高性能计算剖析

AWS 架构完善的框架

2019 年 12 月

声明

客户有责任对本文档中的信息进行独立评估。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 协议决定，本文档与 AWS 和客户之间签订的任何协议无关，亦不影响任何此类协议。

目录

摘要

本文档描述了适用于 AWS 架构完善的框架的**高性能计算 (HPC) 剖析**。文档涵盖了常见的 HPC 场景，并指明了确保工作负载架构设计符合最佳实践的关键元素。

简介

AWS 架构完善的框架能够帮助您认识到您在 AWS 上构建系统时所做决策的优缺点。[1] 使用此框架即可了解在云中设计和运行可靠、安全、高效且经济实惠的系统的架构最佳实践。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。我们相信，拥有架构完善的系统能够大大提高实现业务成功的可能性。

在这篇“剖析”中，我们重点介绍如何在 AWS 云中设计、部署**高性能计算 (HPC) 工作负载**，以及进行架构设计。HPC 工作负载在云中的运行状况极佳。HPC 工作负载的自然消长和爆发特征，使其非常适合按使用量付费的云基础设施。能够微调云资源和创建原生云架构，进而自然而然地实现 HPC 工作负载周转的加速。

为简洁起见，我们仅提供架构完善的框架中特定于 HPC 工作负载的详细信息。我们建议您在设计架构时能够考虑 AWS 架构完善的框架白皮书[2]中的最佳实践和问题。

本白皮书的目标读者是技术岗位的人员，如首席技术官 (CTO)、架构师、开发人员和运维团队成员。阅读本白皮书后，您将了解在云环境中设计和运行 HPC 时可以采用的 AWS 最佳实践和策略。

定义

AWS 架构完善的框架建立在五大支柱的基础上，它们是卓越运营、安全性、可靠性、性能效率和成本优化。在设计解决方案时，您会基于您的业务环境在各个要素之间做出权衡。这些业务决策可以确定设计优先事项。您可以在开发环境中以牺牲一部分可靠性来降低成本；而对于关键业务生产环境，您可以通过增加成本来提高可靠性。对于安全性和卓越运营，一般不会在它们和其他支柱之间进行权衡。

在本文中，我们介绍了松散耦合（有时指社区中的高吞吐量计算 (HTC)）与紧密耦合的工作负载之间的重要区别。我们还将介绍基于服务器和无服务器的设计。请参阅“场景”部分，了解有关这些区别的详细讨论。

AWS 云的某些词汇可能与常见的 HPC 术语不同。例如，HPC 用户可能将服务器称为“节点”，而 AWS 会将虚拟服务器称为“实例”。 通常情况下，当 HPC 用户谈到“作业”时，AWS 会将其称为“工作负载”。

AWS 文档将使用术语“vCPU”作为“线程”或“超线程”（或*半个*物理内核）同义词。在量化 AWS 上的 HPC 应用程序的性能或成本时，切勿遗漏该因数 2。

集群置放群组是一种 AWS 方法，可对具有最高网络需求的应用程序的计算实例进行分组。置放群组并不是一个物理硬件元素，而只是一个逻辑规则，可将所有节点保持在网络的低延迟半径内。

AWS 云基础设施围绕**区域**和**可用区**构建。区域是指全球范围内的某个物理位置，每个区域由多个可用区组成。可用区由一个或多个分散的数据中心组成，每个都拥有独立的配套设施，其中包括冗余电源、联网和连接。根据 HPC 工作负载的特征，您可能希望您的集群可覆盖多个可用区（提高可靠性）或停留在单个可用区内（强调低延迟）。

一般设计原则

在传统计算环境中，架构决策通常为静态的一次性事件，且有时在计算系统的生命周期内无需进行重大软件或硬件升级。随着项目及其环境的发展演进，这些初始决策可能无法适应不断变化的业务能力需求。

云中的情况则截然不同。云基础设施可以随着项目的增长而增长，从而可以实现不断优化的功能。在云中，自动化和按需测试能力将显著降低基础设施设计变更带来的影响与风险。这使系统能够随时间推移不断演进，以便项目能够不断地发展创新。

架构完善的框架提供了一系列一般性设计原则，以促进良好的高性能计算云端设计：

- **动态架构**：避免使用稳态模型的冻结、静态架构和成本估算。您的架构必须是动态的：随时间变化增加和缩小架构数量，以满足您的 HPC 需求。您的架构设计和成本分析必须与 HPC 活动的自然周期完全匹配。例如，随着工作从设计阶段移至实验室阶段，在经过一段时间紧张的模拟工作后，需求可能会有所减少。或者，在经过长期且稳定的数据累积阶段后，可能迎来大规模分析和数据缩减阶段。与许多传统的超级计算中心不同，AWS 云可助您避免排长队，冗长的配额应用程序，以及自定义和软件安装的限制。许多 HPC 工作本质上是突发性的，并且可与弹性和按使用量付费的云范例完美匹配。借助 AWS 的弹性和按使用量付费模型，即无需在超额订购系统（队列等待）或空闲系统（浪费金钱）之间进行痛苦抉择。而是可以在任何给定时间根据给定需求调整环境大小，例如计算集群。
- **使采购模型与工作负载相符**：AWS 提供了一系列的计算采购模型，可满足各类 HPC 使用模式。选择正确的模型可确保您仅为所需内容付费。例如，一家研究机构可能以不同的方式运行相同的天气预报应用程序：
    - 一个学术研究项目研究了具有大量参数扫描和系综的天气变量的作用。这些模拟并不紧迫，而且成本才是首要问题。它们非常适合 Amazon EC2 Spot 实例。借助 Spot 实例，您可以利用 Amazon EC2 的未使用容量，并且与按需价格相比，您可享受高达 90% 的折扣。
    - 在森林火灾频发季期间，通过最新的当地风力预报，可以确保消防人员的安全。模拟中的每分钟延迟都会降低其安全疏散的可能性。对于这些模拟，必须使用按需实例，以允许爆发分析并确保持续获得结果。
    - 每天早上，都会选出要在下午的电视广播中播出的天气预报。计划预留实例可用于确保每天能在正确的时间提供所需的容量。与按需实例相比，使用此定价模型可提供折扣。
- **以数据为始**：在开始设计架构之前，您必须清楚地了解数据。考虑数据来源、规模、速度和更新。性能和成本的整体优化侧重于计算上，并且包括数据考虑事项。AWS 提供了强大的数据和相关服务，包括数据可视化，从而使您能够从数据中获取最大价值。

- **自动简化架构实验**：通过代码实现自动化，您可以低成本创建和复制系统，避免人力支出。您可以跟踪代码变更，审核其产生的影响，并在必要时恢复到以前的版本。能够使用基础设施轻松进行实验，从而允许您针对性能和成本优化架构。AWS 提供了 AWS ParallelCluster 等工具，可帮助您开始处理 HPC 云基础设施即代码。

- **启用协作**：HPC 工作通常是在协作环境中开展，有时会遍及全球许多国家/地区。除即时协作外，通常还会与更广泛的 HPC 和科学社区共享方法和结果。务必提前考虑可以共享哪些工具、代码和数据，以及可与谁共享。交付方式应该属于此设计流程的一部分。例如，可以在 AWS 上以多种方式共享工作流程：您可以使用 Amazon 系统映像 (AMI)、Amazon Elastic Block Store (Amazon EBS) 快照、Amazon Simple Storage Service (Amazon S3) 存储桶、AWS CloudFormation 模板、AWS ParallelCluster 配置文件、AWS Marketplace 产品和脚本。充分利用 AWS 安全和协作功能，使 AWS 成为您和您的协作者解决 HPC 问题的绝佳环境。如此一来，即可在选定组中进行安全共享或与更广泛的社区开展公开共享，进而帮助您的计算解决方案和数据集产生更大的影响。

- **使用原生云设计**：当您将工作负载迁移到 AWS 时，通常没有必要复制本地环境，且可能效果欠佳。利用 AWS 服务的广度和深度，即能使用新的设计模式和原生云解决方案，以全新的方式运行 HPC 工作负载。例如，每个用户或组可以使用独立的集群，而该集群可以根据负载独立进行扩展。用户可以依靠托管服务（如 AWS Batch）或无服务器计算（如 AWS Lambda）来管理底层基础设施。考虑不使用传统的集群计划程序，而是仅在工作负载需要时才使用计划程序。在云中，HPC 集群无需一致续存，可以是临时性资源。在自动化集群部署时，您可以终止一个集群并使用相同或不同的参数快速启动一个新集群。此方法可根据需要创建环境。

- **测试实际的工作负载**：了解您的生产工作负载在云中的表现情况的唯一方法是在云上对其进行测试。大多数 HPC 应用程序都较为复杂，而它们的内存、CPU 和网络模式通常无法精简为一个简单的测试。此外，基础设施的应用程序要求会根据模型使用的应用程序求解器（数学方法或算法）、模型的大小和复杂性等而有所不同。因此，通用基准并非实际 HPC 生产性能的可靠预测指标。同样，测试具有较小基准集或“玩具问题”的应用程序几乎没有价值。 利用 AWS，您只需按实际使用量付费；因此，用您自己的代表性模型执行真实的概念验证是可行方案。基于云的平台的主要优势在于，可以在迁移之前执行真实全面的测试。

