Amazon EMR 迁移指南

如何将 Apache Spark 和 Apache Hadoop 从本地迁移到 AWS

2019 年8 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，并且 (c) 不构成 AWS 及其附属公司、供应商或许可方的任何承诺或保证。AWS 产品或服务均“按原样”提供，没有任何明示或暗示的担保、表态或条件。AWS 对其客户的责任和义务受 AWS 协议约束，本文档不属于 AWS 和客户之间签订的任何协议，亦不构成对任何此类协议的修订。

© 2019 Amazon Web Services, Inc. 或其附属公司。保留所有权利。

目录

简介 1

开启您的旅程 3

迁移方法 3

原型设计 5

选择团队 7

迁移的一般最佳实践 9

收集需求 11

获取本地指标 11

成本估算和优化 12

优化成本 12

存储优化 13

计算优化 16

成本估算概述 18

优化基于 Apache Hadoop YARN 的应用程序 18

Amazon EMR 集群拆分方案 21

集群特征 21

常见集群分拆方案 22

拆分集群的其他注意事项 24

保护 Amazon EMR 上的资源 25

EMR 安全最佳实践 25

身份验证 25

授权 29

加密 34

边界安全 35

网络安全 36
审计 37
软件补丁 38
软件升级 39
常见客户使用案例 40
数据迁移 44
使用 Amazon S3 作为中央数据存储库 44
持续传输大量数据 47
连续传输事件和流数据 50
优化基于 Amazon S3 的中央数据存储库 51
优化成本和性能 54
数据目录迁移 58
Hive metastore 元存储部署模式 58
Hive metastore 元存储迁移选项 63
EMR 上的多租户 66
筒仓模式 66
共享模式 68
在 Amazon EMR 上实施多租户的注意事项 69
Amazon EMR 上的提取、转换、加载 (ETL) 74
Amazon EMR 上的编排 74
迁移 Apache Spark 84
迁移 Apache Hive 87
提供 Ad Hoc 查询功能 95
Presto 注意事项 95
Amazon EMR 上的 HBase 工作负载 97
迁移 Apache Impala 102

卓越运营 103
升级 Amazon EMR 版本 103
卓越运营的通用最佳实践 107
测试和验证 108
数据质量概述 108
检查您的提取管道 109
整体数据质量策略 110
评估数据质量的影响 111
有助进行数据质量检查的工具 112
支持您的迁移 114
AWS 专业服务 114
AWS 合作伙伴 116
AWS Support 117
贡献者 118
其他资源 119
文档修订 120
附录 A：需求收集调查问卷 121
安全要求 123
TCO 注意事项 123
附录 B：EMR Kerberos 工作流程 124
具有单向信任机制的 KDC 的 EMR Kerberos 集群启动流程 124
通过 Hue 访问的 EMR Kerberos 流 126
与 HiveServer2 直接交互的 EMR Kerberos 流 127
EMR Kerberos 集群启动流程 128
附录 C：LDAP 配置示例 130
Hadoop Group Mapping 的示例 LDAP 配置 130

Hue 的 LDAP 配置示例 131
附录 D：数据目录迁移常见问题 133

关于本指南

许多客户在迁移到 Amazon EMR 时都会遇到不少问题，涉及评估、计划、架构选择以及如何满足将 Apache Spark 和 Apache Hadoop 等分析应用程序从本地数据中心迁移到新的 AWS 云环境的诸多要求。许多客户都对分销商的可行性或纯开源软件方法心存疑虑，他们需要切实可行的更改建议。本指南包括迁移的所有步骤，并提供了我们积累的最佳实践，可帮助客户完成迁移过程。

简介

全球各地的企业都在探索 Apache Hadoop 和 Apache Spark 等新型大数据处理和分析框架的强大功能，但他们也发现在本地数据湖环境中运用这些技术会面临一些挑战。当前，在大数据行业飞速发展的同时，一些供应商处境艰难。因此，对于许多客户来说，选择一个安全、长期稳定的平台尤为重要。

由于 IT 部门需要不断地预置资源、处理大量不均衡的工作负载，并跟上快速变化、社区驱动的开源软件创新的步伐，因此通常会遇到缺乏敏捷性、成本过高和管理困难等问题。对底层硬件和软件基础设施进行评估、选型、采购、验收、部署、集成、预置、修补、维护、升级和支持导致的重负，使得许多大数据规划都会面临交付延期和负担过重的问题。

还有一个不易察觉的问题（即使同样重要），那就是各公司的数据中心对 Apache Hadoop 和 Apache Spark 的部署模式，他们直接将计算和存储资源绑定放置在同一组服务器中。而这种捆绑的模式要求必须同时扩展计算和存储资源，因此缺乏灵活性。这意味着几乎所有本地环境都要为大量未充分使用的磁盘容量、处理能力或系统内存买单，因为每种工作负载对这些组件的要求都各不相同。

睿智的企业如何通过大数据归划走向成功?

将大数据（和机器学习）迁移到云具有诸多优势。Amazon Web Services (AWS) 等云基础设施服务提供商可提供选择多样的按需和弹性计算资源、灵活而价优的永久存储以及托管服务（这些服务可提供被广泛认可且持续保持更新的环境以供于开发和运行大数据应用程序）。因此，数据工程师、开发人员、数据科学家和 IT 人员可以集中精力去准备数据和提取有价值的见解。

Amazon EMR、AWS Glue 和 Amazon S_3 等服务让您能够独立地对计算和存储进行解耦和扩展，同时提供一个集成的、托管的高弹性环境，从而即刻减少本地部署模式存在的许多问题。这种模式可以让大数据和数据湖项目变得更快、更敏捷、更易于使用和更经济高效。

然而，在基于云的部署中，来自传统本地部署模式的 Apache Hadoop 和 Apache Spark 的常识经验并非总是最佳实践。使用简单的*直接迁移方法（lift and shift）*复制照搬本地集群节点在云中运行，

从概念上讲很容易，但实践起来并不能达到理想效果。在将大数据迁移到云架构时，您需要投入更多的架构思考，采用不同的设计决策来实现利益最大化。

本指南提供了以下几个主题的最佳实践：

- 迁移数据、应用程序和数据目录
- 使用永久性和临时性资源
- 配置安全策略、访问控制和审计日志
- 估算并最大限度降低成本，同时最大限度提高价值
- 利用 AWS 云实现高可用性 (HA) 和灾难恢复 (DR)
- 实现常见管理任务的自动化

尽管本指南无意代替专业服务，但它涵盖了大量常见问题，以及将大数据和数据湖计划迁移到云的场景。

开启您的旅程

迁移方法

在开启从大数据平台到云的迁移之旅时，您首先必须决定如何进行迁移。一种方法是*重新架构(re-architect)*平台，以充分实现云带来的优势。另一种方法称为*直接迁移(lift and shift)*，即采用现有架构完成向云的直接迁移。最后一种方案是混合方法，也就是将直接迁移与重新架构结合使用。由于两种方法各有利弊，因此做出选择并非易事。

通常来讲，直接迁移方法更简单，不确定性和风险也更小。另外，若工期紧张（例如数据中心租约即将到期），这种方法会更好。但是，直接迁移也有缺点，它并不总是最经济高效的，而且现有架构可能难以映射到即刻可用的云中解决方案。

重新架构具有许多优势，包括对成本和效率的优化。通过重新架构，您可以使用最新、最优的软件，可以与云原生工具更好地集成，还可以利用云原生产品和服务来降低后期运维的负担。

本文将从 Apache Hadoop 生态系统的角度阐述每种迁移方法的利弊。如需借助综合性资源来确定哪种方法适合您的工作流程，请参阅 An E-Book of Cloud Best Practices for Your Enterprise，从更加全面的视角了解迁移到云的最佳实践。

重新架构

若您希望充分实现迁移到云的优势，重新架构将是理想之选。重新架构需要调研、规划、试验、培训、实施和部署。这些工作虽然会耗费资源和时间，但是由于降低了硬件和存储成本、减少了运营维护并实现了最大的灵活性，可以满足未来的业务需求，因此通常可以带来最大的回报率。

重新架构的迁移方法可以为应用程序带来以下优势：

- 由于存储和计算资源分离，因此可独立扩展组件。
- 利用最新功能和软件，提高生产效率并降低成本。
- 由于资源预置速度快，能够快速进行原型设计和实验。

- 可以自由选择垂直扩展（通过请求功能更强大的硬件）和水平扩展（通过请求更多硬件设备）。
- 无需再对集群生命周期的许多方面（包括更换故障节点、升级、修补等）进行管理，从而降低了运营负担。由于可将集群视为临时性资源，因此可以将其退役销毁再重新启动新的资源。
- 使用数据湖架构实现数据共享访问，数据存储在中央存储系统中，可供各种服务和工具使用，以针对不同的使用案例提取和处理数据。例如，使用 AWS Glue 和 Amazon Athena 等无服务器模式运行的服务可以大大减轻运营负担并降低成本，并且只有将数据存储在 S_3 上时才能从这种模式中获益。
- 可以将计算实例视为临时性资源，仅在您主动需要时使用且用量按需求足够即可。

重新架构的最佳实践

重新架构系统以使用 Amazon EMR 时，请考虑以下最佳实践：

- 阅读本指南中的文档，获取迁移成功案例所采用的参考架构和方法。
- 提前与 AWS 代表联系，获取满足您的使用案例和目标要求的架构路线图。

直接迁移

在时间紧迫且存在大量不确定性的情况下，将工作负载从本地迁移到云的理想之选是采用直接迁移方法。直接迁移是指将现有应用程序从一种基础设施迁移到另一种基础设施。这种方法的优势有：

- 更改较少。由于目标是将应用程序迁移到与现有环境相似的环境中，因此只需做出相应更改，使应用程序可以在云中运行即可。
- 由于更改较少，因此减少了未知因素和意外工作，从而降低了风险。
- 由于更改数量较少，因此减少了工程师所需的培训量，从而缩短了产品上线时间。

直接迁移的最佳实践

- 可以考虑使用 Amazon S3 代替 HDFS 进行存储，因为这种方法可以显著降低成本，并且可以根据实际有效数据量需求配置计算资源。使用 HDFS 时，必须至少有一份副本数据，这需要更多的存储空间。使用 HDFS 作为存储方案 时，主要的成本动因是需要使用昂贵的基于本地磁盘存储的 EC2 实例，或使用大型 EBS。使用 AWS 成本计算器进行的快速计算显示，HDFS 的存储成本可达 Amazon S3 成本的三倍之多。有关指南，请参阅使用 Amazon S3 作为中央数据存储库。

- Amazon EMR 可将多个版本的应用程序捆绑到您选择的单个 Amazon 系统映像 (AMI) 中。若您选择将某个应用程序版本更新至较新版本，建议提前了解版本之间的更改并查找已知问题。对于开源应用程序来说，升级到较新版本可能会引入 bug 错误。

- Amazon EMR 集群使用默认值配置，这些默认值取决于所选的实例类型。有关 Hadoop 任务级默认值，请参阅任务配置；有关 Apache Spark 默认值，请参阅 Amazon EMR 设置的 Spark 默认设置。这些默认值适用于大多数工作负载，但是也有部分作业可能需要您修改这些默认值。

- 默认情况下，Amazon EMR 集群会使用容量调度器 (capacity scheduler)用作 Apache Hadoop 资源调度器。请验证此调度器配置与您正在迁移的使用案例相匹配。

混合架构

混合架构会综合利用直接迁移和重新架构两种方法。对于现有应用程序，采用*直接迁移*方法可以快速完成迁移。然后，任何新的应用程序都可以使用*重新架构*的架构。这种混合方法的优势在于，能够提前进行试验，并在全面迁移到云之前获得有关云技术和范式的经验。

原型设计

迁移到不熟悉的新产品或服务时，总是需要一段时间的学习。通常，最好的学习方法是进行原型设计并从实践中学习，而不是单纯进行研究，这有助于尽早发现流程中的未知因素，以便您稍后对此进行规划。建议您将原型设计变成强制要求以对假设进行验证。比较常见的使用新产品和服务时的待验假设包括：

某个特定的数据格式是我的使用场景下的最佳格式。 很多时候，客户认为特定的数据格式是最好的。然而，并没有适用于所有使用案例的最佳数据格式。有些数据格式的性能优于大多数其他数据格式，例如 Apache Parquet 和 Apache ORC。验证这一假设也相对较容易，但可能带来巨大的正向影响。在迁移到生产环境后再切换数据格式比使用实际数据运行快速原型验证要昂贵得多。强烈推荐使用原型设计提前进行验证。

处理特定工作流程时，特定应用程序的性能会优于其他应用程序。(如 spark vs hive)这种情形与上述数据格式的常见假设是一样的。在迁移后期，更改应用程序可能会耗费昂贵的成本，并且会影响新应用的采用。

使用某特定实例类型是运行特定工作流程最经济高效的方法。 很多时候，若另一实例类型针对工作流程进行了调优，则其性能会更好。例如，若启用溢出到磁盘功能(spill-to-disk)，则 C 系列 EC2 实例的性能可能比 R 系列 EC2 实例更好，成本更低。这种方案在迁移后也比较容易更改，若成本和性能是优先级较高的要求，建议使用此方案。

在本地运行的特定应用程序在云上应有相同运行方式及性能表现。 影响工作负载运行的因素有很多，比如实例类型、存储类型、应用程序版本、基础设施配置等。推荐您使用预计会在生产环境中运行的实际数据来运行各种作业来实现最全面的验证。

对于云上，在工作负载可能运行的环境中存在多个影响因素。例如，在一天的不同时间里，请求 Amazon S3 的流量可能会有所不同，或者数据可能会在不需要时进行缓存。因此，原型设计可降低开发和部署过程中意外事件的数量和严重性，而且回报率可能会很高。最后，在开发周期中，越早发现问题，成本越低。

原型设计的最佳实践

- 确定原型设计的内容，集思广益，讨论所有可能的假设和未知因素。优先考虑可能产生最大影响的假设和未知因素。
- 尽早开始，首先选择风险最大的方面进行原型设计。
- 进行原型设计的环境需与实际运行环境类似。从较小的环境或特征子集开始，然后再扩大规模。

- 预先确定测试目标并获得利益相关方的支持。测试目标可以是验证关于某框架如何工作的原理，也可以是验证某个设计。
- 测试要具有确定性并易于重复。使用自动化方法（例如运行脚本或者持续集成的环境），以便在不同的环境中运行测试。例如，在不同的实例类型上或针对多个 AMI 运行测试。这些脚本稍后可以用作部署测试。
- 与其他人一起验证您的测试设置、环境和结果，以确保所有因素都考虑在内。例如，若您对相同的 S_3 对象运行下载测试，可能会导致 Amazon S_3 缓存对象。当实际的工作流程是随机下载对象时，此方案会给出错误的结果。
- 充分运行测试，消除依赖项可能带来的性能差异。例如，Amazon S_3 的多变性可能是其他用户的流量负载或一天中的不同时间造成的。查看不同的百分比（例如 P_{50}、P_{90}、P_{99} 和 P_{100}），并确定差异性可能会对用户体验产生的影响。
- 记录结果，并让团队成员复核。该复核可确保测试的正确运行和结果的一致性。
- 不要自作假设。在大数据分析领域，影响性能、成本和功能的变量太多，这意味着一个看似明显的假设可能并不成立。始终通过测试来验证您的假设。例如，许多人可能很自然的会认为，与选择其他实例类型相比，他们选择和本地使用的硬件十分相近的特定实例类型可能会更合适。
- 原型设计的目的是验证对某设计或模式的假设，使结论具有高度的确定性。通常，若在原型设计中投入更多精力考虑多种因素，则设计在生产环境中正常运作的确定性就会越高。在流程开始时确定目标将确保您的输出保持合理水准。
- 踊跃地通过在论坛上发帖、咨询 AWS 合作伙伴以及与您的客户团队联系来寻求帮助。

选择团队

在准备迁移到云时，您必须谨慎选择项目团队来研究、设计、实施和维护新系统。我们建议您的团队可以有人承担以下角色，同时明白一个人可承担多个角色：

- 一名**项目领头人**，能够端到端管理项目，并具有足够的技术专业知识和大数据技术背景来制定决策。

- 一名**大数据应用程序工程师**，在 Apache Hadoop 和当前迁移涉及到的其他技术方面具有深厚背景。该背景有助于理解基础应用程序的工作方式、预期用途及局限性。
- 一名**基础设施工程师**，精通如何在 AWS 上自动预置基础设施，熟悉 AWS CloudFormation、Terraform（由 HashiCorp 开发）等工具。此人还应该熟悉测试自动化，包括 CI/CD 方法和工具。
- 一名**安全工程师**，能够就公司指定的安全需求提供指导。此人需充分了解相关知识，可以将组织的安全要求与 AWS 提供的安全功能进行配对。此人还需在安全设计方面做到灵活变通，只要符合预期需求即可。
- 一**组工程师**，具有出色的快速学习能力，对于不熟悉的领域勇于深入探索。

迁移团队中的所有成员都必须具备以下特质：

- 必须具有开放的思想和从不同角度思考的能力。云基础设施需要在资源处理方式上进行范式转换。
- 他们必须能够深入研究基础框架、架构和问题。
- 他们必须就项目的任务和方向达成共识。

迁移的一般最佳实践

将大数据和分析工作负载从本地迁移到云需要谨慎决策。以下是将这些工作负载迁移到 Amazon EMR 时需要考虑的一般最佳实践：

考虑使用 AWS Glue、Amazon Redshift 或 Amazon Athena。 虽然 Amazon EMR 具有出色的灵活性且可以提供最大程度的定制和控制，但是管理 Amazon EMR 集群、升级等任务也存在相关成本。考虑使用其他符合您要求的托管 AWS 服务，因为它们通常运维负担较低，并且在某些情况下还可以降低成本。若这些服务均无法满足您的使用案例要求，请使用 Amazon EMR。

使用 Amazon S3 进行存储（数据湖基础设施）。数据湖是一个集中式存储库，可让您以任何规模存储所有结构化和非结构化数据。数据可以按原样存储，而不必先进行结构化。您可以对数据执行不同类型的分析：从驾驶舱仪表盘和可视化，到大数据处理、实时分析和机器学习等。

Amazon S3 的架构具有高耐用性和高可用性，并支持生命周期策略来实现分层存储的设计。更多详情，请参阅使用 Amazon S3 作为中央数据存储库。

将计算与存储解耦，以便对它们进行独立扩展。数据进入 Amazon S3 后，您可以使用 Amazon Elastic Compute Cloud (EC2) 按需启动计算容量。更多信息，请参阅使用 Amazon S3 的益处。

使用多个 Amazon EMR（临时和长期运行）集群，实现可按需启动和关闭集群的设计。 分析当前工作负载，并根据使用模式将工作负载分配给不同的集群。

- 将批处理作业（提取、转换、加载 [ETL]、聚合、数据清理、汇总、机器学习）和交互式查询（一次性分析）分离出来。
- 分别使用预留实例和 Spot 实例分别来降低基线负载要求和可变工作负载的成本。

- 在集群内使用自动扩展功能。自动扩展允许根据 Amazon CloudWatch 指标和扩展策略中指定的其他参数，以编程方式缩减和扩展核心节点和任务节点。
- 合理调整集群大小（实例类型和实例数量）。EMR 提供了多种 Amazon EC2 实例类型，请确保根据工作负载选择正确的实例类型。更多详情，请参阅成本估算和优化。

实施自动化和持续集成/持续交付 (CI/CD) 实践，以进行测试和提升效率。 与其他资源（例如角色和安全组）一起自动预置 EMR 集群是卓越运营的最佳实践。推荐将通常用于应用程序代码的工程规范应用于基础设施。将构建基础设施的代码检入代码存储库，并构建 CI/CD 管道以测试代码。此外，实施*基础设施即代码*还允许在另一个可用区或 AWS 区域中预置 EMR 集群，以防当前使用的可用区或 AWS 区域出现问题。更多详情，请参阅卓越运营。

在迁移过程中尽早让安全和合规工程师参与其中，并确保 EMR 环境符合组织的安全指示。充分利用多种安全相关服务（例如 AWS Identity and Access Management [IAM] 和 AWS Key Management Service [KMS]）和功能（例如 EMR 中的安全配置）。Amazon S3 也包括许多安全相关功能。确保所有静态数据和传输中的数据都已加密。最后，确保在适当情况下启用身份验证和授权。更多详情，请参阅保护 Amazon EMR 上的资源。

收集需求

获取本地指标

以下指标列表有助于估算成本、规划架构和选择实例类型。

在现有 Hadoop 集群上捕获所有这些指标，可帮助在迁移过程中推进决策流程。

- 合计物理 CPU 数量
- CPU 时钟速度和内核数量
- 合计内存大小
- HDFS 存储量（不包含副本）
- 合计最大网络吞吐量
- 上述所用资源至少历时一周的利用率图表

如在全面了解本地架构和可能的迁移要求方面需要帮助，请参阅附录 A：需求收集调查问卷。

成本估算和优化

使用 Amazon EMR，您只需以秒为单位支付集群使用费用。影响费用的因素包括：实例类型、部署的 Amazon EC2 实例数量以及启动集群时所在的 AWS 区域。Amazon EMR 成本是除 Amazon EC2 成本和 Amazon EBS 成本（若使用 EBS 卷）之外的成本，这二者也按秒计费。本章将介绍如何优化 Amazon EMR 集群以及如何利用 Auto Scaling 和实例队列等功能来降低成本。

优化成本

Amazon EMR 提供了多种有助于降低成本的功能。要充分利用这些功能，请考虑以下因素。

工作负载类型

您可以在 Amazon EMR 上运行不同的应用程序和工作负载类型。对于仅在特定时段内运行的应用程序，可以使用*临时性* EMR 集群。对于长时间运行的应用程序，可以使用*长时间运行*集群。下图列出了典型的工作负载类型，以及它们是被归类为临时性还是长时间运行。

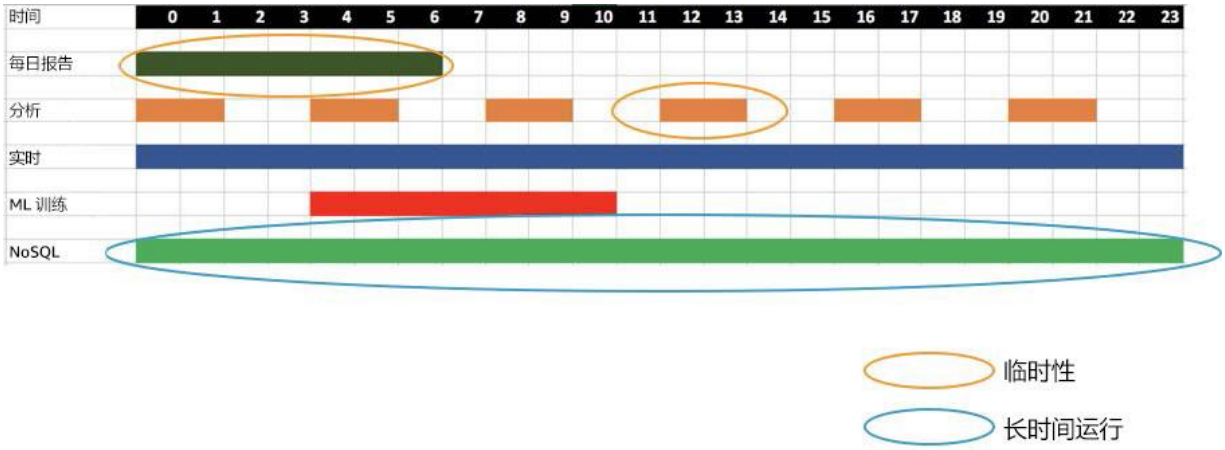


图1：常见工作负载及其集群类型

实例类型

大多数 Amazon EMR 集群都可以在通用 EC2 实例类型/系列（即 M5.large 和 M5.xlarge）上运行。对计算密集型集群而言，在高性能计算 (HPC) 实例（例如计算优化的实例系列 [即 C5]）上运行可

能比较有益。对数据库和内存缓存应用程序而言，在内存增强型实例（例如内存优化的实例系列 [即 R5]）上运行可能会比较有益。对于主节点不会有巨大的计算型请求。对于节点数不超过 50 的大多数集群，可以使用通用实例类型，例如 m5.large（**注意：**对于节点数超过 50 的集群，请考虑使用较大的实例类型，例如 m5.xlarge）。您可以处理的数据量取决于核心节点的容量和输入数据的大小、处理期间的数据大小，以及输出数据的大小。

应用程序设置

此外，作业性能还取决于应用程序设置。使用案例不同，应用程序设置也有所不同。例如，默认情况下，安装有 Apache HBase 的 EMR 集群会为 HBase 分配一半内存，为 Apache Hadoop YARN 分配另一半内存。若您使用 Apache HBase 和 Apache Spark 之类的应用程序，我们建议您不要把两个应用程序运行在单个较大的集群。而要在单独的、较小的集群上运行每个应用程序。

存储优化

使用 Amazon EMR 时，您可以通过将 Amazon S3 用作持久数据存储来解耦计算和存储。优化存储后，作业性能将得到提升。通过这种方法，您可以减少硬件的使用并缩短集群的运行时间。以下策略可帮助您优化集群 (Amazon S3) 存储：

对数据进行分区

当您的数据已分区，并且您基于分区列读取数据时，查询只会读取所需文件。这可减少查询期间扫描的数据量。例如，下图显示了对两个相同大小的数据集执行的两个查询。一个数据集已分区，而另一个数据集未分区。

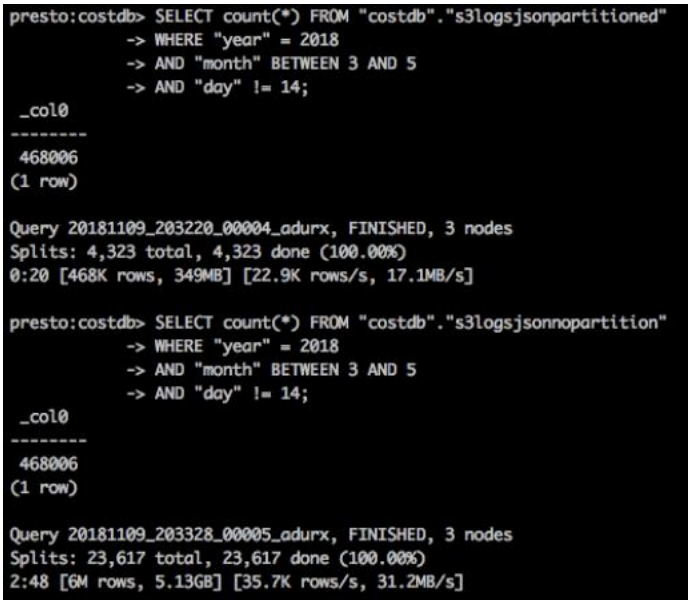


图2：对已分区和未分区数据的查询

对已分区数据（`s3logsjsonpartitioned`）执行的查询用时 20 秒，扫描了 349 MB 数据。

对未分区数据（`s3logsjsonnopartition`）执行的查询用时 2 分 48 秒，扫描了 5.13 GB 数据。

优化文件大小

避免使用过小的文件（通常是指小于 128 MB 的文件）。若拥有尺寸较大、数量较少的文件，您可以减少 Amazon S3 LIST 请求的数量，并提升作业性能。为了说明文件过多对性能的影响，下图显示了对包含 50 个文件的数据集执行的查询，以及对大小相同但包含 25000 个文件的数据集执行的查询。

```
presto:costdb> select count(*) from fewfilesjson;
   _col0
-----------
 200000012
(1 row)

Query 20181108_230835_00008_5ma32, FINISHED, 3 nodes
Splits: 567 total, 567 done (100.00%)
2:31 [200M rows, 25.6GB] [1.33M rows/s, 174MB/s]

presto:costdb> select count(*) from manyfilesjson;
   _col0
-----------
 200000012
(1 row)

Query 20181108_232128_00009_5ma32, FINISHED, 3 nodes
Splits: 25,017 total, 25,017 done (100.00%)
3:47 [200M rows, 25.6GB] [882K rows/s, 116MB/s]
```

图3：不同数目文件的查询时间差异

对包含 50 个文件的数据集（`fewfilesjson`）执行的查询用时 2 分 31 秒。

对包含 25000 个文件的数据集（`manyfilesjson`）执行的查询用时 3 分 47 秒。

压缩数据集

通过压缩数据，您可以减少数据所需的存储量，并且可以最大程度地降低 S3 和 EMR 节点之间的网络流量。压缩数据时，请确保使用的压缩算法允许拆分文件，或者针对集群上的并行化将每个文件处理为最佳大小。默认情况下，Apache Parquet 或 Apache ORC 等文件格式均可提供压缩。下图显示了 Parquet（已启用压缩）和 JSON（文本格式，未启用压缩）两种文件格式之间的大小差异。Parquet 数据集几乎只有 JSON 数据集的五分之一：

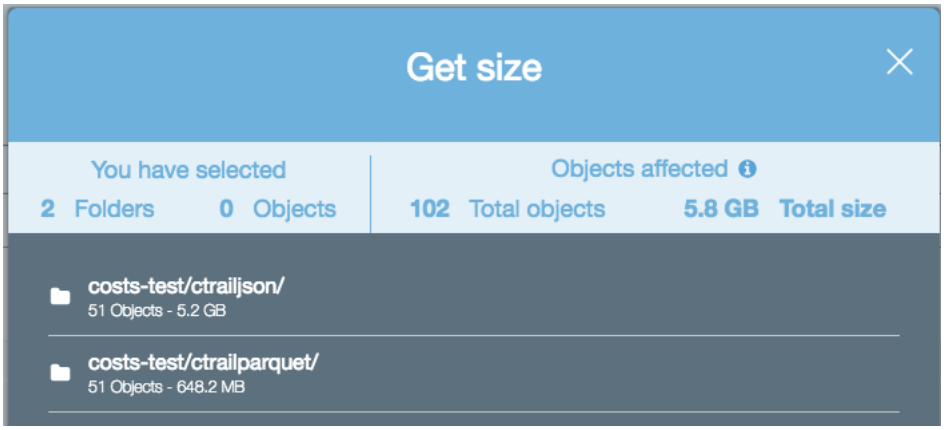


图4：已压缩和未压缩的数据集

优化文件格式

Parquet 和 ORC 等列式文件格式可以提高读取性能。若大多数查询仅选择一个列子集，则列式格式是理想之选。对于主要选择所有列、但只选择一个行子集的使用案例，请选择行优化的文件格式（例如 Apache Avro）。下图对 Parquet 和 JSON（文本）文件格式的 `select count(*)` 查询性能进行了比较。

```
[hadoop@ip-172-31-0-49 ~]$ presto-cli --schema costdb --catalog hive
presto:costdb> select count(*) from ctrailjson;
  _col0
---------
 8006149
(1 row)

Query 20181108_215514_00002_5ma32, FINISHED, 3 nodes
Splits: 217 total, 217 done (100.00%)
0:56 [8.01M rows, 5.21GB] [143K rows/s, 95.3MB/s]

presto:costdb> select count(*) from ctrailparquet;
  _col0
---------
 8006149
(1 row)

Query 20181108_215627_00003_5ma32, FINISHED, 3 nodes
Splits: 67 total, 67 done (100.00%)
0:02 [8.01M rows, 0B] [4.47M rows/s, 0B/s]
```

图 5：不同文件格式的性能比较

查询 JSON 数据集用时 56 秒，扫描了 5.21 GB 数据。

查询 Parquet 数据集用时 2 秒，无需扫描任何数据。

计算优化

在上一节中，我们介绍了一些可用于优化 Amazon S3 存储成本和性能的策略。在本节中，我们将介绍一些优化 Amazon EC2 集群计算的功能和方法。

Amazon EC2 提供了多种购买选项供您选择。启动 Amazon EMR 集群时，您可以使用按需、Spot 或预留 EC2 实例。与按需实例相比，Amazon EC2 Spot 实例可以折扣价提供备用计算容量。与按需定价相比，Amazon EC2 预留实例可让您以较大折扣预留 EC2 实例。更多详情，请参阅实例购买选项。

Spot 实例

使用 Spot 实例运行 EMR 集群适用于很多场景。但是，在为特定工作负载选择 Spot 实例之前，您必须考虑以下几点。例如，若您正在运行的作业需要预计完成时间或具有服务等级协议 (SLA) 要求，则 Spot 实例可能不是最佳选择。对于成本比完成时间更为重要的工作负载，或者可能超出 SLA 限制的工作负载，可以将 Spot 实例用于整个集群。

对于某些工作负载，您也可以将 Spot 实例和按需实例结合使用。例如，若成本比完成时间更重要，但是您不能接受部分工作丢失（即终止整个集群），则可以将 Spot 实例用于*任务节点*，将按需/预留实例用于*主节点*和*核心节点*。

Spot 实例还非常适用于测试和开发工作负载。您可以将 Spot 实例用于整个测试集群，帮助您在测试新应用程序时降低成本。

预留实例

与按需实例相比，使用预留实例 (RI) 时，您可以较低的价格购买/预留 EC2 容量。请注意，要降低预留实例的成本，您必须确保一年内的 RI 使用率高于 70%。例如，若您使用临时性 EMR 集群且集群每天总共仅运行 12 个小时，则您的年使用率为 50%。这意味着预留实例可能无法帮助您降低该工作负载的成本。预留实例可以帮助您为长期运行的集群和工作负载降低成本。

实例队列

实例队列是 Amazon EMR 的一项功能，可为预置 EC2 实例提供多种选项。这种方法让您可以轻松使用 Spot 实例、按需实例或者结合使用二者预置 EMR 集群。使用实例队列启动集群时，您可以选择按需实例和 Spot 实例的目标容量，还可以指定实例类型和可用区列表。使用实例队列可选择最能够满足您启动请求的实例类型和可用区。

实例队列还提供了许多用于预置 Spot 实例的功能。其中包括指定保持 Spot 实例运行所需的定义持续时间（也称作 Spot 限制）、您愿意支付的最高 Spot 价格，以及预置 Spot 实例的超时期限。

Amazon EMR Auto Scaling

您可以使用 Amazon EMR 自动扩展功能来动态扩展集群，从而降低成本。Amazon EMR Auto Scaling 允许您通过选择 Amazon CloudWatch 指标和指定扩展策略来自动扩展（向集群添加更多节点）和缩减

（从集群中删除节点）核心节点和任务节点。通过这种方式您只需使用应用程序需要的适当资源来运行 EMR 集群。并且这种方法同样有助于在集群使用率存在峰值时提升性能（如用户提交新的作业），您希望集群根据该应用程序的要求自动扩展。更多信息，请访问 AWS 大数据博客，参阅 Best practices for resizing and automatic scaling in Amazon EMR。

成本估算概述

在估算 Amazon EMR 集群的成本时，需要考虑多重因素。这些因素包括 EC2 实例（计算层）、EBS 卷和 Amazon S3 存储。由于 Amazon EMR 采用按秒计价，运行持续时间较短的大型 EMR 集群的成本与运行持续时间较长的小型集群的成本相差无几。例如，运行 10 小时 10 节点集群的成本与运行 1 小时 100 节点集群的成本相同。小时费率取决于所使用的实例类型（例如，标准实例、高 CPU 实例、内存增强型实例、高存储实例等）。有关定价详情，请参阅 Amazon EMR 定价。

