E-MapReduce

用户指南

用户指南

在用户开通 E-MapReduce 服务时，需要授予一个名称为“AliyunEMRDefaultRole”的系统默认角色给 E-MapReduce 的服务账号，当且仅当该角色被正确授予后，E-MapReduce 才能正常地调用相关服务（ECS，OSS 等），创建集群以及保存日志等。

角色授权流程

当用户创建集群或创建按需执行计划时，如果没有正确地给 E-MapReduce 的服务账号授予默认角色，则会看到如下提示，此时需单击**前往 RAM 进行授权**，进行角色授权。

单击**同意授权**，将默认角色 AliyunE-MapReduceDefaultRole 授予给 E-MapReduce 的服务账号。

当完成以上授权步骤后，用户需刷新 E-MapReduce 的控制台，然后就可以进行操作了。如果您想查看 AliyunE-MapReduceDefaultRole 相关的详细策略信息，可以登录 RAM 的控制台查看，也可以单击查看链接。

默认角色包含的权限内容

默认角色 AliyunEMRDefaultRole 包含的权限信息如下：

- ECS 相关权限

权限名称（Action）	权限说明
ecs:CreateInstance	创建 ECS 实例
ecs:RenewInstance	ECS 实例续费
ecs:DescribeRegions	查询 ECS 地域信息
ecs:DescribeZones	查询 Zone 信息
ecs:DescribeImages	查询镜像信息
ecs:CreateSecurityGroup	创建安全组
ecs:AllocatePublicIpAddress	分配公网 IP
ecs:DeleteInstance	删除机器实例
ecs:StartInstance	启动机器实例
ecs:StopInstance	停止机器实例
ecs:DescribeInstances	查询机器实例
ecs:DescribeDisks	查询机器相关磁盘信息
ecs:AuthorizeSecurityGroup	设置安全组入规则
ecs:AuthorizeSecurityGroupEgress	设置安全组出规则
ecs:DescribeSecurityGroupAttribute	查询安全组详情
ecs:DescribeSecurityGroups	查询安全组列表信息

- OSS 相关权限

权限名称（Action）	权限说明
oss:PutObject	上传文件或文件夹对象
oss:GetObject	获取文件或文件夹对象
oss:ListObjects	查询文件列表信息

集群规划

EMR集群中由多个不同的节点实例类型组成，他们分别是主实例节点（Master），核心实例节点（Core）和计算实例节点（Task），每一种不同的实例在部署的时候会部署完全不同的服务进程，以完成完全不同的任务。举例来说，我们会在主实例节点（Master）上部署 Hadoop HDFS 的 Namenode 服务，Hadoop YARN 的 ResourceManager 服务，而在核心实例节点（Core）上部署 Datanode 服务，Hadoop YARN 的 NodeManager 服务，在计算实例节点（Task）顾名思义，只进行计算，部署 Hadoop YARN 的 NodeManager 服务，不部署任何 HDFS 相关的服务。

在创建集群的时候需要确定对应的三种实例类型的 ECS 规格，相同实例类型的 ECS 在同一个实例组内。并且可以在后期通过扩容来扩容对应实例组内的机器数量（主实例组除外）。

注意：计算实例节点（Task）从3.2.0及以后版本开始支持

Master 主实例

主实例是集群服务的管控等组件的部署的节点，举例来说，Hadoop YARN 的 ResourceManager 就部署在主实例节点上。用户可以通过 SSH 的方式连接到主实例上，通过软件的 Web UI 来查看集群上的服务的运行情况。同时当需要进行快速的测试或者运行作业的时候，也可以登录到主实例上，通过命令行来直接提交作业。当集群开启了高可用的时候会有2个主实例节点，默认只有1个。

Core 核心实例

核心实例是被主实例管理的实例节点。上面会运行 Hadoop HDFS 的 Datanode 服务，并保存所有的数据。同时也会部署计算服务，比如 Hadoop YARN 的 NodeManager 服务，来执行计算任务。为满足存储数据量或者是计算量上升的需要，核心实例可以随时的扩容，不影响当前集群的正常运行。核心使用可以使用多种不同的存储介质来保存数据。参考磁盘介绍。

Task 计算实例

计算实例是专门负责计算的实例节点，是一个可选的实例类型。如果核心实例的计算能力足够的情况下，可以不使用计算实例。计算实例可以在任何时候快速的为集群增加额外的计算能力，如 Hadoop 的 MapReduce tasks， Spark executors 等。在计算实例上不会保存 HDFS 的数据，因此在计算实例上不运行 Hadoop HDFS 的 Datanode服务。计算实例可以随时的新增和减少，都不会影响到现有集群的运行。计算实例节点的减少可能会引起 MapReduce 和 Spark 的作业的失败，能否成功取决于该计算服务的重试容错能力。

EMR目前支持的ECS实例类型

通用型

VCPU与Memory比为1：4，如32核128G，使用云盘作为存储。

计算型

VCPU与Memory比为1：2，如32核64G，使用云盘作为存储，提供了更多的计算资源。

内存型

VCPU与Memory比为1：8，如32核256G，使用云盘作为存储，提供了更多的内存资源。

大数据型

使用本地SATA盘作为数据存储，拥有很高的存储性价比，是大数据量（T级别的数据量）场景下的推荐机型。

本地SSD型

使用本地SSD盘，拥有极高的本地iops和吞吐能力。总体的存储

共享型（入门级）

共享CPU的机型，在大计算量的场景下，稳定性不够。入门级学习使用，不推荐企业客户使用。

GPU

使用GPU的异构机型，可以用来运行机器学习等场景。

实例类型适用场景

Master 主实例

适合通用型或内存型实例，数据直接使用阿里云的云盘来保存，有三个备份的保证，数据高可靠。

Core 核心实例

小数据量（T以下）或者是使用OSS作为主要的数据存储时，可以使用通用型，计算型或内存型。当数据量较大的情况下，如10T或以上，推荐使用大数据机型，会获得极高的性价比。使用本地盘会有一个数据可靠性的挑战，会由 EMR 平台来进行维护和保证。

Task 计算实例

作为集群的计算能力的补充，可以使用除大数据型以外的所有的机型。目前本地SSD型尚未支持，后续会加入到 Task 中。

云盘与本地盘

在节点上存在2种角色的磁盘，一类是系统盘，用来安装操作系统。一类是数据盘，用来保存数据。系统盘默认都是一块，必须使用云盘。而数据盘可以有很多块，目前上限可以一个节点挂16块。每一块都可以有不同的配置，类型和容量都可以不同。EMR默认使用SSD云盘作为集群的系统盘。EMR 默认挂载4块云盘，目前的内网带宽的情况下4块云盘是比较合理的配置。

有2种类型的磁盘可以用作数据的存储

云盘

包括，SSD 云盘，高效云盘，普通云盘

特点是，磁盘并不直接挂载在本地的计算节点上，通过网络访问远端的一个存储节点。每一份数据在后端都有2个实时备份，一共三份数据。所以当一份数据损坏的时候（磁盘损坏，不是用户自己的业务上的破坏），会自动的使用备份数据恢复

本地盘

包括，大数据型的 SATA 本地盘，和本地 SSD 的本地 SSD 盘

直接挂载在计算节点上的磁盘，拥有超过云盘的性能表现。使用本地盘的时候不能选择数量，只能使用默认配置好的数量，和线下物理机一样，数据没有后端的备份机制，需要上层的软件来保证数据可靠性。

适用的场景

在EMR中，所有云盘和本地盘都会在节点释放的时候清除数据，磁盘无法独立的保存下来，并再次使用。Hadoop HDFS 会使用所有的数据盘作为数据存储。 Hadoop YARN 也会使用所有的数据作为计算的临时存储。

当业务数据量并不太大（T级别以下）的时候，可以使用云盘，iops 和吞吐相比本地盘都会小些。数据量大的时候，推荐都使用本地盘，EMR 会来维护本地盘的数据可靠性。如果发现在使用中明显的吞吐量不够用，可以切换到本地盘的存储上。

OSS

在 EMR 中可以将 OSS 作为 HDFS 使用。 用户可以非常方便的读写OSS，所有使用 HDFS 的代码也可以简单的修改就能访问 OSS 上的数据了。

比如：

Spark中读取数据

```
sc.Textfile("hdfs://user/path")
```

替换存储类型 hdfs -> oss

```
sc.Textfile("oss://user/path")
```

对于MR或者Hive作业也是一样

HDFS命令直接操作OSS数据

```
hadoop fs -ls oss://bucket/path
hadoop fs -cp hdfs://user/path oss://bucket/path
```

这个过程，不需要输入AK和endpoint，EMR都会自动替用户使用当前集群所有者的信息补全。

但 OSS 的 iops 不高，在一些需要高 iops 的场景，不适合使用，比如流式计算 Spark Streaming 或 HBase。

阿里云为了满足大数据场景下的存储的需求，在云上推出了本地盘的机型：D1系列。这个系列提供了本地盘而非云盘作为存储。解决了之前使用云盘的多份冗余数据导致的成本高问题，同时数据的传输不再需要全部通过网络，从而提高了磁盘的吞吐能力。同时还能发挥 Hadoop 的就近计算的优势。