- **在获得结果的时间和成本降低之间取得平衡：**使用最有意义的参数来分析性能：时间和成本。对于对时效要求不高的工作负载，应侧重于成本优化。对于对时效要求不高的工作负载，Spot 实例通常是最便宜的方法。例如，如果研究人员在明年会议召开之前的某个时候必须分析大量实验室测量值，那么 Spot 实例可助力分析固定研究预算内可能的最大测量值。相反，对于对时效要求严格的工作负载（例如紧急响应建模），可用成本优化换取性能，而实例类型、采购模型和集群大小应选择为最短和最即时的执行时间。如若比较平台，则必须考虑整个解决方案的时间，包括非计算方面，例如预配置资源、暂存数据，或是在较传统的环境中作业队列所耗费的时间。

场景

HPC 案例通常是复杂的计算问题，需要并行处理技术。为支持计算，架构完善的 HPC 基础设施能够在计算期间保持持续性能。HPC 工作负载可跨越传统的应用程序（如基因组、计算化学、财务风险建模、计算机辅助工程、天气预测和地震成像）以及新兴应用程序（如机器学习、深度学习和自动驾驶汽车）。尽管如此，支持这些计算的传统网格或 HPC 集群在架构上仍然非常相似，其中针对特定工作负载优化了所选的集群属性。在 AWS 中，可战略性地选择网络、存储类型、计算（实例）类型甚至部署方法，从而针对特定工作负载优化性能、成本和可用性。

根据同时运行的并行流程之间的交互程度，HPC 可分为两类：松散耦合的工作负载和紧密耦合的工作负载。松散耦合的 HPC 案例是指，在整个模拟过程中，多个或并行流程彼此之间没有强烈交互作用的案例。紧密耦合的 HPC 案例是指，在模拟的每个迭代或步骤中，同时运行及彼此定期交换信息的并行流程。

对于松散耦合的工作负载，完成整个计算或模拟通常需要数百至数百万个并行流程。在整个模拟过程中，可以任意顺序和速度运行这些流程。如此一来，即可为松散耦合模拟所需的计算基础设施提供灵活性。

紧密耦合的工作负载在模拟的每个迭代中都均设有定期交换信息的流程。通常情况下，这些紧密耦合模拟会在同构集群上运行。如果基础设施允许，核心或处理器的总数可能达到数十、数万乃至数十万。在模拟过程中，流程的交互对基础设施（例如计算节点和网络基础设施）提出了额外的要求。

用于运行各种松散和紧密耦合的应用程序的基础设施不尽相同，主要区别在于其在节点之间进行流程交互的能力。有一些基本方面既适用于场景，也适用于每一场景的特定设计考虑事项。在 AWS 上选择 HPC 基础设施时，请考虑以下适用于两种场景的基础知识：

- **网络**：网络要求的范围很广，从低要求的情况（例如通信流量极少的松散耦合应用程序）到需要具备大带宽和低延迟的高性能网络的紧密耦合和大规模并行应用程序。
- **存储**：HPC 计算可以独特的方式使用、创建和移动数据。存储基础设施必须在计算的每个步骤中支持这些要求。启动时通常会存储输入数据，运行时会创建和存储更多数据，而运行完成后则会将输出数据移动到存储位置。要考虑的因素包括数据规模、媒体类型、传输速度、共享访问和存储属性（例如，持久性和可用性）。在节点之间使用共享文件系统非常有帮助。例如，使用网络文件系统 (NFS) 共享（如 Amazon Elastic File System (EFS)）或 Lustre 文件系统（如 Amazon FSx for Lustre）。
- **计算**：Amazon EC2 实例类型定义了可用于您的 HPC 工作负载的硬件功能。硬件功能包括处理器类型、核心频率、处理器功能（例如矢量扩展）、内存核心比以及网络性能。在 AWS 上，实例被认为与 HPC 节点相同。在本白皮书中，这些术语可以交换使用。
    - AWS 为托管服务提供了访问计算的能力，且无需选择底层的 EC2 实例类型。AWS Lambda 和 AWS Fargate 为计算服务，可让您运行工作负载而无需预置和管理底层服务器。
- **部署**：AWS 提供了许多可用于部署 HPC 工作负载的选项。可以从 AWS 管理控制台手动启动实例。对于自动化部署，可以使用各种软件开发工具包 (SDK)，进而用不同的编程语言对端到端解决方案进行编码。常见的 HPC 部署选项结合了 bash shell 脚本与 AWS 命令行界面 (AWS CLI)。
    - AWS CloudFormation 模板允许将为应用程序定制的 HPC 集群规范描述为代码，以便可以在几分钟内启动它们。AWS ParallelCluster 是开源软件，可通过 CloudFormation 与已安装的软件（例如，编译器和计划程序）协调集群的启动，从而获得传统的集群体验。
    - AWS 为基于容器的工作负载提供托管部署服务，例如 Amazon EC2 Container Service (Amazon ECS)、Amazon Elastic Kubernetes Service (Amazon EKS)、AWS Fargate 和 AWS Batch。
    - 其他软件选项可从 AWS Marketplace 和 AWS 合作伙伴网络 (APN) 中的第三方公司获得。

通过云计算，可轻松使用基础设施组件和架构设计进行实验。AWS 强烈建议测试实例类型、EBS 卷类型、部署方法等，从而以最低的成本实现最佳性能。

松散耦合场景

松散耦合的工作负载需要处理大量较小的作业。通常，较小的作业会在一个节点上运行，使用共享内存并行 (SMP) 可消耗一个流程或多个流程，进而在节点内进行并行化。

对并行流程或模拟中的迭代进行后处理，即可从模拟中创建一个解决方案或发现。许多领域中都存在松散耦合应用程序，包括蒙特卡洛模拟、图像处理、基因组学分析和电子设计自动化 (EDA)。

在松散耦合的工作负载中丢失一个节点或作业通常不会延误整个计算。丢失的工作可以稍后重拾，也可以完全省略。计算中所涉节点的规范和功率可能有所不同。

适用于松散耦合工作负载的架构具有以下考虑事项：

- **网络**：由于并行流程通常不会彼此交互，因此工作负载的可行性或性能对实例之间的网络的带宽和延迟功能并不敏感。因此，对于此场景而言，集群置放群组不是必需的，因为它们会削弱弹性且无法提高性能。
- **存储**：松散耦合的工作负载因存储需求而异，并受数据集大小及传输、读取和写入数据的所需性能驱动。
- **计算**：每个应用程序都各不相同，但总的来说，应用程序的内存与计算比可驱动底层的 EC2 实例类型。某些应用程序已进行了优化，可利用 EC2 实例上的图形处理单元 (GPU) 或现场可编程门阵列 (FPGA) 加速器。
- **部署**：松散耦合模拟同上会在许多（有时是数百万）个计算核心上运行，这些计算核心可以分布在各个可用区中，并且不会牺牲性能。松散耦合模拟可以使用端到端服务和解决方案（例如 AWS Batch 和 AWS ParallelCluster）进行部署，也可以通过结合使用 AWS 服务（例如 Amazon Simple Queue Service (Amazon SQS)、Auto Scaling、AWS Lambda 和 AWS Step Functions）进行部署。

紧密耦合场景

紧密耦合的应用程序包含相互依赖的并行流程，可执行计算。与松散耦合计算不同，紧密耦合模拟的所有流程会一同迭代，并且需要彼此通信。迭代是指整体模拟中的一个步骤。紧密耦合计算依赖于一到数百万次迭代中的数十到数千个流程或核心。一个节点的故障通常会导致整个计算的失败。为了减少完全故障的风险，在计算过程中会定期进行应用程序级检查点操作，以允许从已知状态重新开始模拟。

这些模拟有赖于消息传递接口 (MPI) 进行进程间通信。通过 OpenMP 的共享内存并行可与 MPI 搭配使用。紧密耦合的 HPC 工作负载的示例包括：计算流体动力学、天气预报和油藏模拟。

适用于紧密耦合的 HPC 工作负载的架构具有以下考虑事项：

- **网络**：紧密耦合计算的网络要求极为严苛。节点之间的通信缓慢将会导致整个计算的速度变慢。为了获得稳定的联网性能，需要最大的实例大小、增强联网以及集群置放群组。这些技术可最大程度地减少模拟运行时并降低总体成本。紧密耦合的应用程序大小不一。分布在大量流程或核心上的大问题通常可以妥善予以并行处理。对于小型场景，总计算需求较低，对网络的需求最大。某些 Amazon EC2 实例会使用 Elastic Fabric Adapter (EFA) 作为网络接口，以便能在 AWS 上大规模运行需要高级别节点间通信的应用程序。EFA 定制的操作系统旁路硬件接口增强了实例间通信的性能，这对于扩展紧密耦合应用程序至关重要。
- **存储**：紧密耦合的工作负载因存储需求而异，并受数据集大小及传输、读取和写入数据的所需性能驱动。临时数据存储或暂存空间需要予以特殊考虑。
- **计算**：EC2 实例可提供各种配置，且这些配置具有可以各种配置提供，这些配置具有不同的核心内存比。对于并行应用程序，将内存密集型并行模拟分布在更多计算节点上有助于减少每核心内存需求，并以性能最佳的实例类型为目标。紧密耦合的应用程序需要从相似的计算节点构建的同构集群。以最大实例大小为目标可最大程度地减少节点间网络延迟，同时在节点之间进行通信时提供最大的网络性能。
- **部署**：提供了多种部署选项。端到端自动化是可以实现的，就像在“传统”集群环境中启动模拟一般。借助云可扩展性，您将能一次启动数百个大型多进程案例，因此无需排队。紧密耦合的模拟可以使用端到端解决方案（例如 AWS Batch 和 AWS ParallelCluster）或者通过基于 AWS 服务的解决方案（例如 CloudFormation 或 EC2 Fleet）进行部署。

