云工作负载的预期服务水平，如响应时间、可用性和数据留存。它们影响云工作负载的架构、资源使用率和环境影响。定期审查 SLA，并做出权衡，显著减少资源使用，以换取可接受的服务水平降低幅度。

实施步骤

- 定义或重新设计 SLA，在支持可持续性目标的同时满足而不是超出您的业务需求。
- 做出权衡，显著降低可持续性影响，以换取可接受的服务水平降低幅度。
  - 可持续性和可靠性：高可用性工作负载往往会消耗更多资源。
  - 可持续发展和性能：使用更多资源来提升性能可能会对环境产生更大影响。
  - 可持续发展和安全：过度安全的工作负载可能会对环境产生更大影响。
- 使用优先考虑业务关键功能的设计模式（例如 AWS 上的微服务），并允许非关键功能具有较低的服务水平（例如响应时间或恢复时间目标）。

资源

相关文档：

- AWS 服务水平协议（SLA）

- Importance of Service Level Agreement for SaaS Providers

相关视频：

- 提供可持续、高性能的架构
- 构建成本、能源和资源高效的计算环境

SUS02-BP03 停止创建和维护未使用的资产

停用您的工作负载中未使用的资产，以便减少支持您的需求所需的云资源数量，并最大限度地减少浪费。

常见反模式：

- 您没有分析应用程序以查找冗余或不再需要的资产。
- 您没有移除冗余或不再需要的资产。

建立此最佳实践的好处：移除未使用的资产可释放资源并提高工作负载的整体效率。

在未建立这种最佳实践的情况下暴露的风险等级：低

实施指导

未使用的资产会消耗存储空间和计算能力等云资源。通过识别和消除这些资产，您可以释放这些资源，从而形成更高效的云架构。定期分析应用程序资产（例如预编制的报告、数据集和静态图像）和资产访问模式，以识别冗余、利用率低下的情况和潜在的淘汰目标。移除这些冗余资产以减少工作负载中的资源浪费。

实施步骤

- 使用监控工具来识别不再需要的静态资产。
- 在移除任何资产之前，评估移除它会对架构产生什么影响。
- 制定计划并移除不再需要的资产。
- 整合生成的重叠资产以消除冗余处理。
- 更新应用程序，以便不再产生和存储不需要的资产。
- 指示第三方停止生成和存储代您管理但不再需要的资产。
- 指示第三方整合代表您生成的多余资产。

- 定期审核工作负载以识别和移除未使用的资产。

资源

相关文档：

- 优化您的 AWS 基础设施以实现可持续性，第 II 部分：存储
- 如何终止我的 AWS 账户中不再需要的活动资源？

相关视频：

- 如何检查我的 AWS 账户中是否有不再需要的活动资源，然后移除它们？

SUS02-BP04 根据其联网要求优化工作负载的地理位置

为工作负载选择可缩短网络流量必须传输的距离的云位置和服务，并减少支持您的工作负载所需的总网络资源。

常见反模式：

- 您根据自己所在的位置选择工作负载的区域。
- 您可以将所有工作负载资源整合到一个地理位置中。
- 所有流量都会流经您现有的数据中心。

建立此最佳实践的好处： 将工作负载放在接近用户的地方可以提供极低的延迟，同时减少网络中的数据移动并减小对环境的影响。

未建立这种最佳实践的情况下暴露的风险等级： 中

实施指导

AWS Cloud 基础设施围绕区域、可用区、置放群组和边缘站点（例如， AWS Outposts 和 AWS Local Zones）等位置选项而构建。这些位置选项负责维护应用程序组件、云服务、边缘网络和本地数据中心之间的连接。

分析您的工作负载中的网络访问模式，以便确定如何使用这些云位置选项和缩短网络流量必须传输的距离。

实施步骤

- 分析您的工作负载中的网络访问模式，以便确定用户如何使用您的应用程序。
  - 使用监控工具，例如 Amazon CloudWatch 和 AWS CloudTrail，以便收集有关网络活动的数据。
  - 分析数据以确定网络访问模式。
- 请根据以下关键元素，为您的工作负载部署选择区域：
  - 您的可持续发展目标： 如 区域选择中所述。
  - 数据所在位置： 对于数据密集型应用程序（如大数据和机器学习），应用程序代码的运行应尽量接近数据。
  - 用户所在位置： 对于面向用户的应用程序，选择接近您工作负载用户的一个或多个区域。
  - 其他制约： 考虑成本和合规性等制约，如 为工作负载选择区域时应考虑的事项中所述。
- 对常用资产使用本地缓存或 AWS 缓存解决方案 ，以提高性能，减少数据移动并减小对环境的影响。

服务	何时使用
Amazon CloudFront	用于缓存静态内容（如图像、脚本和视频）以及动态内容（如 API 响应或 Web 应用程序）。
Amazon ElastiCache	用于缓存 Web 应用程序的内容。
DynamoDB Accelerator	用于将内存中加速添加到您的 DynamoDB 表。

- 使用有助于您在更接近工作负载用户的位置运行代码的服务：

服务	何时使用
Lambda@Edge	用于执行计算密集型操作，当对象不在缓存中时启动这些操作。
Amazon CloudFront Functions	用于处理简单使用场景，如 HTTP(S) 请求或响应操作，这些操作可由短期运行的函数启动。
AWS IoT Greengrass	用于为互联设备运行本地计算、消息收发和数据缓存。

- 使用连接池来允许连接重用并减少所需资源。
- 使用不依赖于持久连接和同步更新的分布式数据存储来保持一致性，从而为区域人口提供服务。
- 用共享的动态容量代替预先配置的静态网络容量，并与其他订阅用户共享网络容量的可持续性影响。

资源

相关文档：

- 优化您的 AWS 基础设施以实现可持续性，第 III 部分：联网
- Amazon ElastiCache 文档
- 什么是 Amazon CloudFront？
- Amazon CloudFront 主要功能

相关视频：

- 揭秘 AWS 上的数据传输
- 在新一代 Amazon EC2 实例上扩展网络性能

相关示例：

- AWS 联网研讨会
- 针对可持续性设计 – 最大限度地减少跨网络的数据移动

SUS02-BP05 针对执行的活动优化团队成员资源

优化提供给团队成员的资源，在支持其需求的同时最大程度地降低对环境可持续性的影响。

常见反模式：

- 忽略了团队成员使用的设备对云应用程序整体效率的影响。
- 手动管理和更新团队成员使用的资源。

建立此最佳实践的好处：优化团队成员资源可以提高支持云的应用程序的整体效率。

在未建立这种最佳实践的情况下暴露的风险等级：低

实施指导

了解您的团队成员用来使用服务的资源、它们的预期生命周期，以及财务和可持续性影响。实施战略以优化这些资源。例如，在利用率高的可扩展基础设施上，而不是在利用率不高的强力单用户系统上，执行渲染和编译等复杂的操作。

实施步骤

- 按照工作站和其他设备的使用方式对它们进行预置。
- 使用虚拟桌面和应用程序串流来限制升级和设备要求。
- 将处理器或内存密集型任务移至云端以利于其弹性。
- 评估流程和系统对您的设备生命周期的影响，并选择在满足业务需求的同时最大限度减少设备更换需求的解决方案。
- 对设备实施远程管理以减少所需的商务旅行。
  - AWS Systems Manager Fleet Manager 是一种统一的用户界面（UI）体验，帮助您远程管理在 AWS 上或在本地运行的节点。

资源

相关文档：

- 什么是 Amazon WorkSpaces？
- Amazon WorkSpaces 的成本优化器
- Amazon AppStream 2.0 文档
- NICE DCV

相关视频：

- 在 AWS 上管理 Amazon WorkSpaces 的成本

SUS02-BP06 实施缓冲和节流以展平需求曲线

缓冲和节流可展平需求曲线，并降低工作负载所需的预置容量。

常见反模式：

- 在不需要的时候立即处理客户端请求。
- 没有分析客户端请求的要求。

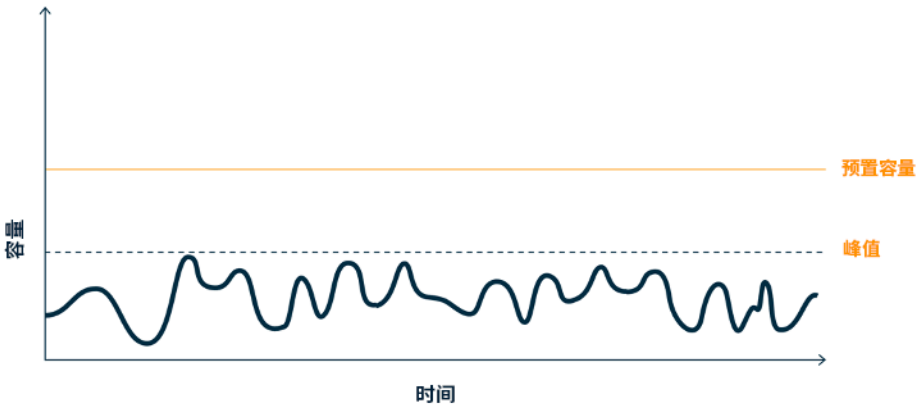
建立此最佳实践的好处：展平需求曲线可降低工作负载所需的预置容量。降低预置容量即可减少能源消耗和减少对环境的影响。