优化基于 Apache Hadoop YARN 的应用程序

关于 Apache Hadoop YARN 和作业优化

Apache Hadoop YARN 可管理虚拟资源。在向 YARN 注册节点时，系统将提供虚拟内存，以及节点可在其中运行的虚拟 CPU 资源。参与作业执行的所有节点的总和即是集群可用的总资源数。理想情况下，YARN 内存和 CPU 资源应与底层硬件的内存和 CPU 资源尽可能接近的匹配，但在某些情况下需要进行调整。使用 Amazon EMR 时，系统将为每个节点提供默认值。有关这些默认值的更多信息，请参阅《*Amazon EMR 发行指南*》中的任务配置。

作业(job)需要使用资源来完成计算。为实现作业并行化，作业要在容器中运行工作子集。作业从 YARN 请求每个容器所需的内存和 CPU 数量。若未指定，则系统会为每个容器分配默认容器大小。若容器大小不合适，作业将因为无法使用分配给它的所有容量而浪费资源，或者由于受到约束而运行缓慢，或者由于资源过于受限而失败。

您必须同时兼顾 YARN 和作业请求，以确保充分使用底层硬件。YARN 可管理虚拟资源，且不一定是和底层硬件为严格映射关系，但是如何配置 YARN 和如何安排调度任务确实会影响底层硬件的使用方式。

优化和监控集群

要调整集群，请首先确保已优化 YARN 的使用。若某些容器始终可用，则由于容器处于闲置状态，缩小集群可以节省成本但不会降低性能。Amazon EMR 向可提供此信息的 Amazon CloudWatch 发出一个 ContainerPending 指标。若容器请求的队列恒定，则增大集群大小有助于应用程序更快地完成，因为它们可以获益于并行性的提高。

为确保您正在使用所有物理资源，请使用 Ganglia 监控软件来提供底层硬件的相关信息。若 YARN 资源（vCPU 和内存）得到 100% 的利用，但 Ganglia 显示实际 CPU 和内存使用率未超过 80%，则您可能需要减小容器大小，以便集群可以运行更多并发容器。

若 Ganglia 显示 CPU 或内存使用率为 100%，但其他资源并未得到充分使用，则可考虑改用另一种性能较高、成本较低的实例类型。例如，若 CPU 使用率为 100%，而在 R4 或 M5 系列实例类型上的内存使用率不到 50%，则改用 C4 系列实例类型或可解决 CPU 瓶颈问题。

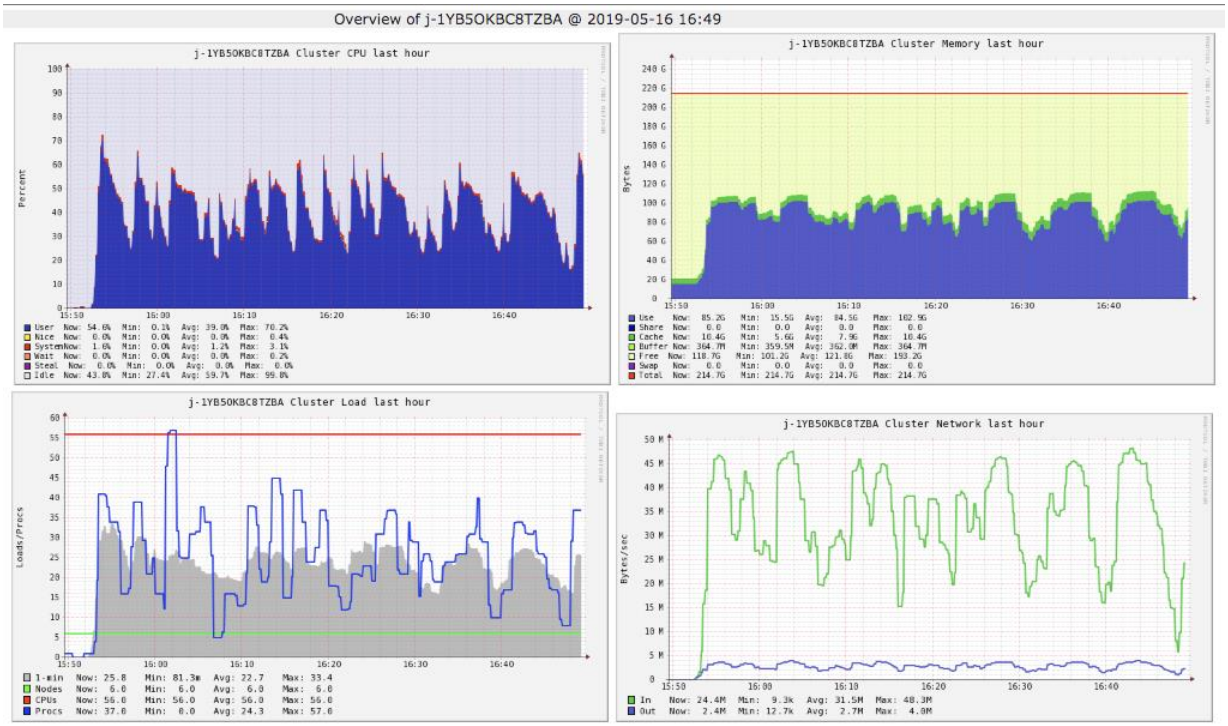


图6：显示CPU 和内存未得到充分利用的示例Ganglia 图表

调整控制机制，实现集群优化

若 CPU 和内存未得到充分利用，则可以调整两个控制机制。第一个控制机制是通过 YARN 请求为每个容器预留的资源量。要调整容器的可用资源，您可以使用 Amazon EMR 应用程序配置更改默认容器大小。同一配置还可用于在应用程序级别控制内存和 CPU，以便更好地控制资源。

第二个控制机制是 YARN 中每个节点可以保留的虚拟资源量。要更改集群中每个节点上可保留的 YARN 内存或 CPU 量，请使用 Amazon EMR 配置 API 设置 `yarn.nodemanager.resource.memory-mb` 和 `yarn.nodemanager.resource.cpu-vcores` 配置。有关默认值的信息，请参阅《Amazon EMR 发行指南》中的 Hadoop 守护程序配置设置。

以下决策图为优化作业提供了建议方法。

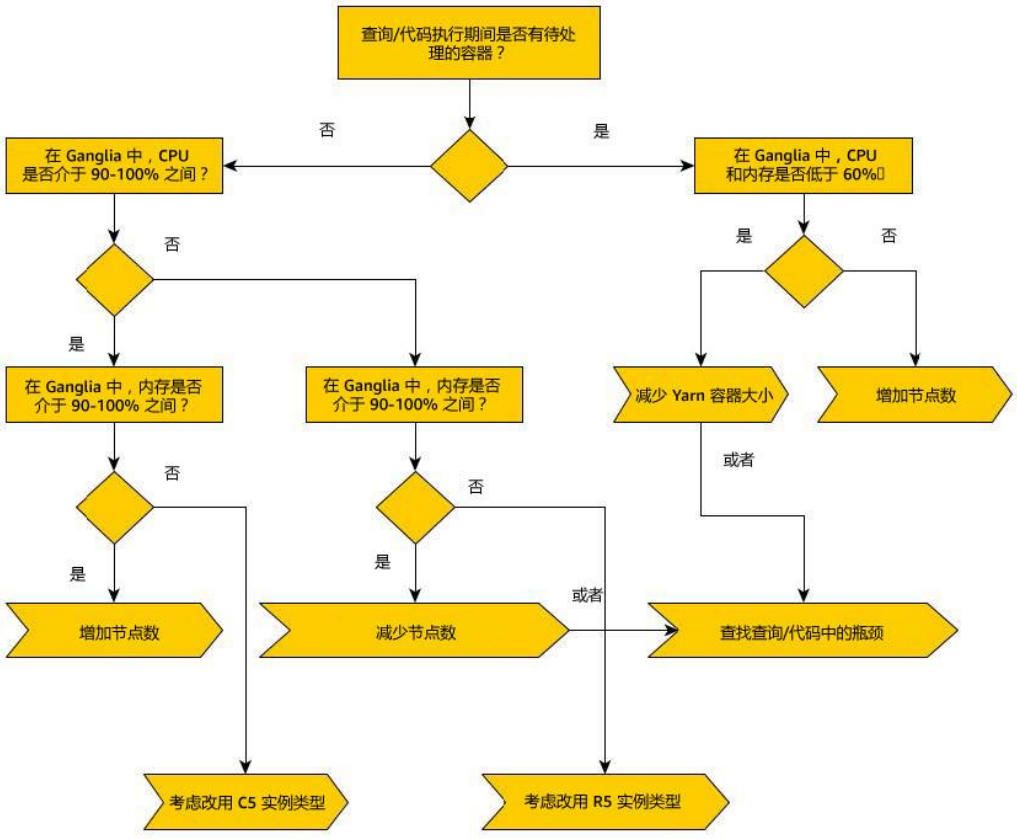


图7：作业优化决策图

Amazon EMR 集群拆分方案

使用 Amazon EMR 的一个主要优势是，您可以轻松启动和终止集群。通过快速启动和终止集群，您可以获得单个长时间运行的集群所无法实现的灵活性，并可以利用 Amazon EC2 Spot 实例节省成本。

本节将介绍将一个永久运行的集群拆分为多个较小集群的几种方法，并列出了这些做法为环境带来的优势。您选择哪种方法取决于您的使用案例和现有工作流程。AWS 可与您合作，帮助您选择有助于实现目标的策略。

集群特征

您可以从不同的角度来处理拆分现有集群的任务，具体取决于您要处理的区域或特征集。您选择的策略取决于您的场景和期望实现的目标。通常要考虑以下集群特征。

机器类型和集群大小

在集群环境中，已构建的不同工作流程可能会在不同资源上遇到瓶颈，例如 CPU 数量、内存大小、网络带宽或网络延迟。在按需启动的小型集群中，您可以选择最适合工作负载的机器类型。同样，还可以选择集群所需的资源数量。例如，您可以使用内存增强型实例来处理机器学习算法，也可为 PrestoDB 等查询引擎运行计算优化的实例。

应用程序、应用程序版本和配置

您可以创建不同的配置并在不同的集群上部署不同的应用程序，从而仅提供用户所需资源。您还可以创建新集群，以便在应用程序版本升级期间运行红黑测试以测试新软件。

安全性

各较小集群的特定 IAM 角色

Amazon EMR 集群分配有一个 EC2 角色，该角色具有访问不同 AWS 资源的权限。您可以为集群分配不同的角色，以便这些角色对各种资源（例如，Amazon S3 存储桶或 Amazon Kinesis Data Streams）具有不同的访问和权限。通过此角色分配，您可以为每个 EMR 集群提供经过严格限制的权限，从而防止意外操作，并在发生任何安全事件时缩小风险范围。

安全控制机制

您可以根据集群所服务的使用案例更改集群的安全级别。例如，若某个集群仅用于从工作流程引擎或通过步骤提交的批处理，则无需将在用户级别保护该集群列为优先级最高的任务。在这种情况下，由于没有人员与集群进行交互，因此可以不用配置 Kerberos。对于服务于交互使用案例的集群，您可能需要配置 Kerberos、限制 SSH 访问并遵循其他安全实践。有关 Amazon EMR 中提供的安全控制机制的信息，请参阅*《Amazon EMR 管理指南》*中的安全性。

网络控制机制

您可以为不同的集群分配不同的安全组，也可以将它们放置在不同的子网中（这些子网可控制来自指定网络源的访问）。

灾难恢复

使用多个集群可提供冗余，在某个集群出现故障或脱机时最大限度地降低影响。以下展示了几个使用案例，它们通过使用多个集群帮助解决问题：

- 由于软件错误、网络问题或其他外部依赖项不可用，导致集群运行不正常。
- 需要执行维护操作，例如软件升级、需要重新启动计算机的补丁或应用程序跳转。

常见集群分拆方案

生命周期阶段

决定如何对集群进行分拆的一种常用方法是为生命周期中的不同阶段（如测试、beta 和生产）提供专用集群。这样，尚未准备进行生产的作业可以在自己的专用集群上运行，且不会干扰生产作

业或与之争用资源或写入数据结果。在不同的阶段使用不同的集群，这还可以让您在安装了较新版本应用程序的集群上测试作业。这种方法让您可以在升级 beta 或生产环境之前对升级进行测试。为了进一步隔离您的工作流程和场景，您可以在 Amazon EMR 中应用一个单独的实例角色，该角色不允许 beta 作业将其结果写入生产 S_3 位置，从而保护数据免受 beta 阶段环境引起的意外删除或修改。

工作负载类型

若集群的服务对象是提交临时查询的最终用户，通常需要更严格的安全控制机制。他们还必须使用不同的应用程序，例如 Apache Hue 和 Apache Zeppelin。这些交互式工作负载集群的使用率通常在工作时间内达到峰值，而在其他所有时间则使用率较低。为这些集群配置自动扩展策略具有诸多益处，尤其是在 Spot 实例上运行时。

用于批处理/ETL 工作负载的集群的峰值使用时间通常也不同于用于查询的交互式工作负载集群。批处理/ETL 工作负载集群可以利用自动扩展，从而使工作负载独立于其他集群进行扩展和缩减。更多信息，请参阅使用 Amazon EMR 中的自动扩展功能。

对时间要求严格的作业

另外一种常见的集群分拆策略是，根据作业是否对时间有严格要求来创建单独的集群。当重复作业的完成时间必须一致时，可以在单独的集群上创建和运行该作业，以确保每次必须运行该作业时，该作业都可获得可预测且一致的资源量。此外，在运行对时间要求严格的作业时，可以使用算力更大、价格更贵的硬件。

作业运行时长

运行时间较长的作业可能会消耗可用资源，并抢夺其他运行时间较短的作业的资源。为长时间运行的作业和短时间运行的作业运行单独的集群，这有助于加速完成短时间运行的作业，提高它们的 SLA，并且如果它们处于关键路径，通常会提高工作流程的 SLA。使用 Spot 实例时，运行短时间运行的作业的集群完成作业的几率更高，因为作业被 EC_2 中断的几率较低。

组/组织类型

某些组织会为具有相同安全要求且共享 EMR 资源所有权的用户组创建集群。这种方法有助于限制对这些组的访问，并且还允许管理员使用资源标记将成本分配给用户组。将几个较小的集群专用于单独的组也可帮助您做资源隔离。通过这种方法，组织中的一个组不会耗尽同一集群中其他组的资源。此外，这种方法还可以降低一个组访问另一个组的数据的风险。

拆分集群的其他注意事项

若集群分拆的太精细，成本会稍有提高。每个集群都需要添加主节点对集群进行管理，尽管这些实例不需要像核心节点和任务节点那样强大。此外，由于必须对资源进行跟踪，因此添加的集群越多，基础设施的复杂性就会越高。用户通常使用以下方法来处理这种复杂性：使用编排器来确保资源已预置并在适当的时候关闭，并构建监控器来确保在空闲一段时间后关闭集群。有关此方法的示例解决方案，请访问 AWS 大数据博客，参阅使用高级 Amazon CloudWatch 指标和 AWS Lambda 通过空闲检查和自动资源终止优化 Amazon EMR 成本。

最后，由于不与其他作业共享，使用多个集群还会降低实例资源利用率。但是，可以通过使用自动扩展来应对这种场景。

保护 Amazon EMR 上的资源

Amazon EMR 提供了各种工具和方法来保护 AWS 云中的数据处理安全。本章将介绍网络安全、身份验证（证明身份）、授权（向资源授予身份）、审计（哪些身份何时访问了哪些资源）、加密和补丁的相关信息。在本章的最后，我们将介绍满足不同使用案例的常见客户设置示例。

EMR 安全最佳实践

在设计早期考虑到安全性。在迁移开始时实施安全设计可以节省时间并降低复杂性，因为在构建架构时考虑了安全性。若在实施完毕、确保安全成为必需任务后再进行大的改动，可能就需要耗费更多精力。

确保支持部门尽早参与安全架构构建流程。 让负责审查和批准安全架构的部门尽早参与构建流程，并向他们及时通报安全相关决策。他们或许能够在流程早期为您提供建议，以减少或避免在流程的后期对设计进行更改。

了解风险。 安全性主要是在系统受到威胁时将受攻击面和影响降到最小。没有任何一个系统是完全安全的。

获取安全例外。安全部门可以为不再适用的规则或危害风险降低的方面提供安全例外。获取安全例外可以大大降低风险以及获得安全部门批准所需执行的工作范围。例如，对于处理批处理作业且与用户之间没有交互性的 Amazon EMR 集群，您可能不需要 SELinux。

针对不同使用案例使用不同的安全设置。没有用户交互的批处理和 ETL 集群的安全设置可能与以交互方式使用的集群不同。交互式集群可能具有多个与集群进行交互的用户和进程，并且每个用户彼此之间需要不同的访问级别。与交互式集群相比，用于批处理的集群的安全控制级别通常要低得多。

身份验证

身份验证（Authentication）是实体（用户或应用程序）向应用程序或系统证明其身份的过程。登录到应用程序或系统时，用户通常会提供登录名和密码，以证明他们是他们自称的用户。另外，还有其他身份验证方法，比如提供证书或票证。

用户与系统之间的身份验证

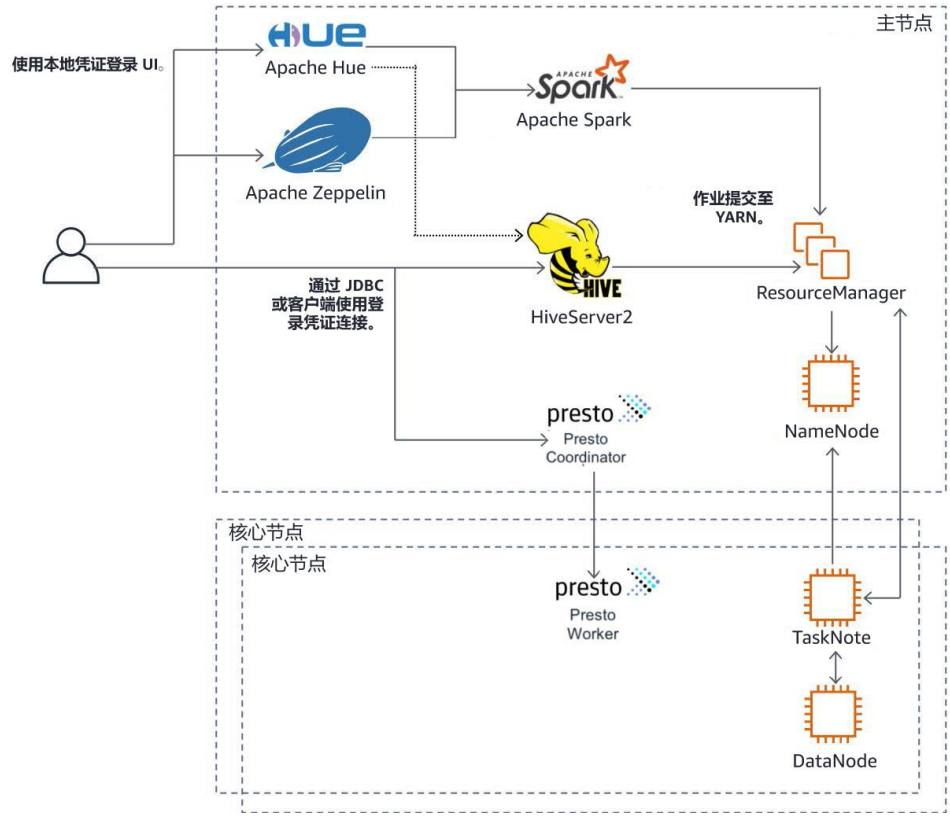


图 8：用户、系统和应用程序三者之间的身份验证

针对 EMR 集群和/或 Amazon EMR 集群上的应用程序对用户进行身份验证的方法有多种。

使用 SSH 密钥进行身份验证

最简单的解决方案是向希望具有 Amazon EMR 集群访问权限的用户提供 SSH 密钥，但这种解决方案也是最不安全的。用户获得密钥后，可通过 SSH 进入集群，并使用集群有权访问的任何资源。由于没有其他安全功能，这些可以登录集群的用户可以以根用户身份运行命令，并且可以访问 Amazon EC2 实例角色有权访问的 AWS 资源。

用户数量或组数量较少时，通常使用此方法。可以通过不同的权限和 SSH 密钥启动多个 EMR 集群，并将这些密钥分发给有权访问每个集群的用户。定期轮换 SSH 密钥，以在密钥泄露时将影响降到最低。若 Amazon EMR 集群有权访问敏感数据，则不建议使用此方法。

轻型目录访问协议 (LDAP) 身份验证

多个应用程序支持使用轻型目录访问协议 (LDAP) 针对 Active Directory 或其他目录服务进行身份验证。使用此方法，用户必须提供他们通常使用的凭证，然后将其发送到本地 Active Directory 服务器进行验证。这些凭证通过验证后，用户即可访问相应的应用程序。[1]

在用户通过验证可以访问应用程序后，可通过两种方式提交作业。第一种方案，应用程序使用自己的身份来提交作业，以进行处理。例如，若 Apache Hue 是应用程序，则 Hue 用户提交作业，并且该作业由定义的用户 *hue* 运行。第二种方法是通过一个应用程序来提交作业，该应用程序模拟用户运行该作业。通过这种方法，管理员可以跟踪用户的使用情况，并可以更轻松地识别系统误用情况。默认情况下，Apache Hue 会提交由用户模拟的作业。

默认情况下，Apache Hadoop 的默认身份验证方法设置为 *simple*，这意味着无需进行用户身份验证。若多个用户使用同一集群，我们建议您更改此设置。 若将 Apache Hadoop 设置为使用 Kerberos，并且通过 Apache Spark 或 Apache Hive 提交 YARN/HDFS 作业，则最终用户必须在每个作为 YARN/HDFS 的节点上拥有操作系统级别的账户。 在某些情况下，必须手动创建这些账户，并使用 SSSD 等工具进行同步。此外，还需要从用户映射到组成员，在此过程中，可以使用 LDAP 或 SSSD。若这是必需的，我们建议您使用 LDAP 对用户进行身份验证，并使用支持 Kerberos 的 Amazon EMR 与操作系统账户自动同步。或者，您也可以通过 LDAP 让 Hadoop 来执行用户到组 (user-to-group)的映射。有关这些映射的内置 Hadoop 实施信息，请参阅 Apache Hadoop 文档中的 Hadoop 组映射。更多信息，请参阅附录 C：LDAP 配置示例。

下表列出了 Amazon EMR 支持的应用程序以及如何启用 LDAP（若支持）的说明。

表1：Amazon EMR 支持的应用程序和LDAP 支持

应用程序	支持 LDAP?	备注
Apache Hive	是	可以使用 HiveServer2 对 LDAP 进行身份验证。关于 HiveServer2 设置的详细信息位于 Apache Hive Wiki 中。配置 LDAP 时，Java 数据库连接 (JDBC) 和 Beeline 连接需要使用用户登录名和密码。 **注意：**我们建议您使用 LDAPS，以确保不共享未加密的凭证。
Presto	是	有关不同配置参数的详细信息，请参阅 Presto 文档中的 LDAP 身份验证。
Apache Spark	否	使用 Hue 或 Apache Zeppelin 对用户进行身份验证，然后将作业提交给 Spark。
Apache Hue	是	有关设置说明，请参阅《Amazon EMR 发行指南》中的为 LDAP 用户配置 Hue；有关分步指南，请访问 AWS 大数据博客，参阅 Using LDAP via AWS Directory Service to Access and Administer Your Hadoop Environment。
Apache Zeppelin	是	Apache Zeppelin 使用 Apache Shiro 来配置身份验证。有关启用 Shiro 的步骤，请参阅适用于 Apache Zeppelin 的 Shiro 身份验证。
Apache HBase	否	使用 Kerberos 作为身份验证机制。

Kerberos

Kerberos 是可在 Amazon EMR 上使用的最安全的身份验证和授权机制。Kerberos 的工作原理是，让用户提供凭证，并从身份认证服务器获取票证来证明身份。然后，该票证可用来授予对集群内资源的访问权限。为了有效使用 Kerberos 进行身份验证，Amazon EMR 集群应该在运行于 Amazon EMR 主节点之上的 Kerberos 服务器与本地 Kerberos 服务器之间创建单向信任。 这里提供了一个 EMR Kerberos 直接与 HiveServer2 进行交互的示例流程。

Amazon EMR 中的大多数应用程序都支持使用 Kerberos 作为身份验证方法。有关完整列表，请参阅《Amazon EMR 管理指南》中的支持的应用程序。

应用程序之间的身份验证

与用户相似，应用程序和服务账户在进行彼此交互时也需要进行身份验证。在 Amazon EMR 中，可以通过以下几种方式执行此类身份验证。（请参阅*图 8。*）

Kerberos

对于应用程序到应用程序的身份验证，推荐使用 Kerberos 方法。使用 Kerberos 身份验证时，应用程序通过安装在主节点上的密钥分发中心 (KDC) 对自身进行身份验证，并对与其连接的其他应用程序进行身份验证。有关 Amazon EMR 上支持 Kerberos 的工作流程的更多信息，请参阅附录 B：EMR Kerberos 工作流程。

Presto

Amazon EMR 版本的 Presto 允许集群中的节点使用 LDAPS 进行身份验证。要设置节点到节点的身份验证，请使用 LDAPS 设置 Presto。有关详细信息，请参阅《Amazon EMR 发行指南》中的在 Amazon EMR 上为 Presto 使用 SSL/TLS 和配置 LDAPS。

节点之间的 SSL/TLS

有关如何通过 Amazon EMR 安全配置在节点之间启用 SSL/TLS，请参阅传输中数据加密。使用 SSL 证书在节点之间进行身份验证可以防止恶意节点加入集群。

授权

授权（Authorization）是允许或拒绝某个身份执行某项操作的行为。要使用身份验证确定某个身份可以执行哪些操作，需先确认该身份已通过验证。本节提供了可以限制某身份可以执行的操作的各种可用机制。

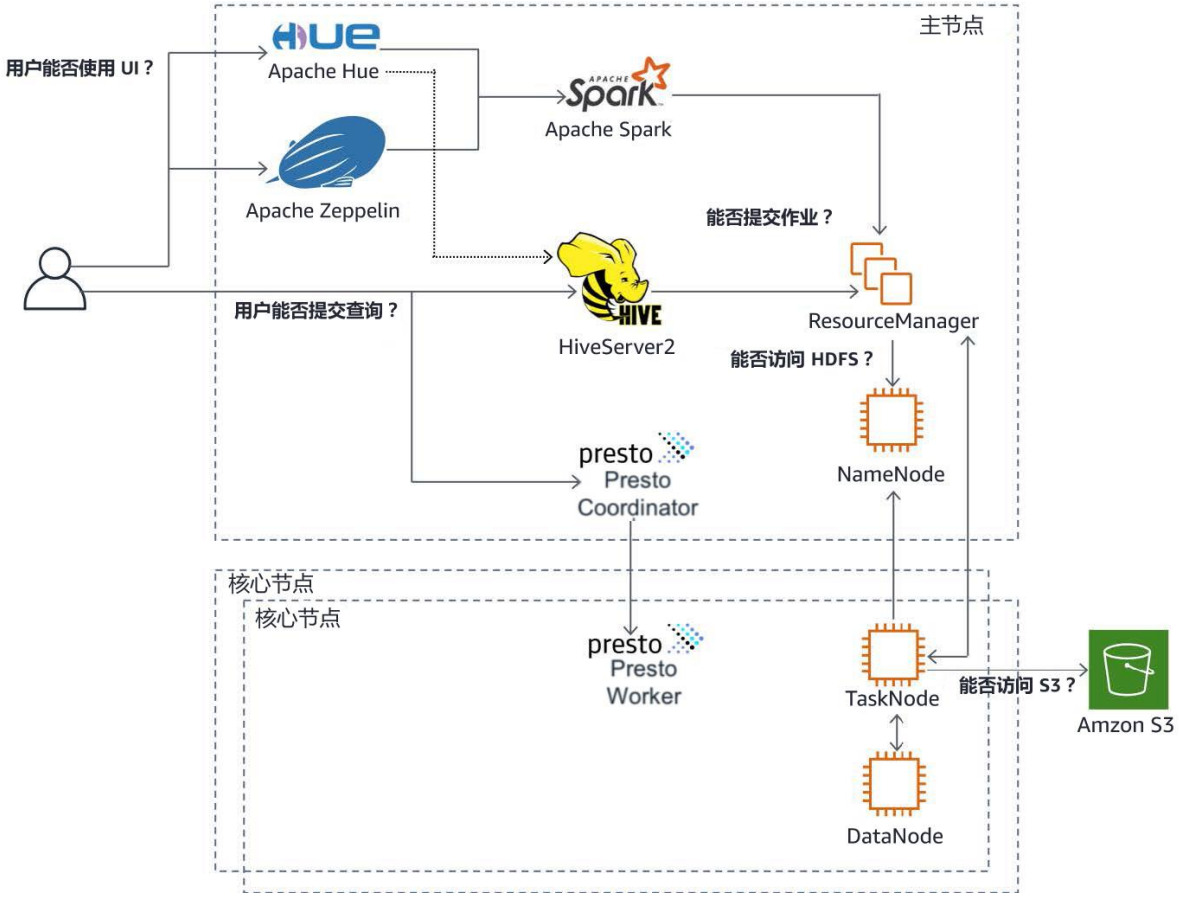


图9：授权流程示例

应用程序授权

对于 Hadoop 生态系统，有几种可能的授权解决方案，例如让各个应用程序控制用户行为，或者让中央流程或应用程序可以跨多个应用程序管理策略。有关跨 Hadoop 生态系统管理策略的应用程序，请参阅 Hadoop Authentication Authorization Data Governance Solutions Comparison Matrix。

Apache Ranger

Apache Ranger 可提供授权策略来管理各种应用程序，包括 YARN 资源队列、Apache HBase 等。

有关如何使用引导操作在 EMR 集群上启用 Apache Ranger 的信息，请参阅 Implementing Authorization and Auditing using Apache Ranger on Amazon EMR。

Apache Knox

Apache Knox 可保护您的 Hadoop 资源的边界安全。用户连接到 Knox，然后可通过 Knox Gateway 进行身份验证和授权。然后，网关将用户流量转发到 Hadoop 资源，而无需直接与其连接。可将 Knox 配置为让组指定用户可以访问的资源，例如 Apache HBase UI。

更多信息，请参阅《Apache Knox Gateway 用户指南》中的身份验证。

Hue、Zeppelin、Hive (LDAP)

能够与 LDAP 集成的应用程序（例如 Hue、Zeppelin 和 Hive），可以被设置为仅允许属于某些组的用户访问。有关实用资源，请参阅以下链接。

- 使用 LDAP 对 Hue 用户进行身份验证
- HiveServer2 身份验证/安全配置
- 配置 Zeppelin 以进行身份验证：LDAP 和 Active Directory

存储级授权 (Amazon S3)

若 EMR 集群具有多个用户，这些用户需要对 Amazon S3 中的数据进行不同级别的访问，则您可以对 EMR 集群进行拆分，或者根据发出请求的用户或组让 EMRFS 继承（assume）不同的 IAM 角色。

对 EMR 集群进行拆分，满足不同权限需求

在不同用户组之间进行权限分段的最简单方法是，创建单独的 EMR 集群，让这些集群承担不同的 Amazon EC2 角色，且拥有相应用户组所需的 S3 权限，然后对用户进行身份验证，以使用相应集群。例如，创建一个专用于 ML 用户的 EMR 集群，ML 用户可以访问相应的数据进行分析。然后，为最终业务用户创建一个单独的集群，该集群仅有权访问 ML 用户创建的分析输出。

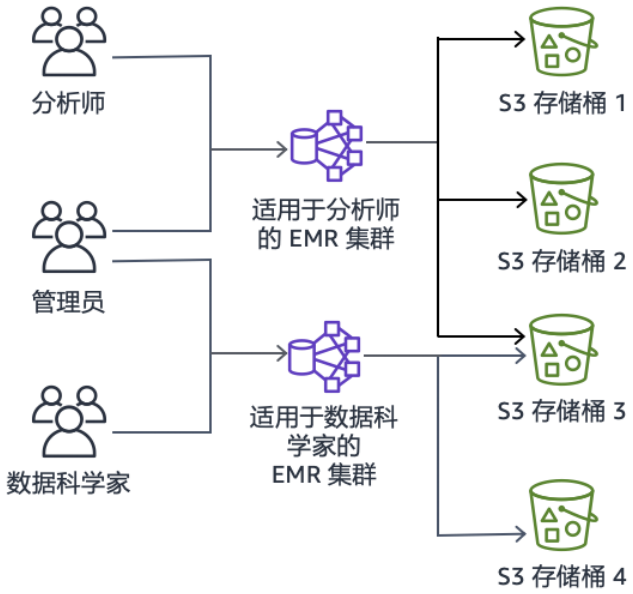


图10：分离 EMR 集群以满足不同权限需求

EMR 文件系统 (EMRFS) 授权

EMRFS 授权允许多个用户使用一个集群，同时为不同用户提供不同的 S3 权限。基于用户、该用户所属的组或 S3 位置的不同，EMRFS 可以继承（assume）不同的角色，并会根据该角色的权限与 S3 进行交互。

用户、组或 S3 位置与 EMRFS 继承的角色之间需要进行相关的角色映射配置。

例如，我们假设有两个用户 Henry 和 Martha，以及两个 AWS IAM 角色：analysts（为分析师创建）和 data_scientist（为数据科学家创建）。某个 EMRFS 映射可以指示，当 Henry 访问 S3 时，ERMFS 继承 analysts 角色，并在 Martha 的数据访问继承 data_scientist 角色。

若将 Kerberos 用于 Hadoop YARN，则需要在 Hadoop 的每个节点上注册用户账户，并且需要使用已定义的方式将用户映射到组。可以通过引导操作手动完成此映射，也可以通过某项机制从本地 Active Directory 访问用户来完成此映射。使用 SSSD 读取身份信息（包括用户到组的映射），可以讲此过程自动化。AWS 为 EMR Kerberos 自动化了设置单向信任。更多详情，请参阅使用 Kerberos 身份验证。

您可以通过在集群的所有节点上更新 emrfs-site.xml，在集群运行时更新规则。EMRFS 会检测并应用新的映射。

注意：若您的用户 user 能够访问实例的实例配置文件（Instance Profile），则他们可以继承 EMRFS 可以继承的任何角色。若用户可以通过 SSH 进入您的集群或运行任意代码（例如运行 Spark、Scala 或 SparkPy 作业），则不建议使用此方法。

AWS Lake Formation

AWS Lake Formation 是一项旨在简化和加速数据湖创建的服务。Amazon EMR 可与 Lake Formation 及其安全模型集成，支持对 Amazon S_3 进行细粒度的访问控制。用户可通过 SAML 使用任一身份提供者进行身份验证，并对主体（principal）判断它是否对 S_3 中 AWS Glue 数据目录中表内的列和分区具有相应的访问权限。