参考架构

许多架构既适用于松散耦合的工作负载，也适用于紧密耦合的工作负载，并且可能需要根据情况进行些许修改。传统的本地集群对集群基础设施强制采用一体适用的方法。不过，云提供了众多可能性并允许优化性能和成本。在云中，您的配置范围可包括，从具有计划程序和登录节点的传统集群经验到原生云架构，后者可利用原生云解决方案获得成本效率优势。下列为五个参考架构：

1. 传统集群环境
2. 基于批处理的架构
3. 基于队列的架构

4. 混合部署
5. 无服务器工作流程

传统集群环境

许多用户的云之旅皆始于与传统 HPC 环境相似的环境。该环境通常涉及具有计划程序的登录节点，可启动作业。

传统集群预置的一种常见方法是基于计算集群的 AWS CloudFormation 模板，并结合针对用户特定任务的自定义。AWS ParallelCluster 是基于 AWS CloudFormation 的端到端集群预置功能的示例。尽管复杂的架构描述隐藏在模板内部，但是典型的配置选项允许用户选择实例类型、计划程序或引导操作，例如安装应用程序或同步数据。可以构建和执行该模板，以提供 HPC 环境，该环境不但具有传统 HPC 集群的“外观和风格”，还具有可扩展性的额外好处。登录节点可维护计划程序、共享文件系统和运行环境。同时，当作业提交到作业队列时，自动扩展机制允许启动其他实例。当实例变为空闲时，它们会自动终止。

集群可以部署为持久配置，也可以视为临时性资源。持久性集群可使用登录实例和计算队列进行部署，该规模可以是固定大小的，也可以关联到 Auto Scaling 群组，而该群组可根据提交的作业数来增加和减少计算队列。持久性集群始终都会运行某些基础设施。或者，可以将集群视为临时性的，且其中每个工作负载都会在其自己的集群上运行。临时性集群可通过自动化实现。例如，将 bash 脚本与 AWS CLI 结合使用，或者将 Python 脚本与 AWS SDK 结合使用以提供端到端的案例自动化。对于每种情况，将会预置和启动资源，在节点上置放数据，跨多个节点运行作业，以及自动检索案例输出或将其发送至 Amazon S3。作业完成后，即会终止基础设施。这些集群将基础设施视为代码，优化成本，并允许对基础设施更改进行完整的版本控制。

传统的集群架构可用于松散和紧密耦合的工作负载。为了获得最佳性能，紧密耦合的工作负载必须在具有同构实例类型的集群置放群组中使用计算队列。

参考架构

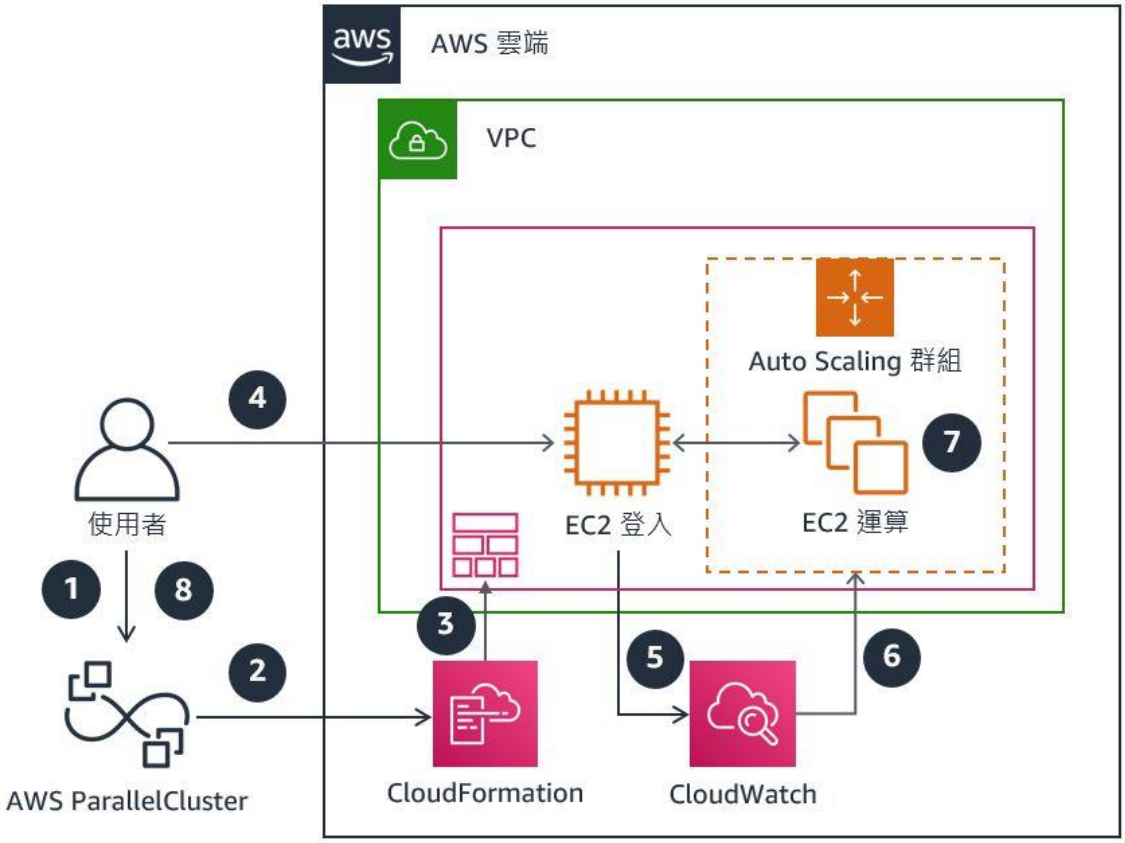

图 1：使用 AWS ParallelCluster 部署的传统集群

工作流程步骤：

1. 用户通过 AWS ParallelCluster CLI 和配置文件中的规范启动集群创建。
2. AWS CloudFormation 按照集群模板文件中的描述来构建集群架构，且用户可在其中提供一些自定义设置（例如，通过编辑配置文件或使用 Web 界面）。
3. AWS CloudFormation 通过使用自定义 HPC 软件/应用程序创建的 EBS 快照来部署基础设施，且集群实例可以通过 NFS 导出访问这些快照。
4. 用户登录到登录实例，并将作业提交至计划程序（例如 SGE、Slurm）。
5. 登录实例可根据作业队列大小向 CloudWatch 发出指标。
6. 如果作业队列大小超过阈值，则 CloudWatch 会触发 Auto Scaling 事件以增加计算实例的数量。
7. 计划的作业由计算队列进行处理。

8. [可选] 用户启动集群删除并终止所有资源。

基于批处理的架构

AWS Batch 是一项完全托管的服务，可让您在云中运行大规模计算工作负载，而无需预置资源或管理计划程序。[3] AWS Batch 让开发人员、科学家和工程师能够轻松高效地在 AWS 上运行成千上万项批处理计算任务。AWS Batch 可以根据提交的批处理作业的数量和特定的资源需求，动态预置最佳计算资源（如 CPU 或内存优化型实例）的数量和类型。其将通过全系列的 AWS 计算服务和功能（如 Amazon EC2[4] 和 Spot 实例）计划、安排和执行批处理计算工作负载。[5] 无需安装和管理运行作业所需的批处理计算软件或服务器集群，因此，您可以可专注于分析结果以及获取新的见解。

借助 AWS Batch，您可以将应用程序打包在一个容器中，指定作业的依赖关系，然后使用 AWS 管理控制台、CLI 或 SDK 提交您的批处理作业。您可以指定执行参数和作业依赖关系，并与各种常见的批处理计算工作流引擎和语言（例如 Pegasus WMS、Luigi 和 AWS Step Functions）集成。AWS Batch 提供了默认作业队列和计算环境定义，有助于您快速入门。

基于 AWS Batch 的架构可用于松散和紧密耦合的工作负载。紧密耦合的工作负载应在 AWS Batch 中使用多节点并行作业。

参考架构

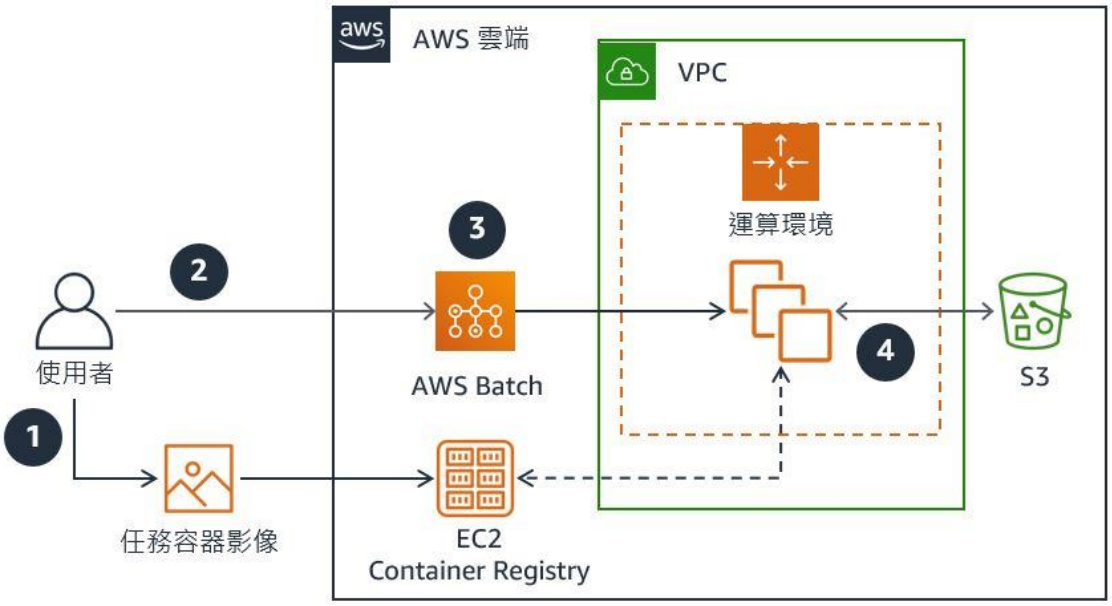

图 2：AWS Batch 架构示例

工作流程步骤：

1. 用户创建作业容器，将该容器上传到 Amazon EC2 Container Registry (Amazon ECR) 或另一个容器注册表（例如 DockerHub），然后为 AWS Batch 创建作业定义。
2. 用户将作业提交到 AWS Batch 中的作业队列。
3. AWS Batch 从容器注册表中提取图像并处理队列中的作业
4. 每个作业的输入和输出数据都存储在 S3 存储桶中。