在未建立这种最佳实践的情况下暴露的风险等级：低

展平工作负载需求曲线有助于降低工作负载的预置容量和减少对环境的影响。假设工作负载的需求曲线如下图所示。此工作负载有两个峰值，为了处理这些峰值，如橙色线所示预置资源容量。因为需要预置容量来处理这两个峰值，所以此工作负载所使用的资源和能量不是由需求曲线下的区域表示，而是由预置容量线下面的区域表示。

需求曲线具有两个不同的峰值，需要高预置容量。

您可以使用缓冲和节流来修改需求曲线和弄平峰值，这意味着可以减少预置容量和消耗的能量。在客户端可以执行重试时实施节流。实施缓冲以存储请求并将处理任务往后推迟一段时间。



节流对需求曲线和预置容量的影响。

实施步骤

- 分析客户端请求以确定如何对它们作出响应。要考虑的问题包括：
  - 是否可以异步处理此请求？
  - 客户端是否具有重试能力？
- 如果客户端有重试能力，则您可以实施节流，它会告诉需求源，如果当前无法处理请求，则应稍后再试。
  - 您可以使用 Amazon API Gateway 来实施节流。
- 对于无法执行重试的客户端，则需要实施缓冲以展平需求曲线。缓冲会延迟请求处理，从而让以不同速率运行的应用程序可以有效通信。基于缓冲的方法使用队列或流来接受来自生产方的消息。然后消息将由使用方读取并处理，这样消息就能够以满足使用方业务需求的速率运行。
  - Amazon Simple Queue Service（Amazon SQS）是一项托管服务，提供允许单个使用方读取单个消息的队列。
  - Amazon Kinesis 提供允许众多使用方读取相同消息的流。
- 分析总体需求、变化率和所需的响应时间，以使所需节流或缓冲的大小适宜。

资源

相关文档：

- 开始使用 Amazon SQS
- 使用队列和消息的应用程序集成

相关视频：

- 为分布式应用程序选择适合的消息传递服务

软件和架构

实施用于执行负载平滑和保持已部署资源始终如一的高利用率的模式，以最大限度地减少资源消耗。由于用户行为会随着时间的推移而发生变化，因此组件可能会因缺乏使用而变得空闲。修改模式和架构以整合未充分利用的组件，从而提高整体利用率。停用不再需要的组件。了解工作负载组件的性能，并优

化消耗资源最多的组件。注意客户用来访问您服务的设备，并实施相应的模式以最大限度地减少设备升级需要。

最佳实践

- SUS03-BP01 针对异步和计划作业优化软件和架构
- SUS03-BP02 删除或重构很少或没有使用的工作负载组件
- SUS03-BP03 优化消耗最多时间或资源的代码区域
- SUS03-BP04 优化对设备的影响
- SUS03-BP05 使用最能支持数据访问和存储模式的软件模式和架构

SUS03-BP01 针对异步和计划作业优化软件和架构

使用高效的软件和架构模式（如队列驱动）来保持所部署资源的始终如一的高利用率。

常见反模式：

- 为了应对不可预见的需求高峰，您过度预置云工作负载中的资源。
- 架构不会通过消息传递组件分离异步消息的发送方和接收方。

建立此最佳实践的好处：

- 高效的软件和架构模式可以最大程度地减少工作负载中未使用的资源，并提高整体效率。
- 可以独立于异步消息的接收来扩缩处理。
- 通过消息传递组件，可以放宽可用性要求，从而能够用更少的资源来满足这些要求。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

使用高效的架构模式（如事件驱动型架构），均匀地利用组件并最大程度地减少工作负载中的过度预置。使用高效的架构模式可以最大程度地减少由于需求随时间变化而导致的闲置资源。

了解工作负载组件的要求，并采用可提高资源总体利用率的架构模式。停用不再需要的组件。

实施步骤

- 分析工作负载的需求，以确定如何响应这些需求。

- 对于不需要同步响应的请求或作业，请使用队列驱动型架构和自动扩缩工作线程来最大限度地提高利用率。以下是一些可以考虑采用队列驱动型架构的示例：

Queuing mechanism	Description
AWS Batch 作业队列	AWS Batch 作业将提交到作业队列，并一直驻留在队列中，直到可以计划在计算环境中运行。
Amazon Simple Queue Service 和 Amazon EC2 竞价型实例	将 Amazon SQS 实例和竞价型实例配对，构建容错又高效的架构。

- 对于可以随时处理的请求或作业，请使用调度机制批量处理作业以提高效率。以下是 AWS 上的调度机制的一些示例：

Scheduling mechanism	Description
Amazon EventBridge 计划程序	Amazon EventBridge 提供的一项功能，允许您大规模创建、运行和管理计划任务。
AWS Glue 基于时间的计划	在 AWS Glue 中为爬网程序和作业定义基于时间的计划。
Amazon Elastic Container Service (Amazon ECS) 计划任务	Amazon ECS 支持创建计划任务。计划任务使用 Amazon EventBridge 规则按计划或响应 EventBridge 事件来运行任务。
实例计划程序	为您的 Amazon EC2 和 Amazon Relational Database Service 实例配置启动和停止计划。

- 如果在架构中使用轮询和 Webhook 机制，请将它们替换为事件。使用事件驱动型架构构建高效的工作负载。
- 利用 AWS 上的无服务器来消除过量配置的基础设施。
- 适当调整架构中各个组件的大小，以防止等待输入的闲置资源。

资源

相关文档：

- 什么是 Amazon Simple Queue Service？
- 什么是 Amazon MQ？
- 基于 Amazon SQS 进行扩缩
- 什么是 AWS Step Functions？
- 什么是 AWS Lambda？
- 将 AWS Lambda 与 Amazon SQS 配合使用
- 什么是 Amazon EventBridge？

相关视频：

- 迁移到事件驱动型架构

SUS03-BP02 删除或重构很少或没有使用的工作负载组件

移除未使用且不再需要的组件，并重构利用率低的组件，以最大限度减少工作负载中的浪费。

常见反模式：

- 没有定期检查工作负载的各个组件的利用率水平。
- 没有检查和分析 AWS 合理调整大小工具（如 AWS Compute Optimizer）的建议。

建立此最佳实践的好处：移除未使用的组件可最大限度减少浪费并提高云工作负载的整体效率。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

检查您的工作负载以识别空闲或未使用的组件。这是一个迭代改进过程，可以通过需求变化或新云服务的发布来触发。例如，AWS Lambda 函数执行时间显著缩短，这可能是需要降低内存大小的指标。此外，随着 AWS 发布新的服务和功能，适用于您的工作负载的最佳服务和架构可能会发生变化。

持续监控工作负载活动并寻找机会来提高单个组件的利用水平。通过删除空闲组件并执行合理调整大小活动，您就可以使用最少的云资源来满足您的业务需求。

实施步骤

- 监控和捕获工作负载关键组件的利用率指标（例如 Amazon CloudWatch 指标中的 CPU 利用率、内存利用率或网络吞吐量）。

- 对于稳定的工作负载，定期检查 AWS 合理调整大小工具（如 AWS Compute Optimizer），以便识别空闲、未使用或未充分利用的组件。
- 对于临时工作负载，请评估利用率指标以识别空闲、未使用或未充分利用的组件。
- 停用不再需要的组件及关联资产（如 Amazon ECR 映像）。
- 重构未充分利用的组件或将其与其他资源整合以提高利用效率。例如，您可以在单个 Amazon RDS 数据库实例中预置多个小数据库，而不是在单个未充分利用的实例中运行数据库。
- 了解您的工作负载为完成一个工作单元而预置的资源。

资源

相关文档：

- AWS Trusted Advisor
- 什么是 Amazon CloudWatch？
- 自动清理 Amazon ECR 中未使用的映像

相关示例：

- Well-Architected 实验室 - 使用 AWS Compute Optimizer 合理调整大小
- Well-Architected 实验室 - 优化硬件模式并遵守可持续性 KPI

SUS03-BP03 优化消耗最多时间或资源的代码区域

优化在架构的不同组件中运行的代码，以最大限度地减少资源使用和提高性能。

常见反模式：

- 忽略为资源使用优化代码。
- 通常通过增加资源来应对性能问题。
- 代码审核和开发过程不会跟踪性能变化。

建立此最佳实践的好处： 使用高效的代码可以最大限度地减少资源使用量并提高性能。

未建立这种最佳实践的情况下暴露的风险等级： 中

实施指导

至关重要的是检查每个功能区域（包括云架构应用程序的代码）以优化其资源使用和性能。持续监控工作负载在构建环境和生产中的性能，并确定改进资源使用率特别高的代码片段的机会。采用定期审核流程来识别代码中资源使用效率低下的错误或反模式。利用可为您的使用场景产生相同结果的简单和高效算法。