元数据授权

Hive、Presto 和 Spark SQL 都利用表元存储（也称为数据目录），它提供了关于可以查询的表的元数据信息。访问元数据可以限制用户，使其仅可访问表数据，从而间接限制他们能够在 S_3 上访问的数据。

Apache Ranger

Apache Ranger 提供了一组插件，您可以在受支持的应用程序（例如 Hive、HDFS 和 YARN）上安装和配置这些插件。可以在用户操作期间调用这些插件，并根据提供给 Ranger 服务器的策略来允许或拒绝操作。 Ranger 可以在 Hive 中的数据库和表上设置策略，以允许用户和组访问 Hive 表和数据库的子集。Ranger 还可以执行列级授权、列掩码和行级过滤。这些策略可在 Ranger Web 控制台上进行管理，且可与 LDAP 服务器交互。

有关如何在 Amazon EMR 上实现此功能的详情，请访问 AWS 大数据博客，参阅 Implementing Authorization and Auditing using Apache Ranger on Amazon EMR。

Hive SQL 授权

Hive SQL 授权可以提供另一种机制，用以控制有权访问表和数据库的用户。更多信息，请参阅 Apache Hive Wiki 上基于 SQL Standard 的 Hive 授权。

AWS 数据目录授权

通过 AWS Glue 数据目录，您可以根据 IAM 委托人 IAM 策略限制对数据库或表的访问。 更多详情，请参阅 AWS Glue 的身份验证和访问控制。

加密

您可以实施数据加密，以帮助保护 Amazon S3 和集群实例存储中的静态数据和传输中的数据。您还可以使用自定义的 Amazon Linux AMI 来加密集群实例的 EBS 根设备卷。

传输中数据加密

Amazon EMR 安全配置让您可以选择一种方法，使用传输层安全 (TLS) 对传输中的数据进行加密。您可以执行以下任意一项操作：

- 手动创建 PEM 证书，将其压缩到一个文件中，然后从 Amazon S3 引用它们。
- 在 Java 中实施证书自定义提供程序，并指定 JAR 的 S3 路径。

有关如何使用 EMR 安全配置来配置传输中加密，以及支持哪些应用程序的详情，请参阅《Amazon EMR 管理指南》中的加密传输中数据和静态数据。 我们强烈建议您加密与集群的所有交互，包括所有基于 UI 的应用程序和终端节点。

静态数据加密

您可以通过多种方案来加密 EMR 集群中的静态数据。默认情况下，EMR 使用 EMR 文件系统 (EMRFS) 从 Amazon S3 读取数据并将数据写入 Amazon S3。要在 Amazon S3 中加密数据，您可以使用以下任一方案：

- **SSE-S3：** Amazon S3 为您管理加密密钥。
- **SSE-KMS：** 您使用 AWS Key Management Service (AWS KMS) 客户主密钥 (CMK) 对 Amazon S3 上的服务器端数据进行加密。确保使用策略支持 Amazon EMR 访问 KMS。
- **CSE-KMS/CSE-C:** 在 Amazon EMR 集群的客户端进行 Amazon S3 加密和解密。您可以使用 AWS KMS (CSE-KMS) 提供的密钥，也可以使用能够提供主密钥 (CSE-C) 的自定义 Java 类。

有关如何使用 EMR 安全配置的详情，请参阅《Amazon EMR 管理指南》中的加密传输中数据和静态数据。

根卷

要加密根卷，必须使用自定义 AMI。有关详情，请参阅《Amazon EMR 管理指南》中的使用加密的 Amazon EBS 根设备卷创建自定义 AMI。

EBS 卷

您可以通过两种机制对非根卷（通常用于存储 HDFS 及其他应用程序数据）上的数据进行加密。这两种机制分别是 HDFS 透明加密（适用于应用程序管理的加密）和 LUKS 加密（适用于操作系统管理的加密）。有关这两种方法的文档，请参阅《Amazon EMR 管理指南》中的适用于本地磁盘的静态加密。

尽管两种机制都可以启用，但是对于 EMR，建议仅启用 LUKS 加密（若可能）。这种方法在加密静态数据的同时，可以最大限度地减少加密开销。

边界安全

确保边界安全是实现 Amazon EMR 安全的基础。此流程包括限制对 EMR 集群本身的访问，并使用“边缘”节点（网关）作为主机。用户和系统可以直接使用此边缘节点，然后将所有其他操作从此网关发送到集群本身。

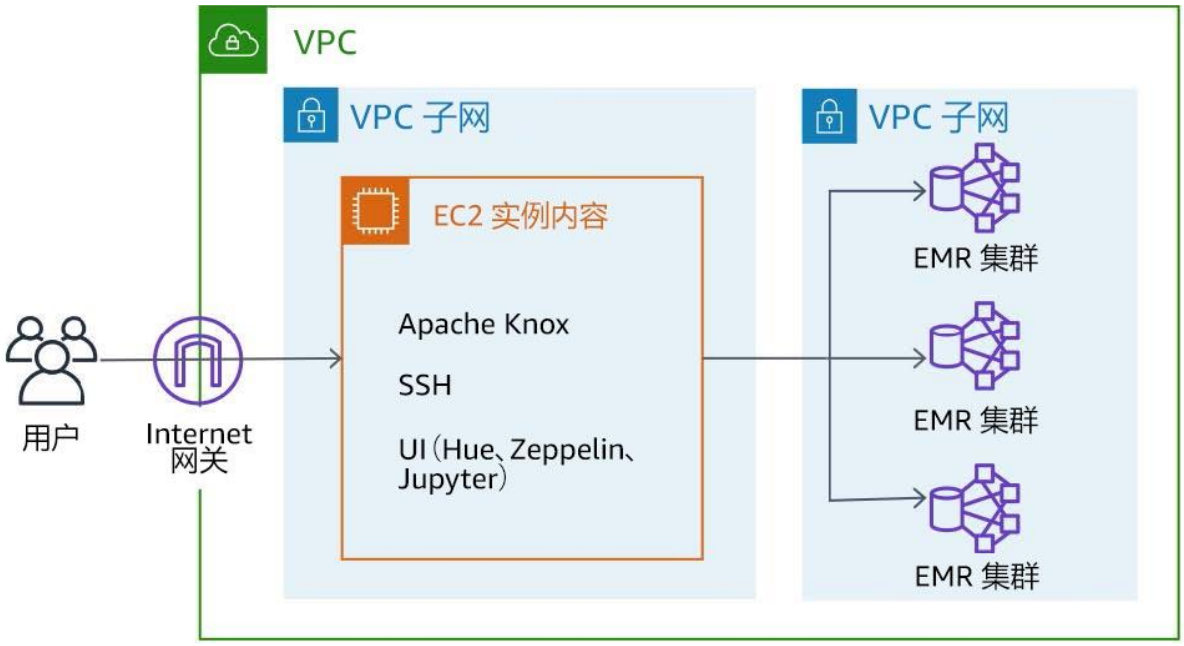


图 11：边界安全

Apache Knox

Apache Knox 是一种应用程序网关，它提供了一个与多个应用程序和多个集群进行交互的终端节点，并为支持 SAML、LDAP 和其他身份验证方法的应用程序和集群提供了一个身份验证层。Knox 可用于以下几种使用案例：

- 若客户端和 Apache Knox 之间需要 HTTPS：Apache Knox 支持 TLS，因此无论集群上的应用程序是否支持 TLS，客户端都可以通过安全通道连接到 Knox。
- 若需要身份验证层：Apache Knox 提供了一个身份验证层，支持多种类型的身份验证机制，这些机制可用于不受集群应用程序支持的实例。例如，Knox 支持可用作 Hadoop 资源管理器的单点登录 (SSO)，但 Hadoop Resource Manager 资源管理器不支持单点登录。
- Apache Knox 记录每个用户请求执行的操作，或由 Knox 内部事件产生的操作。

但是，Apache Knox 确实存在一些限制，而且它不提供细粒度的访问控制。

网络安全

对于如何正确设置 VPC、子网、安全组和网络 ACL，本文档未提供相关信息。有关设置 VPC 的信息，请参阅使用安全组控制网络流量。

网络安全的最佳实践

- 若您的集群用于批处理，并且用户或其他应用程序不需要访问它，请通过 S3 终端节点和任何其他必需的终端节点在私有子网中启动集群。更多信息，请参阅配置网络。
- 若您的集群位于私有子网中，且需要访问没有 VPC 终端节点的 AWS 服务或需要访问 Internet，请创建一个 NAT 实例。更多信息，请参阅场景 2：带有公有子网和私有子网 (NAT) 的 VPC。
- 若您使用私有子网并创建 S3 终端节点，请对终端节点使用限制性 S3 策略。Amazon EMR 2.x 和 3.x AMI 版本中列出了最低策略要求。

- 若要求集群位于私有子网中，但需要以交互方式访问该集群，则创建一个位于公有子网中的堡垒主机（也称为边缘节点），然后从该主机连接到 EMR 集群。此设置可让您在边缘节点上强制实施安全策略，但在 EMR 集群上不实施这些策略。有关分步指南，请访问 AWS 大数据博客，参阅 Securely Access Web Interfaces on Amazon EMR Launched in a Private Subnet。
- 若您能放弃对非交互式集群的 SSH 访问，则可以简化 EMR 集群上的安全要求。

审计

许多客户都有这样一个关键要求：能够审计计算环境，并了解集群中的数据来自何处，以及现正如何使用这些数据。在 EMR 中，可通过各种方式支持此要求。

Amazon S3

通过启用日志记录，可将 Amazon EMR 配置为向 Amazon S3 推送所有应用程序日志、守护程序日志和预置日志。更多信息，请参阅配置集群日志记录和调试。

Apache Ranger

Apache Ranger 插件提供向 Apache Solr 推送审计数据以便将来进行分析的功能。Apache Ranger 还会在 Ranger UI 中推送审计信息。

AWS CloudTrail

AWS CloudTrail 记录对 AWS 服务进行的每次 API 调用，并包含多种信息，例如调用者 AWS 访问密钥、源 IP、用来执行调用的代理等。使用 Amazon EMR 时，有多组 API 调用会存储起来。第一组处于 EMR 服务级别，其中的 API 被调用来进行集群管理，例如创建集群、终止集群和运行步骤。其次，当正在运行的集群尝试访问 S3、Glue 数据目录和 KMS 等资源时，该集群使用的凭证会存储在 AWS CloudTrail 中。有关如何启用和使用 CloudTrail 的更多详情，请参阅在 AWS CloudTrail 中记录 Amazon EMR API 调用。

为帮助满足合规性要求，可将 CloudTrail 日志导出到 Amazon S3 中，然后通过 Amazon Athena 查询这些日志，从而能够检测未经授权或可疑的活动。有关如何通过 Amazon Athena 进行查询的更多信息，请参阅查询 AWS CloudTrail 日志。

Amazon CloudWatch Logs

您可使用 Amazon CloudWatch Logs 代理向 CloudWatch Logs 推送日志，而不仅是使用 EMR 日志推送器向 S3 推送日志。此方法的好处是使这些日志变得可以检索，并能将它们流式传输到其他系统（例如 Amazon Elasticsearch Service）或使用 AWS Lambda 向其他系统（例如 Splunk）推送日志。

要使用此方法，必须安装并配置 Amazon CloudWatch 代理，将其配置为监视 Hadoop 日志目录。之后，该代理开始向 Amazon CloudWatch Logs 推送日志，在该服务中可以根据审计要求设置日志保留和导出目标。

Apache Knox

若您使用 Apache Knox 作为边界安全或代理的网关，还可以将它配置为记录用户在它与之交互的所有资源上执行的所有操作。

软件补丁

Amazon EMR 定期提供新版本，增加新功能、新应用程序和常规更新。我们建议尽可能使用最新版本来启动集群。

操作系统级别的补丁

在 EC2 实例启动时，它会应用操作系统的所有非内核补丁。要查看已安装的补丁，可在 AWS 命令行界面 (AWS CLI) 中运行 `yum updateinfo list security installed`

要安装操作系统级别的其他补丁，我们建议采用以下三种方法：

- 升级到较新的 AMI – 最新的补丁（包括 Linux 内核补丁）将应用到较新的 AMI。查看我们的发行说明，了解哪些版本具有已升级的应用程序，并查看应用程序发行说明，了解是否有与您的问题相关的解决方案。
- 若您需要更灵活地控制所应用的补丁，则使用自定义 AMI。利用此方法，您可以将新补丁应用到 AMI，然后通过重启集群部署这些补丁。
- 与 AWS Support 联系，说明您的问题和所需的补丁。支持人员可能提供解决方法。

若有长时间运行的集群，则可以使用 EC2 Patch Manager。此方法需要您的实例启动并运行 SSM。EMR 集群修补需要您在维护时段将补丁应用到集群，以便能批量重启节点（特别是主节点）。也可以将补丁一次应用到一个节点，以确保不会中断 HDFS 和作业。应用任何补丁前，务必与 AWS Support 联系。若在修补过程中必须重启实例，请务必在修补前启用终止保护。有关详情，请参阅使用终止保护。

应用程序级别的补丁

Amazon EMR 中提供了几种应用程序补丁选择：

- 升级到较新的 AMI – 较新的 AMI 包含最近的补丁。查看发行说明，了解哪些版本具有已升级的应用程序，并查看应用程序发行说明，了解是否有与您的问题相关的解决方案。
- 与 AWS Support 联系，说明您的问题和所需的补丁。根据严重程度，支持人员可能会提供其他解决方法，或者，在极端情况下或可创建将提供补丁的引导操作。

注意：除非您已联系 AWS Support 和 EMR 服务团队，否则我们建议不要在集群上应用和安装补丁。在某些软件升级方案中，您可能需要承担维护补丁带来的运营负担。

软件升级

下面是有关升级 Amazon EMR AMI 的一般建议：

- 阅读 Amazon EMR 发行说明，了解 Hive、HBase 等应用程序的版本变化。
- 阅读 Hive、HBase 等开源软件包的发行说明，并查看可能影响现有工作流程的任何已知问题。
- 使用测试环境来测试升级后的软件。对于将测试工作负载与生产工作负载分隔开，此方法很重要。例如，Hive 的某些版本升级会更新 Hive 元存储架构，这可能会造成兼容性问题。若有人启动 Hive 并将其与较新的版本一起使用，则可能会影响生产。
- 在新旧版本之间测试性能和数据质量。在 Hive 中迁移版本时，可能会出现兼容性问题或错误，从而可能影响数据质量和性能。

常见客户使用案例

私有子网中的批处理 EMR 集群

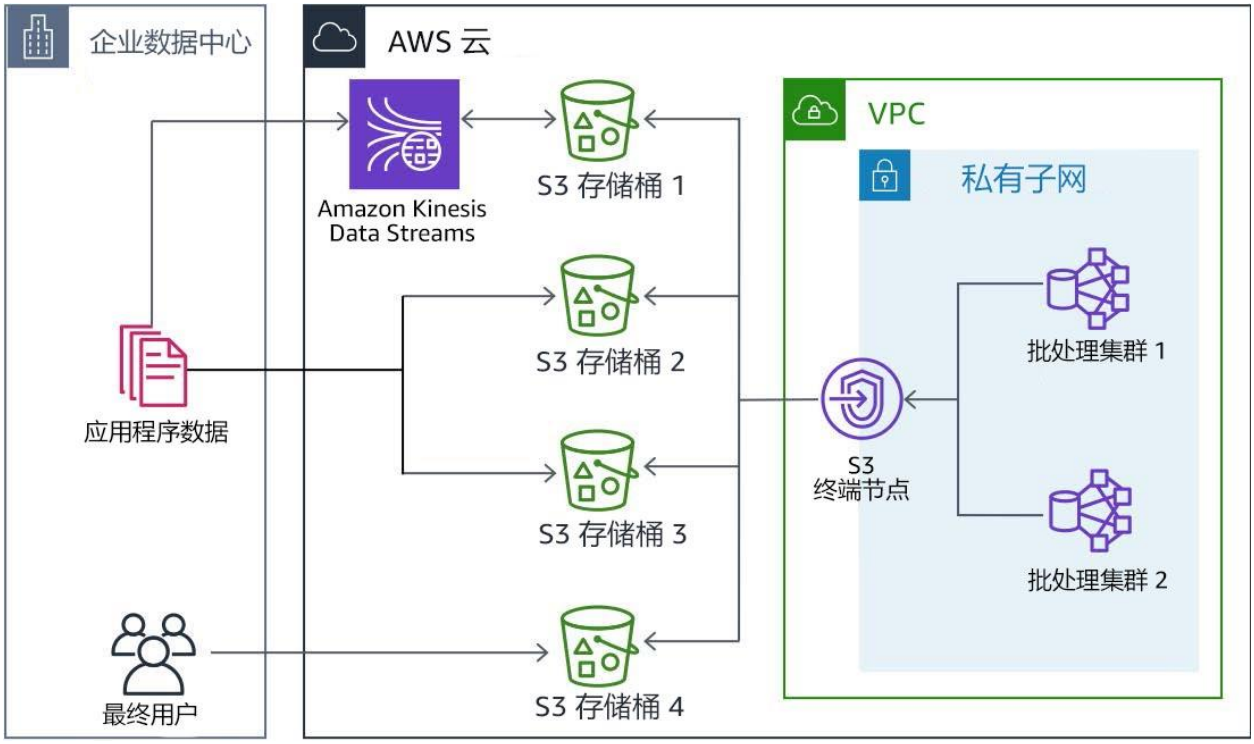


图 12：私有子网中的批处理 EMR 集群

使用案例

- 受保护的 EMR 集群以 ETL 方式对数据进行批处理。
- 用户无需直接访问集群。

何时使用

- 与用户在其中进行交互式访问的集群相比，仅限于 ETL 的使用案例可以降低安全要求。
- 此设计在将 Amazon EMR 集群用作临时性集群时很有效。

实施

1. 创建 VPC 和带有一个 S3 终端节点的私有子网。

2. 禁止对 EMR 集群进行 SSH 访问。
3. 若需要额外的安全性，请通过安全配置完成以下步骤：
   a. 配置无单向信任的 Kerberos，以实现应用程序到应用程序的安全性。
   b. 启用对传输中数据和静态数据加密的功能。
4. 实施 S3 存储桶策略，以便对来自 VPC 的请求进行只读访问，以及对来自最终用户和写入 S3 的应用程序的请求进行写入访问。
5. 使用 IAM 策略来保护 EMR API 的调用，仅将访问权限授予需要访问集群的主体。
6. 通过以下方式之一将工作提交到 EMR 集群：
   - 使用 EMR Steps API 提交工作。
   - 在主节点上安装 Oozie、Airflow 或类似的应用程序，并让其协调处理工作负载。

使用堡垒主机/边缘节点

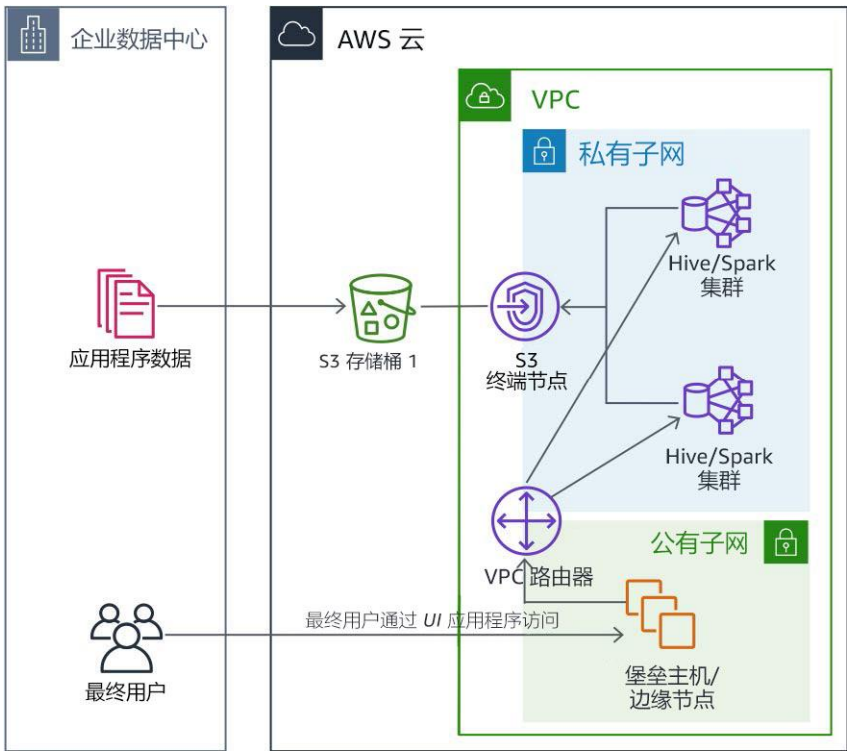


图13：使用堡垒主机/边缘节点

使用案例

- 无需保护 EMR 集群，原因是它在访问严格受控的私有子网中运行。
- 您可以控制您想在堡垒主机上使用的 AMI。例如，若用户只能访问内部管理的 Red Hat Linux，则堡垒主机可以运行该 AMI，同时 EMR 集群继续使用 Amazon Linux。
- 在堡垒主机上对用户进行身份验证后，用户应能访问 EMR 集群上的所有资源。

何时使用

- 您希望控制对 EMR 集群的访问，并让所有用户通过堡垒主机/边缘节点与 EMR 进行交互。
- 您希望将在主节点和数据节点上运行的作业提交进程与 EMR 进程分隔开。

实施

1. 在带有 VPC 终端节点的私有子网中创建 EMR 集群，这些终端节点允许流量流向所需的 AWS 服务（例如 Amazon S3、AWS KMS、Amazon DynamoDB）。
2. 在公有子网中的 EC2 实例上创建堡垒主机。来自公有子网的流量必须能够访问私有子网中的集群。
3. 禁止对 EMR 集群进行 SSH 访问。
4. 通过 Hue、Zeppelin 或 Jupyter 等 UI 来访问这些集群，或者提供对堡垒主机/边缘节点的 SSH 访问。
5. 若允许 SSH 访问，则确保堡垒主机/边缘节点使用 SSSD 或其他机制来允许用户使用其本地数据中心凭证进行登录。
6. 使用 Elastic Load Balancer (ELB) 将流量路由到集群，以实现负载均衡和灾难恢复，需要时在堡垒主机上更改配置。

用于不同使用案例的独立 EMR 集群

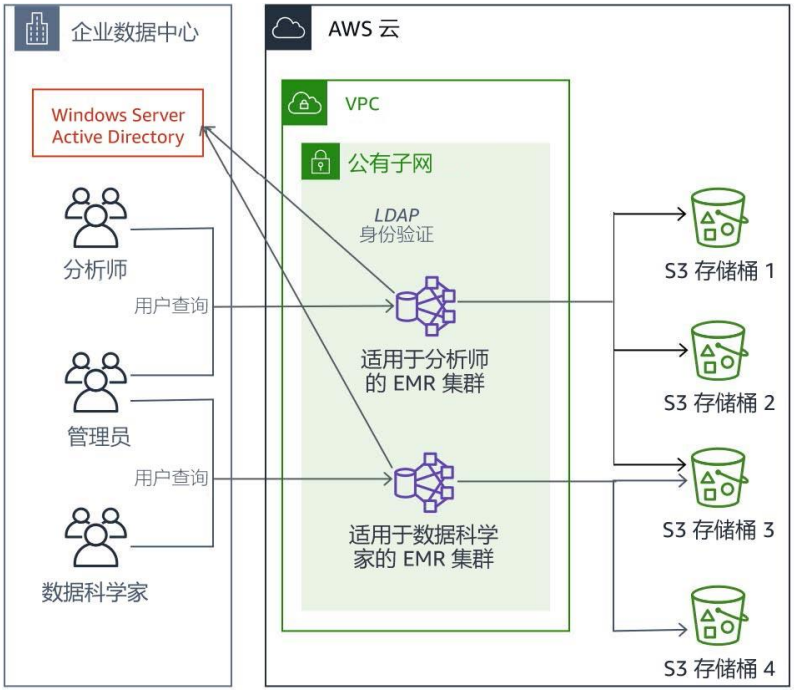


图14：独立EMR集群

使用案例

- 有少量的最终用户组，它们全都共享对相同数据的相同权限级别。
- 所访问的数据不是高度敏感的数据。

实施

1. 为代表您希望最终用户获得的权限的每个 EMR 集群创建不同的 EC2 实例角色。
2. 允许对最终用户使用的应用程序（例如 Hue 和 Zeppelin）进行 LDAP 身份验证。您不希望 Hue 和 Zeppelin 模拟终端用户。若需要模拟，您必须手动创建用户和组，或者在所有节点上设置 SSSD 并转到您的本地 LDAP。
3. 禁止对集群进行 SSH 访问。

数据迁移

使用 Amazon S3 作为中央数据存储库

AWS 的许多客户都将 Amazon S3 用作中央存储库（更常见的称呼是数据湖），以便安全地大规模收集、存储和分析数据。[2] 拥有数据仓库的组织正认识到数据湖的好处，并利用数据湖来实现不同的查询功能、数据科学使用案例和用于发现新信息模型的高级功能。为了在 AWS 上构建数据湖，客户通常将 Amazon S3 而不是 Hadoop 分布式文件系统 (HDFS) 用作中央存储库。

通过构建 Amazon S3，可以存储和检索任意数量的数据，并能提供无与伦比的可用性和 99.999999999%（11 个 9）的持久性。Amazon S3 提供全面的安全性与合规性能力，可满足最严苛的监管要求。

下图显示了数据湖架构模式的概览，在其中 Amazon S3 用作中央存储库。

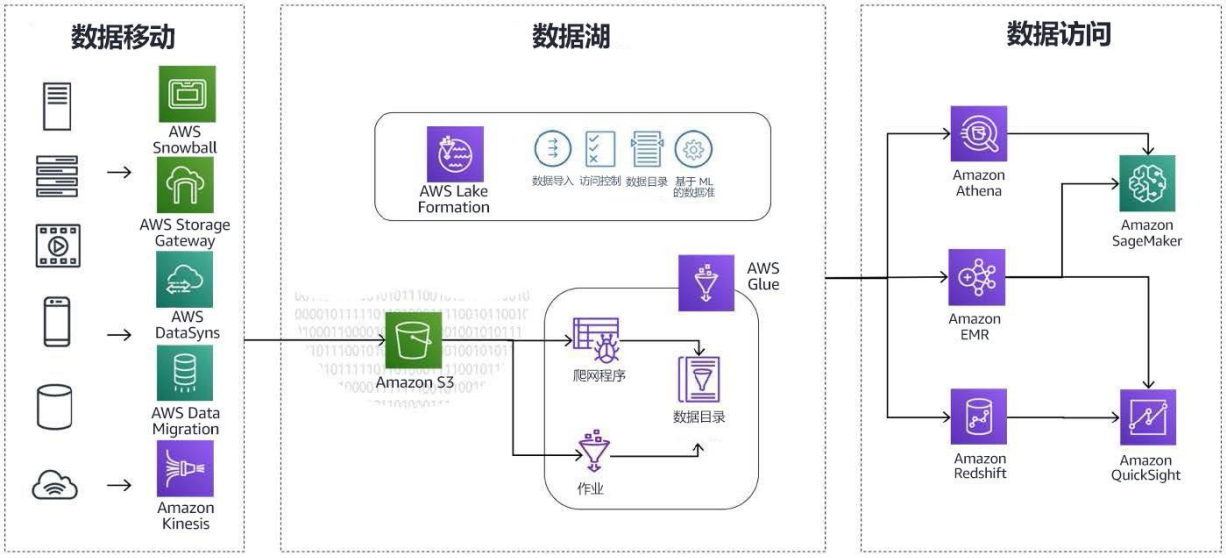


图 15：采用 Amazon S3 的数据湖架构

使用 Amazon S3 的益处

将 Amazon S3 用作存储平台的一些重要益处包括：

- **将存储与计算和数据处理解耦** – 在传统的 Apache Hadoop 和数据仓库解决方案中，存储和计算资源紧密耦合，导致难以优化成本和数据处理工作流程。通过将数据存储在 Amazon S3 中，可以使用 Amazon Elastic Compute Cloud (EC2) 启动所需数量的计算容量，并且可以使用 AWS 分析服务来处理数据。可以优化 EC2 实例，以便提供恰当比例的 CPU、内存和带宽（以实现最佳性能），甚至利用 Amazon EC2 Spot 实例。
- **集中式数据架构** – Amazon S3 可帮助轻松构建多租户环境，其中，许多用户可以使用自己的数据分析工具来分析一组公共数据。传统解决方案需要将多个数据副本分布在多个处理平台上，与之相比，此方法提供更低的成本和更好的数据监管。
- **与 AWS 服务集成** – 将 Amazon S3 与 Amazon Athena、Amazon Redshift Spectrum、Amazon Rekognition 和 AWS Glue 结合使用，以查询和处理数据。Amazon S3 还与 AWS Lambda 无服务器计算集成，无需预置或管理服务器即可运行代码。借助所有这些功能，您只需为处理的实际数据量和耗用的计算时间付费。
- **标准化的 API** – Amazon S3 RESTful API 简单易用，并获得大多数主要第三方独立软件供应商 (ISV) 的支持，其中包括领先的 Apache Hadoop 和分析工具供应商。这可让客户使用他们最习惯和最熟悉的工具，来帮助他们对 Amazon S3 中的数据执行分析。

从本地迁移数据

在 AWS 上构建数据湖的第一步是将数据迁移到云。将数据迁移到 Amazon S3 时，由于带宽和传输速度的物理限制，可能会限制数据迁移能力并导致重大业务中断、高昂成本和较长用时。为帮助客户将数据从其数据中心迁移到 AWS，AWS 提供了多种选择，包括：

- 使用 AWS Snowball 和 AWS Snowmobile 设备将 PB 级到 EB 级的数据迁移到 AWS。
- 使用 AWS Direct Connect 在您的数据中心与 AWS 之间的专用网络连接上迁移大量数据。
- 使用数据库迁移服务 DMS 从关系数据库以及使用 Amazon Kinesis 从网站和移动应用程序等互联网来源实时迁移数据。

一次性传输 PB 级到 EB 级的数据

AWS Snowball 和 AWS Snowmobile

AWS Snowball 是一种 PB 级数据传输解决方案，它使用安全设备将大量数据传入和传出 AWS。通过 Snowball 客户端（安装在托管您要传输的数据的物理工作站上）将数据传输到 AWS Snowball。有关最佳实践，请参阅《AWS Snowball 用户指南》中的如何高效传输 PB 级数据。

使用 Hadoop 分布式文件系统 (HDFS) 的客户还可以使用 Snowball 客户端将数据直接从 HDFS 复制到 AWS Snowball。要将数据从 HDFS 复制到 Snowball，您必须在可以访问 HDFS 集群的本地主机上安装 Snowball 客户端，然后通过 Snowball 将文件从 HDFS 复制到 S_3。此外，适用于 Snowball 的 S_3 SDK Adapter 向 Snowball 客户端提供了兼容 S_3 的接口，用于在 Snowball 上读取和写入数据。 喜欢更紧密集成的客户可以使用 S_3 Adapter 轻松扩展现有应用程序和工作流程，以无缝地与 Snowball 集成。

AWS Snowmobile 是一种用于将大量数据迁移到 AWS 的 EB 级数据传输服务。对于存放在单个位置的大型数据集（10PB 或更大），我们建议使用 AWS Snowmobile 来迁移。对于小于 10PB 或分布在多个位置的数据集，请使用 AWS Snowball。请务必评估主干网络中的可用带宽量。若您有高速主干网络，而且其备用吞吐量达到数百 Gb/s，则可以使用 Snowmobile 同时迁移所有大型数据集。若主干网络的带宽有限，则应考虑使用多个 Snowball 来逐步迁移数据。

> **注意：**一般而言，若使用现有互联网连接的备用容量将数据上传到 AWS 所需的时间超过一周，则应考虑使用 Snowball。例如，若您有可以单独用于传输数据的 100Mb 连接，并且需要传输 100TB 数据，则在该连接上完成数据传输将需要超过 100 天的时间。另一方面，通过使用多个 Snowball，可以在大约一周内传输相同数量的数据。

有关何时使用 Snowball 来传输数据的更多信息，请参阅 AWS Snowball 常见问题。

持续传输大量数据

AWS Direct Connect 和 VPN 连接

AWS Direct Connect 提供专用于从本地向 AWS 网络传输数据的快速连接。若需要持续向 AWS 传输大量数据，则选择 AWS Direct Connect 可能很合适。 利用 AWS Direct Connect，您可以在 AWS 网络与某个 AWS Direct Connect 位置之间建立 1 Gbps 或 10 Gbps 的专用网络连接（或多个连接）。它使用业界标准的 VLAN，可让您使用私有 IP 地址访问 Amazon VPC 中运行的资源。

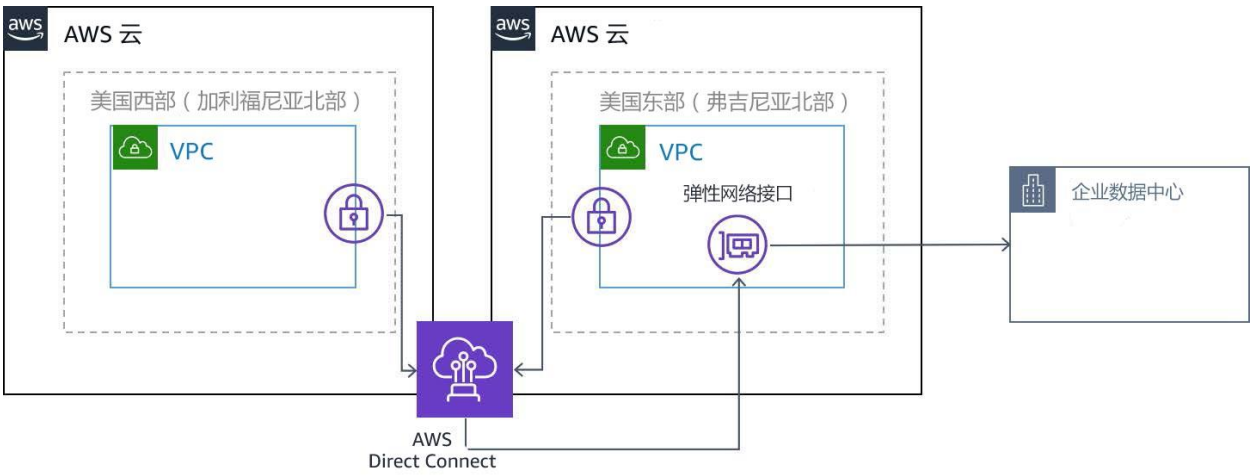


图16：AWS Direct Connect

除了使用 AWS Direct Connect 之外，您还可以通过创建名为 VPN 连接的加密连接来启用远程网络与 VPC 之间的通信。使用 Internet 协议安全性 (IPsec) 在互联网上创建 VPN 连接。可以按照以下方式创建 VPN 连接：将虚拟私有网关连接到 VPC，创建自定义路由表，更新安全组规则，然后创建 AWS 托管 VPN 连接。