基于队列的架构

Amazon SQS 是完全托管的消息队列服务，可轻松从计算步骤和后处理步骤解耦预处理步骤。[6] 从执行离散函数的单个组件构建应用程序，进而提高可扩展性和可靠性。将组件去耦是设计现代应用程序的最佳实践。Amazon SQS 通常是原生云松散耦合解决方案的核心。

Amazon SQS 通常会使用 AWS CLI 或 AWS SDK 脚本化解决方案进行编排，以从桌面部署应用程序，而无需用户直接与 AWS 组件进行交互。与服务托管的部署（例如 AWS Batch）相比，具有 SQS 和 EC2 的基于队列的架构需要自行管理的计算基础设施。

基于队列的架构最适合于松散耦合的工作负载，如果将其应用于紧密耦合的工作负载，则可能会迅速增加复杂性。

参考架构

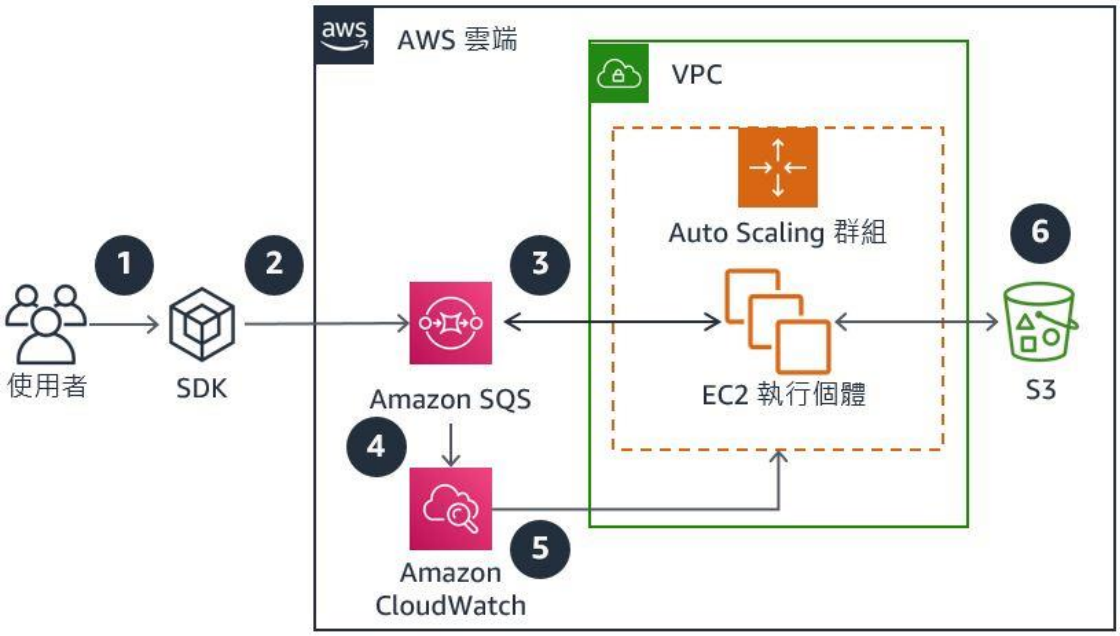

图 3：为松散耦合的工作负载部署的 Amazon SQS

工作流程步骤：

1. 多个用户使用 AWS CLI 或 SDK 提交作业。
2. 作业在 Amazon SQS 中作为消息进行排队。
3. EC2 实例轮询队列并开始处理作业。
4. Amazon SQS 根据队列中的消息（作业）数量发出指标。
5. Amazon CloudWatch 警报配置为在队列长于指定长度时通知 Auto Scaling。Auto Scaling 可增加 EC2 实例的数量。
6. EC2 实例可提取源数据并将结果数据存储在 S3 存储桶中。

混合部署

通常情况下，投资其本地基础设施且还希望使用 AWS 的组织会考虑混合部署。这种方法使组织可以增加本地资源，并创建 AWS 的备用路径，而不是立即进行全面迁移。

混合场景的范围从最小的协调（如工作负载分离）到紧密集成的方法（如计划程序驱动的作业置放）。例如，组织可以分离其工作负载并在 AWS 基础设施上运行某一类型的所有工作负载。或者，在其本地流程和基础设施上进行大量投资的组织可能希望通过其作业计划软件和潜在的作业提交门户网站来管理 AWS 资源，从而为其最终用户带来更加无缝的体验。多个作业计划程序（商业和开源）提供了可根据需要动态预置和取消预置 AWS 资源的功能。底层资源管理有赖于本机 AWS 集成（例如 AWS CLI 或 API），并且允许高度自定义的环境，具体取决于计划程序。尽管作业计划程序有助于管理 AWS 资源，但计划程序只是成功部署的其中一方面。

成功运营混合场景的关键因素是数据局部性和数据移动。某些 HPC 工作负载不需要或不会生成重要的数据集；因此，数据管理已不成问题。但是，作业或会成为一个瓶颈，因为其需要大量的输入数据或生成重要的输出数据。解决数据管理的技术因组织而异。例如，有些组织可能让他们的最终用户在其作业提交脚本中管理数据传输，有些组织可能仅在数据集所在的位置运行某些作业，有些组织可能选择在两个位置都复制数据，还有一些组织可能会选择组合使用多个选项。

根据数据管理方法，AWS 提供了多种有助于混合部署的服务。例如，AWS Direct Connect 在本地环境和 AWS 之间建立专用的网络连接，并且 AWS DataSync 会自动将数据从本地存储移动到 Amazon S3 或 Amazon Elastic File System。其他软件选项可从 AWS Marketplace 和 AWS 合作伙伴网络 (APN) 中的第三方公司获得。

混合部署架构可用于松散和紧密耦合的工作负载。但是，为实现最佳性能，单个紧密耦合的工作负载应驻存在本地或 AWS 中。

参考架构

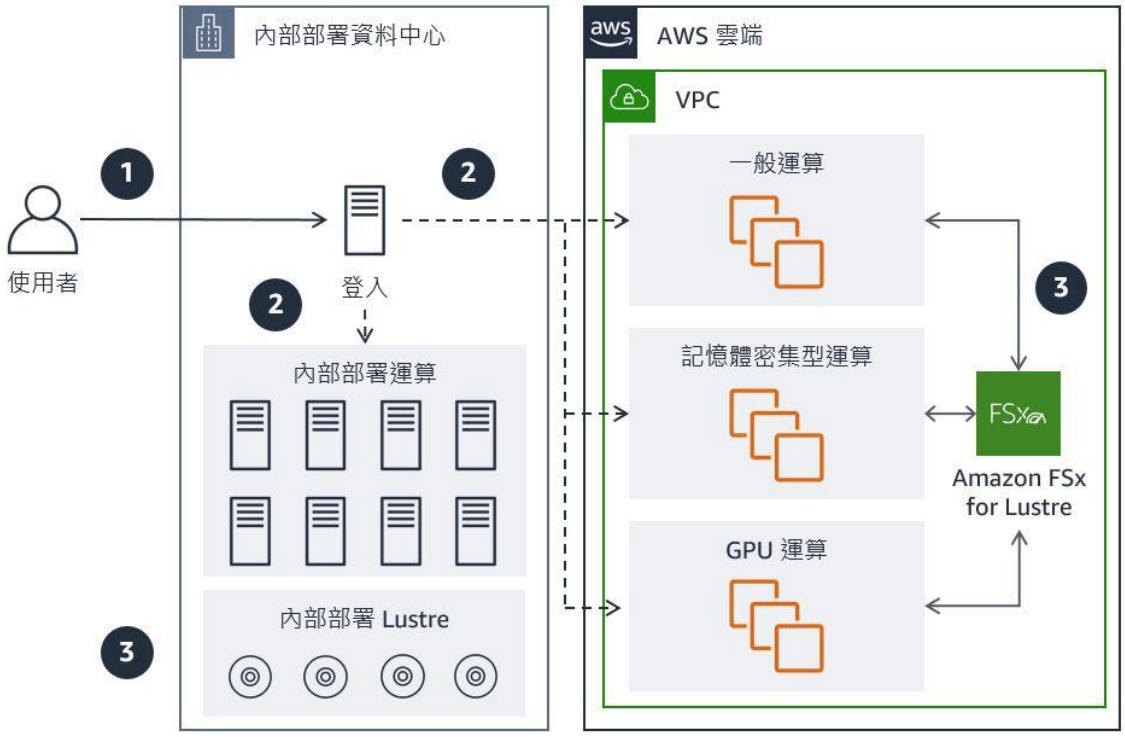

图 3：基于计划程序的混合部署示例

工作流程步骤：

1. 用户将作业提交到本地登录节点上的计划程序（例如 Slurm）。
2. 计划程序可在本地计算或 AWS 基础设施上执行作业，具体取决于相关配置。
3. 作业根据其运行位置访问共享存储。