实施步骤

- 在开发工作负载时，采用自动化代码审核流程来提高质量并识别错误和反模式。
  - 使用 Amazon CodeGuru Reviewer 自动审查代码
  - 使用 Amazon CodeGuru 检查并发错误
  - 使用 Amazon CodeGuru 提高 Python 应用程序的代码质量
- 在运行工作负载时，监测资源以将单个工作单元的资源需求高的组件确定为代码审核目标。
- 对于代码审核，使用代码分析器确定使用时间最长或使用资源最多的代码区域作为优化目标。
  - 借助 Amazon CodeGuru Profiler 减少组织的碳排放
  - 使用 Amazon CodeGuru Profiler 了解 Java 应用程序中的内存使用
  - 通过 Amazon CodeGuru Profiler 改进客户体验并降低成本
- 对工作负载使用最高效的操作系统和编程语言。有关节能编程语言（包括 Rust）的详细信息，请参阅 Rust 可持续性。
- 使用可产生相同结果的更简单、更高效算法取代计算密集型算法。
- 删除排序和格式等不必要的代码。

资源

相关文档：

- 什么是 Amazon CodeGuru Profiler？
- FPGA 实例
- 在 AWS 上进行构建所需工具的 AWS SDK

相关视频：

- 使用 Amazon CodeGuru Profiler 提高代码效率

- 使用 Amazon CodeGuru 自动提供代码审核和应用程序性能建议

SUS03-BP04 优化对设备的影响

了解您的架构中使用的设备，并使用策略来减少其使用。这可以最大限度地减少云工作负载对环境的整体影响。

常见反模式：

- 忽略客户所用设备对环境的影响。
- 手动管理和更新客户使用的资源。

建立此最佳实践的好处：实施针对客户设备优化的软件模式和功能可以减少云工作负载对环境的整体影响。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

实施针对客户设备优化的软件模式和功能可以从几个方面减少对环境的影响。

- 实施向后兼容的新功能可以减少硬件更换次数。
- 优化应用程序以在设备上高效运行，这有助于降低能耗和延长电池寿命（如果它们由电池供电）。
- 针对设备优化应用程序还可以减少网络上的数据传输。

了解架构中使用的设备、其预期生命周期以及更换这些组件产生的影响。实施软件模式和功能，有助于最大程度地降低设备能耗、减少客户更换设备和手动升级设备的需求。

实施步骤

- 列出您架构中使用的设备。设备可以是移动设备、平板电脑、物联网设备、智能灯，甚至是工厂中的智能设备。
- 优化设备上运行的应用程序：
  - 使用策略（例如在后台运行任务）来降低能耗。
  - 在构建有效负载时考虑网络带宽和延迟，并实施有助于您的应用程序在低带宽、高延迟链路上良好运行的功能。

  - 将有效负载和文件转换为设备所需的优化格式。例如，您可以使用 Amazon Elastic Transcoder 或 AWS Elemental MediaConvert 将大型、高质量的数字媒体文件转换为用户可以在移动设备、平板电脑、Web 浏览器和联网电视上播放的格式。
  - 在服务器端执行计算密集型活动（例如图像渲染），或使用应用程序串流来改善旧设备上的用户体验。
  - 对输出进行分段和分页，尤其是对于交互式会话，以管理有效负载并限制本地存储要求。
- 使用自动化空中下载（OTA）机制将更新部署到一个或多个设备。
  - 您可以使用 CI/CD 管道更新移动应用程序。
  - 您可以使用 AWS IoT Device Management 大规模地远程管理互联设备。
- 要测试新功能和更新，请使用具有代表性硬件集的托管式设备场，并迭代开发以最大限度增加支持的设备数。有关更多详细信息，请参阅 SUS06-BP04 使用托管式设备场进行测试。

资源

相关文档：

- 什么是 AWS Device Farm？
- Amazon AppStream 2.0 文档
- NICE DCV
- 在运行 FreeRTOS 的设备上更新固件的 OTA 教程

相关视频：

- AWS Device Farm 简介

SUS03-BP05 使用最能支持数据访问和存储模式的软件模式和架构

了解数据在工作负载中的使用方式、用户使用数据的方式，以及数据的传输和存储方式。使用最能支持数据访问和存储的软件模式和架构，最大限度地减少支持工作负载所需的计算、网络和存储资源。

常见反模式：

- 假设所有工作负载都具有相似的数据存储和访问模式。
- 假设所有工作负载都位于一个存储层，且只使用该存储层。
- 假设数据访问模式会随着时间的推移保持一致。

- 您的架构支持潜在的高数据访问突发，这会导致资源大部分时间都处于空闲状态。

建立此最佳实践的好处：根据数据访问和存储模式选择和优化架构将有助于降低开发复杂性并提高总体利用率。了解何时使用全局表、数据分区和缓存将帮助您减少运营开销，并根据您的工作负载需求进行扩展。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

使用最符合您的数据特性和访问模式的软件和架构模式。例如，使用 AWS 上的现代数据架构使您可以使用为您的独特分析应用场景优化的专用服务。这些架构模式可提高数据处理效率和减少资源使用。

实施步骤

- 分析您的数据特性和访问模式，以便确定云资源的适合配置。要考虑的关键特征包括：
  - 数据类型：结构化、半结构化、非结构化
  - 数据增长：有界、无界
  - 数据耐久性：持久、短暂、瞬时
  - 访问模式：读或写、更新频率、峰值或一致
- 使用最能支持数据访问和存储模式的架构模式。
  - 让我们来构建！ 现代数据架构
  - AWS 上的数据库：根据作业选择合适工具
- 使用可以原生处理压缩数据的技术。
- 为您架构中的数据处理使用专用分析服务。
- 使用最能支持您的主导查询模式的数据库引擎。管理您的数据库索引以确保高效的查询执行。有关更多详情，请参阅 AWS 数据库。
- 选择可减少架构中所用网络容量的网络协议。

资源

相关文档：

- Athena 压缩支持文件格式
- 使用 Amazon Redshift 从列数据格式复制

- 在 Firehose 中转换您的输入记录格式
- AWS Glue 中 ETL 输入和输出的格式选项
- 通过转换为列格式提高 Amazon Athena 上的查询性能
- 使用 Amazon Redshift 从 Amazon S3 加载压缩数据文件
- 使用 Amazon Aurora 上的 Performance Insights 监视数据库负载
- 使用 Amazon RDS 上的 Performance Insights 监视数据库负载
- Amazon S3 Intelligent-Tiering 存储类

相关视频：

- 在 AWS 上构建现代数据架构

数据管理

实施数据管理实践以减少支持工作负载所需的预置存储，以及使用存储所需的资源。了解您的数据，并使用最能支持数据的商业价值及其使用方式的存储技术和配置。当需求减少时，将数据移到更高效、性能更低的存储中，并删除不再需要的数据。

最佳实践

- SUS04-BP01 实施数据分类策略
- SUS04-BP02 使用支持数据访问和存储模式的技术
- SUS04-BP03 使用策略管理数据集的生命周期
- SUS04-BP04 使用弹性和自动化来扩展数据块存储或文件系统
- SUS04-BP05 删除不需要或多余的数据
- SUS04-BP06 使用共享文件系统或存储来访问通用数据
- SUS04-BP07 最大限度地减少跨网络的数据移动
- SUS04-BP08 仅在难以重新创建时备份数据

SUS04-BP01 实施数据分类策略

对数据进行分类，以了解其对业务成果的重要性，并选择合适的节能存储层来存储数据。

常见反模式：

- 您没有识别正在处理或存储的具有类似特征（如敏感性、业务关键性或监管要求）的数据资产。
- 您没有实施数据目录来清点您的数据资产。

建立此最佳实践的好处：实施数据分类策略使您能够为数据确定最节能的存储层。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

数据分类涉及识别在由组织拥有或运营的信息系统中正在处理和存储的数据类型。它还涉及到对数据的重要性以及数据泄露、丢失或滥用的可能影响进行判断。

实施数据分类策略时，要从数据的使用情境进行反推，并创建一个分类方案，该方案考虑到特定数据集对组织运营的重要程度。

实施步骤

- 对您的工作负载存在的各种数据类型进行清点。
  - 有关数据分类类别的更多详情，请参阅《数据分类》白皮书。
- 根据给组织带来的风险，确定数据的重要性、机密性、完整性和可用性。使用这些要求将数据分组到您采用的数据分类层之一。
  - 例如，请参阅将数据分类并保护初创公司的四个简单步骤。
- 针对未标记和未分类的数据定期审核您的环境，并对数据进行适当的分类和标记。
  - 例如，请参阅 AWS Glue 中的数据目录和爬网程序。
- 建立一个提供审计和治理功能的数据目录。
- 确定并记录每个数据类的处理过程。
- 使用自动化来持续审计您的环境，以识别未标记和未分类的数据，并适当地对这些数据进行分类和标记。