> **注意：**我们建议使用 AWS Direct Connect 来满足巨大的持续数据传输需求，原因是 AWS Direct Connect 可以减少成本、增加带宽并提供比基于 Internet 的 VPN 连接更加稳定的网络体验。 若您有迫切的需求，对带宽的要求较低或不太高，并且可以忍受基于 Internet 的连接固有的变化性，则 VPN 连接是很好的解决方案。

在您使用以上任一方法于本地环境之间建立的连接中，可以使用以下任何方法利用 AWS Direct Connect 轻松地持续将数据迁移到 Amazon S3 中。

从 Hadoop 访问 Amazon S3

Apache Hadoop 具有适用于 Amazon S3 的连接器（称为 S3N 或 S3A），该连接器可与 DistCp 工具一起用来从 HDFS 迁移数据。 数据传输命令通常类似以下所示：

```
hadoop distcp hdfs://source-folder s3a://destination-bucket
```

许多时候，迁移数据的原因是本地集群缺乏计算容量。在此类情况下，客户利用 Amazon EMR 提供的 S3DistCp 工具将数据从 HDFS 提取到 Amazon S3 中。有关此情况下最佳实践的更多信息，请参阅 AWS 大数据博客文章 Seven Tips for Using S3DistCp on Amazon EMR to Move Data Efficiently Between HDFS and Amazon S3。可以利用 WANDisco Fusion 和 Cloudera BDR 等市售解决方案将数据从 HDFS 迁移到 Amazon S3。

还可以利用 Apache Hadoop 和 Amazon EMR 与 Amazon S3 的集成，并让数据处理工作流程直接写入 Amazon S3。例如，可以在 Amazon EMR 集群上运行 Apache Sqoop 作业，以便从关系数据库中提取数据，然后将数据写入 Amazon S3。

AWS Glue

AWS Glue 是一种完全托管的提取、转换和加载 (ETL) 服务，可让您更轻松地准备和加载要分析的数据。AWS Glue 还可以发现数据，并将关联的元数据（例如表定义和结构）存储在 AWS Glue 数据目录中。AWS Glue 旨在简化与迁移和转换要分析的数据集相关的任务。它是无服务器的完全托管服务，建立在流行的 Apache Spark 执行框架基础之上。

AWS Glue 可通过 Java 数据库连接 (JDBC) 访问本地关系数据库，以爬取数据存储并将其元数据编目到 AWS Glue 数据目录中。任何将数据存储用作源或目标的 ETL 作业（例如将数据写回 Amazon S3）也可使用该连接。下图显示了这一工作流程：从关系数据库提取数据，转换数据，然后将结果存储在 Amazon S3 中。

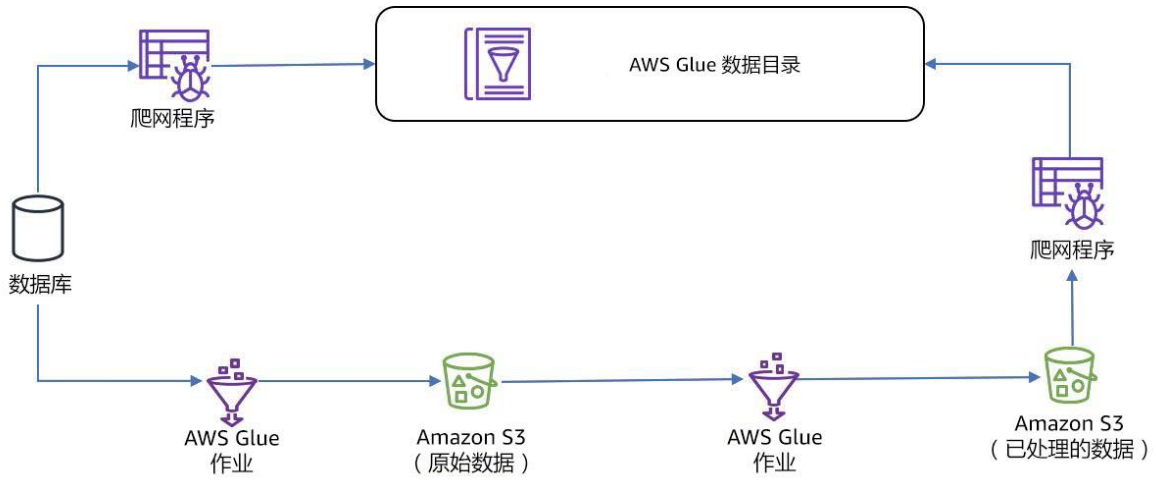


图17：AWS Glue 数据目录

有关更多信息，请参阅 AWS 大数据博客文章 How to extract, transform, and load data for analytic processing using AWS Glue。

AWS DataSync

AWS DataSync 是一种托管的在线数据传输服务，它简化、自动执行和加速执行此过程：通过 Internet 或 AWS Direct Connect，在本地存储系统与 AWS 存储服务（例如 Amazon S3 和 Amazon EFS）之间迁移和复制大量数据。DataSync 会自动扩展和处理所有涉及以下操作的任务：迁移数据、监控传输进度、加密和验证数据传输，以及将任何失败情况通知客户。DataSync 使用标准存储协议 (NFS) 连接到现有存储系统和数据源，并使用专用的网络协议和横向扩展的架构加快将数据传入和传出 AWS 的过程。

客户可以使用 HDFS NFS Gateway（随 Apache Hadoop 提供）和 AWS DataSync 将本地 Hadoop 集群中的数据传入 Amazon S3。

AWS Storage Gateway

AWS Storage Gateway 可让您将旧版本地应用程序与 Amazon S3 集成。它用于对 Amazon S3 中存储的对象进行备份、分层和本地访问。Storage Gateway 的文件网关配置可让本地设备和应用程序通过 NFS 连接，对 Amazon S3 中的数据进行低延迟的访问。这意味着您可以轻松集成没有原生 Amazon S3 功能的应用程序和平台（例如本地实验室设备、大型机、数据库和数据仓库），以直

接将它们的文件写入 Amazon S3。 写入此挂载点的文件将转换为以其原始格式（无任何专有的修改）存储在 Amazon S3 中的对象。

> **注意：**我们建议使用 AWS DataSync 快速将数据从 Hadoop 集群复制到 Amazon S3 中。 可以使用 DataSync 将现有数据快速传输到 Amazon S3，之后使用 Storage Gateway 的文件网关型配置，从本地应用程序对这些数据进行低延迟的访问。

连续传输事件和流数据

AWS Database Migration Service

AWS Database Migration Service (AWS DMS) 可帮助您轻松安全地将数据库迁移到 AWS。AWS DMS 使用变更数据捕获 (CDC) 执行连续的数据复制。大部分数据库管理系统都管理着一个事务日志，该日志记录了对数据库内容和元数据所做的变更。通过使用特定于引擎的 API 操作和函数，AWS DMS 可读取事务日志，并以非侵入方式捕获对数据库所做的变更。

您可以使用 AWS DMS 将任何支持的数据库源中的数据迁移到 Amazon S3。将 Amazon S3 用作 AWS DMS 任务中的目标时，全量导入和变更数据捕获 (CDC) 数据均写入到逗号分隔值 (CSV) 格式文件中。有关如何查询这些数据的更多信息，请参阅 AWS 数据库博客文章 Using AWS Database Migration Service and Amazon Athena to Replicate and Run Ad Hoc Queries on a SQL Server Database。

> **注意：**若使用案例需要将数据库从本地迁移到 AWS，或需要在本地源与 AWS 上的源之间复制数据库，我们建议使用 AWS DMS。数据位于 AWS 中后，即可使用 AWS Glue 迁移和转换数据。

Amazon Kinesis Data Firehose

Amazon Kinesis Data Firehose 是一种完全托管的服务，用于将实时的流数据直接传输到 Amazon S3。Amazon Kinesis Data Firehose 会自动扩展，以匹配流数据的数量和吞吐量，且不需要持续管理。Amazon Kinesis Data Firehose 可以将多个传入的记录合并起来，然后将它们作为单个 S3 对象传输到 Amazon S3。这是一项重要功能，因为它减少了 Amazon S3 的事务处理成本和每秒处理的事务负载。

Amazon Kinesis Data Firehose 还可以配置先执行转换流数据任务，再将流数据存储在 Amazon S3 中。它的转换功能包含压缩、加密、数据批处理和 Lambda 函数。Amazon Kinesis Data Firehose 可以先压缩数据，然后再将数据存储在 Amazon S3 中。它目前支持 GZIP、ZIP 和 SNAPPY 压缩格式。有关示例工作流程，请参阅下图。

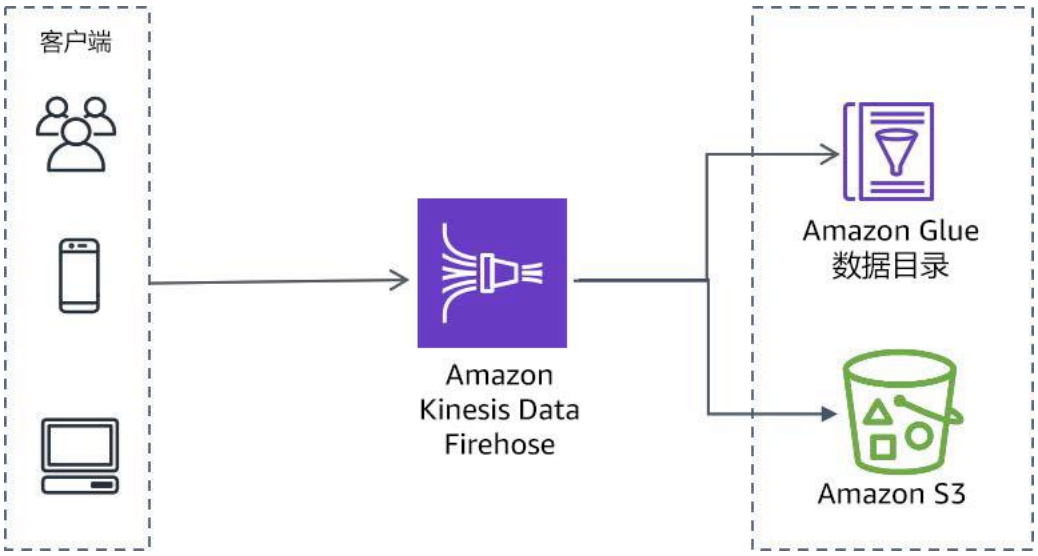


图18：Amazon Kinesis Data Firehose 架构

在图 18 中，代理将源中的数据写入 Amazon Kinesis Data Firehose。此类代理的一个示例为 Amazon Kinesis 代理，它可将一组文件中的数据写入 Amazon Kinesis 数据流。AWS 数据库博客文章 Streaming Changes in a Database with Amazon Kinesis 提供了此类代理的另一个示例。将事件流式传输到 Amazon Kinesis Data Firehose 数据流中之后，Kinesis Data Firehose 会根据配置将数据传输到 Amazon S3 存储桶。还可以配置 Kinesis Data Firehose，让其使用 AWS Glue 数据目录中的表结构将数据传输到 Apache Parquet 格式文件中。

优化基于 Amazon S3 的中央数据存储库

通过将 Amazon S3 用作数据湖的中央存储库，您已将存储和计算解耦，并且根据实际需求和使用情况支付计算费用，从而优化了总体拥有成本 (TCO)。AWS 和 Amazon S3 包含多项功能，可以优化数据湖中的存储占用空间，从而进一步降低成本。通过使用列式文件格式，不仅能加快查询速度，还能降低成本。在本节中，我们将介绍一些此类优化措施，您可以用来优化基于 Amazon S3 的中央数据存储库。

优化存储成本

Amazon S3 提供一系列为频繁访问和不频繁访问的数据设计的存储类。此外，客户还可以使用 Amazon S3 Glacier 来保存长期存档。Amazon S3 提供可配置的生命周期策略，帮助您在数据的整个生命周期中管理数据。下一节介绍 AWS 中提供的优化选项，它们通过跨存储类和 Amazon S3 Glacier 进行数据分层来优化存储成本。

Amazon S3 存储类

Amazon S3 标准（S3 标准）为频繁访问的数据提供高持久性、高可用性和高性能的对象存储。由于 S3 标准提供低延迟和高吞吐量，因此它很适合存储频繁访问的数据（例如大数据分析）。

Amazon S3 智能分层（S3 智能分层）提供与 S3 标准相同的低延迟和高吞吐量性能，但可让您通过自动在两层之间移动数据来进一步优化存储成本 – 一层为频繁的访问进行优化，而成本较低的另一层为不频繁的访问进行优化。支付少量的每对象月度监控和自动化费用后，Amazon S3 会监控 S3 智能分层中的对象的访问模式，并会将连续 30 天未被访问的对象移到不频繁访问层。 此存储类最适合具有未知或不可预测访问模式且长期存在的数据。

Amazon S3 标准 – 不频繁访问（S3 标准 – IA）提供 S3 标准的高持久性、高吞吐量和低延迟，还提供较低的每 GB 存储价格和每 GB 检索费。“S3 标准 – IA”最适合不频繁访问但必要时需要快速访问的数据。

Amazon S3 单区 – 不频繁访问（S3 单区 – IA）将数据存储在单个可用区 (AZ) 中，因而为不需要 S3 标准或“S3 标准 – IA”存储类的可用性和弹性的不频繁访问的数据提供了成本较低的选择（成本比在“S3 标准 – IA”中存储数据低 20%）。 “S3 单区 – IA”最适合存储备份副本或可轻松重建的数据。

Amazon S3 Glacier

Amazon S3 Glacier 是低成本的存储服务，提供带安全功能的持久存储，适用于数据存档和备份。Amazon S3 Glacier 的数据持久性 (99.999999999%) 以及与 AWS 安全功能的集成均与 Amazon S3 相同。通过对存储在 Amazon S3 中的数据资产使用 Amazon S3 生命周期管理，可将 Amazon S3 Glacier 与 Amazon S3 进行集成，从而可将数据资产从 Amazon S3 无缝迁移到 Amazon S3 Glacier。在低存储成本至关重要、极少检索数据资产和可以接受从几分钟到几小时的检索延迟的场景下，Amazon S3 Glacier 是很好的存储选择。

Amazon S3 生命周期管理

Amazon S3 生命周期管理可让您创建数据生命周期规则，这些规则可用于自动将数据资产迁移到成本较低的存储层（例如"Amazon S3 标准 – 不频繁访问"或 Amazon S3 Glacier），或者让它们在不再被需要时过期。由一个 XML 文件组成的生命周期配置包含一组规则，其中包含您希望 Amazon S3 在数据资产的生命周期中对其执行的预定义操作。生命周期配置可根据数据资产的年限和名称执行操作，但也可与 S3 对象标记结合起来执行精细的数据资产管理。

Amazon S3 Storage Class Analysis 存储类分析

为数据湖制定和配置生命周期规则时面临的一个挑战是了解随着时间的推移如何访问数据资产。若并不频繁访问数据资产，则将这些对象转换为更经济高效的存储或存档层将具有经济意义。否则，与这些更经济高效的存储类关联的数据访问费用可能会抵销任何可能节省的费用。Amazon S3 提供 Amazon S3 Storage Class Analysis 存储类分析，可帮助您了解如何使用数据湖数据资产。Amazon S3 存储类分析使用机器学习算法来分析所收集的访问数据，以帮助您制定能优化成本的生命周期规则。

将数据湖存储分层

数据湖通常包含从许多源提取的原始数据，并且在提取后会针对临时查询和持续进行的分析转换和优化这些数据。但是，许多高级应用（例如机器学习和人工智能）包含以下步骤：首先构建数据模型，然后使用原始历史数据训练和改进这些模型。此外，通过保存历史原始数据，您可以回头重新处理历史数据，以在转换后的数据中提供新的见解。由于这些历史数据资产并不频繁受到访问，而且其大小可能很大，因此，它们通常十分适合存储在针对不频繁的数据访问进行了优化的"S3 标准 – IA"存储类中。之后，可以使用生命周期策略将这些数据进一步移动到 Amazon S3 Glacier 中，以用于长期存档。

在数据湖中，可以使用的另一种分层方法是在 Amazon S3 Glacier 文件库中保存经处理的数据和结果，例如合规与审计用途所需的数据和结果。借助 Amazon S3 Glacier 文件库锁定，数据湖管理员得以通过可锁定的策略，轻松在各个 Glacier 文件库上部署和强制实施合规性控制措施。管理员可以在文件库锁定策略中指定像"一次写入多次读取"(WORM) 这样的控制措施，然后锁定该策略以防将来对其进行编辑。锁定后，该策略将保持不变，而且 Amazon S3 Glacier 将强制实施指定的控制措施，以帮助您实现合规性目标，并使用 AWS CloudTrail 提供这些资产的审计跟踪。

优化成本和性能

您还可以围绕着成本和性能来优化 Amazon S3 的使用。Amazon S3 为数据湖提供了高性能基础，原因是它的巨大规模提供了几乎不受限制的吞吐量和很高的事务处理速率。使用 Amazon S3 最佳实践来命名数据资产可确保实现高水平的性能。Amazon Simple Storage Service 开发人员指南中介绍了这些最佳实践。

将数据分区

分区是一种组织数据集以便对其进行高效查询的重要技术。分区根据一列或多列的不同值在分层目录结构中组织数据。 例如，您可能决定按日期（按年、月和日细分）在 Amazon Simple Storage Service (Amazon S3) 中将应用程序日志分区。之后，将相当于一天数据的日志文件置于某个前缀之下，例如 `s3://my_bucket/logs/year=2018/month=01/day=23/`。Amazon Athena、Amazon Redshift Spectrum 和 AWS Glue 之类的服务可以使用此布局按分区值来筛选数据，而无需从 Amazon S3 中读取所有基础数据。

使用列式数据格式

另一个优化领域是使用最佳数据格式来存储经转换和标准化的数据集。列式格式可以更好地压缩数据，从而降低存储成本，另外还能显著提高查询性能。Apache Parquet 和 Apache ORC 是此类列式格式中的两种，它们可以减小所需的存储占用空间，改善查询性能，以及降低 Amazon Athena 和 Amazon Redshift Spectrum 之类的 AWS 服务的查询成本（这些服务按扫描的数据量进行收费）。

偏向使用大文件

若可以在多个文件上并行化多项任务，并且可以进一步拆分每个文件，以进一步提高并行性，则能更高效地运行大数据处理。但是，若文件太小（通常小于 64MB 左右），则处理引擎可能要花费更多的时间，并在打开 Amazon S3 文件、列出前缀(List)、获取对象元数据、设置数据传输、读取文件头和读取压缩字典方面产生相关的开销。不过，若无法拆分文件，并且文件太大，则查询处理会一直等到某个读取器完成整个文件的读取。这可能会降低并行性。

小文件问题的解决方法之一是在 Amazon EMR 上使用 S3DistCP 实用程序。可以使用它将较小的文件组合为较大的对象。还可以使用 S3DistCP 以优化的方式将大量数据从 HDFS 移动到 Amazon S3、从 Amazon S3 移动到 Amazon S3，以及从 Amazon S3 移动到 HDFS。

使用 AWS Glue 转换和标准化数据

AWS Glue 作业可帮助您转换和标准化数据，以优化查询性能。AWS Glue 支持将数据写入 Parquet 和 ORC 格式文件中，这可让您更快速轻松地将数据转换为最适合 Amazon Athena 的格式。此外，AWS Glue 爬网程序不仅会推断文件类型和架构，还会在填充 AWS Glue 数据目录时自动识别数据集的分区结构。可在 AWS Glue ETL 作业或 Amazon Athena 之类的查询引擎中查询结果分区列。

下图显示了如何将提取的 CSV 格式的数据转换为 Apache Parquet 以供 Amazon Athena 查询，同时改善性能和降低成本。有关此架构的更多信息，请参阅 Build a Data Lake Foundation with AWS Glue and Amazon S3。

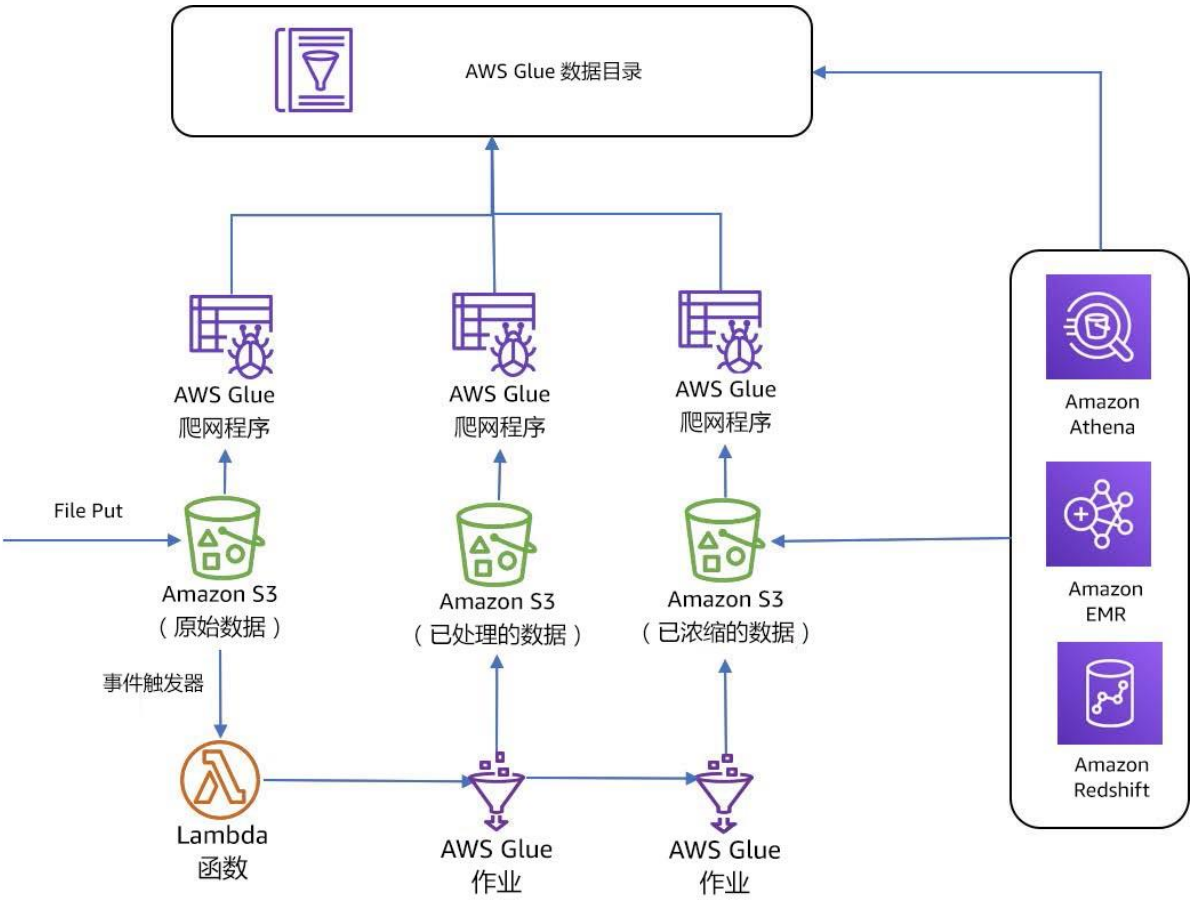


图19：使用AWS Glue 转换数据

了解应用程序如何与 Amazon S3 配合工作

正如我们在本指南前面强调的，Hive、Spark 和 MapReduce 之类的应用程序通过将 HDFS API 映射到 Amazon S3 API（如 Amazon EMR 随附的 EMRFS）与 Amazon S3 配合工作。对大多数 API 而言，HDFS API 与 Amazon S3 之间存在直接映射，但某些文件系统 API（如 rename 和 seek）在 Amazon S3 中并没有等效的 API，因此它们以间接方式映射。此外，Amazon S3、对象存储、HDFS 和文件系统之间存在语义差异，例如前缀(prefixes)与目录(directories)。"Hadoop integration with AWS"一文的"Overview"部分充分介绍了此类语义差异。

由于这些间接映射或语义差异，在 Amazon EMR 上使用开源应用程序时，有时您可能会看到性能问题或作业失败。许多应用程序还需要特定的自定义才能与 Amazon S3 配合工作，例如 Hive Blobstore 优化或 Amazon S3 优化的提交程序（随附于 Amazon EMR 上的 Spark）中的自定义。

本指南中特定于应用程序的章节也列出了其中的一些注意事项，并说明了如何优化这些应用程序，以便与 Amazon S3 配合工作。

数据目录迁移

Apache Hive 是一种开源数据仓库和分析包，在 Apache Hadoop 集群之上运行。Hive 是可以在 Amazon EMR 上运行的应用程序之一。Hive 元存储包含与表和用于构建表的基础数据有关的说明（包括分区名称和数据类型）。有三种 Hive 元存储部署模式：嵌入式(embedded)元存储、本地(local)元存储和远程(remote)元存储。将 Hadoop 本地集群迁移到 Amazon EMR 时，必须遵循不同的策略，具体取决于如何部署 Hive metastore。本章说明用于部署 Hive metastore 的不同模式，以及如何将现有元存储迁移到 Amazon EMR。

Hive metastore 元存储部署模式

Apache Hive 附带 Derby 数据库，它是由本地磁盘提供支持的嵌入式数据库。此数据库用于嵌入式元存储，但我们不建议使用 Derby，原因是它无法通过扩展来适应生产级别的工作负载。在 Amazon EMR 中，默认情况下 Hive 会在主节点文件系统上的 MySQL 数据库中记录元存储信息。在集群终止时，包括主节点在内的所有集群节点都会关闭。发生此情况时，由于节点的文件系统使用临时存储，因此本地数据将会丢失。为避免这种情况，我们建议在集群之外创建外部 Hive 元存储。有两种适用于 Amazon EMR 外部元存储的选择：

- AWS Glue 数据目录
- Amazon RDS 数据库或 Amazon Aurora 数据库

有关这两种选择的常见问题，请参阅附录 D：数据目录迁移常见问题。

AWS Glue 数据目录

AWS Glue 数据目录提供一个覆盖各种数据源和数据格式的统一元数据存储库，并且与 Amazon EMR、Amazon RDS、Amazon Redshift、Redshift Spectrum、Amazon Athena 以及任何与 Apache Hive 元存储兼容的应用程序相集成。使用 AWS Glue 数据目录的好处是：您无需单独管理 Hive 元存储数据库实例，无需维护持续进行的复制，也无需纵向扩展实例。AWS Glue 数据目录具有完全托管、无服务器、高可用性和容错的特点，它维护数据副本以避免故障，而且还根据使用情况扩展硬件。创建新的 Amazon EMR 集群时，可以选择 AWS Glue 数据目录作为 Hive 元存储。（**注意：**此选项仅在 Amazon EMR 版本 5.8.0 或更高版本上可用。）

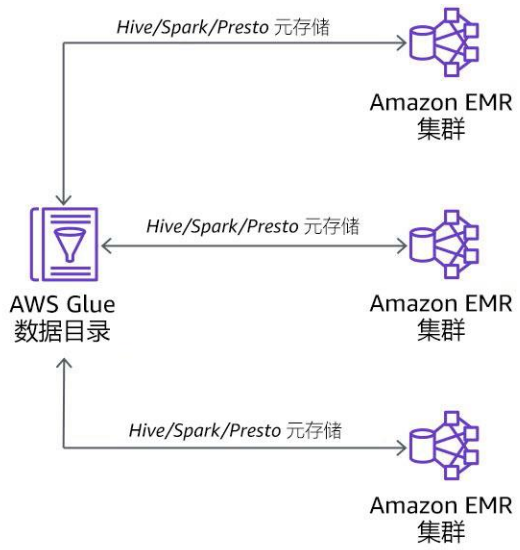


图 20：使用 AWS Glue 数据目录作为 Hive 元存储

Amazon EMR 控制台中的配置

Apache Hive、Presto 和 Apache Spark 均使用 Hive metastore。在 Amazon EMR 控制台中，可以选择将 AWS Glue 数据目录用于这三个应用程序中的任何一个。下图显示了此设置（它出现在控制台中的"高级选项"下）。

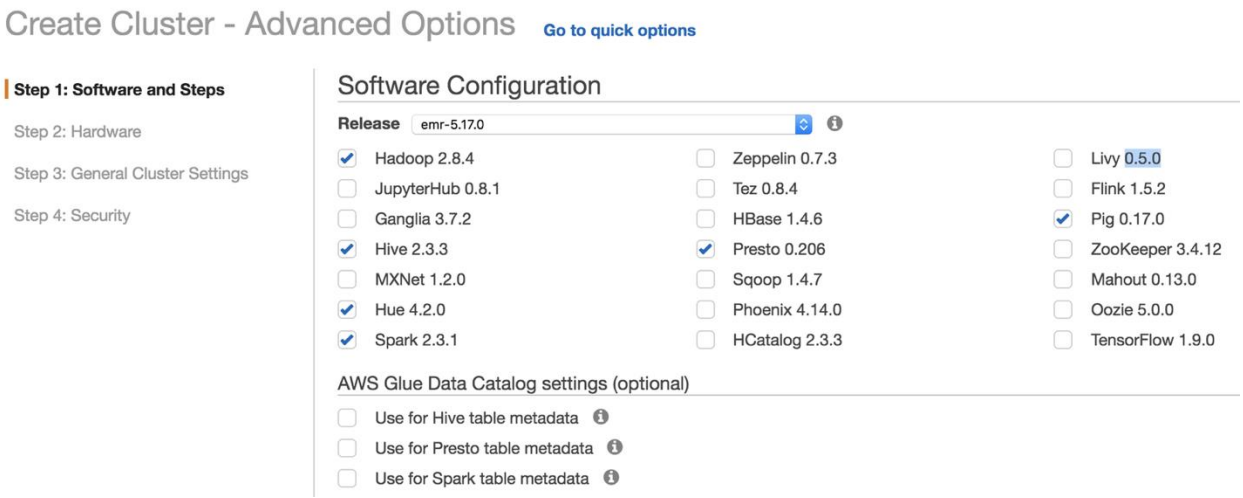


图 21：Amazon EMR 中的 AWS Glue 数据目录设置

AWS CLI 或 Amazon EMR API 中的配置

Hive 站点配置文件必须反映用于 Hive 元存储的 AWS Glue 数据目录。为此，请将 `hive.metastore.client.factory.class` 属性指向 `AWSGlueDataCatalogHiveClientFactory`，如下所示：

```
[
  {
    "Classification": "hive-site",
    "Properties": {
      "hive.metastore.client.factory.class":
"com.amazonaws.glue.catalog.metastore.AWSGlueDataCatalogHiveClientFactory"
    }
  }
]
```

创建 Amazon EMR 集群时，可将此设置作为应用程序配置的一部分。有关如何设置应用程序配置的更多详情，请参阅配置应用程序。

注意事项

- 可以为 AWS Glue 数据目录启用加密。有关详情，请参阅在 AWS Glue 中设置加密。
- 目前不支持列统计数据(Column statistics)、Hive Authorizations 授权和 Hive constraints 约束。要查看 AWS Glue 数据目录的约束列表，请参阅使用 AWS Glue 数据目录作为 Hive 的元存储。
- AWS Glue 数据目录具有多个版本，这表示一个表可能有多个结构版本。AWS Glue 将这些信息（包括 Hive 元存储数据）存储在 AWS Glue 数据目录中。

Amazon RDS 或 Amazon Aurora

部署外部 Hive 元存储要执行两个主要步骤：

1. 创建 Amazon RDS（MySQL 数据库）或 Amazon Aurora 数据库。
2. 配置 `hive-site.xml` 文件，使其指向 MySQL 或 Aurora 数据库。

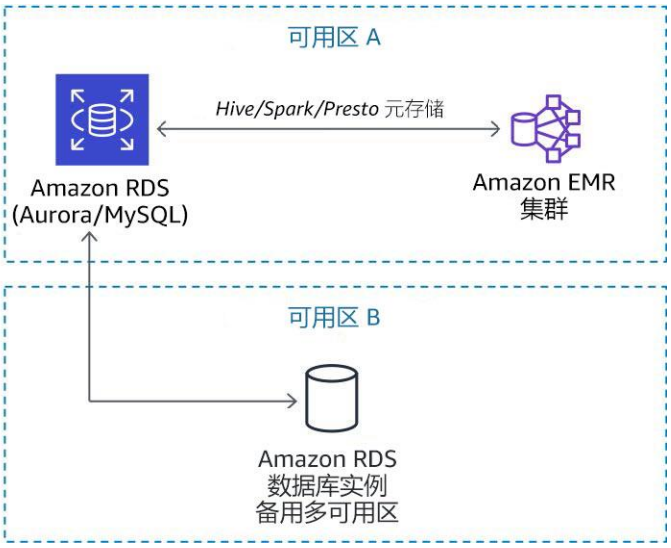


图 22：在 Amazon RDS 或 Amazon Aurora 上创建 Hive 元存储

创建 MySQL 或 Amazon Aurora 数据库

1. 创建数据库。
   - 要在 Amazon RDS 上创建 MySQL 数据库，请参阅使用 Amazon RDS 创建和连接 MySQL 数据库。
   - 要创建 Amazon Aurora 数据库，请参阅创建 Amazon Aurora 数据库集群。
2. 记下 URL、用户名、密码和数据库名称。
3. 更新安全组，以允许在 Amazon EMR 集群与 MySQL 数据库之间建立连接。

为外部 Hive 元存储配置 Amazon EMR

1. 创建一个配置文件，它要包含下面提供的 hive-site 分类信息。
   - `javax.jdo.option.ConnectionDriverName` 应反映驱动程序 `org.mariadb.jdbc.Driver`（首选驱动程序）
   - 将以下三个设置指向新创建的数据库：
     `javax.jdo.option.ConnectionURL`
     `javax.jdo.option.ConnectionUserName`
     `javax.jdo.option.ConnectionPassword`

```
[
    {
      "Classification": "hive-site",
      "Properties": {
        "javax.jdo.option.ConnectionURL":
"jdbc:mysql:\/\/hostname:3306\/hive?createDatabaseIfNotExist=true",
        "javax.jdo.option.ConnectionDriverName":
"org.mariadb.jdbc.Driver",
        "javax.jdo.option.ConnectionUserName": "username",
        "javax.jdo.option.ConnectionPassword": "password"
      }
    }
]
```