相比于使用云盘的方式，极大的提高了存储的性能，并降低了存储的单价，达到和线下物理机几乎相同的成本。

本地盘机型在提供了大量的优势的情况下，也带来了一个问题：数据可靠性。对于云盘来说，由于有阿里云默认的磁盘多备份策略，所以用户可以说完全感知不到磁盘的损坏，由云盘自动保证数据可靠，当使用了本地盘以后这个就需要由上层的软件来保证。同时如果有磁盘与节点的故障情况，也需要进行人工的运维处理。

EMR + D1 方案

EMR 产品针对本地盘机型，如 D1，推出了一整套的自动化运维方案，帮助阿里云用户方便可靠的使用本地盘机型，不需要关心整个运维的过程的同时，做到数据高可靠，服务高可用。

主要的一些点如：

- 强制节点的高可靠分布
- 本地盘与节点的故障监控
- 数据迁移时机自动决策
- 自动的故障节点迁移与数据平衡
- 自动的HDFS数据检测
- 网络拓扑调优

EMR通过整个后台的管控系统的自动化运维，协助用户更好的使用本地盘机型，实现高性价比的大数据系统。

注意： 如需使用 D1 机型搭建 Hadoop 集群，请工单联系我们协助操作。

目前阿里云存在2种网络类型，一个是经典网络，一个是VPC，很多用户的业务系统因为历史的原因还会在经典

网络中，而EMR集群是在VPC中，本节将会介绍如何使经典网络的ECS可以和VPC网络下的EMR集群网络互访
，

ClassicLink

为了解决这个问题，阿里云推出了ClassLink方案

大致步骤如下，详细的请参考上面的ClassLink文档：

1. 首先按照上面文档中指定网段创建vswitch。
2. 在创建集群的时候，请使用该网段的vswitch来部署集群。
3. 在ECS控制台将对应的经典网络节点连接到这个VPC
4. 设置安全组访问规则

然后就可以让经典网络的ECS和VPC的集群互访了。

集群

进入创建集群页面

登录阿里云 E-MapReduce 控制台集群列表。

完成 RAM 授权，操作步骤请参见角色授权。

在上方选择所在的地域（Region），所创建集群将会在对应的地域内，一旦创建后不能修改。

单击右上方的**创建集群**按钮，进行创建。

创建集群流程

注意：集群除了名字以外，一旦创建完成就无法被修改。所以在创建时请仔细确认需要的配置。

要创建集群，您需要继续完成以下 3 个步骤：

- 软件配置
- 硬件配置
- 基础配置

步骤1：软件配置

配置项说明：

产品版本：E-MapReduce 产品的主要版本，代表了一整套的开源软件环境，它会定时的根据内部组成软件的升级进行升级。一般如果 Hadoop 相关的软件有进行升级，E-MapReduce 也会升级，这个时候就会升级这个主版本号。低版本的集群无法自动的升级到一个高版本上。

集群类型：目前的EMR提供了

- Hadoop标准的 Hadoop 集群，包含了大部分的 Hadoop 相关的组件，具体的组件信息可以在选择界面的列表中查看。
- Kafka独立的 Kafka 集群，提供消息服务。

包含配置：展示选择的集群类型下的所有的软件组件列表，包括名称和版本号。根据需求，您可选择不同的组件，被选中的组件会默认启动相关的服务进程。

注意：您选择的组件越多，对您机器的配置要求就越高，否则很可能无法有足够的资源来运行这些服务。

安全模式：是否开启集群的 Kerberos 认证功能。

软件配置（可选）：可以对集群中的基础软件例如 Hadoop、Spark、Hive 等进行配置，详细使用方法请参见软件配置。

步骤2：硬件配置

配置项说明：

付费配置

付费类型包年包月是一次性支付一个长期的费用，价格相对来说会比较便宜，特别是包三年的时候折扣会很大。按量付费是根据实际使用的小时数来支付费用，每个小时计一次费用。适合与短期的测试或者是灵活的动态任务，价格相对来说会贵一些。

购买时长：您可选择购买 1 个月、2 个月、3 个月、6 个月、9 个月、1 年、2 年、3 年。

集群网络配置

集群可用区：选择集群所在的可用区（Zone），不同的可用区会有不同的机型和磁盘。在每个 Region 内存在多个可用区。可用区在物理上属于不同的区域，一般来说如果需要较好的网络，推荐您选择相同的可用区，但是这样也会使创建集群失败的风险增大，因为单个可用区的存库不一定那么充足。如果需要大量的机器可以工单咨询我们。

网络类型：可以选择经典网络和专有网络（VPC），专有网络需要额外提供所属 VPC 以及子网（交换机），若还未创建，可前往VPC控制台进行创建。E-MapReduce 专有网络详细使用说明查看专有网络。

注意：经典网络与专有网络不互通，购买后不能更换网络类型。

ECS 实例系列：不同的可用区有不同的实例系列，系列 I、II、III等。尽量使用最新的系列。

VPC：选择在该地域的VPC。

交换机：选择在对应的VPC下的在对应可用区的交换机，如果在这个可用区没有可用的交换机，那么就需要前往去创建一个新的使用。

新建安全组：一般用户初次来到这里还没有安全组，打开“新建安全组”开关，在“安全组名称”里面填上新的安全组的名字。

选择安全组：集群所属的安全组。这里只展示用户在 E-MapReduce 产品中创建的安全组，目前尚不支持选择在 E-MapReduce 外创建的安全组。如果需要新建安全组，可以选择“新建安全组”选项，同时输入安全组的名字完成新建。长度限制为 2-64 个字符，以大小写字母或中文开头，可使用中文、字母、数字、 “-”和“_”。

集群节点配置

高可用集群：打开后，Hadoop 集群会有 2 个 master 来支持 Resource Manager 和 Name Node 的高可用。HBase 集群原来就支持高可用，只是另一个节点用其中一个 core 节点来充当，如果打开高可用，会独立使用一个 master 节点来支持高可用，更加的安全可靠。默认为非高可用模式，master节点数量为1。

节点类型：

- Master主实例节点，主要负责Resource Manager，Name node等控制进程的部署
- Core核心实例节点，主要负责集群所有数据的存储，可以按照需要进行扩容
- Task纯计算节点，不保存数据。调整集群的计算力使用。

节点配置：不同规格的机型的选择。各个机型有各自比较适用的场景，可以根据需要选择。

数据盘类型：集群的节点使用的数据盘类型，数据盘有 3 种类型，普通云盘、高效云盘和 SSD 云盘，根据不同机型和不同的 Region，会有不同。当用户选择不同的区的时候，该区支持什么盘，下拉框就会展示什么类型的盘。数据盘默认设置为随着集群的释放而释放。本地盘的计算节点，磁盘是默认选定的，无法修改。

数据盘容量：目前推荐的集群容量最小是 40G 单机，最大可以到32T单节点。本地盘的容量是默认的，无法调整。

实例数量：需要的总的节点的台数。一个集群至少需要 3 台实例（高可用集群需增加 1 个 Master 节点，至少 4 台）。按量集群目前最大台数是 50 台，如果需要超过 50 台，请工单联系我们。包月集群最大100台，超过50台请工单联系我们。

步骤3：基础配置

配置项说明：

基本信息

- **集群名称**：集群的名字，长度限制为 1-64 个字符，仅可使用中文、字母、数字、 “-”和 “_”。

运行日志

运行日志：是否保存作业的日志，日志保存默认是打开的。开启后会需要您选择用来保存日志的 OSS 目录位置，会将您的作业的日志保存到该 OSS 存储目录上。当然，您要使用这个功能必须先开通 OSS，同时上传的文件会按照使用的量来计算用户的费用。强烈建议您打开 OSS 日志保存功能，这会对您的作业调试和错误排查有极大的帮助。

日志路径：保存日志的 OSS 路径。

统一Meta数据库：将你所有的 Hive 的元数据都保存到集群外部的数据库上，由EMR产品提供。推荐当集群使用 OSS 作为主要的存储的时候，使用这个功能。

权限设置

- **服务角色**：这个是用户将权限授予EMR服务，允许 EMR 代表用户调用其他阿里云的服务，例如 ECS 和 OSS
- **ECS应用角色**：这个是当用户的程序在 EMR 计算节点上运行的时候，可以不填写阿里云的 AK 来访问相关的云服务，比如OSS。EMR 会自动的申请一个临时 AK 来授权这次访问。而这个 AK 的权限控制将由这个角色来控制。

登录设置

- **登录密码**：设置 master 节点的登录密码。8 - 30 个字符，且必须同时包含大写字母、小写字母、数字和特殊字符!@#$%^&*

引导操作（可选）： 您可以在集群启动 Hadoop 前执行您自定义的脚本，详细使用说明请参见引导操作。

配置清单和集群费用

页面右边会显示您所创建集群的配置清单以及集群费用。根据付费类型的不同，会展示不同的价格信息。按量付费集群显示每小时费用，包年包月显示总费用。

确认创建

当所有的信息都有效填写以后，“创建”按钮会亮起，确认无误后单击**创建**将会创建集群。

注意：

> 若是按量付费集群，集群会立刻开始创建。页面会返回集群列表页，就能看到在列表中有一个“集群创建中”的集群。请耐心等待，集群创建会需要几分钟时间。完成之后集群的状态会切换为“集群空闲”。
>
> 若是包年包月集群，则会先生成订单，在支付完成订单以后集群才会开始创建。