资源

相关文档：

- 利用 AWS Cloud 支持数据分类
- AWS Organizations 中的标记策略

相关视频：

- 在 AWS 上实现敏捷的数据治理

SUS04-BP02 使用支持数据访问和存储模式的技术

使用最能支持您的数据访问和存储方式的存储技术，以在支持您的工作负载的同时最大限度地减少预置资源。

常见反模式：

- 假设所有工作负载都具有相似的数据存储和访问模式。
- 假设所有工作负载都位于一个存储层，且只使用该存储层。
- 假设访问模式始终保持不变。

建立此最佳实践的好处： 根据数据访问和存储模式选择和优化您的存储技术，有助于您减少满足业务需求所需的云资源，并提高云工作负载的整体效率。

未建立这种最佳实践的情况下暴露的风险等级： 低

实施指导

选择最适合您的访问模式的存储解决方案，或者考虑根据存储解决方案更改访问模式，以便尽可能提高性能和效率。

- 评估您的数据特征和访问模式，以收集您的存储需求的关键特征。要考虑的关键特征包括：
  - 数据类型： 结构化、半结构化、非结构化
  - 数据增长： 限界、不限界
  - 数据持久性： 持久、短暂、瞬时
  - 访问模式： 读或写、频率、峰值或一致
- 将数据迁移到支持您的数据特征和访问模式的适当存储技术。下面是 AWS 存储技术的一些示例以及它们的关键特征：

类型	技术	主要特征
对象存储	Amazon S3	一项对象存储服务，具有无限的可扩展性、高可用性和多

类型	技术	主要特征
		种可访问性选项。在 Amazon S3 内外传输和访问对象可以使用诸如 Transfer Acceleration 或 Access Points之类的服务，来满足您的位置、安全需求和访问模式。
存档存储	Amazon S3 Glacier	Amazon S3 的存储类，用于数据归档。
共享文件系统	Amazon Elastic File System（Amazon EFS）	可由多种类型的计算解决方案访问的可挂载文件系统。Amazon EFS 会自动增大和缩小存储，并进行性能优化以提供一致的低延迟。
共享文件系统	Amazon FSx	基于最新 AWS 计算解决方案而构建，支持四种常用文件系统：NetApp ONTAP、OpenZFS、Windows 文件服务器和 Lustre。Amazon FSx 延迟、吞吐量和 IOPS 因文件系统而不同，因此，在为您的工作负载需求选择合适的文件系统时应考虑这些因素。
数据块存储	Amazon Elastic Block Store（Amazon EBS）	设计用于 Amazon Elastic Compute Cloud（Amazon EC2）的可扩展、高性能的块存储服务。Amazon EBS 包括用于事务性、IOPS 密集型工作负载的基于 SSD 的存储，以及用于吞吐量密集型工作负载的基于 HDD 的存储。

类型	技术	主要特征
关系数据库	Amazon Aurora、Amazon RDS、Amazon Redshift	旨在支持 ACID（原子性、一致性、隔离性、持久性）事务，并保持参照完整性和数据强一致性。许多传统应用程序、企业资源规划（ERP）、客户关系管理（CRM）和电子商务系统都使用关系数据库来存储数据。
键值数据库	Amazon DynamoDB	已针对常见的访问模式进行优化，通常用于存储和检索大量数据。键值数据库的典型使用案例包括高流量 Web 应用程序，电子商务系统和游戏应用程序。

- 对于固定大小的存储系统（例如 Amazon EBS 或 Amazon FSx），请监控可用的存储空间，并在达到阈值时自动分配存储空间。您可以利用 Amazon CloudWatch 来收集和分析 Amazon EBS 和 Amazon FSx的不同指标。
- Amazon S3 存储类可以在对象级别进行配置，一个桶可以包含存储在所有存储类中的对象。
- 您也可以使用 Amazon S3 生命周期策略，在不对应用程序进行任何更改的情况下，于存储类之间自动转换对象或删除数据。通常来说，在考虑这些存储机制时，您必须在资源效率、访问延迟和可靠性之间做出取舍。

资源

相关文档：

- Amazon EBS 卷类型
- Amazon EC2 实例存储
- Amazon S3 Intelligent-Tiering
- Amazon EBS I/O 特性
- 使用 Amazon S3 存储类
- 什么是 Amazon S3 Glacier？

相关视频：

- AWS 上数据湖的架构模式
- 深入讨论 Amazon EBS（STG303-R1）
- 利用 Amazon S3 优化存储性能（STG343）
- 在 AWS 上构建现代数据架构

相关示例：

- Amazon EFS CSI 驱动程序
- Amazon EBS CSI 驱动程序
- Amazon EFS 实用程序
- Amazon EBS 自动扩展
- Amazon S3 示例

SUS04-BP03 使用策略管理数据集的生命周期

管理所有数据的生命周期并自动执行删除，以最大限度地减少工作负载所需的总存储。

常见反模式：

- 手动删除数据。
- 不删除任何工作负载数据。
- 不根据数据的保留和访问要求将数据移动到更节能的存储层。

建立此最佳实践的好处：使用数据生命周期策略可确保在工作负载中高效访问和保留数据。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

数据集在其生命周期中通常具有不同的保留和访问要求。例如，应用程序可能需要在有限的时间段内频繁访问某些数据集。之后，这些数据集很少被访问。

要在数据集的整个生命周期内高效管理数据集，请配置生命周期策略，这些策略是定义如何处理数据集的规则。

使用生命周期配置规则，您可以指示特定存储服务将数据集转换到更节能的存储层、将其存档或删除。

实施步骤

- 对工作负载中的数据集进行分类。
- 定义每个数据类的处理过程。
- 设置自动化生命周期策略以强制实施生命周期规则。以下是如何为不同 AWS 存储服务设置自动化生命周期策略的一些示例：

Storage service	How to set automated lifecycle policies
Amazon S3	您可以使用 Amazon S3 生命周期在对象的整个生命周期中对其进行管理。如果访问模式未知、不断变化或不可预测，可以使用 Amazon S3 智能分层，此功能可监控访问模式，并自动将尚未访问的对象移动到成本较低的访问层。您可以利用 Amazon S3 Storage Lens 指标来识别生命周期管理中的优化机会和差距。
Amazon Elastic Block Store	您可以使用 Amazon Data Lifecycle Manager 自动创建、保留和删除 Amazon EBS 快照和 Amazon EBS 支持的 AMI。
Amazon Elastic File System	Amazon EFS 生命周期管理可自动管理文件系统的文件存储。
Amazon Elastic Container Registry	Amazon ECR 生命周期策略通过根据期限或计数使映像过期来自动清理容器映像。
AWS Elemental MediaStore	您可以使用对象生命周期策略来控制对象应在 MediaStore 容器中存储多长时间。

- 删除未使用的卷、快照和超出保留期的数据。利用本机服务功能（如 Amazon DynamoDB 生存时间或 Amazon CloudWatch 日志保留）进行删除。
- 在适当情况下根据生命周期规则汇总和压缩数据。

资源

相关文档：

- 通过 Amazon S3 存储类分析优化 Amazon S3 生命周期规则
- 使用 AWS Config 规则 评估资源

相关视频：

- 利用 Amazon S3 生命周期简化数据生命周期并优化存储成本
- 使用 Amazon S3 Storage Lens 降低存储成本

SUS04-BP04 使用弹性和自动化来扩展数据块存储或文件系统

随着数据的增长，使用弹性和自动化来扩展数据块存储或文件系统，以便最大限度减少总预置存储。

常见反模式：

- 购买大型数据块存储或文件系统以备将来需要。
- 过度预置文件系统的每秒输入和输出操作数（IOPS）。
- 不监控数据卷的利用率。

建立此最佳实践的好处：最大限度地减少存储系统的过度预置可减少空闲资源并提高工作负载的整体效率。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

根据适合工作负载的大小分配、吞吐量和延迟，创建数据块存储和文件系统。随着数据的增长，使用弹性和自动化来扩展数据块存储或文件系统，而无需过度预置这些存储服务。

实施步骤

- 对于固定大小存储（例如 Amazon EBS），请确保您正在监控已使用存储量与总体存储量大小之间的关系，如果可以，请创建自动化流程，以便在达到阈值时增加存储大小。
- 使用弹性卷和托管式数据块数据服务，随着持久性数据的增长自动分配额外的存储。例如，您可以使用 Amazon EBS 弹性卷来更改卷大小、卷类型或调整 Amazon EBS 卷的性能。

- 为您的文件系统选择适合的存储类、性能模式和吞吐量模式，以满足您的业务需求，不要超过这个需求。
  - Amazon EFS 性能
  - Linux 实例上的 Amazon EBS 卷性能
- 为您的数据卷设置目标利用率水平，并调整超出预期范围的卷大小。
- 合理调整只读卷的大小以适应数据。
- 将数据迁移到对象存储，以避免使用数据块存储上的固定卷大小预置多余容量。
- 定期检查弹性卷和文件系统，终止空闲卷并缩减过度预置的资源，以适应当前数据大小。