2. 在控制台中使用 Amazon EMR 时，将此信息作为来自 Amazon S3 的 JSON 或嵌入式文本传递：

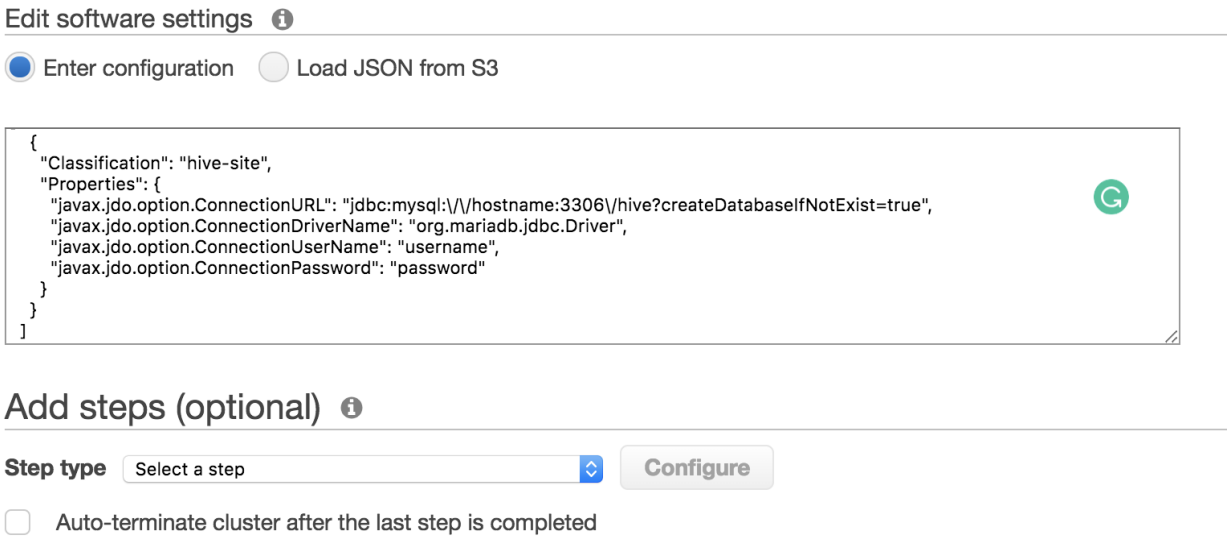


图 23：作为嵌入式文本的配置设置

3. 使用 AWS CLI 时，将 `hive-configuration.json` 配置文件作为本地文件或来自 Amazon S3 的文件传递：

```
aws emr create-cluster --release-label emr-5.17.0 --instance-type
m4.large --instance-count 2 \
--applications Name=Hive --configurations ./hive-configuration.json
--use-default-roles
```

图 24：作为本地文件的配置文件

```
aws emr create-cluster --release-label emr-5.17.0 --instance-type
m4.large --instance-count 2 \
--applications Name=Hive --configurations s3://emr-sample/hive-
configuration.json --use-default-roles
```

图 25：来自 Amazon S3 的配置文件

注意事项

Hive metastore 元存储是单一故障点。Amazon RDS 不会自动复制数据库，因此，强烈建议您在使用 Amazon RDS 时启用复制功能，以避免任何故障。有关如何在不同的可用区中创建数据库副本的更多信息，请参考以下来源：

- 如何为 Amazon RDS 数据库创建只读副本？
- 使用只读副本

Hive metastore 元存储迁移选项

将基于 Hadoop 的工作负载从本地迁移到云时，还必须迁移 Hive metastore 元存储。根据迁移计划和要求，可以通过两种不同方式迁移元存储：将现有 Hive 元存储完全迁移到 AWS 的一次性迁移，或者迁移 Hive metastore、但在本地保留副本的持续性迁移。在这种情况下，两个元存储在迁移阶段会实时同步。下一节将详细讨论这两种方案。

一次性元存储迁移

本部分重点讨论在将现有 Hive 元存储完全迁移到 AWS 时要考虑的一组选项。这种情况适用于组织计划在 AWS 上使用 Hive 元存储的方案。下图说明了此方案：

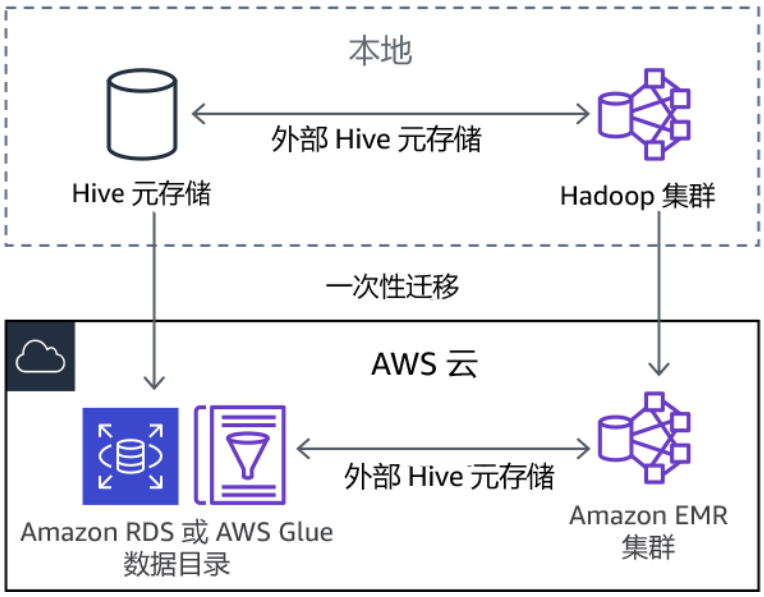


图 26：一次性元存储迁移

现有的 Hive 元存储迁移到 AWS Glue 数据目录

在此情况下，目标是将现有的 Hive 元存储从本地迁移到 AWS Glue 数据目录。可以使用 AWS Glue ETL 作业从 Hive 元存储中提取元数据，并使用 AWS Glue 作业加载元数据及更新 AWS Glue 数据目录。要了解有关这些选项的更多信息，请参阅 GitHub 上的 Migration between the Hive Metastore and the AWS Glue Data Catalog 一文。

现有的 Hive 元存储迁移到 Amazon RDS

在此情况下，您不使用 AWS Glue 数据目录，而是将 Hive 元存储数据从本地数据库迁移到 Amazon RDS。根据目前用于存储 Hive 元存储数据的数据库，您需要执行不同的步骤将它们迁移到相应的 Amazon RDS 实例。例如：

- 本地的 MySQL → Amazon RDS 或 Amazon Aurora 上的 MySQL
- 本地的 PostgreSQL → Amazon RDS 或 Amazon Aurora 上的 PostgreSQL
- 本地的 Oracle → Amazon RDS 上的 Oracle

下面这些资源介绍如何将这些数据库迁移到 AWS：

- Migrate On-Premises MySQL Data to Amazon RDS
- 将数据导入 Amazon RDS 上的 PostgreSQL
- 将数据从 PostgreSQL 数据库实例迁移到 Aurora PostgreSQL 数据库集群
- 将数据从 MySQL 数据库实例迁移到 Amazon Aurora MySQL 数据库集群

持续的元存储同步

此模式主要用于大规模迁移 – 您希望将本地 Hive 元存储迁移到 AWS，但也希望在迁移阶段在数据中心和云中继续运行 Hive 元存储。在这种情况下，需要进行持续同步，以便两个 Hive 元存储均为最新状态。对于指定的时间，仅应使用一个应用程序来更新 Hive 元存储，否则元存储将会不同步。

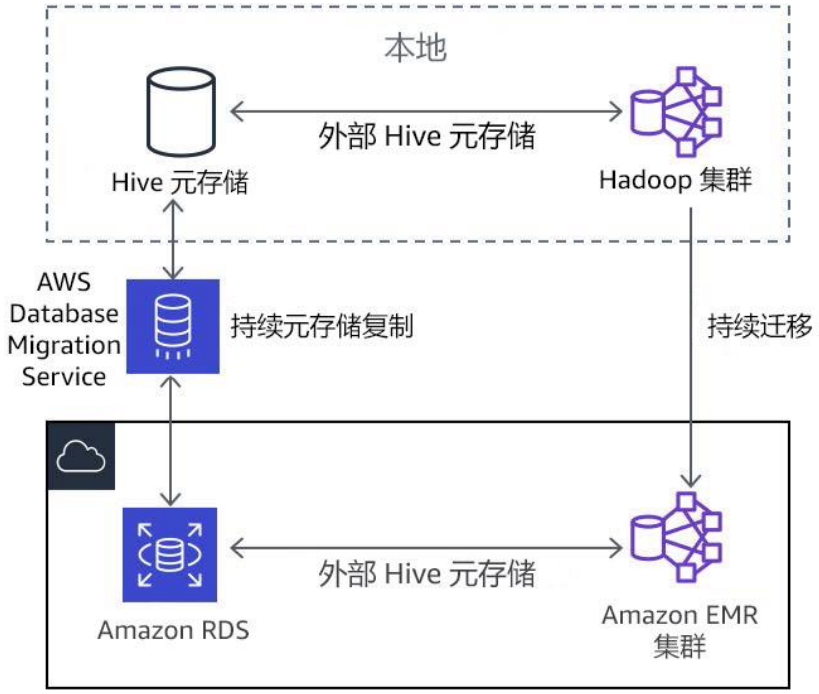


图 27：持续的元存储同步

AWS Database Migration Service 是一种数据迁移服务，可用于创建持续的复制。博客文章 Replicating Amazon EC2 or On-Premises SQL Server to Amazon RDS for SQL Server 讨论了如何为 SQL Server 实现持续的复制，不过同一方法也适用于其他数据库。

EMR 上的多租户

Amazon EMR 提供一整套功能来构建非常安全的多租户 Hadoop 集群资源和数据。Amazon EMR 上的多租户提供将数据和资源在不同租户之间相隔离的机制，还提供控制措施来防止单个应用程序、用户和队列垄断集群资源。多租户自身带来了一系列挑战。例如，在管道的所有阶段实现多租户需要了解涉及到的流程和工具的细微差别；可能难以测量租户的资源使用量，特别是在租户共享元数据和计算资源时；新租户的加入可能会导致难以扩展和投入时间；以及全面应用可靠的安全控制措施可能是很艰巨的任务。

本章讨论在 Amazon EMR 上实现多租户所需执行的步骤，以及用户、数据和资源隔离等重要方面，之后会提出一些最佳实践建议。

大体上说，在 Amazon EMR 上构建多租户分析时隔离用户、数据和资源涉及到两种不同的概念模式。这两种模式是：

- 筒仓模式(Silo Mode)
- 共享模式 (Shared Mode)

筒仓模式

在筒仓模式下，每个租户都获得自己的 Amazon EMR 集群，以及用于处理和分析数据集的特定工具。数据存储在租户的 S_3 存储桶或集群上的 HDFS 中。Hive 元存储通常位于集群上，或存储在外部 Amazon RDS 上。

筒仓方案示例

下图显示了 Amazon EMR 中的筒仓方案示例。在此方案中，有三位不同用户，即一位数据工程师、一位分析师和一位数据科学家，他们各自启动自己的集群。数据工程师安装 Spark 和 Hive 之类的工具，以操控处理后的结果，并将之存储在 S_3 中。分析师运行 Spark SQL 和 Presto 之类的工具，以研究数据集并将查询结果发送到他自己的 S_3 存储桶。最后，数据科学家可能使用 EMR 集群来运行 ML 或深度学习框架。

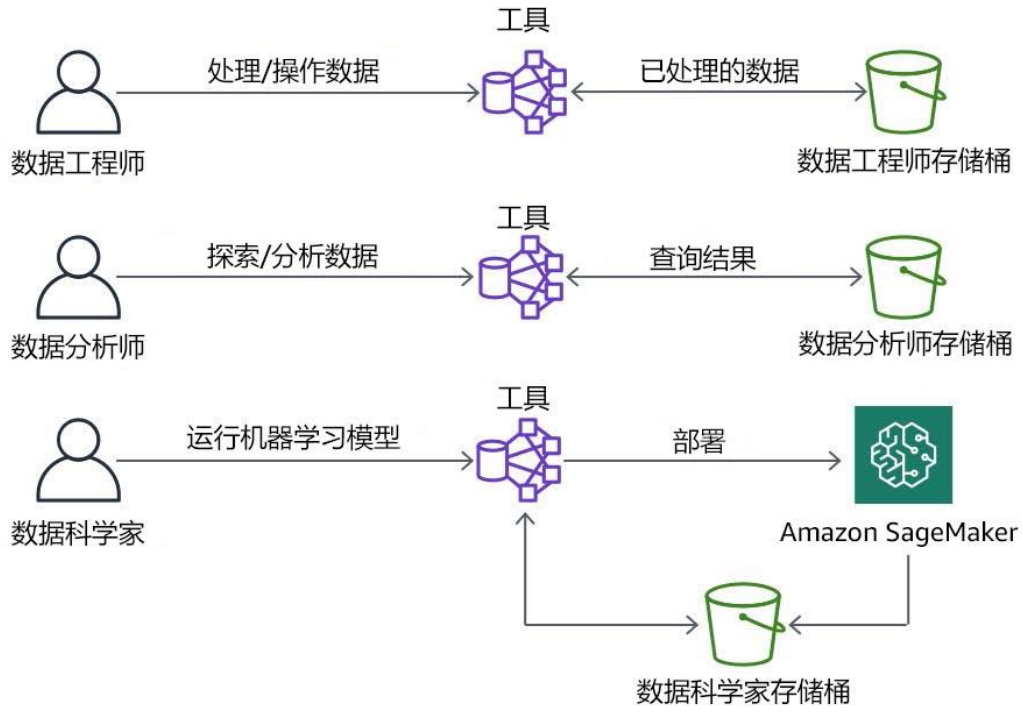


图 28：筒仓模式方案示例

在此模式下，您可将集群配置为在所有处理步骤均完成后自动终止。此设置称为临时性集群。临时性集群不仅能实现每个租户的总体隔离，还能降低成本，原因是仅按照集群的运行时间长度向集群收费。*表 2* 列出了将筒仓模式与 Amazon EMR 一起使用的优缺点。

表 2：筒仓模式的优缺点

优点	缺点
完全隔离数据和资源。	可能难以跨集群共享数据（特别是在使用 HDFS 时）
在与 Spot 实例和临时性集群一起使用时可能较为经济高效。	启动单独的集群可能费用高昂。
易于测算每个租户的资源使用量。	

共享模式

在共享模式下，租户共享 Amazon EMR 集群（安装了用于处理/分析/数据科学的工具 – 全都在一个集群中）。数据集存储在租户的 S_3 存储桶或租户在集群上的 HDFS 文件夹中。Hive 元存储可以在集群上，也可以在外部 Amazon RDS 或 AWS Glue 数据目录上。在许多组织中，此共享方案更为常见。通过在组织之间共享集群，可以经济高效地运行安装了 Hadoop 的大型设施，原因是组织无需创建私有集群即可获得规模经济的好处。

安装了所有工具和框架的大型多节点集群可以支持各种用户。此外，最终用户也能使用此基础设施，他们可以启动边缘节点来运行数据科学平台。

虽然共享集群经济高效，但这样做可能会引发担忧，这是因为某个租户可能会垄断资源，导致无法为其他租户实现 SLA。例如，分析师可能会在 Presto 或 Hive 和大部分集群资源上发布长时间运行的查询。或者，数据科学家可能会使用大量数据来训练一个模型。

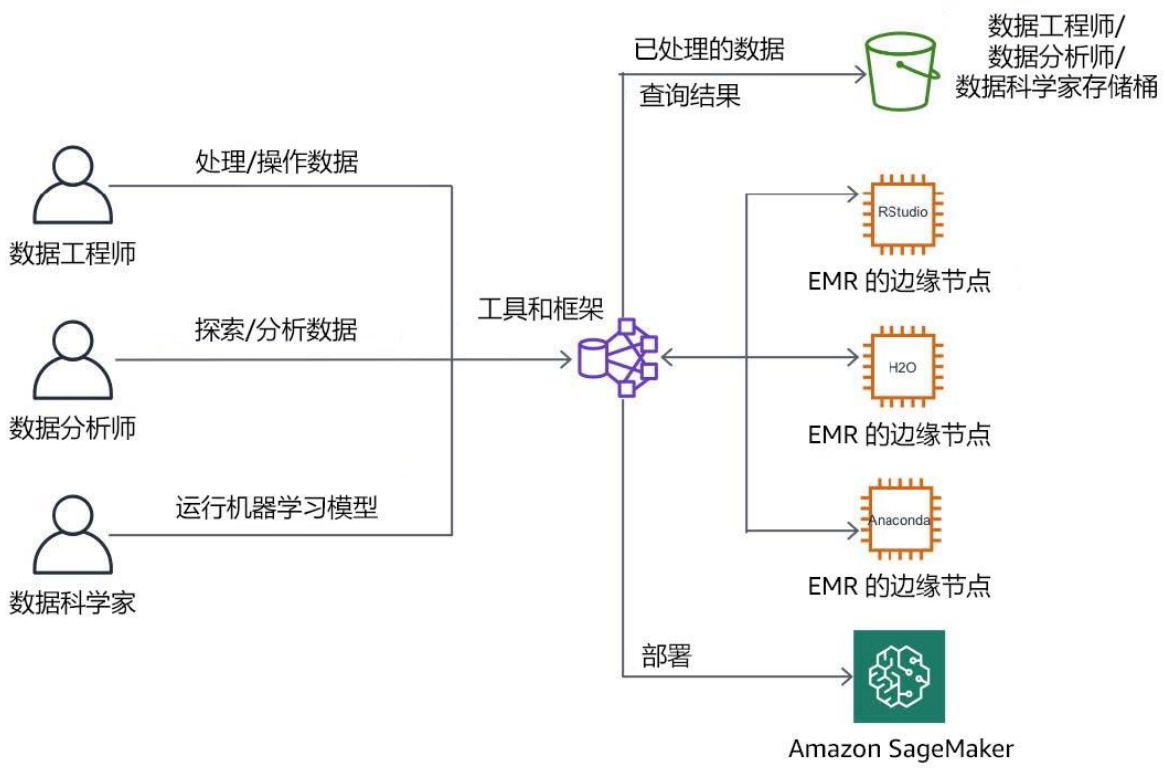


图 29：共享模式方案示例

表 3 列出了在共享模式下启动 Amazon EMR 集群的优缺点。

表 3：使用共享模式的优缺点

优点	缺点
运营负担较小，原因是只需维护一个集群。	在具有许多租户时难以测算使用量和资源
若充分利用集群，则可能较为经济高效。	配置 YARN scheduler 调度器可能既困难又复杂。
	无法针对各项工作负载自定义集群。数据工程师和数据科学家无法拥有适用于其特定工作负载的不同配置（实例类型、卷等）。
	一种配置适应所有使用案例。集群配置不会改变，团队必须努力进行应用程序优化，以改善工作负载的性能，而不是调整集群配置。
	必须先升级所有应用程序，然后才能升级软件。
	若集群出错，影响范围将会很大。

在 Amazon EMR 上实施多租户的注意事项

在 Amazon EMR 上实施多租户架构需要仔细思考和规划，以及考虑包括用户、数据和资源隔离在内的重要方面。

用户隔离

身份验证

用户身份验证是保护集群资源及防止未经授权访问数据的重要环节。在 Amazon EMR 上，可以通过 LDAP 服务器验证用户身份，或者设置 Kerberos 以通过私钥加密技术提供强身份验证。使用 Kerberos 身份验证时，Amazon EMR 会为它在集群上安装的应用程序、组件和子系统配置 Kerberos，以便它们相互进行身份验证。

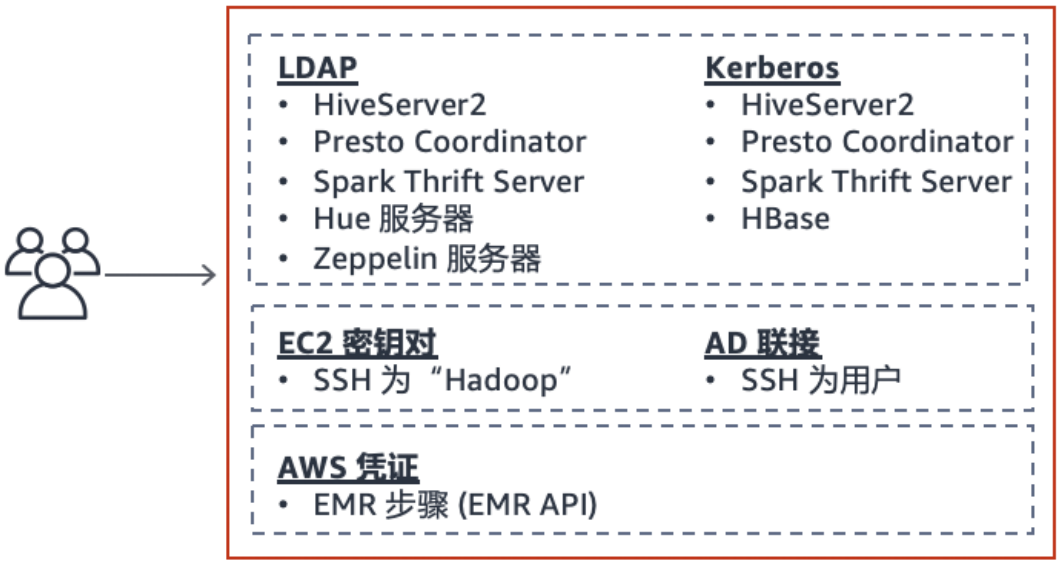


图30：通过LDAP 或Kerberos 验证用户身份

将 Kerberos 与 Amazon EMR 一起使用时，可以选择将其设置为集群专用 KDC，或具有不同架构选项的外部 KDC。无论选择何种架构，配置 Kerberos 的步骤都相同。首先，创建安全配置。创建集群时，您要指定该安全配置和兼容的特定于集群的 Kerberos 选项。然后，在集群上为 Linux 用户创建与 KDC 中的用户主体(principal)相匹配的 HDFS 目录。

之后，可以使用 Amazon EC2 密钥对集群实例的与建立的 SSH 客户端连接间进行授权。

数据隔离

在用户通过身份验证后，您必须考虑授权他们使用哪些数据资产。可以选择在 Amazon EMR 上的存储层或服务器层实施授权。默认情况下，附加到集群上的 EC2 角色的策略可确定可在 Amazon S3 中访问的数据。借助 EMRFS 授权，您可以指定用户或组在使用 EMRFS 访问 S3 时所继承的 IAM 角色。为每个用户或组选择 IAM 角色将允许在多用户 Amazon EMR 集群上实施精细的 S3 访问控制。

注意：对于运行 Spark 应用程序的用户，或通过 SSH 访问 EMR 集群的用户，EMRFS 无法阻止其绕过 EMRFS 并继承不同的 IAM 角色。此局限性能在 AWS Lake Formation 中得到解决。

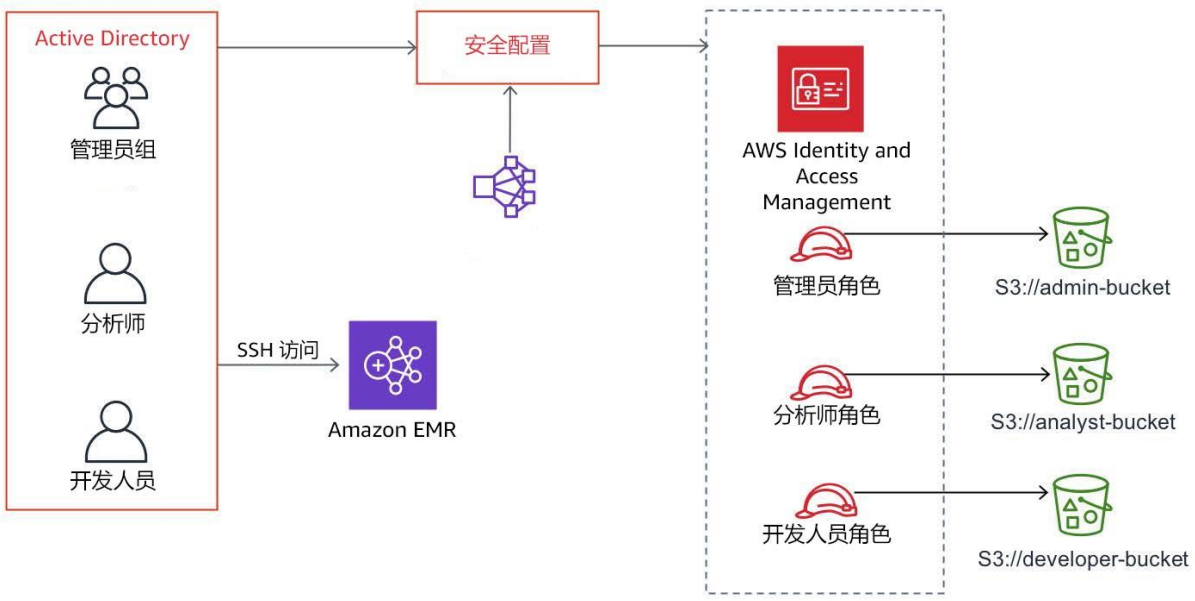


图31：存储层和服务器层的授权

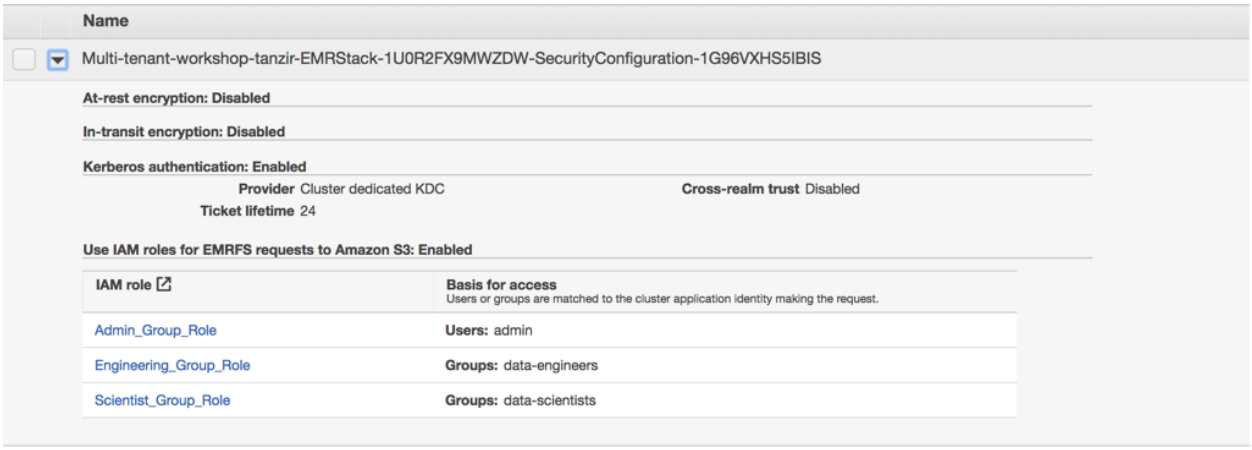


图32：借助 EMRFS 授权实现的精细访问控制

资源隔离

在 Amazon EMR 上，可以使用不同的 YARN 队列来提交作业。每个 YARN 队列都可能有不同的资源容量，并可能与 Amazon EMR 集群上的特定用户和组关联。YARN 资源管理器 UI 显示可用队列的列表。

注意：YARN 队列仅应用于在 YARN 上运行的应用程序（例如，Presto 应用程序不会使用 YARN 队列）。

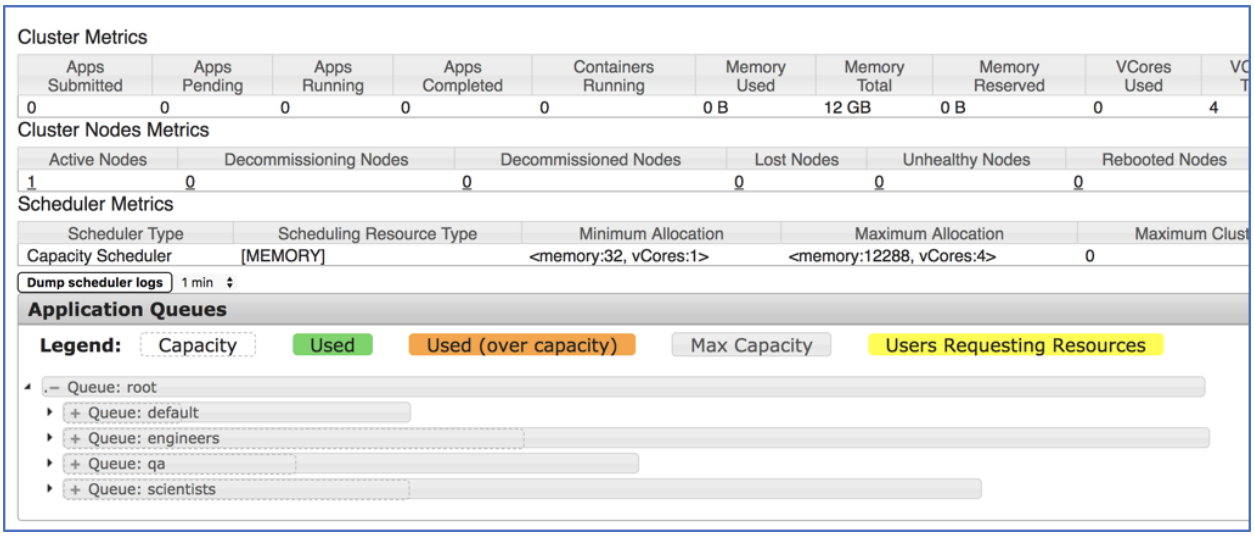


图33：YARN 资源管理器 UI

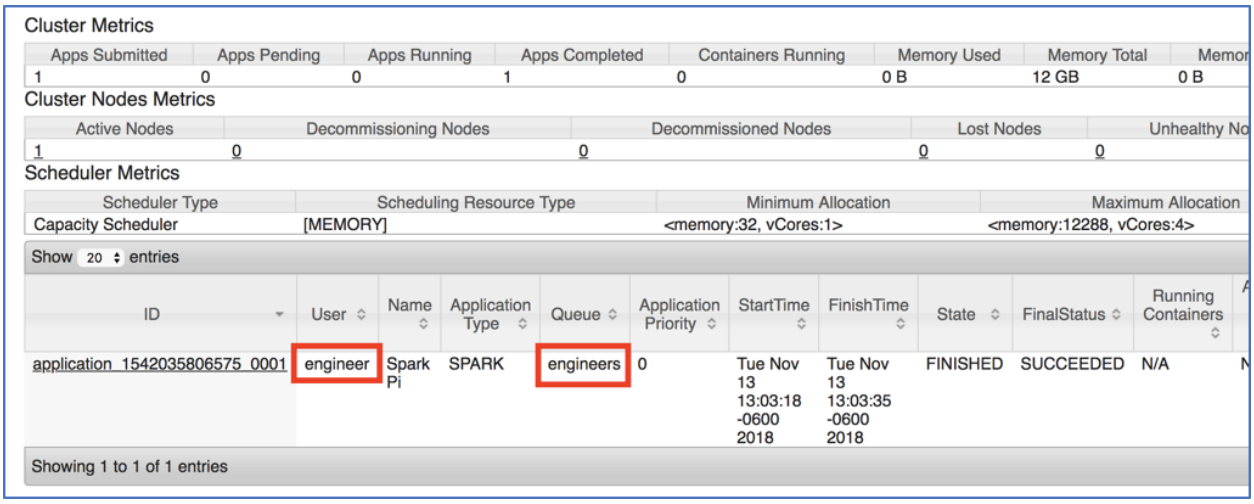
例如，**engineers** 组中的用户 **engineer** 可以登录 EMR 主节点，然后向 **engineers** YARN 队列提交作业：

```
ssh -l engineer <<emr-dns>>
[engineer@ 10-10-1-222 ~] spark-submit --queue engineers --class
org.apache.spark.examples.SparkPi --master yarn
/usr/lib/spark/examples/jars/spark-examples.jar
```

```
[engineer@ip-10-0-1-222 ~]$ spark-submit --queue engineers --class org.apache.spark.examples.SparkPi --master yarn /usr/lib/spark/examples/jars/spark-examples.jar
18/11/13 19:03:11 INFO SparkContext: Running Spark version 2.3.1
18/11/13 19:03:11 INFO SparkContext: Submitted application: Spark Pi
18/11/13 19:03:11 INFO SecurityManager: Changing view acls to: engineer
18/11/13 19:03:11 INFO SecurityManager: Changing modify acls to: engineer
18/11/13 19:03:11 INFO SecurityManager: Changing view acls groups to:
18/11/13 19:03:11 INFO SecurityManager: Changing modify acls groups to:
18/11/13 19:03:11 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: Set(engineer); groups with view permissions: Set();
 users  with modify permissions: Set(engineer); groups with modify permissions: Set()
```

图34：Amazon EMR 上的示例代码

在此例中，用户 **engineer** 提交 Spark 作业，并通过参数"**—queue**" 传递，以反映它应该用来运行该 Spark 作业的队列。YARN 资源管理器 UI 显示同一个作业。



Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memor
1	0	0	1	0	0 B	12 GB	0 B

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy No
1	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:32, vCores:1>	<memory:12288, vCores:4>

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers
application_1542035806575_0001	engineer	Spark Pi	SPARK	engineers	0	Tue Nov 13 13:03:18 -0600 2018	Tue Nov 13 13:03:35 -0600 2018	FINISHED	SUCCEEDED	N/A

Showing 1 to 1 of 1 entries

图35：用户engineer 在YARN 中提交Spark 作业

Amazon EMR 上的提取、转换、加载 (ETL)

Amazon EMR 上的编排

编排应用程序在 Apache Hadoop 生态系统中被重度使用，旨在集成和集中监控多个应用程序。它们可管理相互依赖的复杂作业，维护应用程序的每种状态，并根据预定义的模式来执行。有多个编排应用程序可用于 Hadoop，其中包括 Apache Oozie，它是获得广泛使用的工作流程计划程序系统。您还可以使用 AWS Step Functions 和 AWS Glue 来编排和计划多个应用程序。将工作负载从本地 Hadoop 集群迁移到 Amazon EMR 时，除了迁移在 Hadoop 集群上运行的应用程序外，还必须迁移编排工具，以便它能有效编排云中的应用程序。本节介绍从本地 Hadoop 集群迁移编排应用程序的步骤，以及可用于 Amazon EMR 的其他编排选项，还提出了有关迁移的最佳实践建议。

将 Apache Oozie 迁移到 Amazon EMR

Apache Oozie 是一种 Java Web 应用程序，用于管理和计划 Apache Hadoop 作业。Oozie 将多个作业组合起来，并创建可根据时间或事件计划的操作的有向无环图 (DAG)。Oozie 使用一个数据库来存储每个作业的元数据。默认情况下，Oozie 配置为使用嵌入式 Derby 数据库。但是，为了改善生产工作负载的可靠性和可用性，请务必使用外部数据库（例如 MySQL、PostgreSQL、Oracle 等）。Oozie 使用 XML 定义其工作流程和协调器。