创建失败

如果创建失败，在集群列表页上会显示“集群创建失败”，将鼠标移动到红色的感叹号上会看到失败原因，如下图所示。

创建失败的集群可以不用处理，对应的计算资源并没有真正的创建出来。这个集群会在停留3天以后自动隐藏。

当您的集群资源（计算资源、存储资源）不足的时候，您可以将您的集群进行水平扩展。目前只能扩展您的 Core 节点，且使用的配置默认与您之前购买的 ECS 配置一致。

扩容入口

在集群列表页上，找到需要扩展的集群条目，单击**调整规模**按钮就会进入集群扩容页面。也可以单击**查看详情**，然后在详情页的 Core 节点信息位置单击**调整集群规模**。

扩容界面

如下图所示：

注意：目前，只支持扩容，不支持缩容。

当前 Master 节点数量：不可以调整，当前默认值是 1 个。

当前 Core 台数：默认显示的是您当前的所有 Core 节点数量。

增加 Core 台数: 输入您实际需要增加的量，右侧会显示扩容后的集群总费用，然后单击确定就会进行扩容。安全起见，强烈推荐您在扩容的时候小批量的进行，比如 1 到 2 台，不要一次扩容太大的数量。

扩容状态

如下图所示：

Core节点信息

基本信息： 当前6台 CPU：8核 内存：16G 硬盘类型：高效云盘 硬盘：80G X 4 块

实例ID	实例状态	内网IP	ECS到期时间	EMR到期时间
i-23ntm2cxv	正常	10.47.111.140	2016/10/18 16:00:00	2016/10/13 00:00:00
i-23pw6vp68	正常	10.45.54.35	2016/10/18 16:00:00	2016/10/13 00:00:00
i-230pvxd3z	正常	10.24.34.86	2016/10/18 16:00:00	2016/10/13 00:00:00
i-23mzhb5jz	扩容中	10.24.30.171	2016/12/09 16:00:00	2016/11/28 00:00:00
i-23sii98i2	正常	10.24.39.125	2016/10/18 16:00:00	2016/10/13 00:00:00
i-23czti1xv	正常	10.25.1.230	2016/10/18 16:00:00	2016/10/13 00:00:00

您可以在详情页的 Core 节点信息上看到集群的扩容情况，正在扩容的节点，其状态会显示为“扩容中”。当这台 ECS 的状态转为“正常”后，该 ECS 即已经加入该集群，并可正常提供服务了。

在集群列表页面，您可以单击集群条目右侧的释放按钮对集群进行释放操作。只有以下状态的集群可以被释放。

创建中

运行中

空闲中

普通释放

释放前会提示您再次确认，一旦确认释放，会发生以下的操作：

所有在集群上的作业都会被强制终止。

如果您选择了保存日志到 OSS，那么当前作业的日志会被保存到 OSS，所需时间取决于日志大小。日志的上传和作业的运行是并行的，作业生成日志的同时，就会进行日志的上传。所以最终作业停止时，需要上传的日志一般不会特别多，正常在几分钟内都会完成。

终止并释放所有的ECS。这个过程取决于集群的大小，越小的集群会越快，正常都在几秒内完成，至多不会超过5分钟。最迟释放的ECS在等待释放时仍然计费。

如果您想要省钱而控制在整点前释放，请务必留出一定的释放时间保证确实在整点前释放

释放集群结束。

强制释放

如果您不需要任何日志，只是想快速结束集群的运行，那么可以开启强制释放。释放过程就会跳过日志收集（如果有打开日志收集的话），直接进入ECS释放阶段。

释放失败的集群

由于系统错误等原因，集群有可能会在确认释放后，释放失败。您单击释放是一个异步的过程，可能等一会儿会发生集群释放失败的情况，但是不用担心，E-MapReduce 会启动后台保护，自动重试释放该集群，直到集群被成功释放为止。

展示您所拥有的所有集群的基本信息，如下图：

列表栏各项说明如下：

实例 ID/集群名称：集群的 ID 以及名称。在这里将鼠标移动到名称上，可以对集群的名称进行修改。

集群类型：目前只有 Hadoop 类型，马上会追加 HBase 类型。

已运行时间：从创建开始到目前的运行时间。集群一旦被释放，计时终止。

状态（默认）：集群的状态，请参见**集群状态**。当集群出现异常的时候，比如创建失败，在右侧会有提示信息，鼠标悬停能够查看到详细的错误信息。您也可以单击状态（默认）对状态进行筛选。

付费类型：集群的付费类型。

操作：当前集群可以被施加的操作，包含以下操作。

查看详情：进入集群的详情页，查看集群建立以后的详细信息。

调整规模：集群扩容功能入口。

释放：释放一个集群，请参见**释放集群**。

展示用户集群的详细信息，包含以下四大部分：

集群信息

ID/名称：集群的实例 ID 及名称。

付费类型：集群的付费类型。

地域：集群所在的 Region。

当前状态：请参见**集群状态**。

开始时间：集群的创建时间。

运行时间：集群的运行时间。

日志开启：是否开启了日志保存功能。

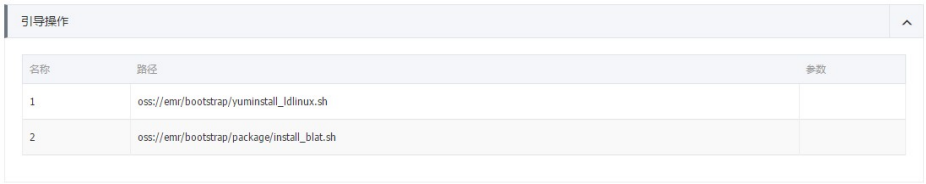
日志位置：如果开启了日志保存，那么这里显示日志保存的位置。

软件配置：软件的配置信息。

引导/软件配置：是否有异常。

高可用集群：是否开启了高可用集群。

引导操作



这里列出了所有配置的引导操作的名称、路径以及参数。

软件信息

主版本：使用的 E-MapReduce 的主版本。

集群类型：选择的集群类型。

软件信息：列出了用户安装的所有的应用程序及其版本，例如，Hadoop 2.6.0，Hive 0.14。

硬件信息

网络类型：集群所在的网络。

所属安全组：集群加入的安全组

所属可用区：集群所在的可用区，例如 cn-hangzhou-b，与 ECS 的一致。

VPC/子网：用户集群所在的 VPC 与子网交换机的 ID。

Master 节点信息：所有 Master 节点对应的配置，包含以下内容。

节点数：当前的节点数量和实际申请的节点数量。理论上这 2 个值一定是一样的，但是在创建过程中，当前节点会小于申请节点，直到创建完成。

CPU：单个节点的 CPU 的核数。

内存：单个节点的内存的容量。

数据盘类型：数据盘类型。

数据盘：单个节点的数据盘容量。

实例状态：包含创建中、正常、扩容中和已释放。

公网 IP：Master 的公网 IP。

内网 IP：机器的内网 IP，可以被集群中的所有节点访问到。

ECS 到期时间：所购买的 ECS 的到期使用时间。

E-MapReduce 到期时间：所购买的 E-MapReduce 的到期使用时间。

Core 节点信息：所有 Core 节点对应的配置，包含以下内容。

节点数：当前的节点数量和实际申请的节点数量。

CPU：单个节点的 CPU 的核数。

内存：单个节点的内存的容量。

数据盘类型：数据盘类型。

数据盘：单个节点的硬盘容量。

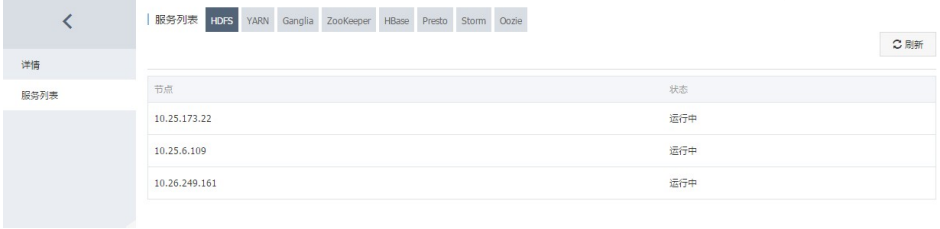
实例状态：包含创建中，正常和扩容中。

内网 IP：机器的内网 IP，可以被集群中的所有节点访问到。

ECS 到期时间：所购买的 ECS 的到期使用时间。

E-MapReduce 到期时间：所购买的 E-MapReduce 的到期使用时间。

集群详情页的tab下面增加了服务列表页，展示了HDFS，YARN等服务在集群节点上的运行状态，如下图所示



只有集群状态是空闲、运行中的集群会展示服务的运行状态，如果点击的服务在创建集群时未勾选，例如Storm，则会提示没有记录。

列表包括节点内网ip和状态，状态有运行中和未运行两种。如果某个节点上的服务状态是未运行，您可以通过master节点跳转，登录到对应节点上查看服务进程情况。

集群脚本的作用

集群，特别是包年包月集群，在使用过程中，可能会有新的安装第三方软件，修改集群运行环境的需求。集群脚本功能可以在集群创建好后批量选择节点，运行您指定的脚本，以实现个性化的需求。