无服务器

松散耦合的云之旅通常会造就一个完全没有服务器的环境，这意味着您可以专注于您的应用程序，而无需将服务器预置责任留给托管服务。AWS Lambda 可以运行代码，而无需预置或管理服务器。您只需为实际使用的计算时间付费 — 代码未运行时不产生费用。您需上传代码，随后 Lambda 就会处理运行和扩展代码所需的一切工作。Lambda 还能够自动触发来自其他 AWS 服务的事件。

可扩展性是无服务器 Lambda 方法的第二项优势。尽管每个工作并不算大（例如，具有一定内存的计算核心），但该架构可以催生数千个并发的 Lambda 工作，从而达到较大的计算吞吐量能力

并获得 HPC 标签。例如，可以通过调用同一算法来分析大量文件，可以并行分析大量基因组，或者可以对基因组内的大量基因位点进行建模。可达到的最大尺寸和扩展速度极为重要。为响应请求，基于服务器的架构需要几分钟时间来增加容量（即使在使用虚拟机（例如 EC2 实例）时）；但无服务器 Lambda 函数可以在几秒钟内完成扩展。AWS Lambda 使 HPC 基础设施能够立即响应针对计算密集型结果的任何无法预见的请求，并且可以满足数量不定的请求，而无需事先浪费任何资源进行预置。

除计算外，还有其他一些可协助 HPC 工作流程的无服务器架构。AWS Step Functions 使您可以通过将不同的 AWS 服务结合在一起来协调管道中的多个步骤。例如，可以使用以下工具针对不同的项目创建自动化基因组学管道：AWS Step Functions 适用于协调、Amazon S3 用于存储、AWS Lambda 适用于小任务、AWS Batch 适用于数据处理。

无服务器架构最适合于松散耦合的工作负载，或者与其他 HPC 架构结合使用时，可以作为工作流程协调。

参考架构

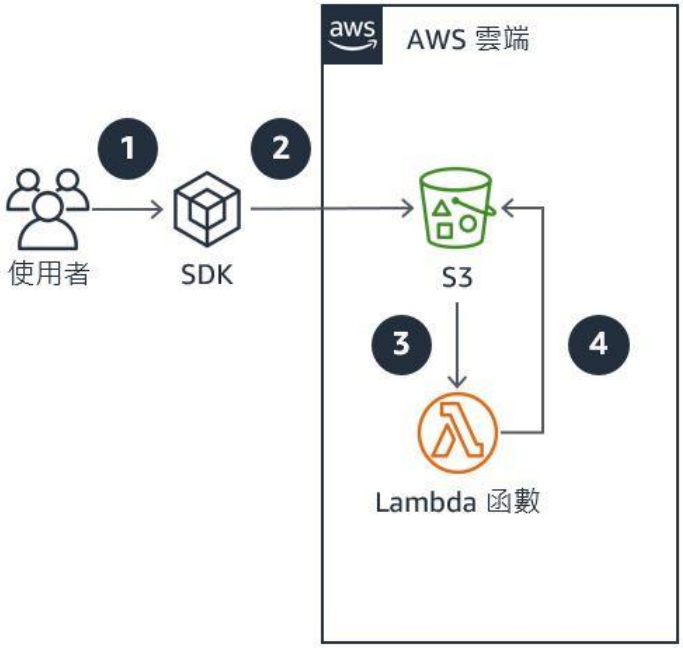

图 4：Lambda 部署的松散耦合工作负载示例

工作流程步骤：

1. 用户通过 AWS CLI 或 SDK 将文件上传到 S3 存储桶。
2. 输入文件使用传入前缀予以保存（例如 input/）。
3. S3 事件自动触发 Lambda 函数处理传入的数据。
4. 输出文件将使用传出前缀保存回 S3 存储桶（例如 output/）。

架构完善的框架的五大支柱

本部分将基于架构完善的框架的五大支柱介绍 HPC。每个支柱都谈及设计原则、定义、最佳实践、评估问题、考虑事项、关键 AWS 服务以及实用链接。

卓越运营支柱

卓越运营支柱包括运行和监控系统以创造业务价值并持续改善支持流程和程序的能力。

设计原则

在云中，有许多原则可推动卓越运营。对于 HPC 工作负载，尤其要注意以下几点。另请参阅 AWS 架构完善的框架白皮书中的设计原则。

- **自动执行集群操作：**在云中，您可以将整个工作负载定义为代码，并使用该代码进行更新。这样一来，您便能自动化重复的流程或程序。您将能始终如一地复制基础设施及实施操作程序，并从中受益。其中包括自动化作业提交流程和对事件的响应，例如作业开始、完成或故障。在 HPC 中，用户通常希望对每个作业重复执行多个步骤，例如上传案例文件、将作业提交到计划程序以及移动结果文件。使用脚本或事件驱动的代码自动执行这些重复步骤，以最大程度地提高可用性及降低成本和故障。
- **使用原生云架构（如适用）：**HPC 架构通常会采用以下两种形式中的一个。第一个是具有登录实例、计算节点和作业计划程序的传统集群配置。第二个是具有自动部署和托管服务的原生云架构。单个工作负载可以针对每个（临时）集群运行，也可以使用无服务器功能。原生云架构可以通过普及先进技术来优化运营；不过，最佳技术方法须与 HPC 用户期望的环境相符。

定义

在云中实现卓越运营有三个领域的最佳实践：

- 准备
- 运营
- 演进

有关**准备**、**操作**和**演进**领域的更多信息，请参阅 AWS 架构完善的框架白皮书。本白皮书中并未介绍演进。

最佳实践

准备

请参阅 AWS 架构完善的框架白皮书的相应部分。

在准备部署工作负载时，请考虑使用专门的软件包（商业或开源）来查看系统信息，并利用这些信息定义工作负载的架构模式。使用自动化工具（例如 AWS ParallelCluster 或 AWS CloudFormation），以可配置变量的方式定义这些架构。

云提供了多种计划选项。一种选择是使用 AWS Batch，其是一项完全托管的批处理服务并且可支持单节点和多节点任务。另一种选择是不使用计划程序。例如，您可以创建一个临时性集群来直接运行单个作业。

> HPCOPS 1：如何标准化跨集群的架构？
>
> HPCOPS 2：如何计划作业，是使用传统计划程序、AWS Batch，还是不使用计划程序而改用临时性集群？

运营

必须对操作进行标准化和常规管理。专注于自动化、频繁进行小规模变更、定期执行质量保证测试以及定义跟踪、审核、回滚和查看变更的机制。变更不应太大且不频繁，不应要求计划的停机时间，也不应要求手动执行。必须收集和审核基于工作负载关键操作指标的各种日志和指标，以确保连续操作。

AWS 提供了使用其他工具来处理 HPC 操作的机会。这些工具包括监控协助，自动化部署等，不一而足。例如，您可以使用 Auto Scaling 重新启动故障实例，使用 CloudWatch 监控集群的负载指标、配置作业完成时的通知，或是或者使用托管服务（例如 AWS Batch）为故障作业实施重试规则。原生云工具可以极大地改善您的应用程序部署和变更管理。

发布管理流程，无论是手动还是自动的，都必须基于小幅度的增量变更和跟踪的版本。您必须能够恢复引入问题的发行版而不会造成操作影响。使用持续集成和持续部署工具（例如 AWS

CodePipeline 和 AWS CodeDeploy）来自动化变更部署。使用版本控制工具（例如 AWS CodeCommit）跟踪源代码变更，并使用自动化工具（例如 AWS CloudFormation 模板）跟踪基础设施配置。

> HPCOPS 3：如何在最大程度地减少变更影响的同时改进工作负载？
>
> HPCOPS 4：如何监控工作负载以确保其按预期运行？

使用云进行 HPC 引入了新的操作注意事项。虽然本地集群的大小是固定的，但云集群可以扩展以满足需求。适用于 HPC 的原生云架构的操作方式也与本地架构截然不同。例如，当作业完成时，他们会使用不同的机制提交作业和预置按需实例资源。您必须采用适应云弹性和原生云架构动态特性的操作程序。

演进

演进实践领域中没有独特的 HPC 最佳实践。有关更多信息，请参阅 AWS 架构完善的框架白皮书的相应部分。

安全性支柱

安全性支柱包括通过风险评估和缓解策略在提供业务价值的同时保护信息、系统和资产的能力。

设计原则

在云中，有许多原则可帮助您增强系统安全性。建议使用 AWS 架构完善的框架白皮书中的设计原则，且这些原则不会因不同的 HPC 工作负载而异。

定义

在云中实现安全性有五个最佳实践领域：

- Identity and access management (IAM)
- 检测性控制
- 基础设施保护
- 数据保护
- 事件响应

在为任何系统设计架构之前，您必须制定安全实践。您必须能够控制权限、识别安全事件、保护您的系统和服务，并通过数据保护机制来保持数据的机密性和完整性。您应该具备一个定义明确且经过实践的流程来响应安全事件。这些工具和方法非常重要，因为它们有助于实现诸如避免数据损失和遵循法律与合规性要求等一系列目标。

借助 AWS 责任共担模式，组织能够利用云服务实现其安全性和合规性目标。AWS 负责保护用于支持云服务的基础设施，因此您能够专注于使用云上的各种服务来实现您的目标。AWS 云提供对安全数据的访问机制以及响应安全性事件的自动化方法。

AWS 架构完善的框架白皮书中详细记录了所有至关重要的安全最佳实践领域。AWS 架构完善的框架白皮书中介绍了**检测控制**、**基础设施保护**和**事件响应**领域。本白皮书中并未对这些领域进行介绍，且无需针对 HPC 工作负载修改这些领域。