资源

相关文档：

- Amazon FSx 文档
- 什么是 Amazon Elastic File System？

相关视频：

- 深入了解 Amazon EBS 弹性卷
- 提高性能和节省成本的 Amazon EBS 和快照优化策略
- 使用最佳实践优化 Amazon EFS 以节省成本和提高性能

SUS04-BP05 删除不需要或多余的数据

删除不需要或多余的数据，以最大程度地减少存储数据集所需的存储资源。

常见反模式：

- 复制可以轻松获取或重新创建的数据。
- 备份所有数据时不考虑其重要性。
- 只不定期地删除数据、操作事件时删除数据，或者根本不删除数据。
- 无论存储服务的持久性如何，都冗余地存储数据。
- 在没有任何业务理由的情况下启用 Amazon S3 版本控制。

建立此最佳实践的好处：删除不需要的数据可减少工作负载所需的存储大小和工作负载对环境的影响。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

请勿存储不需要的数据。自动删除不需要的数据。使用技术在文件和数据块级别进行重复数据删除。利用服务的本机数据复制和冗余功能。

实施步骤

- 评估是否可以通过使用 AWS Data Exchange 中的现有公开可用数据集，以及 AWS 上的开放数据来避免存储数据。
- 使用可以在数据块和对象级别删除重复数据的机制。以下是有关如何删除 AWS 上的重复数据的一些示例：

Storage service	Deduplication mechanism
Amazon S3	使用 AWS Lake Formation FindMatches 通过新的 FindMatches ML 转换在数据集（包括没有标识符的数据集）中查找匹配的记录。
Amazon FSx	在 Amazon FSx for Windows 上启用重复数据删除。
Amazon Elastic Block Store 快照	快照是增量备份，这意味着只保存拍摄最新快照之后出现更改的设备上的数据块。

- 分析数据访问以识别不需要的数据。自动执行生命周期策略。利用本机服务功能（如 Amazon DynamoDB 生存时间、Amazon S3 生命周期或 Amazon CloudWatch 日志保留）进行删除。
- 使用 AWS 上的数据虚拟化功能在源头维护数据并避免数据重复。
  - AWS 上的云原生数据虚拟化
  - 实验：使用 Amazon Redshift 数据共享优化数据模式
- 使用可进行增量备份的备份技术。
- 利用 Amazon S3 的持久性和 Amazon EBS 的复制性来实现持久性目标，而不是使用自我管理技术[如独立磁盘冗余阵列（RAID）]。
- 集中日志和跟踪数据，对相同的日志条目进行重复数据删除，并在需要时建立调整详细程度的机制。
- 仅在合理的情况下预填充缓存。

- 建立缓存监控和自动化以相应地调整缓存大小。
- 推送新版本的工作负载时，从对象存储和边缘缓存中删除过时的部署和资产。

资源

相关文档：

- 更改 CloudWatch Logs 中的日志数据留存
- Amazon FSx for Windows File Server 上的重复数据删除
- Amazon FSx for ONTAP 的功能，包括重复数据删除
- 使 Amazon CloudFront 上的文件失效
- 使用 Amazon EFS 备份和还原 AWS Backup 文件系统
- 什么是 Amazon CloudWatch Logs？
- 在 Amazon RDS 上使用备份

相关视频：

- 使用适用于 AWS Lake Formation 的 ML Transforms 进行模糊匹配和重复数据删除

相关示例：

- 如何使用 Amazon Athena 分析我的 Amazon S3 服务器访问日志？

SUS04-BP06 使用共享文件系统或存储来访问通用数据

采用共享文件系统或存储以避免数据重复，并为您的工作负载提供更高效的基础设施。

常见反模式：

- 为每个客户端预置存储。
- 未卸下不活动的客户端的数据卷。
- 不提供跨平台和系统的存储访问。

建立此最佳实践的好处：使用共享文件系统或存储实现将数据共享到一个或多个使用者，而无需复制数据。这有助于减少工作负载所需的存储资源。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

如果您有多个用户或应用程序访问同一个数据集，则使用共享存储技术很重要，这可以为工作负载提供高效的基础设施。共享存储技术提供一个位置来集中存储和管理数据集并避免数据重复。它还加强了不同系统之间数据的一致性。此外，因为多个计算资源会同时并行访问和处理数据，所以利用共享存储技术可以更高效地使用计算能力。

仅在需要时才从这些共享存储服务中提取数据，并卸下未使用的卷以释放资源。

实施步骤

- 当数据具有多个使用者时，将数据迁移到共享存储。下面是 AWS 上的共享存储技术的一些示例：

Storage option	When to use
Amazon EBS 多重挂载	Amazon EBS 多重挂载使您可以将单个预调配 IOPS SSD（io1 或 io2）卷挂载到同一可用区中的多个实例。
Amazon EFS	请参阅何时选择 Amazon EFS。
Amazon FSx	请参阅选择 Amazon FSx 文件系统。
Amazon S3	不需要文件系统结构而旨在与对象存储一起使用的应用程序可以使用 Amazon S3 作为可大规模扩展、持久、低成本的对象存储解决方案。

- 仅在需要时将数据复制到共享文件系统或从共享文件系统提取数据。例如，您可以创建采用 Amazon S3 的 Amazon FSx for Lustre 文件系统，并仅将处理作业所需的数据子集加载到 Amazon FSx。
- 根据您的使用模式适当删除数据，如SUS04-BP03 使用策略管理数据集的生命周期所述。
- 将卷与未积极使用它们的客户端分离。

资源

相关文档：

- 将文件系统链接到 Amazon S3 存储桶

- 在无服务器应用程序中使用适用于 AWS Lambda 的 Amazon EFS
- Amazon EFS Intelligent-Tiering 通过改变访问模式优化工作负载成本
- 将 Amazon FSx 与本地数据存储库配合使用

相关视频：

- 使用 Amazon EFS 进行存储成本优化

SUS04-BP07 最大限度地减少跨网络的数据移动

使用共享文件系统或对象存储来访问通用数据，并最大限度地减少支持工作负载数据移动所需的总网络资源。

常见反模式：

- 不管数据用户位于何处，将所有数据存储在同一个 AWS 区域。
- 在网络中移动数据之前不优化数据大小和格式。

建立此最佳实践的好处： 优化跨网络的数据移动可以减少工作负载所需的总网络资源，并降低对环境的影响。

未建立这种最佳实践的情况下暴露的风险等级： 中

实施指导

在组织中移动数据需要计算、网络和存储资源。使用相应的技术最大程度地减少数据移动并提高工作负载的整体效率。

实施步骤

- 在 为工作负载选择区域时，考虑将与数据或用户的接近程度作为决策因素。
- 按区域对使用的服务进行分区，以便将其特定于区域的数据存储在使用它的区域内。
- 使用高效的文件格式（如 Parquet 或 ORC），并在通过网络移动数据之前先对其进行压缩。
- 不移动未使用的数据。一些让您能够避免移动未使用数据的示例：
  - 将 API 响应缩减到仅针对相关数据。
  - 聚合详细数据（不需要记录级别信息）。

  - 请参阅 Well-Architected 实验室 – 使用 Amazon Redshift 数据共享优化数据模式。
  - 考虑 AWS Lake Formation 中的跨账户数据共享。
- 使用有助于您在更接近工作负载用户的位置运行代码的服务。

服务	何时使用
Lambda@Edge	用于计算密集型操作，当对象不在缓存中时会运行这些操作。
CloudFront Functions	用于简单使用场景（如 HTTP(S) 请求/响应操作），这些操作可由短期运行的函数启动。
AWS IoT Greengrass	为互联设备运行本地计算、消息收发和数据缓存。

资源

相关文档：

- 优化您的 AWS 基础设施以实现可持续性，第 III 部分：联网
- AWS 全球基础设施
- Amazon CloudFront 主要功能，包括 CloudFront 全球边缘网络
- 在 Amazon OpenSearch Service 中压缩 HTTP 请求
- 使用 Amazon EMR 进行中间数据压缩
- 将压缩数据文件从 Amazon S3 加载到 Amazon Redshift
- 通过 Amazon CloudFront 提供压缩文件

相关视频：

- 揭秘 AWS 上的数据传输

相关示例：

- 针对可持续性设计 – 最大限度地减少跨网络的数据移动

SUS04-BP08 仅在难以重新创建时备份数据

避免备份没有商业价值的数据，尽量减少工作负载的存储资源需求。

常见反模式：

- 没有为数据制定备份策略。
- 备份可以轻松重新创建的数据。

建立此最佳实践的好处：避免备份非关键数据可减少工作负载所需的存储资源并降低其对环境的影响。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