集群脚本类似引导操作，您可以在集群创建好后安装很多目前集群尚未支持的软件到您的集群上，例如：

使用 yum 安装已经提供的软件。

直接下载公网上的一些公开的软件。

读取 OSS 中您的自有数据。

安装并运行一个服务，例如 Flink 或者 Impala，但需要编写的脚本会复杂些。

强烈建议您先用单个节点进行集群脚本的测试，测试都正确以后再在整个集群上操作。

如何使用

集群状态是空闲或者运行中的集群可以运行集群脚本，集群列表页面点击对应集群的**查看详情**按钮

左侧菜单单击**集群脚本**，即会进入该集群的集群脚本执行界面，右侧是已经执行过的集群脚本列表。

单击右上角**创建并执行**，进入创建界面。

填写创建界面上的配置项，选择执行的节点，点击执行，完成添加并执行操作。

集群脚本列表可以看到新创建的集群脚本，点击刷新可以更新集群脚本的状态。

点击**查看详情**可以看到脚本在各个节点上的运行情况，点击刷新可以更新脚本在各个节点上的运行状态。

只有空闲或者运行中的可用集群才能使用集群脚本功能。集群脚本适用于长期存在的集群，对按需创建的临时集群，应使用引导操作来完成集群初始化工作。

集群脚本会在您指定的节点上下载oss上的脚本并运行，根据返回值是否为0判断执行成功还是失败。如果运行状态是失败，您可以登录到各个节点上查看运行日志，运行日志记录在每个节点的/var/log/cluster-scripts/**clusterScriptId**目录下。如果集群配置了oss日志目录，运行日志也会上传到**osslogpath/clusterId/ip**/cluster-scripts/**clusterScriptId**目录下方便查看。

默认会使用 root 账户执行您指定的脚本，您可以在脚本中使用 su hadoop 切换到 Hadoop 账户。

集群脚本可能在部分节点上运行成功，部分节点上运行失败，例如节点重启导致的脚本运行失败。您可以在解决异常问题后，单独指定失败的节点再次运行。当集群扩容后，您也可以指定扩容的节点单独运行集群脚本。

1个集群同一时间只能运行一个集群脚本，如果有正在运行的集群脚本，无法提交执行新的集群脚本。每个集群最多保留10个集群脚本记录，超过10个需要将之前的记录删除才能创建新的集群脚本。

脚本的例子

类似引导操作的脚本，您可以在脚本中指定从 OSS 下载需要的文件，下面的例子会将 oss://yourbucket/myfile.tar.gz 这个文件下载到本地，并解压到 /yourdir 目录下：

```
#!/bin/bash
osscmd --id=<yourid> --key=<yourkey> --host=oss-cn-hangzhou-internal.aliyuncs.com get
oss://<yourbucket>/<myfile>.tar.gz ./<myfile>.tar.gz
mkdir -p /<yourdir>
tar -zxvf <myfile>.tar.gz -C /<yourdir>
```

osscmd 已预安装在节点上，可以直接调用来下载文件。

注意：OSS 地址 host 有内网地址、外网地址和 VPC 网络地址之分。如果用经典网络，需要指定内网地址，杭州是 oss-cn-hangzhou-internal.aliyuncs.com。如果用 VPC 网络，要指定 VPC 内网可访问的域名，杭州是 vpc100-oss-cn-hangzhou.aliyuncs.com。

脚本也可以通过 yum 安装额外的系统软件包，下面的例子会安装 ld-linux.so.2：

```
#!/bin/bash
yum install -y ld-linux.so.2
```

当您的包年包月集群服务即将到期的时候，您需要执行集群的续费操作，以继续您的 E-MapReduce 集群服务。集群续费包括 E-MapReduce 服务费的续费以及集群中 ECS 的续费。

续费入口

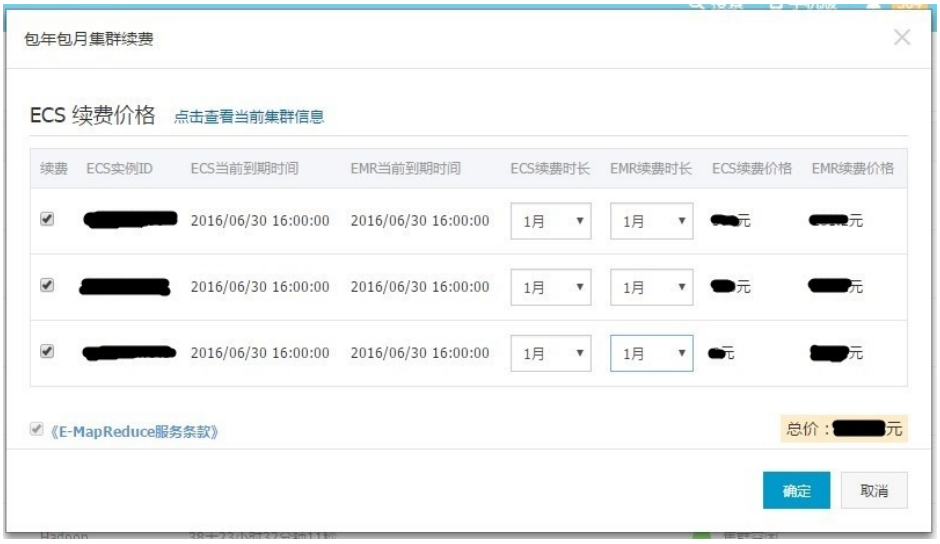
登录阿里云 E-MapReduce 集群列表页。

找到需要续费的集群条目。

单击相应集群条目操作栏下的**续费**按钮，进入集群续费界面。

续费界面

如下图所示：



续费：勾选需要续费的机器。

ECS 实例 ID：集群中这台机器的 ECS 实例 ID。

ECS 当前到期时间：这台 ECS 到期的时间。

E-MapReduce 当前到期时间：E-MapReduce 服务到期的时间。

ECS 续费时长：需要对这个节点续费的时长，目前支持 1-9 个月、1 年、2 年和 3 年的续费时长。

E-MapReduce 续费时长：对应的这个节点的 E-MapReduce 服务费的时长，推荐和 ECS 的设置一致。

ECS续费价格：ECS 节点对应的续费价格。

E-MapReduce 续费价格：E-MapReduce 服务对应节点的续费价格。

支付订单

注意：集群续费的费用为 ECS 续费价格与 E-MapReduce 服务产品价格的总和。当集群列表中存在未支付的订单时，您将无法执行集群的扩容和续费操作。

在单击**确定**按钮后，会弹出下单成功提示的提示框（该提示信息可能会有较长的时间延迟,请耐心等待）。

单击**支付订单**，跳转到订单支付页面。支付页面会显示您应付的总金额以及各订单的详情，其中一个为 E-MapReduce 产品费用订单，其它的为集群续费的 ECS 订单。

单击**确认支付**，完成付款。

完成支付之后，单击**已完成**，返回集群列表页面。

此时，在集群列表页上续费成功集群的到期时间会更新为续费之后的到期时间，对应的 ECS 的到期时间更新目前存在一定的延迟，一般约 3-5 分钟之后会更新成续费之后的到期时间。

若您只是确认了续费订单但并未进行支付，您可在集群列表页面中找到该集群条目，在其右侧的操作栏中会出现“前往支付”和“取消订单”的按钮。您可单击**前往支付**，完成对应的订单支付和集群扩容流程；或单击**取消订单**，以取消本次续费操作。

目前 E-MapReduce 创建集群的时候需要使用在 E-MapReduce 中创建的安全组。

主要是因为 E-MapReduce 创建的集群只开放了 22 端口，推荐您将 ECS 实例也按照功能划分，放于不同的用户安全组中。例如，E-MapReduce 的安全组为“E-MapReduce 安全组”，而您已有的安全组为“用户-安全组”，每个安全组按照不同的需要开启不同的访问控制。

如果需要和已有集群进行联动，请参考如下做法。

将 E-MapReduce 集群加入现有安全组

登录阿里云 E-MapReduce 控制台集群列表。

找到需加入安全组的集群条目，单击其操作中的**查看详情**，即会进入集群详情页。

在集群详情页上，找到“网络信息”下的所属安全组，查看集群所有 ECS 实例所在的安全组名称。

前往阿里云ECS的管理控制台，单击页面左侧的**安全组**按钮，在列表中找到步骤 3 中查看到的安全组条目。

单击该安全组操作中的**管理实例**，您会看到很多名字以 emr-xxx 开头的 ECS 实例，这些就是对应的 E-MapReduce 集群中的 ECS。

全选这些实例并单击移入安全组，选择想要加入的新的安全组即可。

将现有集群加入 E-MapReduce 安全组

和上面的操作一样，先找到现有集群所在安全组，重复如上的操作，移入 E-MapReduce 的安全组即可。如果是一些零散的机器，也可以直接在 ECS 的控制台界面上选择机器，然后通过下方的批量操作将集群移入 E-MapReduce 的安全组。