最佳实践

Identity and Access Management (IAM)

Identity and Access Management 是信息安全计划的重要组成部分。他们确保只有经过授权和身份验证的用户才能访问您的资源。例如，您需要定义一些身份主体（在您的账户中进行操作的用户、组、服务和角色），根据这些主体创建策略，并实施严格的凭证管理。这些权限管理元素构成了身份验证和授权的核心概念。

自主及临时性运行 HPC 工作负载可限制敏感数据的泄露。通过自主部署，人员对实例的访问可降至最低，进而可最大程度地减少资源暴露。HPC 数据是在有限的时间内生成的，因此可最大程度地减少未经授权访问数据的可能性。

> HPCSEC 1：如何使用托管服务、自治方法和临时性集群，以最大程度地减少人员对工作负载基础设施的访问？

HPC 架构可以使用各种托管（例如 AWS Batch、AWS Lambda）和非托管计算服务（例如 Amazon EC2）。当架构要求直接访问计算环境时，例如连接到 EC2 实例，用户通常要通过安全外壳 (SSH) 连接并使用 SSH 密钥进行身份验证。在传统集群方案中，此访问模型较为典型。所有凭证（包括 SSH 密钥）都必须得到适当保护并定期轮换。

另外，AWS Systems Manager 包含完全托管的服务 (Session Manager)，可提供基于浏览器的交互式 shell 和 CLI 体验。它提供了安全且可审核的实例管理，且无需打开入站端口，维护堡垒主机以及管理 SSH 密钥。可以通过任何支持 ProxyCommand 的 SSH 客户端访问 Session Manager。

> HPCSEC 2：可使用哪些方法保护和管理您的凭证？

检测性控制

检测性控制最佳实践领域中没有独特的 HPC 最佳实践。请参阅 AWS 架构完善的框架白皮书的相应部分。

基础设施保护

基础设施最佳实践领域中没有独特的 HPC 最佳实践。请参阅 AWS 架构完善的框架白皮书的相应部分。

数据保护

在为任何系统设计架构之前，您必须制定基本安全实践。例如，数据分级提供了一种基于敏感程度对组织数据进行分类的方法，加密通过让未经授权的用户无法获知数据的真正内容来保护数据。这些工具和方法非常重要，因为它们有助于实现诸如避免数据损失或遵循法律与合规性要求等一系列目标。

> HPCSEC 3：您的架构可如何在结果的整个生命周期内满足存储可用性和持久性的数据要求？

除了敏感度和监管义务级别外，还可以根据下次使用数据的时间和方式对 HPC 数据进行分类。最终结果通常会保留下来，而中间结果（会在必要时重新创建）则可能不需要保留。仔细评估和分类数据允许将重要数据的编程数据迁移到更具弹性的存储解决方案，例如 Amazon S3 和 Amazon EFS。

了解数据寿命并使用编程处理数据，即可将架构完善的基础设施的数据泄露降至最低并提供最大限度的保护。

事件响应

事件响应最佳实践领域中没有独特的 HPC 最佳实践。请参阅 AWS 架构完善的框架白皮书的相应部分。

可靠性支柱

可靠性支柱包含系统从基础设施中断或服务中断恢复、动态获取计算资源以满足需求以及减少中断（如错误配置或暂时性网络问题）的能力。

设计原则

在云中，有许多原则可帮助您提高可靠性。对于 HPC 工作负载，尤其要注意以下几点。有关更多信息，请参阅 AWS 架构完善的框架白皮书中的设计原则。

- **横向扩展以提高聚合系统的可用性**：务必要考虑可能会降低单个故障对整体系统的影响的横向扩展选项。例如，与其让一个大型的共享 HPC 集群运行多个作业，不如考虑在整个 Amazon 基础设施中创建多个集群，以进一步隔离潜在故障的风险。由于可以将基础设施视为代码，因此您可以横向扩展单个集群中的资源，还可以横向扩展运行个别案例的集群的数量。
- **无需预估容量**：可以设置一组 HPC 集群来满足当前需求，并可根据需求的增加或减少手动或自动扩展集群。例如，在不使用时终止空闲的计算节点，然后运行。例如，在不使用时终止空闲的计算节点，然后运行并发集群以处理多个计算，而不是在队列中等待。
- **管理自动化中的变更**：通过自动化对基础设施的更改，您可以根据版本控制置放集群基础设置以及完整复制之前创建的集群。必须要管理自动化变更。

定义

在云中有三个领域的可靠性最佳实践：

- 基础
- 变更管理
- 故障管理

AWS 架构完善的框架白皮书中介绍了变更管理领域。

最佳实践

基础

> HPCREL 1：如何管理您账户的 AWS 服务限制？

AWS 会设置服务限制（您的团队可以请求的每种资源的数量上限），以防止您意外超额预置资源。

HPC 应用程序通常同时需要大量的计算实例。HPC 工作负载非常需要横向扩展的能力和优势。但是，在将大型工作负载一次部署到一个大型集群或多个较小的集群之前，横向扩展可能需要增加 AWS 服务限制。

通常必须将服务限制增加至大于默认值，以便处理大型部署的要求。联系 AWS Support，申请提高限制。

变更管理

变更管理最佳实践领域中没有独特的 HPC 最佳实践。请参阅 AWS 架构完善的框架白皮书的相应部分。

故障管理

任何复杂的系统都可以预期偶尔会发生故障，因此了解这些故障，对其进行响应并防止其再次发生至关重要。故障场景可以包括集群无法启动或特定工作负载的故障。

> HPCREL 2：您的应用程序如何利用检查点来从故障中恢复？

可以通过多种方式提高故障容错能力。对于长时间运行的情况，可在代码中加入常规检查点，以便您在发生故障时能从部分状态继续运行。检查点是许多 HPC 应用程序中内置的应用程序级故障管理的常见功能。最常见的方法是使应用程序定期写出中间结果。中间结果可提供对应用程序错误的潜在洞察，并可以根据需要重新启动案例，同时仅会丢失部分工作。

当您使用成本效益高但可能会中断的实例时，检查点对 Spot 实例很有用。此外，某些应用程序可能会受益于更改默认的 Spot 中断行为（例如，停止或休眠实例而不是终止实例）。当依赖检查点进行故障管理时，务必要考虑存储选件的耐用性。

HPCREL 3：您如何针对架构制定故障容错计划？

部署到多个可用区时，可以提高故障容错能力。根据紧密耦合的 HPC 应用程序的低延迟要求，每个单独案例都应位于单个集群置放群组和可用区中。或者，松散耦合的应用程序不具有如此低的延迟要求，并且可以通过部署到多个可用区来改善故障管理。

做出此设计决策时，请考虑可靠性和成本支柱之间的权衡。重复计算和存储基础设施（例如，头节点和附加存储）会产生额外的成本，并且可能存在将数据移动到可用区或另一个 AWS 区域的数据传输费用。对于非紧急使用案例，作为灾难恢复 (DR) 事件的一部分，最好仅移至另一个可用区。

性能效率支柱

性能效率支柱专注于有效利用计算资源来满足需求的能力，以及如何在需求发生变化和技术不断演进的情况下保持这种效率。

设计原则

在云中设计 HPC 时，有许多原则可以帮助您实现性能效率：

- **设计应用程序的集群**：传统集群是静态的且需要针对集群设计应用程序。AWS 提供了为应用程序设计集群的功能。对于每个应用程序，单独的集群不再需要一体适用的模型。在 AWS 上运行各种应用程序时，可以使用不同的架构来满足每个应用程序的需求。这样一来，既可以降低成本，又可获得最佳性能。
- **借助有意义的使用案例测试性能**：测量特定架构上的 HPC 应用程序的性能的最佳方式是，对应用程序本身运行有意义的演示。一个或大或小的演示案例（没有预期的计算、内存、数据传输或网络流量）将无法对 AWS 上的应用程序性能进行有意义的测试。尽管系统特定的基准可助您了解底层计算基础设施的性能，但它们并不能反映应用程序的总体性能。AWS 按使用付费模型可加快概念验证速度且更具成本效益。
- **使用原生云架构（如适用）**：借助云中的托管、无服务器原生云架构，您无需运行和维护服务器即可执行传统计算活动。HPC 的原生云组件可用于计算、存储、作业编排以及数据和元数据的组织。借助各种 AWS 服务，工作负载流程中的每个步骤都可以进行解耦和优化，进而实现更高的性能。

- **频繁进行实验：**利用虚拟和可自动化资源，您可以使用不同类型的实例、存储或配置来快速进行比较测试。

定义

在云中实现性能效率包括四个方面的最佳实践：

- 选择
- 审核
- 监控
- 权衡

AWS 架构完善的框架白皮书中介绍了**审核**、**监控**和**权衡**领域。

最佳实践

选择

特定系统的最佳解决方案会根据您的工作负载类型而有所不同。架构完善的系统会使用多种解决方案，并且利用各种不同的功能来提高性能。HPC 架构可以依赖一个或多个不同的架构元素，例如排队、批处理、集群计算、容器、无服务器和事件驱动。

计算

> HPCPERF 1：如何选择计算解决方案？

针对特定 HPC 架构的最佳计算解决方案取决于工作负载部署方法、自动化程度、使用模式和配置。可以为流程的每个步骤选择不同的计算解决方案。选择错误的计算解决方案可能会降低性能效率。