Oozie 随附在 Amazon EMR 发布版本 5.0.0 和更高版本中。您可以在 Amazon EMR 控制台的“软件配置”中选择 Oozie：

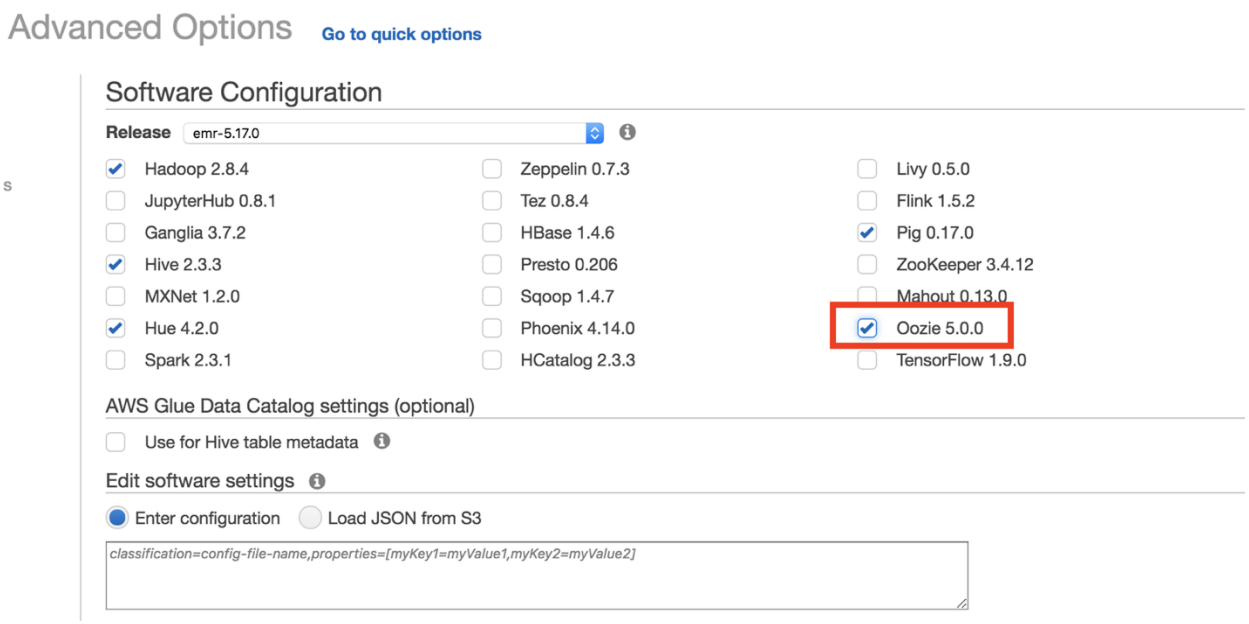


图36：在Amazon EMR 软件配置过程中选择Oozie

也可以在通过 AWS CLI 配置选项时选择 Oozie：

```
aws emr create-cluster --release-label emr-5.17.0 --instance-type
m4.large --instance-count 2 --applications Name=Hadoop Name=Hive
Name=Oozie --name "TestCluster"
```

将 Apache Oozie 从本地 Hadoop 集群迁移到 Amazon EMR 时，按以下顺序进行：

1. 迁移 Oozie 数据库（可选）。
2. 迁移 Oozie 作业。

迁移 Oozie 数据库

仅在您想将 Oozie 作业历史记录从本地集群迁移到 Amazon EMR 时，才需要执行以下步骤。在此情况下，某些 Oozie 数据库链接可能会中断，原因是它们指向旧集群的主机地址。因此，在迁移到 Amazon EMR 上的 Apache Oozie 时，我们建议从空的 Oozie 数据库开始。

Apache Oozie 附带 dump 导出和加载实用程序，可用于将现有数据库转储到文件，然后将该文件加载到新数据库中。

1. 登录正在运行现有 Oozie 服务器的主机，然后执行以下命令。

   注意：在 Amazon EMR 中，`oozie-setup.sh` 文件位于此文件路径下：`/usr/lib/oozie/bin/`

```
./oozie-setup.sh export /folder/oozie_db.zip
```

2. 使用 AWS CLI 将 `oozie_db.zip` 文件上传到 Amazon S3：

```
aws s3 cp oozie_db.zip s3://migration/oozie/oozie_db.zip
```

3. 登录 Amazon EMR 主节点，然后从 Amazon S3 下载 `oozie_db.zip` 文件。

```
ssh -i <<key>> hadoop@<<amazon-emr-master-node>>
aws s3 cp s3://migration/oozie/oozie_db.zip oozie_db.zip
```

4. 更改 `oozie-site.xml` 文件，使其指向新的目标数据库。以下是与 Oozie 数据库设置相关的默认配置：

```
oozie.db.schema.name=oozie
oozie.service.JPAService.create.db.schema=false
oozie.service.JPAService.validate.db.connection=false
oozie.service.JPAService.jdbc.driver=org.apache.derby.jdbc.EmbeddedDriver
oozie.service.JPAService.jdbc.url=jdbc:derby:${oozie.data.dir}/${oozie.db.schema.name}-db;create=true
oozie.service.JPAService.jdbc.username=sa
oozie.service.JPAService.jdbc.password=
oozie.service.JPAService.pool.max.active.conn=10
```

   例如，若您计划在 Amazon RDS 上使用 MySQL，则创建 Amazon RDS 实例，然后更新 `oozie-site.xml` 文件以映射到 RDS 配置。

```
<property>
    <name>oozie.service.JPAService.jdbc.driver</name>
    <value>com.mysql.jdbc.Driver</value>
</property>
<property>
    <name>oozie.service.JPAService.jdbc.url</name>
    <value>jdbc:mysql://<<rds-host>>:3306/oozie</value>
    </property>
<property>
    <name>oozie.service.JPAService.jdbc.username</name>
```

```
    <value>mysql-username</value>
</property>
<property>
    <name>oozie.service.JPAService.jdbc.password</name>
    <value>mysql-password</value>
</property>
```

5. 在新的 RDS 实例中创建 Oozie 数据库，并酌情向 Oozie 用户授予权限：

```
$ mysql -u root -p
Enter password:

mysql> create database oozie default character set utf8;
Query OK, 1 row affected (0.00 sec)

mysql> grant all privileges on oozie.* to 'oozie'@'localhost'
identified by 'oozie';
Query OK, 0 rows affected (0.00 sec)

mysql> grant all privileges on oozie.* to 'oozie'@'%' identified by
'oozie';
Query OK, 0 rows affected (0.00 sec)
```

6. 将先前的 `oozie_db.zip` 数据库转储加载到 AWS 上的这个新数据库：

```
./oozie-setup.sh import oozie_db.zip
```

导入后，CLI 会显示您已导入的数据库行数及其各自的表名称。

```
4 row(s) imported to table WF_JOBS.
Persisting last batch. [batch.size=12]
12 row(s) imported to table WF_ACTIONS.
0 row(s) imported to table COORD_JOBS.
0 row(s) imported to table COORD_ACTIONS.
0 row(s) imported to table BUNDLE_JOBS.
0 row(s) imported to table BUNDLE_ACTIONS.
0 row(s) imported to table SLA_REGISTRATION.
0 row(s) imported to table SLA_SUMMARY.
Cleanup not necessary, no entities skipped.
3239 [main] INFO org.apache.oozie.service.Services - Shutdown
```

图37：成功的数据库导入

7. 重启 Oozie 服务器，以反映新的配置和数据库：

```
sudo restart oozie
```

注意：根据数据库类型，可能需将驱动程序 driver 的 jar 放在 Amazon EMR 上的 Oozie 类路径下，具体位置是：/var/lib/oozie。

迁移 Oozie 作业

将 Oozie 作业从本地 Hadoop 集群迁移到 Amazon EMR 非常简单。若您尚未将 Oozie 数据库迁移到 Amazon EMR，则只需将所有与 Oozie 作业相关的文件迁移到 Amazon EMR。迁移与作业相关的文件包括迁移以下文件：

- job.properties
- workflow.xml
- coordinator.xml
- 捆绑包文件 Bundle files
- 外部参数文件
- 任何依赖文件

要快速迁移这些文件，请执行以下步骤：

1. 将所有与 Oozie 相关的文件压缩为一个存档文件。
2. 将该存档文件上传到 Amazon S3。
3. 将该文件下载到 Amazon EMR 主节点。
4. 将压缩文件提取到合适的文件夹中。
5. 修改各个文件，以反映 Amazon EMR 集群设置（即 nameNode、resourceManager 等）。
6. 在 Amazon EMR 上重新提交这些 Oozie 作业。

注意事项

迁移 Oozie 数据库和作业时，要考虑以下问题。

- Amazon EMR 上不支持 Oozie 原生 Web 接口。要将前端接口用于 Oozie，请使用在 Amazon EMR 上运行的 Hue 应用程序。有关更多信息，请参阅 Hue。
- 像 Amazon EMR 一样，默认情况下 Oozie 配置为使用嵌入式 Derby 数据库。我们建议使用 Amazon RDS 实例来托管 Oozie 数据库。
- 您可以使用 Amazon S3 存储工作流程 XML 文件。为此，必须将 S3 库文件放在 Oozie 类路径下。请按照以下步骤执行操作：

  a. 打开命令行，然后使用以下命令找到 Amazon EMR 主节点中的 `emrfs-hadoop-assembly` 文件。

```
[hadoop@ip-10-0-30-172 spark]$ locate emrfs-hadoop-assembly
/usr/share/aws/emr/emrfs/lib/emrfs-hadoop-assembly-2.26.0.jar
```

  b. 将 emrfs-hadoop-assembly 文件复制到 `/usr/lib/oozie/lib` 目录。

```
[hadoop@ip-10-0-30-172 spark]$ sudo cp
/usr/share/aws/emr/emrfs/lib/emrfs-hadoop-assembly-2.26.0.jar
/usr/lib/oozie/lib
```

c. 重启 Oozie 服务器。

```
[hadoop@ip-10-0-30-172 spark]$ sudo restart oozie
```

以下是 `workflow.xml` 示例文件，它使用 Oozie 的 Spark action 执行 SparkPI。在此示例中，`spark-examples-jar` 和 `workflow.xml` 存储在 Amazon S3 存储桶 (`s3://tm-app-demos/oozie`) 中。

```
<workflow-app xmlns='uri:oozie:workflow:1.0' name='SparkPi-S3'>
<start to='spark-node' />

<action name='spark-node'>
<spark xmlns="uri:oozie:spark-action:1.0">
<resource-manager>${resourceManager}</resource-manager>
<name-node>${nameNode}</name-node>
<master>${master}</master>
<name>SparkPi</name>
<class>org.apache.spark.examples.SparkPi</class>
<jar>s3://tm-app-demos/oozie/lib/spark-examples.jar</jar>
</spark>
<ok to="end" />
<error to="fail" />
</action>

<kill name="fail">
<message>Workflow failed, error
message[${wf:errorMessage(wf:lastErrorNode())}]
</message>
</kill>
<end name='end' />
</workflow-app>
```

以下示例是对应的 `job.properties` 文件。请参见*图 38*，以了解为 Oozie 文件创建的 Amazon S3 存储桶。

```
nameNode=hdfs://ip-10-0-30-172.ec2.internal:8020
resourceManager=ip-10-0-30-172.ec2.internal:8032
master=yarn-cluster
queueName=default
```

```
examplesRoot=examples
oozie.use.system.libpath=true
#oozie.wf.application.path=${nameNode}/user/${user.name}/oozie/spark
oozie.wf.application.path=s3://tm-app-demos/oozie
```

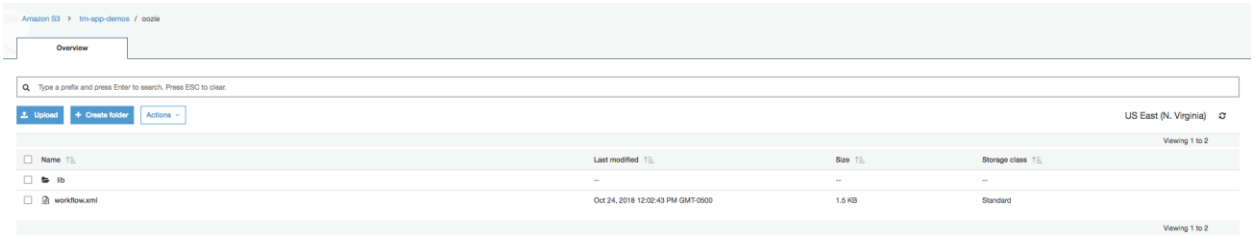


图38：用于Oozie 相关文件的Amazon S3 存储桶

用于编排的 AWS 服务

可使用 AWS 服务为基于 Hadoop 的作业创建编排。以下服务是 AWS 上的一些常见编排选项。

AWS Step Functions

AWS Step Functions 可让您协调多项 AWS 服务，以建立无服务器的工作流程，从而能快速构建和更新应用程序。通过使用 Step Functions，您可以设计和运行将多项服务整合起来的工作流程。工作流程由一系列步骤组成，而且上一个步骤的输出充当下一个步骤的输入。Step Functions 可让您更轻松直观地开发应用程序，原因是它将工作流程转换为易于理解、易于向他人解释和易于更改的状态机图。您可以在各个步骤执行时进行监控，这意味着可以快速识别并解决问题。Step Functions 可以自动触发和跟踪各个步骤，并在出现错误时重试，以便您的应用程序按照预期顺序执行。

下图是 AWS Step Functions 如何编排多个 Apache Spark 作业的示例。



图39：使用AWS Step Functions 编排的多个Apache Spark 作业

有关如何集成 AWS Step Functions 以便为基于 Hadoop 的作业创建编排的更多信息，请参阅以下博客文章：

- Orchestrate Apache Spark applications using AWS Step Functions and Apache Livy
- Orchestrate multiple ETL jobs using AWS Step Functions and AWS Lambda

AWS Data Pipeline

AWS Data Pipeline 是一种服务，可帮助您可靠地处理数据，并按指定的间隔在不同 AWS 计算与存储服务以及本地数据源之间移动数据。利用 AWS Data Pipeline，您可以定期在存储数据的地方访问数据，大规模地转换和处理数据，以及高效地将结果传输给各种 AWS 服务（例如 Amazon S3、Amazon RDS、Amazon DynamoDB 和 Amazon EMR）。AWS Data Pipeline 帮助您轻松创建具有容错、可重复和高可用性特点的复杂数据处理工作负载。

有关如何为 Amazon EMR 配置 AWS Data Pipeline 及如何使用它进行编排的更多信息，请参阅以下资源：

- ETL Processing Using AWS Data Pipeline and Amazon Elastic MapReduce
- 使用 AWS Data Pipeline 控制台启动集群
- 通过 AWS Data Pipeline 自动处理定期集群

其他编排选项

因被社区采用和包含丰富的功能，某些用于编排的开源应用程序受到追捧。本节将介绍 Apache Airflow 和 Luigi，并说明如何在 AWS 上使用这两种应用程序来为基于 Hadoop 的作业创建编排。

Apache Airflow

Airflow 是一种开源任务计划程序，可帮助管理 ETL 任务。可从一个中央位置计划和管理 Apache Airflow 工作流程。利用 Airflow 的配置即代码方法，可以轻松地自动生成工作流程、ETL 任务和依赖关系。它可帮助开发人员将焦点从构建和调试数据管道转向关注业务问题。

Apache Airflow 可安装在 Amazon EC2 实例上，也可通过引导程序安装在 Amazon EMR 主节点上。它附带各种连接器，这些连接器可帮助将其与不同的 AWS 服务进行集成。

有关如何在 Amazon EMR 上安装和集成 Airflow 以便运行作业的更多信息，请访问 AWS 大数据博客，参阅 Build a Concurrent Data Orchestration Pipeline Using Amazon EMR and Apache Livy。

Luigi

Luigi 是另一种开源应用程序，可用于构建复杂的批处理作业管道。它处理计划、依赖关系解析和工作流程管理，并包含命令行工具集成。

有关编排的最佳实践

在为基于 Hadoop 的作业构建可靠和容错的编排时，要考虑一些要点。像 Hadoop 一样，编排工具应能够扩展，以便能处理大量作业，另外还要能够随着 Hadoop 的扩展而相应地扩展。下面提供一些在为 Hadoop 作业创建编排时要考虑的最佳实践：

- 大多数编排应用程序均使用默认的嵌入式数据库来存储作业元数据。对于生产工作负载，我们建议使用单独的数据库，以改善性能和可用性。
- 尽可能使用托管或无服务器编排服务，以减少持续不断的人工参与。
- 与通知服务（例如 Amazon SNS 和 Amazon CloudWatch）集成，以便相关各方在故障发生时立即收到通知并积极参与排障。
- 确保编排应用程序可以处理异步和同步的作业和任务执行，以改善性能并减少开销。
- 编排应用程序应同时能监控作业的执行和管理，以便开发人员可以集中监控一切。
- 编排应用程序应提供优雅处理故障的能力。
- 若使用 Apache Airflow，请务必对生产工作负载使用集群模式配置。

使用下表来确定最适合您的使用案例的编排应用程序。

表 4：编排应用程序

区分因素/使用案例	AWS Step Functions	Apache Oozie	Apache Airflow
无服务器	是	否	否
基于 Spark 的作业	是	是	是
丰富的 UI 和问题排查工具	否	否	是
与其他监控工具集成	否	否	是
执行类型	状态机/Lambda	DAG	DAG
与 AWS 服务交互	大规模	非常有限	一般
管理责任	小	小	巨大
混合环境 – AWS 服务和非 AWS 服务	仅在云中	仅适用于 Hadoop 作业	广泛覆盖
成本	2c/1000 转换	EMR 使用情况	EC2 使用情况

迁移 Apache Spark

Apache Spark 应用程序是 Amazon EMR 上常见的工作负载。由于 Amazon EMR 集群可配置为使用特定的实例类型，并能轻松地横向扩展到大量的工作负载，因此 Apache Spark 能有效地围绕性能和成本优化计算资源。

有关迁移 Apache Spark 的使用案例

迁移到 Amazon EMR 上的 Spark 时，要考虑多个使用案例。许多时候，现有环境都是一个大型集群，具有专用于执行 Spark 作业的特定数量的资源。借助 Amazon EMR，您可以继续使用一个大型共享集群，也可以使用按需实例集群，以按作业隔离资源。按需实例方法不仅可以确保 Spark 充分利用每个集群的资源，还可以让您利用不同的实例类型。但是，这种方法要求在运行 Spark 作业之前围绕创建 Amazon EMR 集群进行更多规划和自动化。

共享集群

在共享集群中，请注意您希望在任一给定时间运行多少个并行作业。默认情况下，EMR 会配置执行程序，以通过使用 maximumResourceAllocation 属性在每个节点上使用最大数量的资源。在共享集群上，您可能需要手动配置 Spark 核心、内存和执行程序。有关详情，请访问 AWS 大数据博客，参阅在 Amazon EMR 上成功管理 Apache Spark 应用程序内存的最佳实践。共享集群适用于交互式使用案例，例如在您使用 Jupyter Notebook 时。

使用共享集群时，建议您使用 Amazon EMR 中的动态分配设置来自动计算默认执行程序大小，并允许将不再使用的资源归还给集群。Amazon EMR 会默认启用动态分配。

每个作业的专用集群

为每个 Spark 作业使用单独的集群对于计划的 Spark 作业颇为有益。这种方法有助于隔离 Spark 作业以防止资源争用，可以根据作业属于 CPU 密集型、GPU 密集型还是内存密集型工作负载来优化作业，并确保仅在作业期间为所使用的资源付费。Amazon EMR `maximumResourceAllocation` 设置有助于确保整个集群的资源专用于相应作业。更多信息，请参阅《Amazon EMR 发行指南》中的使用 maximumResourceAllocation。

使用 Amazon EC2 Spot 实例优化成本

降低 EMR 上 Spark 作业成本的一种常用方法是使用 EC2 Spot 实例。使用 EMR 发布版本 5.9.0 或更高版本时，EMR 上的 Spark 包含一组功能，可确保在手动调整大小或自动缩放策略请求的情况下，Spark 能够妥善处理节点终止。[3] Amazon EMR 5.11.0 包含 `spark.decommissioning.timeout.threshold`，可在使用 Spot 实例时进一步提高 Spark 的弹性。因此，若您使用 Spot 实例，请确保您使用的是 Amazon EMR 发布版本 5.11.0 或更高版本。

使用实例队列

实例队列是 Amazon EMR 的一项功能，可让您基于一组特定的单位指定目标容量。这些单位可以代表核心、内存或任何所指。若您确知自己的 Spark 作业需要一定数量的资源，并且想要混搭使用 EC2 实例类型和可用区中的容量，则实例队列非常有用。使用实例队列，您可以选择一个 VPC 网络和一组 EC2 子网，然后该功能会在您选择的子网中搜索可用区，直至找到所需容量。

遗憾的是，实例队列不支持多个核心或任务组，也不支持自动扩展。若您具有需要这些功能之一的动态工作负载，则必须使用实例组。有关配置集群的更多信息，请参阅《Amazon EMR 管理指南》中的集群配置指南和最佳实践。

Spark 文件写入性能

在 Amazon EMR 5.14.0 中，默认 `FileOutputCommitter` 算法已更新为使用版本 2 而非版本 1。此更新可减少重命名(rename)的数量，从而提高应用程序的性能。任何要迁移到 EMR 的 Spark 应用程序都应使用最新的 Amazon EMR 版本来利用此更新和其他性能改进。此外，Amazon EMR 版本 5.20.0 还包括一个 S3 优化型提交程序，该提交程序默认处于启用状态。EMRFS S3 优化型提交程序是 OutputCommitter 的一种替代实现，专为在使用 EMRFS 时将文件写入 Amazon S3 而优化。

S3 Select 与 Spark

在某些情况下，将 S3 Select 与 Spark 结合使用可提高性能，并减少在 Amazon EMR 和 Amazon S3 之间传输的数据量。S3 Select 允许应用程序仅从对象检索数据子集。从 EMR 5.17.0 开始，S3 Select 支持 CSV 和 JSON 文件。若您的查询将原始数据集的一半以上过滤掉，且您在 Amazon S3 与 EMR 之间的网络连接具有良好的传输速度，则 S3 Select 可能适合您的应用程序。

EMRFS 一致视图

当多个相互依赖的 Spark 作业都需要将数据写入 S3 时，请确保一个作业写入的数据与后续作业读取的数据相同。由于 S3 数据一致性模型，若在大量写入后立即列出对象，则对象元数据可能并非总是可用。启用一致视图后，EMRFS 会维护一个预期位于 Amazon S3 中的对象的元数据存储，并为后续作业返回该元数据存储中列出的对象集。

> **注意：**一致视图适用于控制所有以 S3 为读写目标的一组相互依赖的作业链或应用程序，例如 S3 EMR 部署中的 Apache HBase。不要将其用于 S3 存储桶中所有对象的全局一致视图。

AWS Glue 数据目录

对于 Amazon EMR 5.8.0 及更高版本，Spark 支持使用 AWS Glue 数据目录作为 Spark SQL 的元存储。若计划迁移到 EMR 上的 Spark，请首先确定是否可以将现有的元存储库迁移到 AWS Glue。使用 AWS Glue 数据目录的好处不仅在于将其用作托管元数据目录，而且还能集成许多不同的 AWS 产品，包括 Presto 和 EMR 上的 Apache Hive、Amazon Athena、Amazon Redshift Spectrum、Amazon SageMaker 和 Amazon Kinesis Data Firehose。与 AWS Glue 爬网程序结合使用，数据目录可以生成 schema 和分区，并提供对数据库和表的细粒度访问控制。

对 EMR 集群上的 Spark 作业进行问题排查

要调试常见问题，请查看以下应用程序的历史记录和日志文件：

- 对于 Spark Web UI，请通过 EMR 集群主节点访问 Spark HistoryServer UI，端口号为 18080。 更多信息，请参阅访问 Spark Web UI。
- 对于 YARN 应用程序（包括 Spark 作业），请访问 Amazon EMR 控制台中的**应用程序历史记录**选项卡。应用程序历史记录最多保留 7 天，包括有关 Spark 任务和阶段完成情况的详细信息。更多信息，请参阅查看应用程序历史记录。
- 默认情况下，通过控制台启动的 Amazon EMR 集群会自动将日志文件存档到 Amazon S3。您可以在 Amazon S3 中找到原始容器日志，并在集群处于活动状态以及终止后查看它们。更多信息，请参阅查看日志文件。

迁移 Apache Hive

Apache Hive 是深受 Amazon EMR 客户青睐的应用程序之一，常被用作数据仓库。与任何迁移一样，您需要考虑多个方面。

Hive metastore 元存储

默认情况下，Amazon EMR 集群配置为使用 MySQL 的本地实例作为 Hive metastore。为了最有效地使用 Amazon EMR，您应该使用共享的 Hive 元存储，例如 Amazon RDS、Amazon Aurora 或 AWS Glue 数据目录。[4] 若您需要一种持久性元存储，或者有一个由其他集群、服务、应用程序或 AWS 账户共享的元存储，我们建议您将 AWS Glue 数据目录用作 Hive 的元存储。更多信息，请参阅为 Hive 配置外部元存储。

升级

Hive 升级伴随着 Hive 元存储的更新。Hive 元数据数据库应备份，并与生产实例隔离，因为 Hive 升级可能会更改 Hive schema，这可能会导致兼容性问题和生产问题。您可以使用 Hive Schema Tool 中的 –`upgradeSchema` 命令执行升级。您还可以使用此工具将 schema 从旧版本升级到当前版本。

Hive 执行引擎

Apache Tez 是受 EMR 上的 Hive 集群支持的默认执行引擎。在大多数情况下，Tez 都能提升性能。但是，若您要从使用 MapReduce 执行 (MR) 引擎的较旧版本的 Hive 迁移，则可能需要更改某些作业。

Hive 2.3.0 添加了支持将 Spark 作为执行引擎的功能，但是若不更改 Spark 中的基础 Hive jar，则 EMR 不支持此设置。这不属于 EMR 上受支持的配置。

Tez 容器大小

若您有无法拆分的大型输入文件，或者作业的 map 部分超出了 container 的默认内存限制，则将需要更大的 Tez 容器。在 Amazon EMR 上，默认容器设置为 **-1**，也就是说，将使用 `mapreduce.map.memory.mb` 的值。该设置的默认值取决于为您的 EMR 集群选择的特定实例类型。有关此设置和其他默认值，请参阅任务配置。

所需配置的值取决于您作业的具体情况。该值必须至少与 `mapreduce.map.memory.mb` 设置相同。

若 Tez 容器耗尽内存，则会出现以下错误消息：

```
Container
[pid=1234,containerID=container_1111222233334444_0007_02_000001] is
running beyond physical memory limits. Current usage: 1.0 GB of 1
GB physical memory used; 1.9 GB of 5 GB virtual memory used.
Killing container.
```

HDFS vs S3 注意事项

EMR 的优点是能够通过将 Amazon S3 用作您的主要数据存储来分离存储和计算需求。这种方法允许您通过独立扩缩存储和计算需求来节省 HDFS 成本。Amazon S3 可提供无限可扩展性、高耐用性和可用性以及其他功能，例如数据加密和生命周期管理。即便如此，Hadoop 在设计时的预期是底层文件系统需支持原子重命名并且保持一致性。若您的工作流程要求列表操作 list 和"写入后读取"即时一致性，则有如下几种选项需要考虑。

EMRFS 一致视图

EMRFS 一致视图是一项可选功能，它允许 EMR 集群检查由 EMRFS 写入或与之同步的 Amazon S3 对象的列表 list 操作和"写入后读取"一致性。对于处理写入 S3 的链式 MapReduce 作业或 EMR 上的特定应用程序（例如控制所有对 S3 的读写操作的 HBase）而言，这是理想之选。

Hive Tez 合并文件

合并文件会执行大量操作，这些操作会导致 S3"写入后读取"一致性失效，并可能导致数据丢失。因此，我们建议您在未启用 EMRFS 一致视图的情况下不要使用 `hive.merge.tezfiles`。另一种选择是对基于 HDFS 的临时性 EMR 集群执行任何 `INSERT OVERWRITE` 或 `ALTER CONCATENATE` 语句，然后使用 `s3-dist-cp` 将结果复制回 S3。

EMR 客户解决大规模一致性问题的其他方法里还包括为其作业配置自定义 manifest 清单文件方法，而非使用 S3 列表操作来检索数据，或通过建立自己的元数据存储（例如 Netflix Iceberg）来解决此类问题。

Hive Blobstore 优化

Hive Blobstore 优化旨在提高 Hive 上中间 MR 作业的性能。但是，在使用基于 S3 的表执行简单的 Hive 查询时（例如 `INSERT OVERWRITE` 或 `ALTER TABLE CONCATENATE`），此设置有时会导致执行时间延长和数据丢失。此问题是由针对非 multi-stage MR 作业类型的查询启用该功能导

致的。这种情况会导致 Hive 在作业期间使用 Amazon S3 作为临时目录 scratch directory。在这种情况下，随着作业的推进，由于写入临时数据、复制到另一个 S3 临时位置以及最终复制到最终 S3 位置，S3 上的重命名数量会增加。

若您禁用此设置，则作业的临时目录 scratch directory 将重定位到 HDFS。若您更喜欢这种方案，请确保在 EMR 集群上分配充足的空间，以适应此更改。默认情况下，在该过程结束时执行的 `distcp` 作业会被限制为最多 20 个映射器（mapper）。若您发现此作业耗时过长（尤其是在处理 TB 级数据时），则可以在 Hive 作业中使用以下代码手动设置 mapper 的最大数量：[5]

```
SET distcp.options.m=500
```

作业吞吐量和计划

默认情况下，EMR 使用 Hadoop CapacityScheduler 在集群中分配资源。CapacityScheduler 允许大型集群共享，并为各组织提供容量保证。CapacityScheduler 支持可以允许组织在多租户集群中授予容量的队列。您可以通过在 EMR 集群创建期间修改 `capacity-scheduler` 分类来配置 CapacityScheduler。

在某些情况下，对于允许作业使用未用资源的集群，使用 FairScheduler 可能更合适。

配置 FairScheduler
创建 EMR 集群时，您可以通过几种方式配置 FairScheduler。

修改 yarn-config.json 文件

1. 在 `yarn-config.json` 文件中使用以下配置。

```
[
  {
    "Classification": "yarn-site",
    "Properties": {
      "yarn.resourcemanager.scheduler.class":
"org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.FairS
cheduler"
    }
```

```
    }
]
```

2. 修改 AWS CLI 命令以匹配以下代码示例，并使用此命令，通过上面创建的配置文件启动集群。

```
aws emr create-cluster \
        --applications Name=Spark Name=Ganglia \
        --ec2-attributes "${EC2_PROPERTIES}" \
        --service-role EMR_DefaultRole \
        --release-label emr-5.20.0 \
        --log-uri ${S3_LOGS} \
        --enable-debugging \
        --name ${CLUSTER_NAME} \
        --region us-east-1 \
        --instance-groups \
        --configurations file://yarn-config.json
InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m3.xlarge
InstanceGroupType=CORE,InstanceCount=4,InstanceType=m3.xlarge)
```

在控制台中编辑软件设置

1. 登录 AWS 管理控制台，并通过以下网址打开 Amazon EMR 控制台：https://console.aws.amazon.com/elasticmapreduce。
2. 选择**创建集群、转到高级选项。**
3. 选择 **Spark。**
4. 在**编辑软件设置**下，将**输入配置**保留为未选中状态，并输入以下配置：

```
classification=yarn-
site,properties=[yarn.resourcemanager.scheduler.class=org.apache.ha
doop.yarn.server.resourcemanager.scheduler.fair.FairScheduler]
```

5. 选择**创建集群。**

维护基于 Hive 的高可用性集群

由于一次性查询具有随时产生的性质，Hive 集群通常会长时间运行。尽管 Amazon EMR 集群主节点丢失的情况并不常见，但在云环境中，仍然存在这样的可能性。维护基于 Hive 的高可用性集群的方法有两种。

热故障转移

在热故障转移方案中，除了主集群之外，还要持续运行一个较小规模的辅助集群。若发生故障，可以采用手动方式或者通过更新 Amazon Route 53 中的条目，将客户端重定向到新集群。您可以为辅助集群配置少量节点，后续若该集群转为主集群，请使用自动扩展功能来增加节点数量。

多集群配置

因为所有数据都存储在 S3 上，所以并非所有客户端都需要通过同一集群处理。您可使用负载均衡器配置多个集群，也可向环境的使用者公开作业提交框架。这些集群可以由所有客户端共享，也可以根据内部需求在每个团队中共享。这种方法的好处之一是，若发生集群故障，则对客户群的总体影响仅限于在发生故障的具体集群上执行的那些查询。另外，您可以配置自动扩展，让每个集群可以彼此独立地进行容量扩展。若按团队对集群进行细分，则此方法可确保任何一个团队的作业都不会影响另一个团队的作业性能。

但是，使用多个集群就意味着使用多个主节点（每个主节点用于一个集群）。因此，您需要额外的 EC2 实例，而在您仅使用单个集群时，不需要为使用这些额外的实例付费。但是，使用 EC2 实例的按秒计费（至少 1 分钟）定价模式，在使用多个集群的情况下，您可以选择仅激活执行任务所需的集群，而非始终运行单个集群，从而实现成本节省。您可以在 AWS Lambda 函数（在管道上调用激活的函数）中配置这种设置的逻辑。进而实现启动或关闭集群且不影响其他集群的活动。

临时设计

临时集群可以降低拥有单个长期运行的集群的成本和运营要求。若您具有可预测的短期作业，则这种方法非常有用，但是若您的用户持续不断地查询数据，则这种方法可能不合适。

注： 2019 年 5 月份 EMR 宣布自版本 5.23.0 开始，已在支持 Amazon EMR 的所有区域推出支持多主节点功能。您现在可以启动包含三个主节点的 EMR 群集以实现集群内 HBase， YARN Resource Manager、HDFS Name Node、Spark、Hive 和 Ganglia 等应用的高可用性。请访问计划和配置主节点，了解有关此功能的更多信息。

常见问题

如何在 Amazon EMR 上实施事务 transactions 和压缩 compactions?

尽管尚未官方宣称在使用 S3 作为主要数据存储的 EMR 集群上被支持，但是若您当前启用了 Hive 事务，请联系您的客户团队或支持人员，并根据您的特定需求进一步研究如何使用 S3 在 EMR 上实现该功能。

如何排查从 Amazon S3 加载数据的问题?