安全组的规则

一个 ECS 实例在多个不同的安全组的时候，安全组的规则是或的关系。举例来说就是，E-MapReduce 的安全组只开放了 22 端口，而“用户-安全组”开放了所有的端口。当 E-MapReduce 的集群加入“用户-安全组”以后，E-MapReduce 中的机器也会开放所有端口，所以在使用上请特别注意。

组件查看快捷入口

当集群创建完成以后，我们会为您的集群创建默认的绑定几个域名，来方便您访问您的开源组件：

- Yarn
- HDFS
- Ganglia

这些链接可以在集群管理中的**组件快捷入口**处找到。

注意：

在2.7.x 以上版本，或者是3.5.x以上版本，可以使用更加安全和方便的 UI 访问方式 KNOX。 参考这里

初次使用

当您初次使用组件快捷方式的时候，因为安全的原因，您需要遵循以下的步骤来打开您的安全组的访问

设置安全组访问

确认需要开放的安全组

emr在创建的时候，会默认为您创建一个emr专用的安全组。这个安全组默认的策略只开放了公网向内网方向的22端口。在集群的**详情**页面的**网络信息**处，找到集群对应所在的安全组名称和id。

获取公网访问ip

为了安全的访问集群组件，我们推荐安全组在设置策略的时候，只针对当前的公网访问ip来开放。要获得当前的访问，请访问http://ip.taobao.com/,在左下角会显示您当前的公网访问ip。

修改安全组策略

i. 前往**ECS的控制台**，在安全组id中输入刚才集群详情中的安全组id进行查找。找到以后点击条目后面的**配置规则**。
ii. 如果集群是**经典网络**，那么配置**公网入方向**
iii. 如果集群是**VPC**，那么配置**内网入方向**

新增三条规则，如下


编辑安全组规则
网卡类型：
内网
规则方向：
入方向
授权策略：
允许
协议类型：
TCP
快速开放用于远程登录的端口：
开放22端口(Linux)
开放3389端口(Windows)
* 端口范围：
80/80
取值范围从1到65535；设置格式例如"1/200"、"80/80"，其中 -1/-1 代表不限制端口。 教我设置
授权类型：
地址段访问
* 授权对象：
42.120.74.109
请根据实际场景设置授权对象的CIDR，另外，0.0.0.0/0代表允许或拒绝所有IP的访问，设置时请务必谨慎。 教我设置
优先级：
1
优先级可选范围为1-100，默认值为1，即最高优先级。


编辑安全组规则
网卡类型：
内网
规则方向：
入方向
授权策略：
允许
协议类型：
TCP
快速开放用于远程登录的端口：
开放22端口(Linux)
开放3389端口(Windows)
* 端口范围：
5901/5901
取值范围从1到65535；设置格式例如"1/200"、"80/80"，其中 -1/-1 代表不限制端口。 教我设置
授权类型：
地址段访问
* 授权对象：
42.120.74.109
请根据实际场景设置授权对象的CIDR，另外，0.0.0.0/0代表允许或拒绝所有IP的访问，设置时请务必谨慎。 教我设置
优先级：
1
优先级可选范围为1-100，默认值为1，即最高优先级。

全部完成以后，在策略列表中会看到新增的3条策略

vi. 这个时候网络部分的设置就完成了。现在已经安全的开放了网络的访问路径。

设置开源组件访问

完成了网络的设置，这时候就可以设置访问权限了。默认的情况下，新集群的初次访问是没有设置用户名和密码的。所以无法通过http认证。您需要参照如下步骤设置您的访问权限

1. 首先打开集群**详情**页面。
2. 点击左侧的**组件快捷入口**。
3. 点击右上方的**访问设置**按钮，会提示您设置访问用户名和密码。
4. 设置完成以后，再点击下方的链接，输入设置好的用户名和密码，就能正常的访问UI界面了。

注意：

1. 只能存在一个用户名和密码，所以每次设置以后都将替换之前设置的用户民和密码。
2. 目前只有2.3及以上的版本才支持此功能。

在实际使用中，我们可能会发现我们集群中的节点，尤其是 Master 节点的 CPU 或者内存不够了。目前 EMR 还不能支持直接的升级配置。我们推荐您可以通过如下的方式来完成节点的配置升级。

注意： 包年包月集群才支持升级配置。

节点配置升级引导

1. 确认要升级的集群的ID，这里举例：C-123456789ABCDF
2. 复制集群名字到 ECS 控制台，选择对应的地域，按照实例名称搜索，输入集群ID，点击搜索，就会展示出所有的集群的节点机器。
3. 点击其中一台 ECS 节点右侧的**升降配**，并选择升级配置。
4. 选择要升级的机型，并支付差价费用。
5. 在 ECS 控制台上选择**更多**，再选择重启，将改节点重启以使新的配置生效
6. 重复以上的3、4、5步，直到所有的节点都升级完毕

   修改集群配置，使得Yarn可以使用新增的资源

   i. 修改Yarn服务的yarn-site.xml文件

   修改 yarn.nodemanager.resource.memory-mb的值为 机器内存 * 0.8 这里单位用MB。修改 yarn.scheduler.maximum-allocation-mb 的值为 机器内存 * 0.8 单位MB。举例来说，我现在新的配置下，内存是32G那么这里就配置为

   ```
   yarn.nodemanager.resource.memory-mb=26214
   yarn.scheduler.maximum-allocation-mb=26214
   ```

   如果您的集群还没有支持页面修改的方式，那么需要登陆到节点上，修改每一个节点的/etc/emr/hadoop-conf/yarn-site.xml文件中对应的配置值。

   iii. 重启Yarn服务，一般只要重启worker节点就行，但是重启后nodemanger的端口变了，在yarn控制台看到很多节点lost，所以最好resourcemanager也重启下。
8. 工单联系 EMR 团队，提供升级以后的新的机型配置信息。EMR 团队会进行配置同步，保证后续的集群续费和操作正常。

作业

登录阿里云 E-MapReduce 控制台作业列表。

单击该页右上角的**创建作业**，进入创建作业页面。

填写作业名称。

选择 Hadoop 作业类型。表示创建的作业是一个 Hadoop Mapreduce 作业。这种类型的作业，其后台实际上是通过以下的方式提交的 Hadoop 作业。

```
hadoop jar xxx.jar [MainClass] -Dxxx ....
```

在**应用参数**中填写提交该 job 需要提供的命令行参数。这里需要说明的是，这个选项框中需要填写的内容从 hadoop jar 后面的第一个参数开始填写。也就是说，选项框中第一个要填写的是运行该作业需要提供的 jar 包所在地址，然后后面紧跟 [MainClass] 以及其他用户可以自行提供的命令行参数。

举个例子，假设用户想要提交一个 Hadoop 的 sleep job，该 jo b不读写任何数据，只是提交一些 mapper 和 reducer task 到集群中，每个 task sleep 一段时间，然后 job 成功。在 Hadoop 中（hadoop-2.6.0 为例）以，该 job 被打包在 Hadoop 发行版的 hadoop-mapreduce-client-jobclient-2.6.0-tests.jar 中。那么，若是在命令行中提交该 job，则命令如下：

```
hadoop jar /path/to/hadoop-mapreduce-client-jobclient-2.6.0-tests.jar sleep -m 3 -r 3 -mt 100 -rt 100
```

要在 E-MapReduce 中配置这个作业，那么作业配置页面的“应用参数”选项框中，需要填写的内容即为：

```
/path/to/hadoop-mapreduce-client-jobclient-2.6.0-tests.jar sleep -m 3 -r 3 -mt 100 -rt 100
```

需要注意的是，这里用的 jar 包路径是 E-MapReduce 宿主机上的一个绝对路径，这种方式有一个问题，就是用户可能会将这些 jar 包放置在任何位置，而且随着集群的创建和释放，这些 jar 包也会跟着释放而变得不可用。所以，请使用以下方法：

用户将自己的 jar 包上传到 OSS 的 bucket 中进行存储，当配置 Hadoop 的参数时，单击**选择 OSS 路径**，从 OSS 目录中进行选择要执行的 jar 包。系统会为用户自动补齐 jar 包所在的 OSS 地址。请务必将代码的 jar 的前缀切换为 ossref（单击**切换资源类型**），以保证这个 jar 包会被 E-MapReduce 正确下载。

单击**确定**，该 jar 包所在的 OSS 路径地址就会自动填充到“应用参数”选项框中。作业提交的时候，系统能够根据这个路径地址自动从 OSS 找到相应的 jar 包。