实例是虚拟化的服务器，且具有不同的系列和大小，可提供各种功能。一些实例系列针对特定的工作负载，例如计算、内存或 GPU 密集型工作负载。其他实例则是通用的。

目标工作负载和通用实例系列对于 HPC 应用程序都非常有用。HPC 特别感兴趣的实例包括计算优化的系列和加速的实例类型，例如 GPU 和 FPGA。

某些实例系列拥有系列内的变体，因此可提供额外功能。例如，实例系列可能拥有本地存储、更大的联网功能或其他处理器的变体。这些变体可在实例类型矩阵[7]中进行查看，且或能提高某些 HPC 工作负载的性能。

在每个实例系列中，一个或多个实例大小允许垂直扩展资源。一些应用程序需要较大的实例类型（例如 24xlarge），而另一些应用程序则需要在较小的类型上运行（例如 large），具体取决于应用程序支持的数量或流程。当处理紧密耦合的工作负载时，使用最大的实例类型可以获得最佳性能。

T 系列实例系列是专为 CPU 利用率适中的应用程序设计的，这些应用程序可以从超出 CPU 性能的基准水平中受益。大多数 HPC 应用程序都属于计算密集型，并且 T 系列实例系列的性能或会有所下降。

应用程序的要求各不相同（例如，所需的核心、处理器速度、内存要求、存储需求和联网规范）。选择实例系列和类型时，请从应用程序的特定需求开始。对于需要特定应用程序组件的目标实例的应用程序，可以混合并匹配实例类型。

容器是一种操作系统虚拟化方法，对许多 HPC 工作负载具有吸引力，尤其是在应用程序已被容器化的情况下。AWS 服务（例如 AWS Batch、Amazon Elastic Container Service (ECS) 和 Amazon Elastic Container Service for Kubernetes (EKS)）有助于部署容器化应用程序。

函数可抽象出执行环境。AWS Lambda 让您可以在不部署、运行或维护实例的情况下执行代码。许多 AWS 服务会根据服务内部的活动发出事件，并且通常可以从服务事件中触发 Lambda 函数。例如，可以在将对象上传到 Amazon S3 之后执行 Lambda 函数。许多 HPC 用户会使用 Lambda 自动执行代码，这是其工作流程的一部分。

启动您选定的计算实例时，有几种选择：

- **操作系统：**当前的操作系统对于实现最佳性能以及确保可访问最新的库而言至关重要。
- **虚拟化类型：**新一代 EC2 实例可在 AWS Nitro 系统上运行。Nitro 系统会将所有主机硬件的计算和内存资源提供给您的实例，进而提高整体性能。专用 Nitro 卡可实现高速联网，高速 EBS 和 I/O 加速。实例不会保留管理软件的资源。

Nitro 管理程序是一种轻量级管理程序，可以管理内存和 CPU 分配，并提供与裸机无异的性能。借助 Nitro 系统，裸机实例还可以在无 Nitro 管理程序的情况下运行。启动裸机实例即可启动底层服务器，其中包括验证所有硬件和固件组件。这意味着与虚拟化实例相比，要启动您的工作负载，裸机实例需要花费更长的时间。在动态 HPC 环境中运行时，必须考虑额外的初始化时间，其中资源会根据需求启动和终止。

HPCPERF 2：如何优化您应用程序的计算环境？

底层硬件功能：除了选择 AMI 外，您还可通过利用底层 Intel 处理器的硬件功能进一步优化您的环境。优化底层硬件时，需要考虑四种方法：

1. 高级处理器功能
2. Intel 超线程技术
3. 处理器关联性
4. 处理器状态控制

HPC 应用程序可从这些高级处理器功能（例如高级矢量扩展）中获益，并且可通过编译适用于 Intel 架构的软件提高其计算速度。[8] 架构特定的指令的编译器选项会因编译器而异（请查看编译器的使用指南）。

默认情况下，AWS 启用了 Intel 超线程技术，其通常被称为“超线程”。超线程可允许每个超线程一个进程（每个核心两个进程），进而提高某些应用程序的性能。大多数 HPC 应用程序可受益于禁用超线程，因此其倾向于成为 HPC 应用程序的首选环境。在 Amazon EC2 中可以轻松禁用超线程。除非已在启用了超线程的情况下对应用程序进行了测试，否则建议禁用超线程，并在运行 HPC 应用程序时启动进程并将其分别固定到核心。借助 CPU 或处理器关联性，可轻松完成进程固定。

处理器关联性可以通过多种方式进行控制。例如，可以在操作系统级别（Windows 和 Linux 均可）进行配置，将其设置为线程库中的编译器标记，或是在执行期间指定为 MPI 标记。选择的控制处理器关联性的方法取决于您的工作负载和应用程序。

借助 AWS，您能在调整某些实例类型的处理器状态控制。[9] 您可以考虑更改 C 状态（空闲状态）和 P 状态（操作状态）设置，进而优化性能。默认的 C 状态和 P 状态设置可提供最佳性能，且对于大多数工作负载而言，此为最佳设置。但是，如果您的应用程序能够受益于以更高的单核或双核频率为代价以降低延迟，或者受益于与睿频加速频率峰值相比，在更低的频率下获得稳定的性能，则请在所选的实例上使用 C 状态和 P 状态设置进行实验。

有许多计算选项可用于优化计算环境。云部署允许在从操作系统到实例类型再到裸机部署的每个级别上进行实验。由于静态集群是在部署之前进行调整的，因此，要实现所需的性能，务必要花些时间对基于云的集群进行实验。

存储

HPCPERF 3：如何选择存储解决方案？

针对特定 HPC 架构的最佳存储解决方案在很大程度上取决于针对该架构的各个应用程序。工作负载部署方法、自动化程度以及所需的数据生命周期模式也属于相关因素。AWS 提供了各种存储选项。与计算一样，针对应用程序的特定存储需求时，可以获得最佳性能。AWS 不需要您为“一体适用的”方法超额配置您的存储，并且也并非总是需要大型、高速的共享文件系统。优化计算选择对于优化 HPC 性能非常重要，并且许多 HPC 应用程序将无法从最快的存储解决方案中受益。

HPC 部署通常需要由集群计算节点访问的共享或高性能文件系统。您可以使用多种架构模式从 AWS Managed Services、AWS Marketplace 产品、APN 合作伙伴解决方案以及部署在 EC2 实例上的开源配置实施这些存储解决方案。特别是，Amazon FSx for Lustre 是一项托管服务，可为需要高性能并行文件系统的 HPC 架构提供极具成本效益的高性能解决方案。此外，您也可以从具有 Amazon EBS 卷或实例存储卷的 Amazon Elastic File System (EFS) 或 EC2 实例创建共享文件系统。通常情况下，简单的 NFS 挂载可用于创建共享目录。

选择存储解决方案时，您可以为本地存储和共享存储中的一个或两个存储选择一个 EBS 支持的实例。EBS 卷通常是 HPC 存储解决方案的基础。可以使用各种类型的 EBS 卷，包括磁性硬盘驱动器 (HDD)、通用固态硬盘 (SSD) 和适用于高 IOPS 解决方案的预置 IOPS SSD。它们在吞吐量、IOPS 性能和成本方面均存在差异。

您可以通过选择 Amazon EBS 优化实例来进一步提高性能。EBS 优化实例使用经过优化的配置堆栈，可以针对 Amazon EBS I/O 提供额外的专用容量。这种优化通过最小化您的 Amazon EBS I/O 与实例的其他网络流量之间的争用，来为 EBS 卷提供最佳性能。为获得稳定一致的性能，以及依赖低延迟网络或对 EBS 卷有大量 I/O 数据需求的 HPC 应用程序，请选择 EBS 优化实例。

要启动 EBS 优化实例，请选择默认情况下启用 EBS 优化的实例类型，或选择允许在启动时启用 EBS 优化的实例类型。

实例存储卷，包括非易失性存储标准 (NVMe) SSD 卷（仅在某些实例系列中可用），可用于临时块级存储。请参阅实例类型矩阵以了解 EBS 优化和实例存储卷支持。[10]

选择存储解决方案时，确保它与您的访问模式保持一致，从而实现预期性能。可以轻松使用不同的存储类型和配置进行实验。对于 HPC 工作负载，最贵的选择并不一定总是性能最佳的解决方案。

联网

HPCPERF 4：如何选择网络解决方案？

适合某个 HPC 工作负载的最佳网络解决方案会根据延迟、带宽、吞吐量要求而有所不同。对于计算节点之间的网络连接，紧密耦合的 HPC 应用程序通常可能需要最低的延迟。对于大小适中、紧密耦合的工作负载，可以选择具有大量核心的大型实例类型，从而使应用程序完全适合该实例，而完全无需跨网络。

或者，某些应用程序会受网络限制并且需要较高的网络性能。可以为这些应用程序选择具有更高网络性能的实例。使用系列中的最大实例类型可以获得最高的网络性能。有关更多信息，请参阅实例类型矩阵。[7]

对于大型紧密耦合的应用程序，需要实例之间具有低延迟的多个实例。在 AWS 上，其可通过将计算节点启动到集群置放群组中予以实现，其中该集群置放群组是可用区中实例的逻辑分组。集群置放群组可提供非阻塞和非超额连接性，包括实例之间的完全等分带宽。将集群置放群组用于跨越多个实例的对延迟敏感的紧密耦合应用程序。

除了集群置放群组外，紧密耦合的应用程序还可以从 Elastic Fabric Adapter (EFA) 中受益，其中 EFA 是可以连接到 Amazon EC2 实例的网络设备。与传统的基于云的 HPC 系统中使用的 TCP 传输相比，EFA 可提供更低且更稳定一致的延迟和更高的吞吐量。其可通过 *Libfabric* API 启用操作系统旁路访问模型，该模型允许 HPC 应用程序直接与网络接口硬件进行通信。EFA 增强了实例间通信的性能，经过优化可在现有的 AWS 网络基础设施上运行，并且对于扩展紧密耦合的应用程序至关重要。[13]