一种常见的错误做法是像使用典型文件系统一样使用 Amazon S3。若要从 HDFS 迁移到 Amazon S3，则必须考虑两者之间存在的一些差别。有关详情，请参阅[您在向 Amazon S3 装载数据或者从中移出数据时是否遇到问题？]。

提供 Ad Hoc 查询功能

Presto 注意事项

在 Amazon Athena 与 PrestoDB 之间选择

Amazon Athena 是一种无服务器交互式查询引擎，可对驻留在 Amazon S3 中的数据执行 SQL 查询。许多客户都将 Athena 用于各种使用案例，包括交互式查询数据以探索数据，支持利用 S3 中保存的运营指标的控制面板，以及支持 Amazon QuickSight 或 Tableau 等可视化工具。对于这些类型的工作负载，我们强烈建议您考虑使用 Amazon Athena。Athena 易于集成，具有多种功能（例如成本管理和安全控制），并且几乎不需要容量规划。所有这些特征都能降低运营负担和成本。但是，在某些使用案例中，PrestoDB 可能比 Amazon Athena 更适用。例如，请考虑以下优先事项：

- **降低成本**：若降低成本是您的主要目标，我们建议您估算这两种方法的成本。您可能会发现，使用基于 Amazon EMR 的 Presto 运行时，加载和查询模式的成本更低廉。考虑成本的增加（若有）是否超出了在 EMR 上运行和维护具备扩展能力和出色可用性的 Presto 的益处，再与 Amazon Athena 提供的功能进行对比。
- **性能要求**：若您的使用案例对性能高度敏感，请选择调整 Presto 集群以满足性能要求。
- **关键功能**：若有 Amazon Athena 当前未提供的功能（例如为自定义数据类型使用自定义序列化器/反序列化器），或者当前不支持的数据存储连接器，则基于 EMR 的 Presto 可能更合适。

有关 Athena 和 Presto 的性能提示和最佳实践，请访问 AWS 大数据博客，参阅 Top 10 Performance Tuning Tips for Amazon Athena。

元数据管理

将 AWS Glue 作为数据目录

从 Amazon EMR 版本 5.10.0 起，Amazon EMR 可以将 AWS Glue 数据目录用作 Presto 的默认 Hive 元存储。有关此方法的更多信息，请参阅数据目录迁移一章。该章中介绍的益处也适用于在 Amazon EMR 上运行的 Presto。

将 AWS Glue 数据目录与基于 Amazon EMR 的 Presto 结合使用时，授权机制（例如 Hive SQL 授权）将被基于 AWS Glue 的策略所取代。

您还需要单独保护 Amazon S3 中的基础数据。您可以使用 S3 存储桶策略或 AWS IAM 策略保护这些数据。您可能会发现使用 IAM 策略更加有效，因为您可以集中管理 Amazon S3 和 AWS Glue 数据目录的访问。

有关使用 AWS Glue 数据目录的更多信息，请访问 AWS 大数据博客，参阅 Easily manage table metadata for Presto running on Amazon EMR using the AWS Glue Data Catalog。有关 Presto 与 AWS Glue 数据目录之间的交互的现有限制，请参阅使用 AWS Glue 数据目录时的注意事项。

EMRFS 和 PrestoS3FileSystem 配置

默认情况下，在 Amazon EMR 发布版本 5.12.0 或更高版本上运行的 Presto 可以使用 EMRFS 访问 Amazon S3 中的数据。在 EMR 的早期版本上运行的 Presto 只能使用 PrestoS3FileSystem（Hive 连接器的一个组件）。

通过 EMRFS 访问数据使您可以在 EMR 安全配置中配置 Amazon S3 加密要求。使用 EMRFS 时，您还可以在 Presto 集群中使用单独的 IAM 角色，以根据用户、组或 Amazon S3 位置控制对数据的访问。

对于 EMR 发布版本 5.12.0 或更高版本，可以从使用 EMRFS 切换到使用 PrestoS3FileSystem。若您的组织仍然依赖 RDMBS 上运行的基于 Hive SQL 的授权和 Hive 元存储，则这种方法可能会很有用。有关在 Amazon EMR 上配置 PrestoS3FileSystem 的其他详情，请参阅 EMRFS 和 PrestoS3FileSystem 配置。

Amazon EMR 上的 HBase 工作负载

使用 Amazon EMR 的 Amazon S3 上的 Apache HBase 可以为涉及到大量读取操作的工作负载节省大量成本。[6] 由于可以使用 Amazon S3 作为主要存储机制，因此 HBase 集群会比典型的 HBase 部署规模小得多。

有关调整 HBase 以使之在 Amazon S3 上实现最优性能的详情，请访问 AWS 大数据博客，参阅 Migrate to Apache HBase on Amazon S3 on Amazon EMR: Guidelines and Best Practices。

HBase 升级

对于 HBase 升级，我们建议您同时运行较新版本的 HBase 和先前版本，直到完成所有测试。从 HBase 获取快照，并使用该快照启动采用升级版本的新集群。您可以同时运行两个集群，并同时通过两个集群执行更新和读取。测试完成后，您可以关闭旧集群。若遇到任何需要回滚升级的问题，则可以回滚到旧集群。

在 HBase on S3 上优化批量加载 bulk load

使用 EMR 将现有集群迁移到 HBase on S3 上时，一种常见方法是使用批量加载。尽管这种方法可以有效启动新集群，但还有几种方法可以优化批量加载，以实现最佳性能。

根据您的 HBase 版本以及生成的 StoreFile 的位置，执行批量加载的命令应类似于以下代码：

```
hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles \
  <s3://bucket/storefileoutput/> <tablename>
```

在节点上运行该命令时，系统会执行以下步骤。

1. 在 storefileoutput 位置列出可用 StoreFile
2. 确定 StoreFile 应该存储到哪个 region，以及该 StoreFile 是否适合给定 region。
3. 若该 StoreFile 过大，不适用于给定 region（根据 `hbase.hregion.max.filesize` 确定），或者 region 自文件创建之后已进行 split 拆分，则主节点会将 StoreFile 拆分为两个文件，并将这两个新文件添加回要处理的 StoreFile 列表。

4. 对于每个有需要加载的 StoreFile 的 region，系统都会向必要的 RegionServer 发出请求，以启动 BulkLoad 操作。
5. RegionServer 将 StoreFile 复制到目标文件系统，然后将 StoreFile 加载到 HBase。

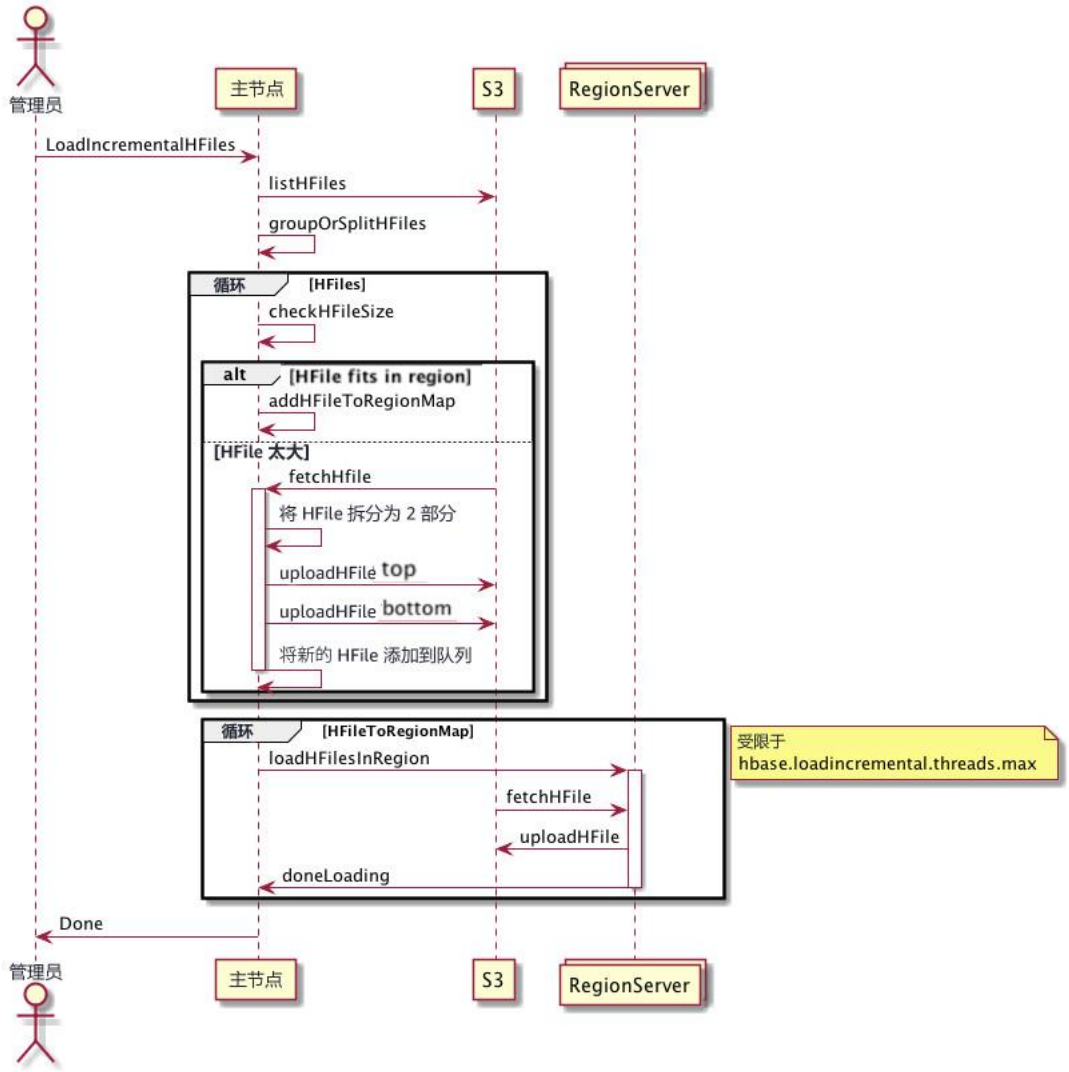


图 40：CompleteBulkLoad 流程概述

与任何分布式系统一样，此流程的性能取决于可用的 CPU 和网络带宽。

优化 S3 上传

您可以通过调整大小设置来优化 S3 上传：

- **fs.s3.threadpool.size**：默认情况下，此设置为 20，但您可以增大此值，以增加所执行的并行分段上传的数量。要增大此设置值，您可能需要使用 CPU 数量更多的实例。
- **fs.s3n.multipart.uploads.split.size**：上传大文件时增大设置值，以避免达到分段上传限制。

增加主节点上的线程数量

主节点（或在其上运行 LoadIncrementalHFiles 的任何节点）执行两个主要步骤：拆分 StoreFile（若它们不适合 Region），以及与每个负责的 RegionServer 协调处理 BulkLoad。这些操作在线程池中发生，默认情况下该线程池的大小为节点上可用 CPU 的数量。例如，若您的主节点是只有两个 vCPU 的小型 m4.large 节点，且有两个 StoreFile 必须要 split 拆分，则您的整个批量加载过程就可能会被阻塞。您可以通过两种方法解决此问题：

- 使用配备更多 vCPU 的更大节点。
- 运行作业时定义 `hbase.loadincremental.threads.max` 变量

但是，即使您有 100 个 RegionServer，LoadIncrementalHFiles 命令也只能使用 `hbase.loadincremental.threads.max` 变量中定义的数量。若您的节点有八个 vCPU，则仅使用八个区域服务器。您可以将该变量增加到您拥有的 RegionServer 的数量，但是若您的主节点必须拆分 StoreFile，则可能会很快陷入困境。

增大 RPC 超时设置

若您生成了较大的 StoreFile（10GB 以上）或某个特定 RegionServer 的 StoreFile 数量繁多，则由于 CallTimeoutException，LoadIncrementalHFiles 命令有时可能会返回"Error connecting to server"消息。这种行为属于正常现象，但是该问题会导致多次尝试执行 BulkLoad，且往往会导致加载失败。

要解决此问题，您可以在开始工作时使用 `hbase.rpc.timeout` 变量来增大 RPC 超时设置。

监控批量加载

在运行批量加载时，HBase UI 未包含足够的指标来严密监控批量加载。为了更好地了解批量加载，请使用 Ganglia 启动 EMR 集群。除了常规的 CPU 和网络指标，Ganglia 还包含许多 HBase 指标。具体来说，`regionserver.Server.Bulkload_count` 可以显示每个 RegionServer 上完成了多少次批量加载。

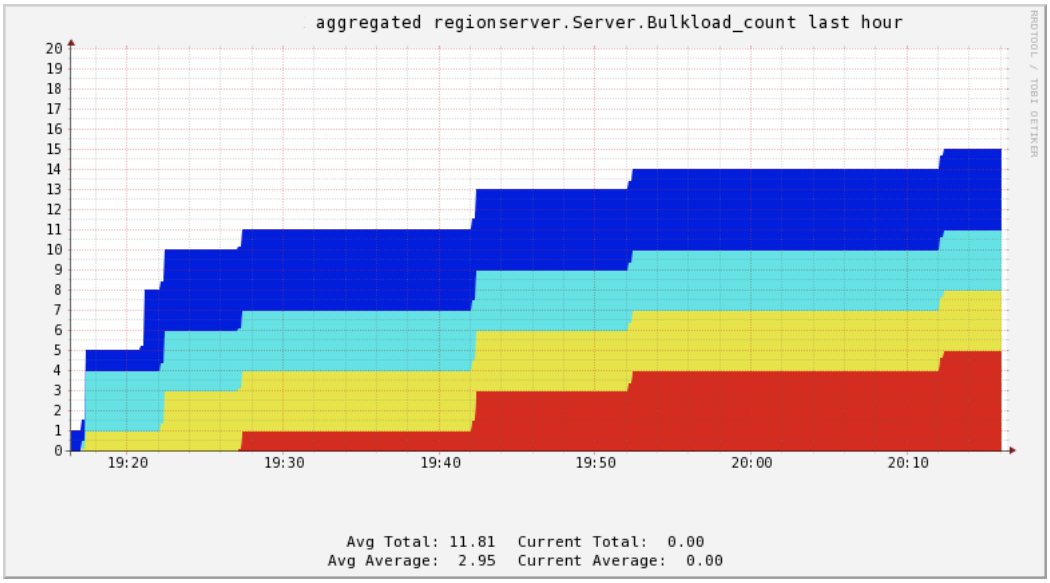


图 41：Ganglia 区域服务器批量加载计数

调试批量加载

若遇到问题，您可以在 HBase 上启用调试以获取有关这些问题的更多详情。要启用调试，请将 `log4j.logger.org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles` 变量设置为 DEBUG。

优化策略摘要和示例

本节包括三种优化批量加载的方法：

- 充分利用 Amazon S3 和网络带宽。
- 充分利用主节点上的 CPU。
- 更改预期工作负载的 HBase 设置。

对于上述所有配置变量，您都可以将它们指定为 LoadIncrementalHFiles 的命令行变量，也可以使用各种 Hadoop 配置文件在集群范围内配置这些设置。请参阅以下示例。

增加线程和 RPC 超时

以下命令显示将线程数设置为 20，将 HBase RPC 超时设置为 10 分钟。

```
hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles \
  -Dhbase.loadincremental.threads.max=20 \
```

```
  -Dhbase.rpc.timeout=600000 \
  <s3://bucket/storefileoutput/> \
  <tablename>
```

定义 HBase on S3 设置并启用调试

若您通过 AWS CLI 创建集群，下面的 JSON 文件显示了一个典型的配置文件。该文件可定义 HBase on S3 设置，并在两个 HBase 类上启用 DEBUG 和 TRACE 日志记录。

```
[
    {
        "Classification": "hbase",
        "Properties": {
            "hbase.emr.storageMode": "s3"
        }
    },
    {
        "Classification": "hbase-site",
        "Properties": {
            "hbase.rootdir": "s3://<bucket>/<hbaseroot>",
            "hbase.hregion.max.filesize": "21474836480",
        }
    },
    {
        "Classification": "hbase-log4j",
        "Properties": {

"log4j.logger.org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFil
es": "DEBUG",
            "log4j.logger.org.apache.hadoop.hbase.ipc.RpcServer":
"TRACE"
        }
    }
]
```

迁移 Apache Impala

若您正在运行 Apache Impala 并希望将其迁移到 Amazon EMR，建议您使用 Amazon Athena 或 Presto。有关如何选择 Amazon Athena 与 PrestoDB 的更多详情，请参阅迁移 Presto。

若由于 Presto 或 Athena 未能涵盖某个使用案例而必须使用 Impala，则可以使用以下三个选项来安装 Impala：

- 在 Amazon EC2 上手动安装 Impala。
- 使用引导操作在 Amazon EMR 上安装 Impala。
- 借助第三方云提供商安装和管理 Impala。

由于 Impala 不是托管应用程序，因此 AWS Support 和 Amazon EMR 服务团队无法为 Impala 安装提供支持。

卓越运营

升级 Amazon EMR 版本

定期升级 Amazon EMR 版本的一种最佳实践。升级集群软件可确保您使用开源应用程序中最新最强大的功能。以下是使用最新软件升级的一些益处：

- 性能增强可以提升应用程序运行速度。
- 错误修复可使基础设施更加稳定。
- 安全补丁可帮助确保集群安全。

这些益处既适用于开源应用程序软件，也适用于管理开源软件所需的 Amazon EMR 软件。

下图是 2018 年 7 月至 2018 年 12 月期间 Amazon EMR 5.x 软件版本和相应的开源应用程序版本的示例。在撰写本文档时，Amazon EMR 大约每 4 至 6 周发布一次新版本，其中包含该软件的最新版本。有关完整的版本列表和发布说明，请参阅 Amazon EMR 5.x 发布版本。

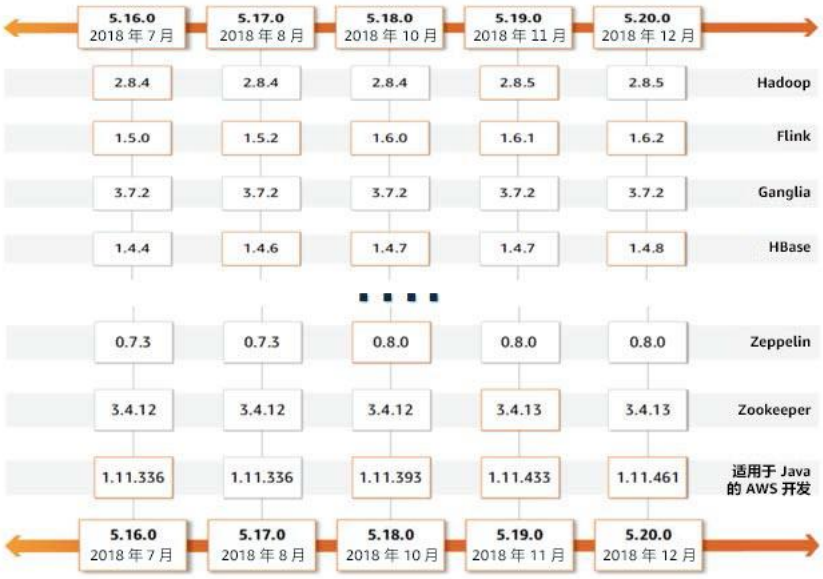


图 42：Amazon EMR 5.x 发布版本示例

有关修补 Amazon EMR 集群上的软件的适当时机，请参阅软件修补。

升级流程

升级软件时，在性能和数据质量方面存在回归(regression)风险。升级可能会更改 API 接口，导致您的代码无法再正常运行，就像换用新框架一样。升级还可能会引入新的错误，这些错误可能会导致应用程序运行失败。在发布 Amazon EMR 之前，AWS 会通过运行大量集成测试尽最大努力识别开源软件中的 regression 问题，但有些 regression 问题可能难以识别。因此，必须在将每个版本提供给用户之前对其进行测试。但是，升级越频繁，版本之间的更改数量就越少，这会减少升级的工作量，同时也能降低 regression 风险。

建议的升级步骤

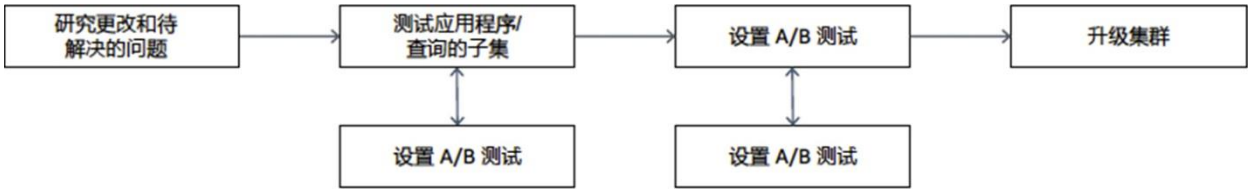


图 43：建议的升级步骤

研究更改和待解决的问题

所有开源应用程序都会提供发布说明，并且大多数还会提供 JIRA 以跟踪问题。在执行升级之前，您可以开展研究以查明是否存在错误、问题或配置更新，从而节省时间。

表 5 列出了常见的开源应用程序、其发布说明和问题跟踪系统。对于 Amazon EMR，请参阅 Amazon EMR 5.x 发布版本。

表 5：发布说明和问题跟踪的应用程序链接

应用程序	发布说明	问题跟踪
Hadoop	https://hadoop.apache.org/old/releases.html	https://issues.apache.org/jira/projects/HADOOP/issues
Hive	https://hive.apache.org/downloads.html	https://issues.apache.org/jira/projects/HIVE/issues
Spark	https://spark.apache.org/releases/	https://issues.apache.org/jira/projects/SPARK/issues

应用程序	发布说明	问题跟踪
Presto	https://prestodb.github.io/docs/current/release.html	https://github.com/prestodb/presto/issues
HBase	https://hbase.apache.org/downloads.html	https://issues.apache.org/jira/projects/HBASE/issues

测试应用程序/查询的子集

在用户测试新版本或配置之前，我们建议您使用一部分能代表整体使用情况的使用案例来测试相应版本。这种方法可确保在部署之前发现任何配置问题。

修复问题

若在测试版本时发现问题，请执行以下步骤：

1. 检查是否可以通过某个配置值解决该问题。例如，看看您能否使用某个配置值来禁用有问题的新功能或增强功能。
2. 检查问题是否已经确定并已在社区更新版本中得到修复。若更新版本中包含相应修复，请通知 AWS Support 工程师。AWS 将评估是否可以将其包含在我们的下一个版本中。
3. 更改应用程序或查询以避免该问题。
4. 联系 **AWS Support**，看看是否有任何解决方法。
5. 若没有解决方法，请放弃升级，然后等待具有必要修复程序的版本。

设置 A/B 测试（建议）

下一步是逐步将工作负载迁移至新配置。若在生产环境中发现严重问题，这种方法可让您选择中止升级。若您使用 Amazon EMR 处理交互式用户查询，则设置路由有助于以受控方式 (*图 44*) 将负载从一个集群转移到另一个集群。您还可以使用支持流量加权和粘性会话的负载均衡器。

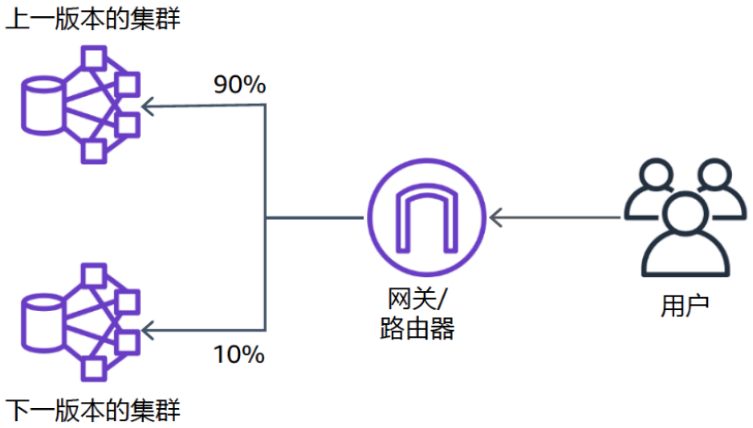


图 44：使用路由逐步迁移负载

完成升级

通过将所有 Amazon EMR 集群迁移至新版本来完成升级。最后，停止使用旧版本。

升级最佳实践

- 升级需要付出时间和精力 – 务必确保您的团队妥善安排升级计划，并留出时间来完成升级。
- 请注意，升级时，某些可能会发生变化的依赖关系。
- 在执行手动测试时，请复制 Hive 元存储以确保 schema 向后兼容。
- 若可以，请跟踪作业的性能以确保未发生显著 regression 回归影响。
- 按应用程序拆分集群。这种方法允许您单独升级组件，而非整体升级。
- 研究版本之间的变化，以便更容易地发现问题。
- 使用 Amazon Route 53 自动注册集群。这种方法可使用户更容易地找到它们。有关设置 Amazon Route 53 的更多信息，请访问 AWS 大数据博客，参阅 Dynamically Create Friendly URLs for Your Amazon EMR Web Interfaces。

卓越运营的通用最佳实践

将所有 EMR 集群配置为临时性集群

将所有 Amazon EMR 集群配置为临时性集群具有诸多益处。临时性集群即启动执行一些工作后关闭的集群。如要将集群设为临时性集群，则不能在集群上保留状态，因为在关闭集群时会删除状态。这种方法可使灾难恢复和 Amazon EMR 升级更加轻松快捷，，并允许您分离存储和计算资源以进行独立扩展。

将预置资源自动化

使用用于调用 AWS 命令行界面 (AWS CLI) 或开发工具包代码的 AWS CloudFormation 脚本自动预置 EMR 集群。这种方法可使用户一致地创建集群，并且更轻松地跟踪集群更改。

确保对所有代码、脚本和其他构件进行源代码管理

通过确保所有代码、脚本和其他构件都存在于源代码管理系统(source control)中，您可以随时间推移跟踪变化情况。
此外，您可以跟踪可能影响您的环境运作的更改。这种方法还允许您在不同部署阶段（例如测试版环境）中部署构件，并降低不良 artifact 从一个阶段转入另一个阶段的几率。

监控集群中的异常行为并采用自动扩展

具体来说，请务必注意长时间运行或卡住的作业以及磁盘或 HDFS 容量问题。

考虑使用 AWS Glue、Amazon Redshift 或 Amazon Athena

虽然 Amazon EMR 具有出色的灵活性且可以提供最大程度的定制和控制，但是管理 Amazon EMR 集群、升级等任务也存在相关成本。考虑使用其他符合您要求的托管 AWS 服务，因为它们通常运营负担较低，并且在某些情况下还可以降低成本。若这些服务均无法满足您的使用案例要求，请使用 Amazon EMR。

测试和验证

在制定从源系统迁移数据的流程并运行分析作业之后，确保作业获得良好的数据并定义数据规则将变得更加重要。

单元测试通常是为代码编写的，但是数据质量测试却经常被忽略。错误或格式不当的数据可能会对生产系统产生影响。数据质量问题包括以下情形：

- 缺少值可能会导致需要非空值的生产系统发生故障。
- 数据分布的变化可能会导致机器学习模型产生意外输出。
- 错误数据的聚合可能会导致在信息不充分的情况下做出业务决策。

本章将介绍多种数据质量检查方法，但是此处介绍的方法并非详尽无遗。许多第三方供应商和合作伙伴都会提供有关数据分析和数据质量的解决方案，不过这些内容不在本文档的讨论范围之内。此处介绍的方法和最佳实践可以并且也应该用于解决数据质量问题。

数据质量概述

数据质量本质上是对数据满足用户期望的情况的一种度量。换句话说，它是数据相对于其要建模的结构或对象的正确性。由于从数据整体来考虑，这是一个很难定义的概念，因此我们从以下维度角度考虑数据质量：

- **完整性**衡量数据的全面程度。
- **唯一性**关注数据中的重复项。
- **及时性**可确保您的数据是最新的，并且在您的需求/SLA 范畴内可用。
- **有效性**衡量您的数据符合已定义的架构、语法和要求的程度。
- **准确性**衡量在要求的范围内表示对象时，您的数据的正确性。
- **一致性**评估对于同一对象的引用或对象之间的关系，数据是否存在差异。

这不是完整的数据维度列表。组织可以使用其他指标和属性，作为自身数据质量要求的维度。

检查您的提取管道

数据完整性可被视为数据质量检查的关键部分。要检查的最重要区域之一是提取的原始数据是否正确。

为了验证数据是否已从源系统正确迁移到目标系统，您首先应了解您的工具用于确保数据有效性的现有机制。例如，假设您在使用 Apache Sqoop。Sqoop 通过使用 `--validate` 标志来验证数据复制作业，该标志会在源和目标之间执行行计数比较。例如：

```
$ sqoop import --connect jdbc:mysql://db.foo.com/corp --table
WIDGETS -validate
```

但是，Sqoop 执行此验证的方式存在各种限制。Sqoop 不支持以下情况：

- 全表选项(all-tables option)
- 自由格式查询选项 (free-form query)
- 导入到 Hive 或 HBase 的数据
- 使用 `--where` 参数的表导入操作
- 增量导入

尽管其中部分问题可以缓解，但根本性问题在于，Sqoop 基本上仅执行行计数比较。为了验证所传输的每个值，您必须使用 Sqoop 反转数据流，并将新复制的数据推回源系统。在进行此反向推送之后，您可以将旧数据与新数据通过哈希值或校验和 checksums 进行比较。这种操作成本高昂，因此，在这种情况下，您必须考虑风险接受程度和数据策略。

在迁移过程中，您必须确定与数据准确性有关的确定性级别，并为此制定一个相应的可量化指标。要求的严格程度决定了迁移策略的成本、复杂性和性能影响，并可能影响整体解决方案的性能。此外，根据可接受风险的级别，可能有其他替代方法（例如抽样或应用程序级别验证）可行。

另一个示例是 AWS CLI，在使用它将数据从源系统传输到 Amazon S_3 时，往往会出现数据质量（特别是完整性）方面的问题。务必要了解传输目标的特征以及 AWS CLI 和其他工具如何帮助您验证所复制的数据。这样，您就可以制定合理的数据质量目标，并彻底解决和回答架构团队与业务拥有者的问题。

以下列表是使用 AWS CLI 和其他数据移动工具时出现的一些常见问题。具体包括：

- 当对象传输到 Amazon S3 中时，AWS CLI 是否会对这些对象进行验证？
- AWS CLI 如何验证对象？
- Amazon S3 是否会公开所用的校验和 checksum？
- 若在 Amazon S3 中采用分段上传，会发生什么情况？ 如何计算部分及整个对象的校验和？
- 如何不依靠 Amazon S3 ETag 验证已上传对象的校验和是否正确？

整体数据质量策略

通常，数据源以及在构建和测试初始分析作业和模型时用于提取数据的流程与生产环境中用于分析或推理的数据源及相应流程有所不同。例如，为了方便快捷地进行访问，初始研究（探究"是否可能？"的问题以及数据发现）常常针对从数据湖或已经建立好的数据源中提取、经过清理、简化并且可能经过充实的样本数据执行。这里隐含的假设在于，在生产环境中操控的数据是相同的，可以足够快地提供这些样本数据以执行操作，但这种方案并非始终适用；尤其是在开发与生产环境之间存在严苛界限的环境中更是如此。您应该对这种假设进行测试，确保分析作业或机器学习模型能够按预期工作。建议您创建用于构建模型的数据管道流程图解，以显示所有数据的来源及转换方式。此图表可帮助您确定潜在挑战。

另外，同时建议创建要在生产中使用的数据管道和数据目录的流程图解。在次之上注意观察两个管道之间的异同。

- 若两者相同，则很可能会遇到相同的错误。这些错误是否重要？
- 若两者不同，则表示采用了不同的来源或不同的处理。这些差异是否会影响您的分析作业或 ML 模型？ 您如何确知这一点？

与相关主题专家和项目发起人共享图解和摘要。讨论差异，并就它们与所有利益相关者达成合理共识。若发现两个管道之间的差距过大（主观评估），请考虑采用能更好地代表生产环境中的数据的其他方法来获取数据。

涉及的数据源越多，要合并的数据源就越分散。同样，涉及的数据转换步骤越多，数据质量挑战就越复杂。

评估数据质量的影响

在初始分析作业期间，最开始往往是使用经过清理的数据（例如，来自数据湖的数据），或者在运行初始作业之前清理数据。例如：

- 合并数据时，若未找到直接的关键字匹配项，则数据可能会被丢弃。
- 具有空值或极值的记录可能会被删除。

通常，在将数据用于分析或 ML 训练之前，需要执行许多单独的清理和转换步骤。但是，在生产环境中使用作业或模型时，所使用的数据通常来自不同的来源（即生产数据），并通过生产数据产生的路径到达模型的推理终端节点。您的分析作业、ETL 工作流程或 ML 模型是期望工作于妥善处理的经过清洗的开发或测试数据。为确保您的作业在生产环境中的表现与在较低层环境中的表现相同，请考虑比较统计数据，并利用未清理过的数据输入验证模型。

比较统计数据

为确保分析作业或模型在生产环境中表现良好，请添加正式的检查点，以将源输入数据与实际用于训练的数据作业模型进行比较。确保从定量和定性的角度评估数据。

定量评估

在定量评估中，除了已定义的任何其他可量化指标外，还应检查计数、数据时间跨度和数据输入的精确度(precision)。

比较*计数*可以以识别、跟踪和突出显示数据丢失情况，并根据合理的条件进行测试。实际构建和测试模型使用了多少百分比的源数据？ 合并中意外丢弃的数据是否会导致任何潜在的偏差？ 一段时间之后（例如，对于日志消息），数据存储子系统是否会对结果进行过滤、求平均值或聚合处理？

检查*数据时间跨度*和保留时间，以确定每个数据集覆盖的时间段。是否包括了所有可能相关的业务周期，尤其是在高峰负载期间？

通过比较数据源和用于训练模型的数据的平均值、中位数和标准偏差来量化*数据精确度*。计算离群值的数量或百分比。对于低维数据或关键变量，箱形图可以快速直观地评估合理性。通过这种方式

进行精确度量化，可以帮助您确定"期望"值的范围，并建立警报，以便在有超出此范围的数据时通知您。

定性评估

准确度(accuracy)与精确度(precision)同等重要，但或许只能进行定性评估。例如，根据经验和样本探索，您对数据的准确度有多大把握？ 是否有足够多的异常注解？ 例如，操作员是否报告此传感器始终处于高负荷运行状态？ 根据这种评估可以采取的措施多种多样。常见的结果是根据分析期间发现的某种因素对数据进行细分，并针对每个细分子集采取不同的措施。例如，在对智能运输系统的一项研究中，经识别发现一组传感器配置错误，随后将这组传感器送修，并使用另一组子集执行交通分析。[7]

根据未经清理的数据输入验证模型

验证数据模型的一种简单而强大的技术是从原始数据中获取在每个清理或转换步骤中删除的数据子集，并将其与最终用于运行作业的数据进行比较。然后，评估输出结果。终端节点是否在所有情况下均提供合理的响应？ 使用结果来确定应在何处添加检查和错误处理机制。是否应为推理终端节点添加错误处理机制？ 还是应该要求调用作业的应用程序识别和删除有问题的输入，或处理有问题的输出？

有助进行数据质量检查的工具

您不必自行实施所有这一切数据质量检查操作，这里的一些供应商和开源工具可帮助您处理数据质量流程：

Apache Griffin

Apache Griffin 是适用于大数据的开源数据质量解决方案，支持批处理和流式传输模式。它提供了一种统一的流程，可以从不同的角度衡量您的数据质量，从而帮助您构建值得信赖的数据资产，增强您对业务的信心。

您可以将 Griffin 与 StreamSets 或 Apache Kafka 之类的工具结合使用，以获得可以为您执行数据质量检查的端到端流式传输工作流。

Deequ

Deequ 是一个在 Apache Spark 的基础之上构建的库，用于定义"数据的单元测试"，可以衡量大型数据集中的数据质量。它使您可以计算自己的数据集的数据质量指标，定义和验证数据质量约束，并了解数据分布的变化。您可以集中精力描述数据应有的呈现方式，而不必自行实施检查和验证算法。Deequ 是在 Apache Spark 的基础之上实现的，旨在随通常位于分布式文件系统或数据仓库中的大型数据集进行扩展。