在该 OSS 的 jar 包路径后面，即可进一步填写作业运行的其他命令行参数。

选择执行失败后策略。

单击**确认**，作业配置即定义完成。

上面的例子中，sleep job 并没有数据的输入输出，如果作业要读取数据，并输出处理结果（比如 wordcount），则需要指定数据的 input 路径和 output 路径。用户可以读写 E-MapReduce 集群 HDFS 上的数据，同样也可以读写 OSS 上的数据。如果需要读写 OSS 上的数据，只需要在填写 input 路径和 output 路径时，数据路径写成 OSS 上的路径地址即可，例如：

```
jar ossref://emr/checklist/jars/chengtao/hadoop/hadoop-mapreduce-examples-2.6.0.jar randomtextwriter  -D
mapreduce.randomtextwriter.totalbytes=320000 oss://emr/checklist/data/chengtao/hadoop/Wordcount/Input
```

E-MapReduce 中，用户申请集群的时候，默认为用户提供了 Hive 环境，用户可以直接使用 Hive 来创建和操作自己的表和数据。操作步骤如下。

用户需要提前准备好 Hive SQL 的脚本，例如：

```
USE DEFAULT;

DROP TABLE uservisits;

CREATE EXTERNAL TABLE IF NOT EXISTS uservisits (sourceIP STRING,destURL STRING,visitDate
STRING,adRevenue DOUBLE,user
Agent STRING,countryCode STRING,languageCode STRING,searchWord STRING,duration INT ) ROW
FORMAT DELIMITED FIELDS TERMI
NATED BY ',' STORED AS SEQUENCEFILE LOCATION '/HiBench/Aggregation/Input/uservisits';

DROP TABLE uservisits_aggre;

CREATE EXTERNAL TABLE IF NOT EXISTS uservisits_aggre ( sourceIP STRING, sumAdRevenue DOUBLE)
STORED AS SEQUENCEFILE LO
CATION '/HiBench/Aggregation/Output/uservisits_aggre';
INSERT OVERWRITE TABLE uservisits_aggre SELECT sourceIP, SUM(adRevenue) FROM uservisits GROUP BY
sourceIP;
```

将该脚本保存到一个脚本文件中，例如叫 uservisits_aggre_hdfs.hive，然后将该脚本上传到 OSS 的某个目录中（例如：oss://path/to/uservisits_aggre_hdfs.hive）。

登录阿里云 E-MapReduce 控制台作业列表。

单击该页右上角的**创建作业**，进入创建作业页面。

填写作业名称。

选择 Hive 作业类型，表示创建的作业是一个 Hive 作业。这种类型的作业，其后台实际上是通过以下的方式提交。

```
hive [user provided parameters]
```

在**应用参数**选项框中填入 Hive 命令后续的参数。例如，如果需要使用刚刚上传到 OSS 的 Hive 脚本，则填写的内容如下：

```
-f ossref://path/to/uservisits_aggre_hdfs.hive
```

您也可以单击**选择 OSS 路径**，从 OSS 中进行浏览和选择，系统会自动补齐 OSS 上 Hive 脚本的绝对路径。请务必将 Hive 脚本的前缀修改为 ossref（单击**切换资源类型**），以保证 E-MapReduce 可以正确下载该文件。

选择执行失败后策略。

单击**确定**，Hive 作业即定义完成。

E-MapReduce 中，用户申请集群的时候，默认为用户提供了 Pig 环境，用户可以直接使用 Pig 来创建和操作自己的表和数据。操作步骤如下。

用户需要提前准备好 Pig 的脚本，例如：

```
```shell
/*
* Licensed to the Apache Software Foundation (ASF) under one
* or more contributor license agreements. See the NOTICE file
* distributed with this work for additional information
* regarding copyright ownership. The ASF licenses this file
* to you under the Apache License, Version 2.0 (the
* "License"); you may not use this file except in compliance
* with the License. You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/
```
```

```
-- Query Phrase Popularity (Hadoop cluster)

-- This script processes a search query log file from the Excite search engine and finds search phrases that
occur with particular high frequency during certain times of the day.

-- Register the tutorial JAR file so that the included UDFs can be called in the script.
REGISTER oss://emr/checklist/jars/chengtao/pig/tutorial.jar;

-- Use the PigStorage function to load the excite log file into the “raw” bag as an array of records.
-- Input: (user,time,query)
raw = LOAD 'oss://emr/checklist/data/chengtao/pig/excite.log.bz2' USING PigStorage('\t') AS (user, time,
query);

-- Call the NonURLDetector UDF to remove records if the query field is empty or a URL.
clean1 = FILTER raw BY org.apache.pig.tutorial.NonURLDetector(query);

-- Call the ToLower UDF to change the query field to lowercase.
clean2 = FOREACH clean1 GENERATE user, time, org.apache.pig.tutorial.ToLower(query) as query;

-- Because the log file only contains queries for a single day, we are only interested in the hour.
-- The excite query log timestamp format is YYMMDDHHMMSS.
-- Call the ExtractHour UDF to extract the hour (HH) from the time field.
houred = FOREACH clean2 GENERATE user, org.apache.pig.tutorial.ExtractHour(time) as hour, query;

-- Call the NGramGenerator UDF to compose the n-grams of the query.
ngramed1 = FOREACH houred GENERATE user, hour,
flatten(org.apache.pig.tutorial.NGramGenerator(query)) as ngram;

-- Use the DISTINCT command to get the unique n-grams for all records.
ngramed2 = DISTINCT ngramed1;

-- Use the GROUP command to group records by n-gram and hour.
hour_frequency1 = GROUP ngramed2 BY (ngram, hour);

-- Use the COUNT function to get the count (occurrences) of each n-gram.
hour_frequency2 = FOREACH hour_frequency1 GENERATE flatten($0), COUNT($1) as count;

-- Use the GROUP command to group records by n-gram only.
-- Each group now corresponds to a distinct n-gram and has the count for each hour.
uniq_frequency1 = GROUP hour_frequency2 BY group::ngram;

-- For each group, identify the hour in which this n-gram is used with a particularly high frequency.
-- Call the ScoreGenerator UDF to calculate a "popularity" score for the n-gram.
uniq_frequency2 = FOREACH uniq_frequency1 GENERATE flatten($0),
flatten(org.apache.pig.tutorial.ScoreGenerator($1));

-- Use the FOREACH-GENERATE command to assign names to the fields.
uniq_frequency3 = FOREACH uniq_frequency2 GENERATE $1 as hour, $0 as ngram, $2 as score, $3 as
count, $4 as mean;

-- Use the FILTER command to move all records with a score less than or equal to 2.0.
filtered_uniq_frequency = FILTER uniq_frequency3 BY score > 2.0;

-- Use the ORDER command to sort the remaining records by hour and score.
ordered_uniq_frequency = ORDER filtered_uniq_frequency BY hour, score;
```

```
-- Use the PigStorage function to store the results.
-- Output: (hour, n-gram, score, count, average_counts_among_all_hours)
STORE ordered_uniq_frequency INTO 'oss://emr/checklist/data/chengtao/pig/script1-hadoop-results'
USING PigStorage();
```

将该脚本保存到一个脚本文件中，例如叫 script1-hadoop-oss.pig，然后将该脚本上传到 OSS 的某个目录中（例如：oss://path/to/script1-hadoop-oss.pig）。

进入阿里云 E-MapReduce 控制台作业列表。

单击该页右上角的**创建作业**，进入创建作业页面。

填写作业名称。

选择 Pig 作业类型，表示创建的作业是一个 Pig 作业。这种类型的作业，其后台实际上是通过以下的方式提交。

```
pig [user provided parameters]
```

在**应用参数**选项框中填入 Pig 命令后续的参数。例如，如果需要使用刚刚上传到 OSS 的 Pig 脚本，则填写如下：

```
-x mapreduce ossref://emr/checklist/jars/chengtao/pig/script1-hadoop-oss.pig
```

您也可以单击**选择 OSS 路径**，从 OSS 中进行浏览和选择，系统会自动补齐 OSS 上 Pig 脚本的绝对路径。请务必将 Pig 脚本的前缀修改为 ossref（单击**切换资源类型**），以保证 E-MapReduce 可以正确下载该文件。

选择执行失败后策略。

单击**确定**，Pig 作业即定义完成。

进入阿里云 E-MapReduce 控制台作业列表。

单击该页右上角的**创建作业**，进入创建作业页面。

填写作业名称。

选择 Spark 作业类型，表示创建的作业是一个 Spark 作业。Spark 作业在 E-MapReduce 后台使用以下的方式提交：

```
spark-submit [options] --class [MainClass] xxx.jar args
```

在**应用参数**选项框中填写提交该 Spark 作业需要的命令行参数。请注意，应用参数框中只需要填写“spark-submit”之后的参数即可。以下分别示例如何填写创建 Spark 作业和 pyspark 作业的参数。

创建 Spark 作业

新建一个 Spark WordCount 作业。

作业名称： Wordcount

类型：选择 Spark

应用参数：

在命令行下完整的提交命令是：

```
spark-submit --master yarn-client --driver-memory 7G --executor-memory 5G --executor-cores 1 --num-executors 32 --class com.aliyun.emr.checklist.benchmark.SparkWordCount emr-checklist_2.10-0.1.0.jar oss://emr/checklist/data/wc oss://emr/checklist/data/wc-counts 32
```