避免备份不必要的数据有助于降低成本和减少工作负载使用的存储资源。仅备份具有商业价值或满足合规性要求所必需的数据。检查备份策略并在恢复方案中排除没有价值的临时存储。

实施步骤

- 实施数据分类策略，如SUS04-BP01 实施数据分类策略中所述。
- 根据重要性对数据进行分类，并根据恢复时间目标（RTO）和恢复点目标（RPO）设计备份策略。避免备份非关键数据。
  - 排除可以轻松重新创建的数据。
  - 从备份中排除临时数据。
  - 排除数据的本地副本，除非从公共位置恢复该数据所需的时间会超过您的服务等级协议（SLA）。
- 使用自动化解决方案或托管服务来备份关键业务数据。
  - AWS Backup 是一项完全托管式服务，可让您轻松集中自动化云端和本地不同 AWS 服务的数据保护。有关如何使用 AWS Backup 创建自动备份的动手实践指导，请参阅 Well-Architected 实验室 - 测试数据的备份和恢复。
  - 使用 AWS Backup 自动备份和优化 Amazon EFS 的备份成本。

资源

相关最佳实践：

- REL09-BP01 识别和备份需要备份的所有数据，或从源复制数据

- REL09-BP03 自动执行数据备份
- REL13-BP02 使用定义的恢复策略来实现恢复目标

相关文档：

- 使用 AWS Backup 备份和还原 Amazon EFS 文件系统
- Amazon EBS 快照
- 在 Amazon Relational Database Service 上使用备份
- APN 合作伙伴：可以帮助进行备份的合作伙伴
- AWS Marketplace：可以用于备份的产品
- 备份 Amazon EFS
- 适用于 Windows File Server 的备份 Amazon FSx
- Amazon ElastiCache for Redis 的备份和还原

相关视频：

- AWS re:Invent 2021 - 使用 AWS 进行备份、灾难恢复和勒索软件防护
- AWS Backup 演示：跨账户和跨区域备份
- AWS re:Invent 2019：深入了解 AWS Backup，主讲：Rackspace (STG341)

相关示例：

- Well-Architected 实验室 - 测试数据的备份与还原
- Well-Architected 实验室 - 面向分析工作负载的备份和还原（具备失效自动恢复功能）
- Well-Architected 实验室 - 灾难恢复 - 备份与还原

硬件和服务

寻找机会，通过更改硬件管理实践来降低工作负载可持续性影响。最大限度地减少预置和部署所需的硬件数量，并为您的各项工作负载选择最高效的硬件和服务。

最佳实践

- SUS05-BP01 使用最少的硬件来满足您的需求

- SUS05-BP02 使用影响最小的实例类型
- SUS05-BP03 使用托管服务
- SUS05-BP04 优化基于硬件的计算加速器的使用

SUS05-BP01 使用最少的硬件来满足您的需求

为您的工作负载使用最少的硬件，高效地满足您的业务需求。

常见反模式：

- 不监控资源使用率。
- 架构中有利用率较低的资源。
- 没有检查静态硬件的利用率以确定是否应调整大小。
- 没有根据业务 KPI 为计算基础设施设置硬件利用率目标。

建立此最佳实践的好处：合理调整云资源的大小有助于减少工作负载对环境的影响、节省资金并保持性能基准。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

以最佳方式选择工作负载所需的硬件总数，以提高其整体效率。AWS Cloud 让您能够通过各种机制（如 AWS Auto Scaling）灵活地动态扩展或缩减资源数，并满足不断变化的需求。它还提供 API 和 SDK，让您可以轻松修改资源。使用这些功能经常更改工作负载实施。此外，按照 AWS 工具中的合理调整大小准则高效地运营您的云资源和满足您的业务需求。

实施步骤

- 选择尽可能满足您的需求的实例类型。
  - 如何为我的工作负载选择适当的 Amazon EC2 实例类型？
  - 基于属性为 Amazon EC2 实例集选择实例类型。
  - 使用基于属性的实例类型选择来创建 Auto Scaling 组。
- 通过小增量扩缩来适应可变的工作负载。
- 使用多个计算购买选项，在实例灵活性、可扩展性和成本节省方面实现平衡。

  - 按需型实例最适合新的、有状态和突增工作负载，这些工作负载不能灵活地调整实例类型、位置或时间。
  - 竞价型实例为容错和灵活的应用程序提供了很好的补充选择。
  - 将 Compute Savings Plans 用于稳定状态的工作负载，当您的需求（如可用区、区域、实例系列或实例类型）发生变化时，这种工作负载可以允许灵活性。
- 使用实例和可用区多样性最大限度地提高应用程序可用性和尽可能利用过剩的容量。
- 使用来自 AWS 工具的合理调整大小建议来调整工作负载。
  - AWS Compute Optimizer
  - AWS Trusted Advisor
- 协商服务等级协议（SLA），允许暂时减少容量，同时利用自动化功能部署替换资源。

资源

相关文档：

- 优化您的 AWS 基础设施以实现可持续性，第 I 部分：计算
- 基于属性为 Amazon EC2 实例集的 Auto Scaling 选择实例类型
- AWS Compute Optimizer 文档
- 运行 Lambda：性能优化
- 弹性伸缩文档

相关视频：

- 构建成本、能源和资源高效的计算环境

相关示例：

- Well-Architected 实验室 - 在启用 AWS Compute Optimizer 和内存利用率的情况下合理调整大小（级别 200)

SUS05-BP02 使用影响最小的实例类型

持续监控和使用新实例类型以充分利用能源效率改进。

常见反模式：

- 您只使用一个系列的实例。
- 您只使用 x86 实例。
- 您在 Amazon EC2 Auto Scaling 配置中指定一种实例类型。
- 您使用 AWS 实例的方式与其预期用途不匹配（例如，您将计算优化的实例用于内存密集型工作负载）。
- 您没有定期评估新的实例类型。
- 您不查看 AWS 合理调整大小工具（如 AWS Compute Optimizer）的建议。

建立此最佳实践的好处： 通过使用节能且大小合适的实例，您可以大大减小工作负载对环境的影响并降低其成本。

未建立这种最佳实践的情况下暴露的风险等级： 中

实施指导

在云工作负载中使用高效的实例对于降低资源使用率和成本效益至关重要。持续监控新实例类型的发布并利用能效改进，包括那些旨在支持特定工作负载（例如机器学习训练和推理以及视频转码）的实例类型。

实施步骤

- 学习和探索可以减小工作负载对环境影响的实例类型。
  - 订阅 AWS 新增功能 及时了解新增的 AWS 技术和实例。
  - 了解不同的 AWS 实例类型。
  - 通过观看如下视频，了解基于 AWS Graviton 的实例（这些实例在 Amazon EC2 中每瓦能耗方面提供出色性能）： re:Invent 2020 - 深入了解 AWS Graviton2 处理器提供支持的 Amazon EC2 实例 和 深入了解 AWS Graviton3 和 Amazon EC2 C7g 实例。
- 规划工作负载并将其转换为影响极小的实例类型。
  - 定义一个流程来评估工作负载的新功能或实例。利用云中的敏捷性，快速测试新的实例类型如何改善工作负载的环境可持续性。使用代理指标来衡量完成一个单元的工作需要多少资源。
  - 如有可能，修改工作负载以使用不同数量的 vCPU 和不同数量的内存，以最大限度地增加您的实例类型选项。
  - 考虑将工作负载转换为基于 Graviton 的实例，以提高工作负载的性能效率。
    - AWS Graviton Fast Start
    - 将工作负载转换为基于 AWS Graviton 的 Amazon Elastic Compute Cloud 实例时的注意事项

    - 适用于 ISV 的 AWS Graviton2
  - 考虑选择 AWS Graviton 选项（在使用 AWS 托管服务时）。
  - 将工作负载迁移到提供对可持续性影响极小的实例且仍满足您的业务要求的区域。
  - 对于机器学习工作负载，请利用特定于工作负载的专用硬件，例如 AWS Trainium、 AWS Inferentia和 Amazon EC2 DL1。 Inf2 等 AWS Inferentia 实例，相比同类同类 Amazon EC2 实例，性能功耗比提升了 50%
  - 使用 Amazon SageMaker Inference Recommender 来合理调整机器学习推理端点的大小。
  - 对于突增工作负载（不经常需要额外容量的工作负载），请使用 可突增性能实例。
  - 对于无状态和容错工作负载，请使用 Amazon EC2 竞价型实例 提高云的整体利用率并减少未使用资源对可持续性的影响。
- 运营和优化您的工作负载实例。
  - 对于临时工作负载，请评估 实例 Amazon CloudWatch 指标 （例如 `CPUUtilization` ），以确定实例是空闲还是未充分利用。
  - 对于稳定的工作负载，请定期检查 AWS 合理调整大小工具（如 AWS Compute Optimizer ），以确定优化和合理调整实例大小的机会。
    - Well-Architected 实验室 - 合理调整大小建议
    - Well-Architected 实验室 - 使用 Compute Optimizer 合理调整大小
    - Well-Architected 实验室 - 优化硬件模式并观察可持续性 KPI