支持您的迁移

AWS 专业服务

AWS 专业服务提供全球专业实践服务，以支持企业云计算的重点领域。这些专业实践服务通过解决方案、技术和行业主题领域的最佳实践、框架、工具和服务提供有针对性的指导。他们拥有深厚的专业背景，可帮助您充分利用 AWS 云提供的业务和技术优势。

具体到 Amazon EMR，数据分析实践的领域专业知识可帮助组织充分利用 AWS 服务，从其数据资产中获得更多价值。具体到 Hadoop 迁移，AWS 专业服务具备规范且经验证的方法。这种方法包括协调阶段和启动阶段，这两个部分将在以下各节中进行介绍。

协调从 Hadoop 到 Amazon EMR 和 Amazon S3 数据湖（及 AWS Stack）的迁移

AWS 专业服务团队将与您合作，以了解并记录您的当前环境以及预期的未来成果。尽管此阶段的工作范围由双方协商确定，但以下列表给出了建议在 Hadoop 迁移协调阶段执行的活动。

- 了解业务：确认业务驱动因素和成功指标。
- 识别和记录价值驱动因素，以支持未来架构决策。此时应考虑内部 SLA 目标以及现有和新的业务价值。
- 数据优先级：使用业务驱动因素和技术考虑事项作为既定工作范围的优先排序标准。
- 数据评估：评估端到端数据流架构，以提供未来 Hadoop 环境相关建议，包括源和目标、文件格式等。
- 理解技术：解构 Hadoop 工作负载，以确定到 Amazon EMR 的迁移路径。
- 现有应用程序（例如，Impala 到 Presto）或使用案例（例如，数据科学 notebooks）对应的动手实验
- 安全深度调研（Ranger、Sentry、Kerberos、LDAP、IAM）
- 确认 AWS Glue 或 Amazon RDS 数据目录选项。

- EMR 服务计价练习/what-if 假设分析
- 提供实现到 Amazon EMR 的迁移详细的工作计划。

考虑到 Amazon EMR 对 Amazon S3 的依赖性，协调阶段第二部分的工作重点是 S3 数据湖。

- 介绍 AWS Lake Formation 服务，说明包含无服务器功能的架构已面向未来做好准备（若客户感兴趣）。服务包括 Amazon QuickSight 和 AWS Lake Formation 以及无服务器服务 AWS Glue 和 Amazon Redshift Athena。
- 提供有关数据摄取架构的建议。
- 提供有关满足安全性、控制和访问要求的数据存储架构方法的建议。
- 提供有关数据目录方法的建议。
- 为下游应用程序或用户提供有关数据服务层的建议，以供其自助服务式访问目录并执行数据探索。
- 为平台的最终用户使用提供建议，以帮助其通过轻松、可重用的方式完成启用。
- 提供有关成本优化型数据湖的计算和存储分离的建议。

Hadoop 迁移协调和 S3 数据湖产出：

- 识别多达五个使用案例
- 终稿将需求与架构设计协调一致的最终架构文档
- 详尽的迁移路径，支持执行 EMR 启动工作
- AWS 服务成本估算 *（平台支出）*
- 根据使用情况制定的选择适当作业工具的建议
- 基于 AWS 的 S3 存储桶策略，用于已定义的工作负载

Amazon EMR 启动

在 Amazon EMR 启动实施期间，AWS 专业服务团队会提供 Amazon EMR 平台基础架构，该架构支持使用基础设施即代码的方式，通过与 Amazon S3 数据湖上的存储分离的计算资源来执行 Hadoop 工作负载。AWS 专业服务团队在以下领域提供帮助：

- 使用 AWS CloudFormation 模板部署 Amazon EMR。
- 提供在 Amazon EMR 上设计和实施数据安全要求的指南。
- 确保 EMR 可以访问定义的 S3 存储桶。
- 使用 CloudFormation 模板设置 IAM 角色、安全组和 Active Directory。
- 为 Amazon EMR 的外部 Apache Hive 元存储设置 AWS Glue 或 Amazon RDS。
- 设置 Amazon EMR 集群并执行一项工作负载调整

Amazon EMR 启动阶段的目标包括：

- 在 EMR 上自动执行和优化调整过的工作负载
- 自动化的 EMR 基础设施即代码服务，可支持临时性集群和最终用户访问，以用于开发和研究。
- 记录翔实的性能和测试结果。
- 优化的 AWS 服务成本估算 *（平台支出）*

有关工作量、时间表和成本的更多信息，请联系您的销售团队或单击此处直接与 AWS 专业服务团队领导联系。

AWS 合作伙伴

除 AWS 专业服务团队之外，AWS 还包括一个强大的合作伙伴公司网络，包括在 AWS 服务交付计划中。AWS 服务交付计划是一项验证计划，旨在识别并认可具有丰富客户经验以及对特定 AWS 服务具有深刻认知的 APN 合作伙伴。符合标准的公司会按地理位置列出，并可在 Amazon EMR 合作伙伴站点上找到。

AWS Support

AWS Support 计划旨在为您提供正确的工具组合以及获取专业知识的渠道，让您可以在优化成本、性能和管理风险的同时成功使用 AWS。

适用于 Amazon EMR 的基本支持

所有 AWS 客户均可获享基本支持，包括：

- 客户服务和社区 – 全天候访问客户服务、文档、白皮书和支持论坛。
- AWS Trusted Advisor – 访问七项核心 Trusted Advisor 检查和指南，以遵循最佳实践来预置资源，提高性能和安全性。
- AWS Personal Health Dashboard – AWS 服务运行状况的个性化视图，用于在资源受到影响时发出警报。

适用于 Amazon EMR 的 AWS Premium Support

对于我们文档中可以找到的应用程序和配置，AWS Support 提供必要的协议来解决您的问题。但是，由于对 Amazon EMR 集群的自定义无穷无尽，因此某些应用程序和配置可能不在我们的文档范围之内，包括自定义 AMI 功能（以前称为 AMI for EMR Bring Your Own AMI 功能）的更改，不过只要 AMI 尚未被弃用，我们就会尽最大努力提供相关支持。[8] Amazon EMR 团队建议您联系技术客户经理，以找到受支持的解决方案或尽可能降低与特定自定义相关的风险。

AWS Premium Support 包括：

- 回答有关 AWS 服务和功能的"如何"问题
- 提供最佳实践，以帮助您在 AWS 云中成功集成、部署和管理应用程序。
- 对 API 和 AWS 开发工具包进行问题排查
- 使用 AWS 资源对操作或系统问题进行问题排查
- 对 AWS 管理控制台或其他 AWS 工具进行问题排查
- 调试 EC2 运行状况检查检测到的问题

- 对第三方应用程序（如操作系统、Web 服务器、电子邮件、VPN、数据库和存储配置）进行问题排查

AWS Premium Support 不包括：

- 代码开发
- 调试自定义软件
- 执行系统管理任务
- 查询调优

第三方和 Marketplace 支持

第三方产品（包括合作伙伴解决方案）通常不在任何 AWS Support 团队的服务范围之内。尽管我们会尽力提供帮助，但建议您从 AWS 合作伙伴网络或直接从提供产品的公司获取指导。

多种 AWS Marketplace 产品可直接在 Marketplace 中提供支持，或通过原始供应商网站提供支持。更多信息，请参阅 New – Product Support Connection for AWS Marketplace Customers。

贡献者

本文档的贡献者包括：

- Nikki Rouda，AWS 首席产品营销经理
- Mert Hocanin，AWS 大数据架构师
- Rajesh Ramchander，AWS 高级大数据顾问
- Bruno Faria，AWS EMR 解决方案架构师
- Rahul Bhartia，AWS 首席大数据架构师
- Tanzir Musabbir，AWS 高级大数据专家 SA
- Radhika Ravirala，AWS 高级专家 SA

- Damon Cortesi，AWS 大数据架构师
- Tony Nguyen，AWS 专业服务 高级顾问
- Drew Alexander，AWS 首席业务拓展人员
- Jason Berkowitz，AWS 数据湖实践经理
- Matthew Liem，AWS 大数据顾问

其他资源

更多信息，请参阅以下资源：

- Amazon EMR 网页
- 关于 Amazon EMR 版本
- AWS 大数据博客

博客文章

- 混合云架构顿悟 时刻
- 4 Dos and Don'ts When Using the Cloud to Experiment
- Top Performance Tuning Tips for Amazon Athena
- Untangling Hadoop YARN
- Orchestrate Apache Spark applications using AWS Step Functions and Apache Livy
- Orchestrate multiple ETL jobs using AWS Step Functions and AWS Lambda
- Top 10 Performance Tuning Tips for Amazon Athena
- Easily manage table metadata for Presto running on Amazon EMR using the AWS Glue Data Catalog

白皮书和指南

- Amazon EMR 版本指南

- Amazon EMR 管理指南
- Amazon EMR 入门
- 在 AWS 上构建多租户存储模型
- AWS 上的大数据分析选项

分析师报告

- 将 Apache Spark 和 Hadoop 迁移到 Amazon EMR 的经济获益
- Forrester Wave：云端 Hadoop/Spark 平台，2019 年第 1 季度

媒体

- AWS re:Invent 大会视频
- Amazon EMR YouTube 视频

文档修订

日期	描述
2020 年 3 月	中文本地化版本
2019 年 8 月	更新了标题和简介。
2019 年 6 月	首次发布。

附录 A：需求收集调查问卷

为了从本地迁移到 Amazon EMR、Amazon Athena 和 AWS Glue，您可以利用这份调查表，帮助您盘点当前架构以及可能的迁移要求。

当前集群设置

- 当前集群是怎样的？
  - 有多少个节点？
  - 集群中存储了多少数据？
- 您正在运行 Apache Hadoop 和/或 Apache Spark 哪个发行版？

集群使用

- 在高峰期和低谷期，平均使用了多少集群资源？
  - 有多少用户？CPU 和内存的占用百分比是多少？
  - 用户位于何处？是在一个时区内还是分散在全球？
- 有多少数据会被定期访问？
- 每月会添加多少新数据？
- 源输出、中间输出和最终输出数据格式是什么？
- 工作负载是否以任何方式隔离？（即使用队列或计划程序隔离）

维护

- 目前是如何管理集群的？
- 升级是如何完成的？
- 是否有独立的开发和生产集群？
- 是否有备用集群或数据备份规程？

使用案例

批处理作业

- 每天有多少作业?
- 它们平均运行多长时间?
- 它们通常何时运行?
- 批处理作业运行什么类型的作业? （即机器学习、聚合、数据格式转换等）。（针对机器类型进行优化。）
- 批处理作业使用什么框架或语言? （即 Apache Spark、HiveQL 等)
- 是否有针对下游消费者的服务等级协议 (SLA)?
- 如何将作业升级到批处理环境?
- 作业如何提交到集群?
- 作业是否设计为幂等?

交互式使用案例

- 是否可以交互方式访问集群?
- 谁在使用集群?
- 如何保护集群?
- 有多少人在使用它们?
- 人们使用哪些工具或应用程序进行连接?
- 若使用 Spark，如何部署和提交 Spark 作业?

Amazon Athena 使用案例

- 查询负载是多少? Amazon Athena 是否更适合您的使用案例?

AWS Glue 使用案例

- AWS Glue 预计会承担怎样的负载?
- 是否有现有的 Hive 数据目录? 数据目录可否迁移到 AWS Glue?

安全要求

- 您目前是否在使用 Kerberos?
- 您目前是否在使用目录服务? 使用的是哪种?
- 是否有细粒度的访问权限控制要求? 如何限制读/写访问权限?
- 是否允许用户创建自己的表? 您如何跟踪数据沿袭?

TCO 注意事项

长期增长预期

- 随着时间的推移，数据会增长多少?
- 随着时间的推移，计算需求将增长多少?
- 是否有需要注意的网络传输成本? （包括迁移期间和迁移之后的成本，例如跨区域复制。）
- 是否有停用现有环境的目标日期?
- 在过渡期间维护现有基础设施的成本是多少?

附录 B：EMR Kerberos 工作流程

MIT 的 KDC 开源版本安装在 Amazon EMR 上。KDC 使用默认设置。有关其配置和操作的详细说明，请参阅 MIT 的 Kerberos 文档。

本附录中的各个小节将展示用户如何与启用了 Kerberos 的 Amazon EMR 集群和消息流进行交互。

具有单向信任机制的 KDC 的 EMR Kerberos 集群启动流程

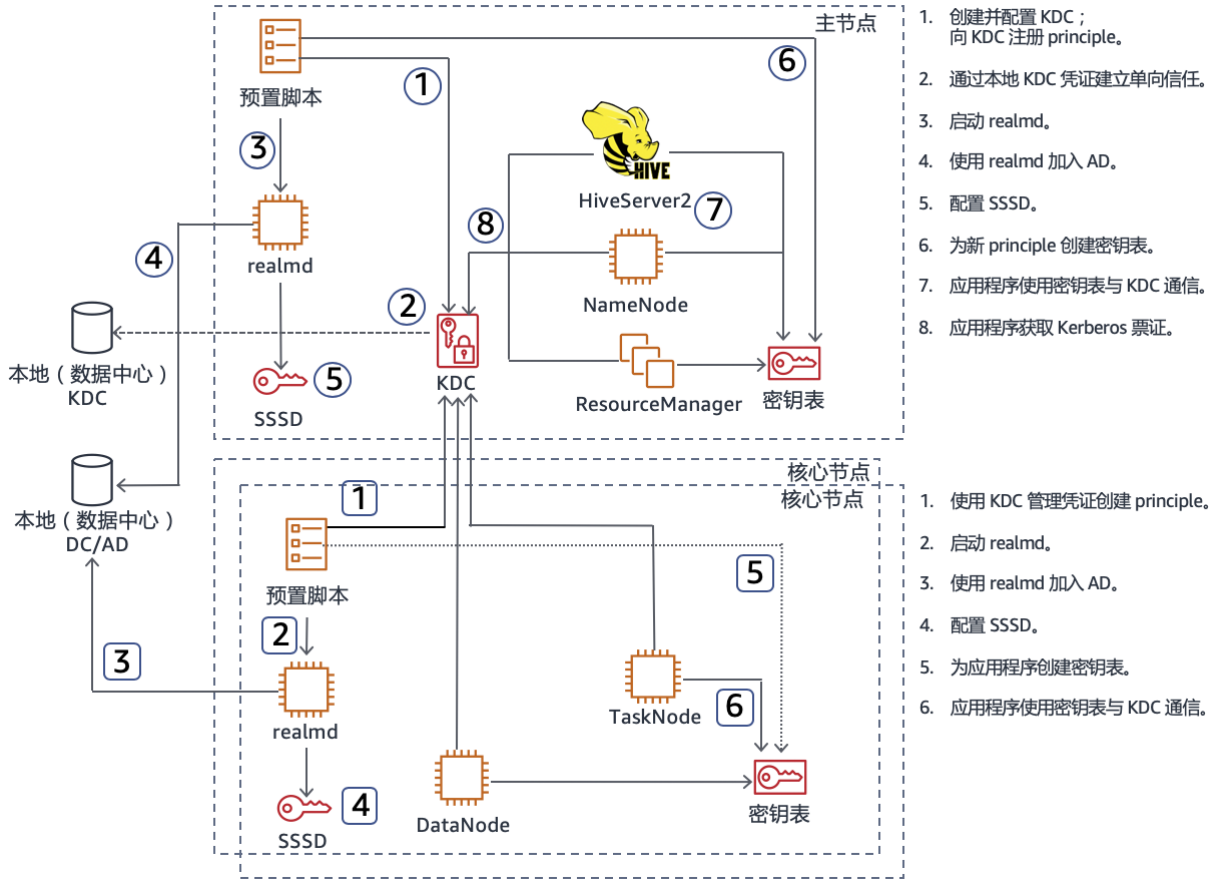


图 45：具有单向信任机制的 KDC 的 EMR Kerberos 集群启动流程

在此流程中，每个节点都有一个预置脚本，该脚本负责：

- （仅主节点）创建 KDC 并将其配置为单向信任。

- （所有节点）启动 realmd 以加入 Active Directory，然后配置 SSSD 以处理用户和组映射请求并创建密钥表。
- （所有节点）为每个应用程序和子应用程序创建应用程序 principle 和密钥表。
- 在集群中出现一个新节点时，其预置脚本将执行相同的操作：在主节点上的 KDC 内为本地运行的所有应用程序创建 principle、创建密钥表文件、将该节点加入 Active Directory（若已配置），并启动应用程序。

通过 Hue 访问的 EMR Kerberos 流

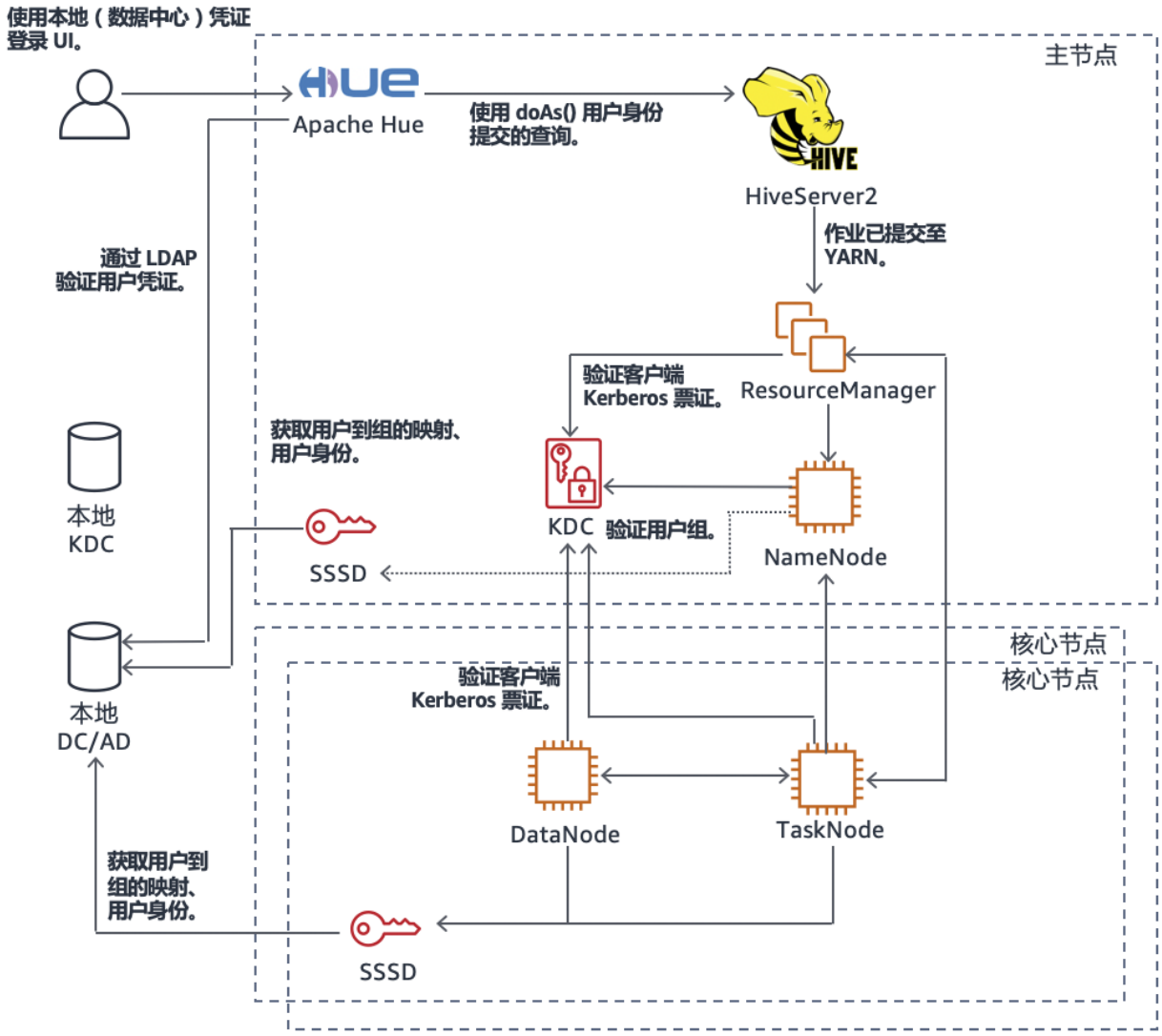


图 46：通过 Hue 访问的 EMR Kerberos 流

1. 用户使用其数据中心凭证登录 Hue（或 Zeppelin）。
2. Hue 使用 LDAP(S) 向数据中心的 Active Directory 验证这些凭证。
3. 通过身份验证后，用户可以提交 Hive 查询。
4. Hue 提交这些查询，并告知 HiveServer2 以用户身份（即模拟身份）运行作业。

5. HiveServer2 将作业提交给资源管理器进行处理。
6. 在作业执行期间，Hadoop 使用 SSSD 验证本地节点上的用户账户，从而对用户进行身份验证和授权。

与 HiveServer2 直接交互的 EMR Kerberos 流

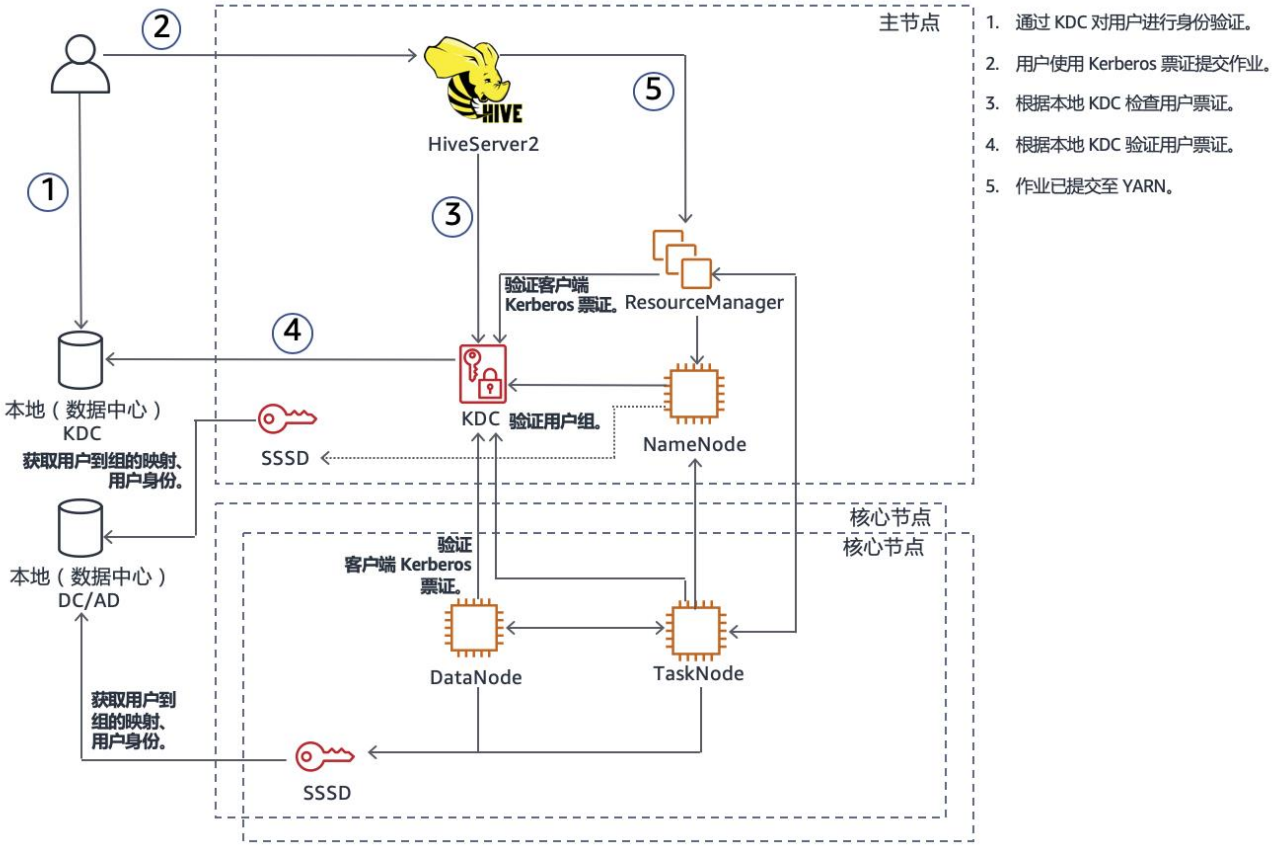


图 47：与 HiveServer2 直接交互的 EMR Kerberos 流

1. 用户使用本地 KDC 的凭证进行身份验证，并收到 Kerberos 票证。
2. 用户向 HiveServer2 提交查询和 Kerberos 票证。
3. HiveServer2 请求本地 KDC 验证 Kerberos 票证。

4. 本地 KDC 请求数据中心的 KDC 验证 Kerberos 票证。
5. HiveServer2 将作业提交给资源管理器，以用户身份进行处理。
6. 在作业执行期间，Hadoop 使用 SSSD 验证本地节点上的用户账户，从而对用户进行身份验证和授权。

EMR Kerberos 集群启动流程

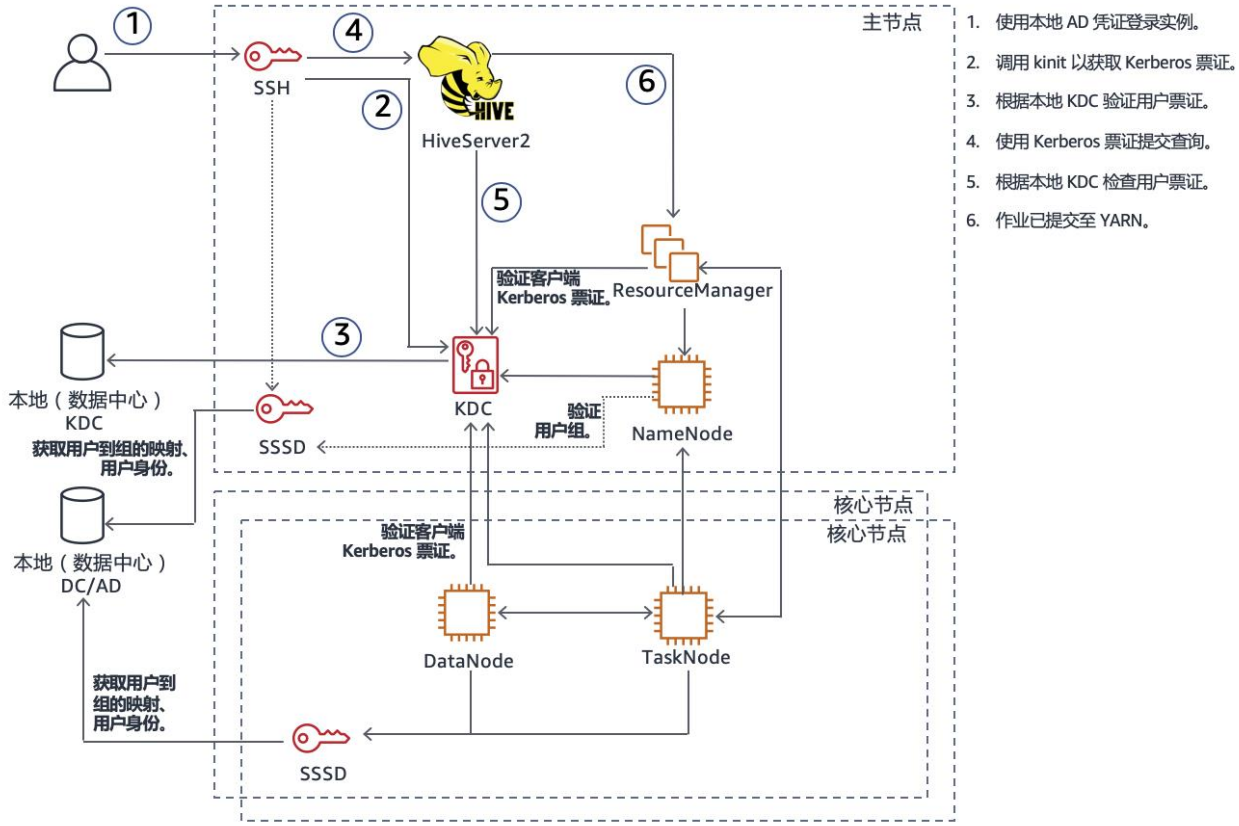


图 48：EMR Kerberos 集群启动流程

1. 用户通过 SSH 访问主节点，并使用用户凭证进行身份验证。
2. 若用户需要 Kerberos 票证，则用户必须执行 kinit (kinit -k -t <keytab> <principal>) 才能使用其凭证从本地 KDC 获得 Kerberos 票证。
3. 本地 KDC 使用数据中心的 KDC 对用户进行身份验证并返回 Kerberos 票证。

4. 用户将查询和 Kerberos 票证提交到 HiveServer2。
5. HiveServer2 请求本地 KDC 验证票证。
6. 本地 KDC 请求数据中心 KDC 验证票证。
7. HiveServer2 将作业提交给资源管理器，以用户身份进行处理。
8. 在作业执行期间，Hadoop 使用 SSSD 对本地节点上的用户账户进行身份验证和授权。

附录 C：LDAP 配置示例

Hadoop Group Mapping 的示例 LDAP 配置

下面显示了一个示例 Amazon EMR 配置文件，用于设置 Hadoop 组映射，以直接使用 LDAP。对于没有 Kerberos 的集群，请使用此设置。

```
[

{
      "classification":"core-site",
      "properties":{
         "hadoop.security.group.mapping.ldap.search.attr.member":"m
ember",
         "hadoop.security.group.mapping.ldap.search.filter.user":"(
objectclass=*)",
         "hadoop.security.group.mapping.ldap.search.attr.group.name
":"cn",
         "hadoop.security.group.mapping.ldap.base":"dc=corp,dc=emr,
dc=local",
         "hadoop.security.group.mapping":"org.apache.hadoop.securit
y.LdapGroupsMapping",
         "hadoop.security.group.mapping.ldap.url":"ldap://172.31.93
.167",
         "hadoop.security.group.mapping.ldap.bind.password":"Bind@U
ser123",
         "hadoop.security.group.mapping.ldap.bind.user":"binduser@c
orp.emr.local",
         "hadoop.security.group.mapping.ldap.search.filter.group":"
(objectclass=*)"
      },
      "configurations":[
      ]
   }
]
```

Hue 的 LDAP 配置示例

```
[
 {
    "classification":"hue-ini",
    "properties":{
    },
    "configurations":[
       {
          "classification":"desktop",
          "properties": { },
            "configurations":[
               {
                  "classification":"auth",
                  "properties":{
                     "backend":"desktop.auth.backend.LdapBackend"
                  },
                  "configurations": []
               },
               {
                  "classification":"ldap",
                  "properties":{
                     "bind_dn":"binduser@corp.emr.local",
                     "trace_level":"0",
                     "search_bind_authentication":"false",
                     "debug":"true",
                     "base_dn":"dc=corp,dc=emr,dc=local",
                     "bind_password":"Bind@User123",
                     "ignore_username_case":"true",
                     "create_users_on_login":"true",
                     "ldap_username_pattern":"uid=<username>,cn=use
rs,dc=corp,dc=emr,dc=local",
                     "force_username_lowercase":"true",
                     "ldap_url":"ldap://172.31.93.167",
                     "nt_domain":"corp.emr.local"
                  },
                  "configurations":[
                     {
                        "classification":"groups",
                        "properties":{
                           "group_filter":"objectclass=*",
                           "group_name_attr":"cn"
```

```
                        },
                        "configurations": []                    },
                    {
                        "classification":"users",
                        "properties":{
                            "user_name_attr":"sAMAccountName",
                            "user_filter":"objectclass=*"
                        },
                        "configurations":[]
                    }
                ]
            }
        ]
    }
]
```

附录 D：数据目录迁移常见问题

使用 AWS Glue 数据目录而非通用 Hive 元存储时，有哪些限制?

AWS Glue 数据目录不支持列统计信息、Hive 授权和 Hive 约束。要查看 AWS Glue 数据目录约束的列表，请参阅使用 AWS Glue 数据目录作为 Hive 的元存储中的*使用 AWS Glue 目录时的注意事项*。

AWS Glue 数据目录可使用哪些类型的安全功能?

您可以在 AWS Glue 数据目录上启用加密，并且可以通过 IAM 策略配置对 AWS Glue 操作的访问。默认的 Amazon EMR EC2 角色 (EMR_EC2_DefaultRole) 允许执行必要的 AWS Glue 操作。但是，若在创建集群时指定了自定义 EC2 实例配置文件和权限，请确保允许执行适当的 AWS Glue 操作。有关可用的 Glue IAM 策略的列表，请参阅 AWS Glue API 权限：操作和资源参考。

多个 Amazon EMR 集群可否使用一个 AWS Glue 数据目录?

可以，可通过一对多方式对 Amazon EMR 集群以及 Amazon Athena 和 Amazon Redshift 使用 AWS Glue 数据目录。

本地 Hadoop 集群可否使用 AWS Glue 数据目录?

不可以，AWS Glue 数据目录库尚未开放源代码，并且无法使用 AWS Glue 数据目录从本地 Hadoop 集群进行连接。

何时应使用 Amazon RDS 上的 Hive 元存储，而非 AWS Glue 数据目录?

若您想要完全控制 Hive 元存储，并希望与其他开源应用程序（例如 Apache Ranger 和 Apache Atlas）集成，请使用 Amazon RDS 上的 Hive 元存储。若您正在寻找托管和无服务器的 Hive 元存储，请使用 AWS Glue 数据目录。

Notes

[1] 有关如何设置 LDAP 服务器并与 Apache Hue 集成的分步指南，请访问 AWS 大数据博客，参阅 Using LDAP via AWS Directory Service to Access and Administer Your Hadoop Environment。

[2] 将 Amazon S3 用作数据湖存储层的示例客户包括 NASDAQ、Zillow、Yelp、iRobot 和 FINRA。

[3] 有关这些功能的更多信息，请参阅配置节点停用行为。

[4] 对于 Amazon EMR 版本 5.8.0 及更高版本，您可以将 Hive 配置为使用 AWS Glue 数据目录作为其元存储。请参阅数据目录迁移中的现有的 Hive 元存储迁移到 AWS Glue 数据目录。

[5] 适用于 Amazon EMR 5.20 及更高版本的软件。

[6] 例如，FINRA 将 700TB 的 HBase 环境迁移到了 Amazon S3 上的 HBase。更多信息，请参阅 Low-Latency Access on Trillions of Records: FINRA's Architecture Using Apache HBase on Amazon EMR with Amazon S3。

[7] V.M.Megler、Kristin Tufte 和 David Maier，*《提高智能交通系统中的数据质量》*，https://arxiv.org/abs/1602.03100（2016 年 2 月 9 日）

[8] 有关支持范围的信息，请参阅 Amazon EMR 新增功能和 Amazon EMR 常见问题。