在 E-MapReduce 作业的应用参数框中只需要填写：

```
--master yarn-client --driver-memory 7G --executor-memory 5G --executor-cores 1 --num-executors 32 --class com.aliyun.emr.checklist.benchmark.SparkWordCount ossref://emr/checklist/jars/emr-checklist_2.10-0.1.0.jar oss://emr/checklist/data/wc oss://emr/checklist/data/wc-counts 32
```

需要注意的是：作业 Jar 包保存在 OSS 中，引用这个 Jar 包的方式是 ossref://emr/checklist/jars/emr-checklist_2.10-0.1.0.jar。您可以单击**选择 OSS 路径**，从 OSS 中进行浏览和选择，系统会自动补齐 OSS 上 Spark 脚本的绝对路径。请务必将默认的“oss”协议切换成“ossref”协议。

创建 pyspark 作业

E-MapReduce 除了支持 Scala 或者 Java 类型作业外，还支持 python 类型 Spark 作业。以下新建一个 python 脚本的 Spark Kmeans 作业。

作业名称：Python-Kmeans

类型：Spark

应用参数：

```
--master yarn-client --driver-memory 7g --num-executors 10 --executor-memory 5g --executor-cores 1  ossref://emr/checklist/python/kmeans.py oss://emr/checklist/data/kddb 5 32
```

支持 Python 脚本资源的引用，同样使用“ossref”协议。

pyspark 目前不支持在线安装 Python 工具包。

选择执行失败后策略。

单击**确定**，Spark 作业即定义完成。

注意：Spark SQL 提交作业的模式默认是 yarn-client 模式。

进入阿里云 E-MapReduce 控制台作业列表。

单击该页右上角的**创建作业**，进入创建作业页面。

填写作业名称。

选择 Spark SQL 作业类型，表示创建的作业是一个 Spark SQL 作业。Spark SQL 作业在 E-MapReduce 后台使用以下的方式提交：

```
spark-sql [options] [cli option]
```

在“应用参数”选项框中填入 Spark SQL 命令后续的参数。

-e 选项

-e 选项可以直接写运行的 SQL，在作业**应用参数**框中直接输入，如下所示：

```
-e "show databases;"
```

-f 选项

-f 选项可以指定 Spark SQL 的脚本文件。通过将编写好的 Spark SQL 脚本文件放在 OSS 上，可以更灵活，建议您使用这种运行方式。如下所示：

```
-f ossref://your-bucket/your-spark-sql-script.sql
```

选择执行失败后策略。

单击**确定**，Spark SQL 作业即定义完成。

注意：目前 Shell 脚本默认是使用 Hadoop 用户执行的，如果需要使用 root 用户，可以使用 sudo。请谨慎使用 Shell 脚本作业。

进入阿里云 E-MapReduce 控制台作业列表。

单击该页右上角的**创建作业**，进入创建作业页面。

填写作业名称。

选择 Shell 作业类型，表示创建的作业是一个 Bash Shell 作业。

在“应用参数”选项框中填入 Shell 命令后续的参数。

-c 选项

-c 选项可以直接设置要运行的 Shell 脚本，在作业**应用参数**框中直接输入，如下所示：

```
-c "echo 1; sleep 2; echo 2; sleep 4; echo 3; sleep 8; echo 4; sleep 16; echo 5; sleep 32; echo 6; sleep 64; echo 8; sleep 128; echo finished"
```

-f 选项

-f 选项可以直接运行 Shell 脚本文件。通过将 Shell 脚本文件上传到 OSS 上，在 job 参数里面可以直接制定 OSS 上的 Shell 脚本，比使用 -c 选项更加灵活，如下所示：

```
```

```
-f ossref://mxbucket/sample/sample-shell-job.sh
```

选择执行失败后策略。

单击**确定**，Shell 作业即定义完成。

注意：只有 E-MapReduce 产品版本 V1.3.0（包括）以上支持 Sqoop 作业类型。在低版本集群上运行 Sqoop 作业会失败，errlog 会报不支持的错误。参数细节请参见数据传输 Sqoop。

进入阿里云 E-MapReduce 控制台作业列表。

单击该页右上角的**创建作业**，进入创建作业页面。

填写作业名称。

选择 Sqoop 作业类型，表示创建的作业是一个 Sqoop 作业。Sqoop 作业在 E-MapReduce 后台使用以下的方式提交：

```
sqoop [args]
```

在**应用参数**选项框中填入 Sqoop 命令后续的参数。

选择执行失败后策略。

单击**确定**，Sqoop 作业即定义完成。

作业的创建

一个新作业可以在任何时候被创建。被创建的作业目前只可以在所创建的 Region 内被使用。

作业的克隆

完全的克隆一个已经存在作业的配置。同样也只限定在同一个 Region 内。

作业的修改

如果要将作业加入到一个执行计划中，需要保证该执行计划当前没有在运行中，同时也需要保证执行计划的周期调度没有在调度中，这个时候才可以修改该作业。

如果要将这个作业加入到多个执行计划中，需停止要加入的所有执行计划的运行和周期调度后才可以修改。因为修改作业会导致所有使用该作业的执行计划也发生变化，可能会导致正在执行的或者周期调度的执行计划的错误。

如果想要进行调试，推荐使用克隆功能，完成调试后，替换执行计划中的原作业。

作业的删除

和修改一样，只有在作业加入的执行计划当前没有在运行中，同时周期调度也没有在调度中的情况下，才能被删除。

在创建作业过程中，支持在作业参数中设置时间变量通配符。

变量通配符格式

E-MapReduce 所支持的变量通配符的格式为${dateexpr-1d} 或者 ${dateexpr-1h} 的格式。例如，假设当前时间为“20160427 12:08:01”：

> 如果在作业参数中写成 ${yyyyMMdd HH:mm:ss-1d}，那么这个参数通配符在真正执行的时候会被替换成 “20160426 12:08:01”，即在当前日期上减了一天并精确到了秒。
>
> 如果写成 ${yyyyMMdd-1d}，则执行时会替换成 “20160426”，表示当前日期的前一天。
>
> 如果写成 ${yyyyMMdd}，则会被替换成“20160427”，直接表示当前的日期。

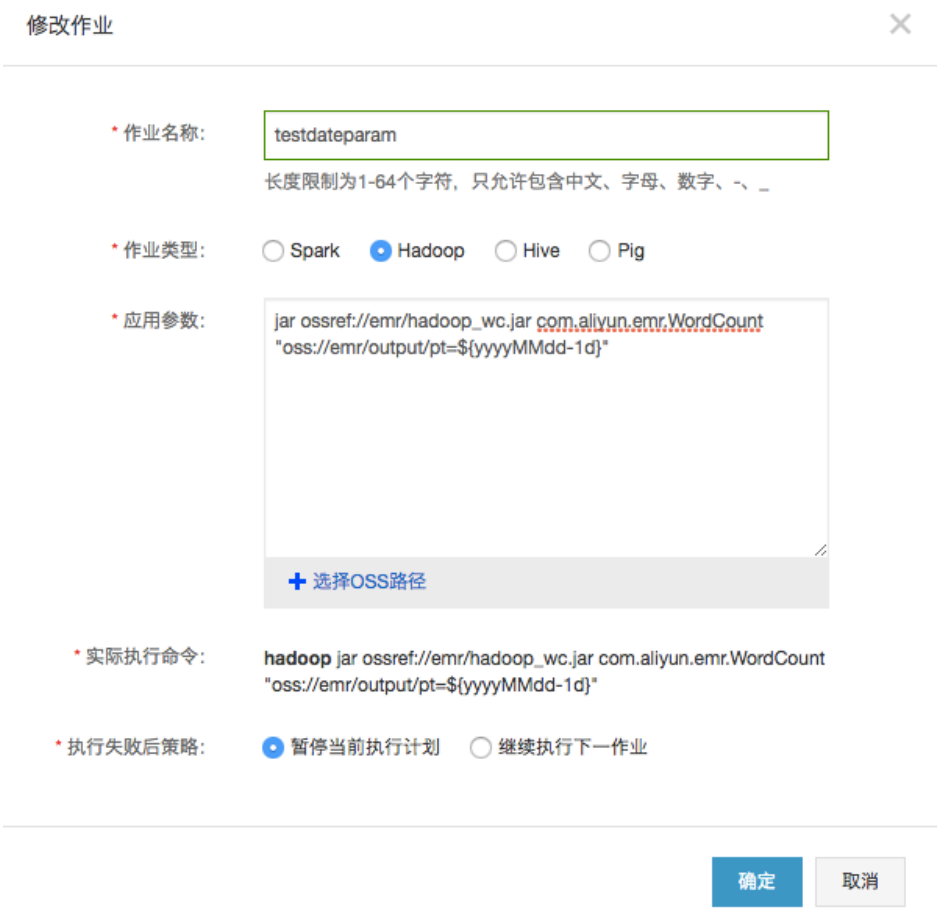
dateexpr 表示标准的时间格式表达式，对应的时间会按照该表达式指定的格式进行格式化，后面可以再跟上对应加减的时间。支持表达式后面的加减 1d（加减1天），也可以写成加减 N 天或者加减 N 小时，例如 ${yyyyMMdd-5d}、${yyyyMMdd+5d}、${yyyyMMdd+5h}、${yyyyMMdd-5h}都可以支持，对应的替换方式和前面描述的一致。