资源

相关文档：

- 优化您的 AWS 基础设施以实现可持续性，第 I 部分：计算
- AWS Graviton
- Amazon EC2 DL1
- Amazon EC2 容量预留实例集
- Amazon EC2 竞价型实例集
- 函数：Lambda 函数配置
- 基于属性为 Amazon EC2 实例集选择实例类型。
- 在 AWS 上构建可持续、高效且优化成本的应用程序
- Contino 可持续发展控制面板如何助力客户减少碳排放

相关视频：

- 深入了解 AWS Graviton2 处理器提供支持的 Amazon EC2 实例
- 深入了解 AWS Graviton3 和 Amazon EC2 C7g 实例
- 构建成本、能源和资源高效的计算环境

相关示例：

- 解决方案：关于在 AWS 上优化深度学习工作负载以实现可持续性的指导
- Well-Architected 实验室 - 合理调整大小建议
- Well-Architected 实验室 - 使用 Compute Optimizer 合理调整大小
- Well-Architected 实验室 - 优化硬件模式并观察可持续性 KPI
- Well-Architected 实验室 - 将服务迁移到 Graviton

SUS05-BP03 使用托管服务

使用托管服务在云中更高效地运营。

常见反模式：

- 使用利用率低的 Amazon EC2 实例来运行应用程序。
- 内部团队仅管理工作负载，而没有时间专注于创新或简化。
- 为可在托管服务上更高效运行的任务部署和维护技术。

建立此最佳实践的好处：

- 使用托管服务将责任转移给 AWS，其拥有对数百万客户的洞察，可以帮助推动新的创新和提高效率。
- 由于使用了多租户控制面板，托管服务将服务的环境影响分散到许多用户。

在未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

托管服务将维持已部署硬件的高利用率和可持续性优化的责任转移给 AWS。托管服务还消除了维护服务的运营和管理负担，让您的团队有更多时间专注于创新。

审核您的工作负载，以便确定可由 AWS 托管服务替换的组件。例如，Amazon RDS、Amazon Redshift 和 Amazon ElastiCache 提供托管数据库服务。Amazon Athena、Amazon EMR 和 Amazon OpenSearch Service 提供托管分析服务。

实施步骤

1. 清点工作负载的服务和组件。
2. 评测和确定可由托管服务替换的组件。以下是一些可以考虑采用托管服务的示例：

Task	What to use on AWS
托管数据库	使用托管 Amazon Relational Database Service（Amazon RDS）实例而不是在 Amazon Elastic Compute Cloud（Amazon EC2）上维护自己的 Amazon RDS 实例。
托管容器工作负载	使用 AWS Fargate，而不是实施您自己的容器基础设施。
托管 Web 应用	使用 AWS Amplify 托管作为完全托管式 CI/CD，为静态网站和服务器端呈现的 Web 应用托管服务。

3. 确定依赖项和创建迁移计划。相应地更新运行手册和行动手册。
   - AWS Application Discovery Service 会自动收集并提供有关应用程序依赖项和利用率的详细信息，帮助您在制定迁移计划时做出更明智的决策
4. 迁移到托管服务之前测试服务。
5. 使用迁移计划，用托管服务来代替自托管服务。
6. 迁移完成后持续监控服务，以便根据需要进行调整并优化服务。

资源

相关文档：

- AWS Cloud 产品
- AWS 总拥有成本（TCO）计算器
- Amazon DocumentDB

- Amazon Elastic Kubernetes Service（EKS）
- Amazon Managed Streaming for Apache Kafka（Amazon MSK）

相关视频：

- 使用 AWS Managed Services 实现大规模云运营

SUS05-BP04 优化基于硬件的计算加速器的使用

优化加速型计算实例的使用，以减少工作负载的物理基础架构需求。

常见反模式：

- 不监控 GPU 使用情况。
- 将通用实例用于工作负载，而专用实例可以提供更高的性能、更低的成本和更高的性能功耗比。
- 使用基于硬件的计算加速器来完成任务，而使用基于 CPU 的替代方案能更高效地完成任务。

建立此最佳实践的好处：通过优化基于硬件的加速器的使用，您能够减少工作负载对物理基础设施的需求。

未建立这种最佳实践的情况下暴露的风险等级：中

实施指导

如果需要高处理能力，可以受益于使用加速型计算实例，这些实例提供对基于硬件的计算加速器的访问，例如图形处理单元（GPU）和现场可编程门阵列（FPGA）。这些硬件加速器能够比基于 CPU 的替代方案更有效地执行某些功能，例如图形处理或数据模式匹配。许多加速工作负载（如渲染、转码和机器学习）在资源使用方面变化很大。仅在需要时运行此硬件，并在不需要时自动停用它们，以最大限度地减少资源消耗。

实施步骤

- 确定哪些 加速型计算实例 可以满足您的要求。
- 对于机器学习工作负载，请利用特定于工作负载的专用硬件，例如 AWS Trainium、 AWS Inferentia和 Amazon EC2 DL1。Inf2 等 AWS Inferentia 实例相比 同类 Amazon EC2 实例，性能功耗比提升了 50%。

- 收集加速型计算实例的使用情况指标。例如，您可以使用 CloudWatch 代理，为 GPU 收集各种指标，例如 `utilization_gpu` 和 `utilization_memory` ，如 使用 Amazon CloudWatch 收集 NVIDIA GPU 指标中所示。
- 优化硬件加速器的代码、网络运营和设置，确保底层硬件得到充分利用。
  - 优化 GPU 设置
  - 深度学习 AMI 中的 GPU 监控和优化
  - 优化 I/O 以实现 Amazon SageMaker 中深度学习训练的 GPU 性能优化
- 使用最新的高性能库和 GPU 驱动程序。
- 使用自动化功能在不使用 GPU 实例时将其释放。

资源

相关文档：

- 加速计算
- 让我们来构建！ 使用自定义芯片和加速器来构建
- 如何为我的工作负载选择合适的 Amazon EC2 实例类型？
- Amazon EC2 VT1 实例
- 选择最佳 AI 加速器和模型编译，以使用 Amazon SageMaker 进行计算机视觉推理

相关视频：

- 如何选择 Amazon EC2 GPU 实例进行深度学习
- 部署经济高效的深度学习推理

流程和文化

寻找机会，通过对开发、测试和部署实践进行更改来降低可持续性影响。

最佳实践

- SUS06-BP01 采用可以快速引入可持续性改进的方法
- SUS06-BP02 让您的工作负载保持最新状态
- SUS06-BP03 提高构建环境的利用率

- SUS06-BP04 使用托管式设备场进行测试

SUS06-BP01 采用可以快速引入可持续性改进的方法

采用方法和流程来验证潜在的改进、最大限度降低测试成本和带来一些小改进。

常见反模式：

- 仅在项目开始时才完成一次审核应用程序可持续性。
- 工作负载变得过时，因为发布过程过于繁琐，无法为提高资源效率而引入微小的更改。
- 未制定相应的机制来提高工作负载的可持续性。

建立此最佳实践的好处：通过建立流程以引入和跟踪可持续性改进，您将能够不断采用新特性和功能、消除问题和提高工作负载效率。

在未建立这种最佳实践的情况下暴露的风险等级：中等

实施指导

在将潜在可持续性改进部署到生产环境之前，对其进行测试和验证。在计算改进的潜在未来收益时，考虑测试成本。开发低成本的测试方法，以实现细微的改进。

实施步骤

- 在您的开发待办事项中添加可持续性改进要求。
- 使用迭代改进流程来识别、评测、测试和部署这些改进并确定其优先顺序。
- 持续改进和简化您的开发流程。例如，使用持续集成和持续交付（CI/CD）管道测试和部署潜在的改进，自动完成软件交付过程，从而减少工作量和减少手动操作引起的错误。
- 使用最小可行代表性组件开发和测试潜在改进，从而降低测试成本。
- 持续评测改进的影响并根据需要作出调整。

资源

相关文档：

- AWS 支持可持续性解决方案

- 基于 AWS CodeCommit 的可扩展敏捷开发实践

相关视频：

- 提供可持续、高性能的架构

相关示例：

- Well-Architected 实验室 - 将成本和使用情况报告转化为效率报告

SUS06-BP02 让您的工作负载保持最新状态

让您的工作负载保持最新状态，采用高效功能、消除问题和提高工作负载的整体效率。

常见反模式：

- 假设当前的架构是静态的，不会随着时间的推移而更新。
- 您没有任何系统（也不会定期）评估更新的软件和软件包是否与您的工作负载兼容。

建立此最佳实践的好处：通过建立一个及时更新工作负载的流程，您可以采用新的特性和功能，解决问题，并提高工作负载效率。

在未建立这种最佳实践的情况下暴露的风险等级：低

实施指导

最新的操作系统、运行时、中间件、库和应用程序可以提高工作负载效率，并简化更高效技术的采用。最新的软件可能还包括更准确地衡量工作负载对可持续性的影响的功能，因为供应商提供的功能是为了满足其自身的可持续性目标。定期更新，以便使用最新的功能和版本让您的工作负载保持最新。