如果应用程序无法利用 EFA 的操作系统旁路功能，或者实例类型不支持 EFA，则可以通过选择支持增强联网的实例类型来获得最佳的网络性能。增强联网可通过使用直通而不是硬件模拟的设备，为 EC2 实例提供更高的网络性能和更低的 CPU 利用率。与传统设备虚拟化相比，此方法允许 EC2 实例实现更高的带宽、更高的每秒数据包数处理和更低的实例间延迟。

增强联网可用于所有最新一代的实例类型，并且需要具有受支持驱动程序的 AMI。尽管大多数最新的 AMI 包含受支持的驱动程序，但是自定义 AMI 可能需要更新的驱动程序。有关启用增强联网和实例支持的更多信息，请参阅增强联网文档。[11]

松散耦合的工作负载通常对极低的延迟网络不敏感，不需要使用集群置放群组，也不需要将实例保持在相同的可用区或区域中。

审核

审核最佳实践领域中没有独特的 HPC 最佳实践。请参阅 AWS 架构完善的框架白皮书的相应部分。

监控

监控最佳实践领域中没有独特的 HPC 最佳实践。请参阅 AWS 架构完善的框架白皮书的相应部分。

权衡

监控最佳实践领域中没有独特的 HPC 最佳实践。请参阅 AWS 架构完善的框架白皮书的相应部分。

成本优化支柱

成本优化支柱包括 HPC 系统在整个生命周期中不断完善和改进的过程。从最开始的概念验证的初始设计到生产工作负载的持续运营，您都可以采用本文档中的实践来构建和运营具有成本意识的系统，从而在实现业务成果的同时最小化成本，使您的企业能够最大限度地提高投资回报率。

设计原则

对于云中的 HPC，您可以遵循以下几个原则来实现成本优化：

- **采用的消费模式**：只需按您使用的计算资源付费。HPC 工作负载时高时低，因此可根据需要增加和减少资源容量，进而降低成本。例如，可以预置和保留低级别运行速度的 HPC 容量，以便从更高的折扣中受益，而爆发需求可以按 Spot 或按需定价进行预置，并且仅在需要时才联机。
- **优化特定作业的基础设施成本**：许多 HPC 工作负载都是数据处理管道的一部分，包括数据数据传输、预处理、数值计算、后处理、数据传输和存储步骤。计算平台可在云中而非大型且昂贵的服务器上对每个步骤进行优化。例如，如果管道中的单个步骤需要大量内存，则您只需要为内存密集型应用程序购买更昂贵的大型内存服务器即可，而所有其他步骤仍可以在较小和较便宜的计算平台上妥当运行。通过针对工作负载的每一步骤优化基础设施，可以降低成本。
- **以最有效的方式突增工作负载**：通过在云中横向扩展，节省 HPC 工作负载。在横向扩展时，可以同时运行整个工作负载中的许多作业或迭代，从而减少总的已用时间。根据应用程序的不同，横向扩展可以保持成本中立，同时可通过在短时间内交付结果间接节省成本。

- **使用 spot 定价：**与按需实例相比，Amazon EC2 Spot 实例可以折扣价在 AWS 上提供备用计算容量。但是，当 EC2 需要回收容量时，可以中断 Spot 实例。对于灵活或容错的工作负载，Spot 实例通常是最具成本效益的资源。HPC 工作负载具有间歇性，因此非常适合 Spot 实例。通过与 Spot Advisor 协作，可以将 Spot 实例中断的风险降到最低，并且可以通过更改默认中断行为并使用 Spot 队列管理您的 Spot 实例，进而减轻中断影响。Spot 实例节省的成本可以轻松抵消偶尔重新启动工作负载的需求。
- **评估成本与时间的权衡：**紧密耦合的大规模并行工作负载能够在各种核心数量上运行。对于这些应用程序，通常情况下，核心数量越多，机箱的运行效率就会越低。如果将运行许多类型和大小相似的案例，则可以创建成本与周转时间的曲线。曲线对于案例类型和应用程序而言是特定的，因为扩展比例很大程度上取决于计算与网络需求的比率。相较于较小的工作负载，较大的工作负载将能进一步扩展。了解成本与周转时间之间的权衡后，对时间有严格要求的工作负载可以在更多的核心上快速运行，而在更少的核心上时可以最高效率运行，进而节省成本。当您想取得时间敏感性与成本敏感性间的平衡时，工作负载可能会介于两者之间。

定义

云中的成本优化包括四个领域的最佳实践：

- 资源成本效益
- 供需匹配
- 支出认知
- 持续优化

AWS 架构完善的框架白皮书中介绍了**供需匹配**、**支出认知**和**持续优化。**

最佳实践

资源成本效益

HPCCOST 1：如何评估工作负载的可用计算和存储选项，进而优化成本？

HPCCOST 2：如何评估作业完成时间和成本之间的权衡？

为系统使用合适的实例、资源和功能是成本管理的关键。实例选择可能会增加或减少运行 HPC 工作负载的总体成本。例如，紧密耦合的 HPC 工作负载可能需要五个小时才能在数个较小服务器的集群上运行；而在数量较少的大型服务器的集群上运行时，每小时可能要花费两倍的成本，但一小时即可计算出结果，因此总体而言节省了成本。存储的选择也会影响成本。考虑作业周转与成本优化之间的潜在权衡，并使用不同的实例大小和存储选项测试工作负载，进而优化成本。

AWS 提供各种灵活且具有成本效益的定价选项，您可以从 EC2 和其他服务获取最符合您需求的实例。按需实例允许按小时支付计算容量的费用，且无需承诺最低用量。预留实例允许预留容量，且与按需定价相比可节约一些成本。使用 Spot 实例，您可以利用未使用的 Amazon EC2 容量，并且与按需定价相比可额外节约成本。

架构完善的系统使用最具有成本效益的资源。您还可以通过将托管服务用于预处理和后处理来降低成本。例如，不必为存储和后处理已完成的运行数据而维护服务器，但是可以将数据存储在 Amazon S3 上，然后使用 Amazon EMR 或 AWS Batch 进行后处理。

许多 AWS 服务都提供了可进一步降低成本的功能。例如，将 Auto Scaling 与 EC2 集成时，可以根据工作负载需求自动启动和终止实例。将 FSx for Lustre 与 S3 本机集成，并将 S3 存储桶的全部内容显示为 Lustre 文件系统。这样一来，您可以通过为您的即时工作负载预置最小的 Lustre 文件系统来优化存储成本，同时在具有经济效益的 S3 存储中维护长期数据。S3 提供了不同的存储类，因此您可以使用最具成本效益的类别来存储数据；Glacier 或 Glacier Deep Storage Class 使您能够以最低成本归档数据。

对不同的实例类型、存储要求和架构进行实验，既可最大程度降低成本，又可维持良好性能。

供需匹配

供需匹配最佳实践领域中没有独特的 HPC 最佳实践。请参阅 AWS 架构完善的框架白皮书的相应部分。

支出认知

支出认知最佳实践领域中没有独特的 HPC 最佳实践。请参阅 AWS 架构完善的框架白皮书的相应部分。

持续优化

持续优化最佳实践领域中没有独特的 HPC 最佳实践。请参阅 AWS 架构完善的框架白皮书的相应部分。

总结

本篇剖析强调了在云中设计和运行可靠、安全、高效且经济实惠的高性能计算工作负载系统的架构。我们介绍了原型 HPC 架构和总体 HPC 设计原则。我们从 HPC 的角度重新审视了架构完善的五大支柱，为您提供了一系列问题，可助您查看现有或建议的 HPC 架构。将框架应用于您的架构可帮助您构建稳定高效的系统，使您可以专注于运行 HPC 应用程序并突破领域的界限。

贡献者

以下是对本文档做出贡献的个人和组织：

- Aaron Bucher，Amazon Web Services HPC 专家解决方案架构师
- Omar Shorbaji，Amazon Web Services 全球解决方案架构师
- Linda Hedges，Amazon Web Services HPC 应用程序工程师
- Nina Vogl，Amazon Web Services HPC 专家解决方案架构师
- Sean Smith，Amazon Web Services HPC 软件开发工程师
- Kevin Jorissen，Amazon Web Services 解决方案架构师 – 气候与天气
- Philip Fitzsimons，Amazon Web Services Well-Architected 高级经理

延伸阅读

有关更多信息，请参阅以下内容：

- AWS 架构完善的框架[12]
- https://aws.amazon.com/hpc
- https://d1.awsstatic.com/whitepapers/Intro_to_HPC_on_AWS.pdf
- https://d1.awsstatic.com/whitepapers/optimizing-electronic-design-automation-eda-workflows-on-aws.pdf
- https://aws.amazon.com/blogs/compute/real-world-aws-scalability/

文档修订

日期	描述
2019 年 12 月	次要更新
2018 年 11 月	次要更新
2017 年 11 月	原始版本

Notes

[1] https://aws.amazon.com/well-architected

[2] https://d0.awsstatic.com/whitepapers/architecture/AWS_Well-Architected_Framework.pdf

[3] https://aws.amazon.com/batch/

[4] https://aws.amazon.com/ec2/

[5] https://aws.amazon.com/ec2/spot/

[6] https://aws.amazon.com/message-queue

[7] https://aws.amazon.com/ec2/instance-types/#instance-type-matrix

[8] https://aws.amazon.com/intel/

[9] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/processor_state_control.html

[10] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSOptimized.html#ebs-optimization-support

[11] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/enhanced-networking.html

[12] https://aws.amazon.com/well-architected

[13] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/efa.html