实施步骤

- 定义一个流程和计划来评估工作负载的新功能或实例。利用云中的敏捷性，快速测试新功能如何改善工作负载以：
  - 减小对可持续性的影响。
  - 提升性能效率。
  - 为计划改进消除障碍。

    - 提高衡量和管理可持续性影响的能力。
- 盘点工作负载软件和架构，并确定需要更新的组件。
    - 您可以使用 AWS Systems Manager 清单从 Amazon EC2 实例收集操作系统（OS）、应用程序和实例元数据，并快速了解哪些实例正在运行您的软件策略所需的软件和配置，以及哪些实例需要更新。
- 了解如何更新工作负载的组件。

Workload component	How to update
机器映像	使用 EC2 Image Builder 来管理适用于 Linux 或 Windows Server 映像的亚马逊云机器镜像（AMI）更新。
容器映像	将 Amazon Elastic Container Registry (Amazon ECR) 与现有管道配合使用以管理 Amazon Elastic Container Service (Amazon ECS) 映像。
AWS Lambda	AWS Lambda 包括版本管理功能。

- 采用自动化更新流程，以减少部署新功能的工作量，并减少手动过程引起的错误。
    - 您可以使用 CI/CD 自动更新 AMI、容器映像以及与您的云应用程序相关的其他构件。
    - 您可以使用 AWS Systems Manager Patch Manager 等工具自动执行系统更新流程，并使用 AWS Systems Manager 维护窗口安排活动。

资源

相关文档：

- AWS 架构中心
- AWS 新增功能
- AWS 开发人员工具

相关示例：

- Well-Architected 实验室 - 清单和补丁管理

- 实验室：AWS Systems Manager

SUS06-BP03 提高构建环境的利用率

提高资源利用率，以开发、测试和构建工作负载。

常见反模式：

- 手动预置或终止构建环境。
- 使构建环境保持独立于测试、构建或发布活动运行（例如，在开发团队成员的工作时间之外运行环境）。
- 为构建环境过度预置资源。

建立此最佳实践的好处：通过提高构建环境的利用率，您可以提高云工作负载的整体效率，同时将资源分配给构建者，以便高效地进行开发、测试和构建。

在未建立这种最佳实践的情况下暴露的风险等级：低

实施指导

使用自动化和基础设施即代码功能，在需要时启动构建环境，并在不使用时将其关闭。一种常见模式是安排与开发团队成员的工作时间相吻合的可用时段。您的测试环境与生产配置非常相似。但是，寻找机会使用具有突增容量的实例类型、Amazon EC2 竞价型实例、自动扩展数据库服务、容器和无服务器技术，以使开发和测试容量与使用容量保持一致。限制数据量，使之刚好满足测试要求。如果在测试中使用生产数据，请探索共享生产数据，而无需四处移动数据的可能性。

实施步骤

- 使用基础设施即代码来预置构建环境。
- 使用自动化功能来管理开发和测试环境的生命周期，并最大限度地提高构建资源的效率。
- 使用策略来最大程度地利用开发和测试环境。
  - 使用最小可行代表性环境来开发和测试潜在的改进。
  - 如果可能，请使用无服务器技术。
  - 使用按需型实例来补充您的开发人员设备。
  - 使用具有容量暴增的实例类型、竞价型实例和其他技术，使构建容量与使用容量保持一致。
  - 采用原生云服务来实现安全的实例 Shell 访问，而不是部署堡垒主机群。
  - 根据构建作业自动扩展构建资源。

资源

相关文档：

- AWS Systems Manager Session Manager
- Amazon EC2 可突增性能实例
- 什么是 AWS CloudFormation？
- 什么是 AWS CodeBuild？
- AWS 上的实例计划程序

相关视频：

- 持续集成最佳实践

SUS06-BP04 使用托管式设备场进行测试

使用托管式设备场在一组具有代表性的硬件上高效地测试新功能。

常见反模式：

- 在各个物理设备上手动测试和部署应用程序。
- 未在真实的物理设备上使用应用测试服务进行测试以及与应用（例如，Android、iOS 和 Web 应用）互动。

建立此最佳实践的好处：使用托管式设备场来测试具有云功能的应用程序，这提供了许多好处：

- 包括可在各种设备上测试应用程序的更高效功能。
- 无需使用内部基础设施进行测试。
- 提供多种设备类型（包括不太常用的较旧硬件），从而不需要进行不必要的设备升级。

在未建立这种最佳实践的情况下暴露的风险等级：低

实施指导

使用托管式设备场有助于简化在一组有代表性的硬件上测试新功能的过程。托管式设备场提供多种设备类型，包括不太常用的较旧硬件，并避免不必要的设备升级对客户可持续性的影响。

实施步骤

- 定义您的测试要求和计划（例如，测试类型、操作系统和测试时间表）。
  - 您可以使用 Amazon CloudWatch RUM 收集和分析客户端数据并制定测试计划。
- 选择可支持您的测试要求的托管式设备场。例如，您可以使用 AWS Device Farm 来测试和了解您的更改对一组具有代表性的硬件的影响。
- 使用持续集成/持续部署（CI/CD）来安排和运行测试。
  - 将 AWS Device Farm 与 CI/CD 管道集成，以便运行跨浏览器的 Selenium 测试
  - 使用 AWS DevOps 和移动服务构建和测试 iOS 和 iPadOS 应用
- 持续审核测试结果，必要时进行改进。

资源

相关文档：

- AWS Device Farm 设备列表
- 查看 CloudWatch RUM 控制面板

相关示例：

- 适用于 Android 的 AWS Device Farm 示例应用
- 适用于 iOS 的 AWS Device Farm 示例应用
- 适用于 AWS Device Farm 的 Appium Web 测试

相关视频：

- 使用 Amazon CloudWatch RUM 通过最终用户洞察优化应用程序

总结

越来越多的组织正在制定可持续发展目标，以应对政府监管、竞争优势以及客户、员工和投资者需求的变化。CTO、架构师、开发人员和运营团队成员正在寻找行之有效的方法来加快实现组织的可持续性目标。通过使用 AWS 服务支持的这些设计原则和最佳实践，您可以做出明智的决策，确保在安全性、成本、性能、可靠性和卓越运营与 AWS Cloud 工作负载的可持续性成果之间达到平衡。您为减少资源使用量和提高工作负载效率而采取的每项行动都有助于减少对环境的影响，并且有助于组织实现更广泛的可持续性目标。

贡献者

本文档的贡献者包括：

- Sam Mokhtari，Amazon Web Services Senior Efficiency Lead Solutions Architect
- Brendan Sisson，Amazon Web Services Principal Sustainability Solutions Architect
- Margaret O'Toole，Amazon Web Services Sustainability Tech Leader
- Steffen Grunwald，Amazon Web Services Principal Sustainability Solutions Architect
- Ryan Eccles，Amazon Sustainability 部门的 Principal Engineer
- Rodney Lester，Amazon Web Services Principal Architect
- Adrian Cockcroft，Amazon Web Services Sustainability Architecture 部门的 VP
- Ian Meyers，Amazon Web Services Solutions Architecture 部门的 Director of Technology

延伸阅读

有关更多信息，请参阅：

- AWS Well-Architected
- AWS Architecture Center
- 云中的可持续性
- AWS 支持可持续性解决方案
- The Climate Pledge
- 联合国可持续发展目标
- 温室气体核算协议

文档修订

要获得有关此白皮书的更新通知，请订阅 RSS 源。

变更	说明	日期
更新了风险等级	对最佳实践风险等级进行了次要更新。	October 3, 2023
更新了最佳实践指南	根据以下领域的新指南更新了最佳实践： 符合需求、软件和架构、数据和 硬件和服务访问 AWS 资源。	July 13, 2023
针对新框架进行了更新	为最佳实践更新了规范性指南并增加了新的最佳实践。	April 10, 2023
已更新白皮书	为最佳实践更新了新的实施指导。	December 15, 2022
已更新白皮书	扩展了最佳实践并增加了改进计划。	October 20, 2022
原始版本	发布了可持续性支柱 – AWS Well-Architected Framework。	December 2, 2021

声明

客户负责对本文档中的信息进行独立评估判断。本文档：（a）仅供参考，（b）代表 AWS 当前的产品服务和实践，如有变更，恕不另行通知，以及（c）不构成 AWS 及其附属公司、供应商或授权商的任何承诺或保证。AWS 产品或服务均“按原样”提供，没有任何明示或暗示的担保、声明或条件。AWS 对其客户的责任和义务由 AWS 协议规定，本文档与 AWS 和客户之间签订的任何协议无关，亦不影响任何此类协议。

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the AWS 词汇表 Reference.