注意：目前 E-MapReduce 仅支持小时和天维度的加减，即只支持在 dateexpr 后面 +Nd、-Nd、+Nh、-Nh 的形式（dateexpr 为时间格式表达式，N 为整数）。

示例

下图作业中的**应用参数**在实际执行时会被替换成：

```
jar ossref://emr/jar/hadoop/hadoop_wc.jar com.aliyun.emr.WordCount  oss://emr/output/pt=20160426
```



执行计划

执行计划是一组作业的集合，他们通过调度上的配置，可以被一次性或者周期性的执行。他可以在一个现有的 E-MapReduce 集群上运行，也可以动态的按需创建出一个临时集群来运行作业。它最大的优势就是跑多少就用多少资源，最大化的节省资源的浪费。

创建执行计划的步骤如下：

登录阿里云 E-MapReduce 控制台执行计划页面。

选择地域（Region）。

单击右上角的**创建执行计划**，进入创建执行计划页面。

在选择集群方式页面上，有两个选项，分别是“按需创建”和“已有集群”。

按需创建：创建一个全新的集群，用来运行作业。

一次性调度的执行计划，会在开始执行的时候创建对应配置的集群，并在运行完成以后释放该集群。具体创建参数说明参考**创建集群**。

周期调度的执行计划，会在每一个调度周期开始时，按照用户的设置创建出一个新的集群运行作业，并在运行结束后，释放集群。

已有集群：使用一个已有的集群，并且该集群要符合以下要求。如果选择“已有集群”，则进入选择集群页面。用户可选择要将该执行计划关联到的集群。

i. 目前只有“运行中”和“空闲”这 2 个状态的集群可以被提交执行计划。

单击**下一步**，进入配置作业页面。左边表中会列出用户所有的作业，可以单击选中需要执行的作业，然后单击中央的右向按钮将作业加入已选作业队列。已选作业队列中的作业会被按排列顺序提交到集群中执行。同一个作业可以被添加多次，就会多次执行。如果您还没有创建任何作业，请您先参见创建作业的操作说明创建作业。

单击**下一步**，进入配置调度方式页面。配置项说明如下：

执行计划名称:长度限制为 1-64 个字符，只允许包含中文、字母、数字、’-‘、’_’。

调度策略

手动执行：创建完执行计划以后，并不会自动执行。需要用户手动执行。一旦已经在运行中了，不可以被再次执行。

周期调度：创建完执行计划以后，周期调度功能会立刻启动。并在用户设置的调度时间点上开始执行。可以在列表页面关闭周期调度。当调度执行开始的时候，上一周期的执行还未结束，本次调度就会被忽略。

调度周期设置：可以有天或小时两种调度的周期。天默认是一天，且无法更改。若选择小时，则可设置具体间隔时间，范围从 1-23。

首次执行时间：调度有效的开始时间。从这个时间开始，按照调度周期进行周期调度。第一次调度按照实际的时间满足要求的最近一个时间点开始调度。

单击**确认提交**，完成执行计划的创建。

其他

周期调度示例

这个设置表示，从 2015 年 10 月 31 日 10 点 0 分开始第一次调度，以后每隔一天调度一次。第二次调度是 2015 年 11 月 1 日 10 点 0 分。

作业的执行顺序

执行计划中的作业，按照用户选择的作业在作业列表中的顺序，从第一个开始一直执行到最后一个。

多个执行计划的执行顺序

每一个执行计划都可以看做是一个整体。当多个执行计划被提交到同一个集群上后，每一个执行计划都会按照自身内部的作业顺序提交作业，和单个执行计划的顺序是一致。而多个执行计划之间的作业是并行的。

实践示例 —— 前期作业调试

在作业的调试阶段，如果经常用按需自动创建集群的方式会比较慢，每次都需要启动集群会花费不少的时间。推荐的方式是：先手动创建一个集群，然后在执行计划中，选择关联该集群来运行作业，并设置调度方式为立即执行。调试的时候，每次都通过单击执行计划列表页上的“立即运行”来多次运行，查看结果。一旦作业调试完成，修改执行计划。将关联现有集群的方式，修改为按需创建新集群。并将调度方式修改为周期调度（视实际情况而定）。后续就可以按需自动跑任务了。

登录阿里云 E-MapReduce 控制台执行计划页面。

找到相应的执行计划条目，单击其操作栏中的**管理**按钮，进入执行计划管理页面。在这里您可以：

查看执行计划详情

您可以查看到该执行计划的名称、关联集群、作业配置等基本信息，还有其调度策略、调度状态、报警信息等。

修改执行计划

注意事项

一个执行计划当前并没有在运行中且它没有在被周期调度中，才能够被修改。如果是一个立即执行的执行计划，只要它当前没有在运行中就可以被修改。如果是一个周期调度的执行计划，首先要等待它当前的运行结束，然后确认它是否正在被周期调度中，如果是请单击“停止调度”，然后才可修改执行计划。

独立修改

每一个单独的模块，都可以被独立的修改。单击条目右侧的**修改**按钮即可进行修改。

配置报警通知

共有三类报警通知：

启动超时通知：周期调度任务，在指定时间点，没有正常调度，并在 10 分钟的超时时间内，仍然没有调度执行，发送报警通知。

执行失败通知：执行计划内有任何一个作业失败，发送报警通知。

执行成功通知：执行计划内的所有作业执行成功，发送通知。

运行与查看结果

在“基本信息”中的“调度状态”右侧，当执行计划可以被运行的时候，会有**立即运行**的按钮，单击后即会产生一次调度执行。

在页面的最下方，是运行记录的部分，会显示执行计划每次被执行的实例，可以方便用户查看对应的作业列表和日志。

展示您所有的执行计划的基本信息，如下图所示：

执行计划 ID/名称：执行计划的 ID 和对应的名称。

最近执行集群：最近一次执行该执行计划的集群，是一个按需创建的集群或是一个关联的已有集群。

如果是按需的，那么在集群的名字下面会显示（自动创建），表示这个集群是有 E-MapReduce 按需自动创建出来的，运行完成以后会自动释放。

最近一次运行情况：最近一次执行计划的运行状态。

开始时间：最近一次执行计划开始的时间。

运行时间：最近一次执行计划的运行时长。

运行状态：最近一次执行计划的运行状态。

调度状态：是否在调度中还是调度已经停止。只有周期作业才会有调度状态。

操作

立即运行：非调度中且非运行中，才能手动运行。单击后，会立刻运行一次执行计划。

启动调度/停止调度：当调度停止状态时出现启动调度，单击即会开始调度。当调度运行中显示停止调度，单击即会停止调度。只有周期执行计划才有此按钮。

运行记录：单击会进入执行计划中的作业日志查看页面。

更多

修改：修改执行计划的配置。调度中或者运行中的执行计划不能被修改。

删除：删除执行计划。调度中或者运行中的执行计划不能被删除。

执行记录查看

登录阿里云 E-MapReduce 控制台执行计划页面。

单击相应执行计划条目右侧操作中的**运行记录**，即可进入执行记录页面。如下图所示：

执行序列 ID： 本次执行记录的执行次数，表明了它在整个执行队列中的顺序位置。比如第一次执行就是1，第n次就是n。

运行状态： 每一次执行记录的运行状态。

开始时间： 执行计划开始运行的时间。

运行时间： 到查看页面当时为止，一共运行的时间。

执行集群： 执行计划运行的集群，可以是按需也可以是一个关联的已有集群。点击可以前往集群的详情页查看。

操作

- 查看作业列表：单击该按钮，即可进入单次执行计划的作业列表查看每个作业的执行情况。

作业记录查看

在这里可以查看单次执行计划的执行记录中的作业列表，以及每一个作业的具体信息，如下图所示：

作业执行 ID：作业每一次执行都会产生一个对应的 ID，它和作业本身的 ID 是不同的。这个 ID 可以想象成作业每运行一次的一个记录的唯一标示，您可用其在 OSS 上进行日志查询。

作业名称：作业的名称。

状态：作业的运行状态。

作业类型：作业的类型。

开始时间：这个作业开始运行的时间，都已经转换为本地时间。

运行时间：这个作业一共运行了多久，以秒为单位。

操作

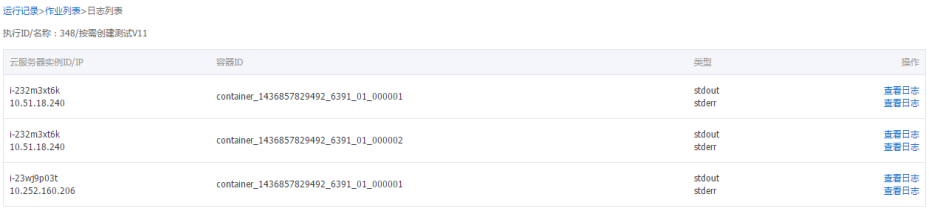
停止作业：无论作业在提交中还是在运行中，都可以被停止。如果是提交中，那么停止作业会让这个作业不执行。如果是在运行中，那么这个作业会被 kill 掉。

stdout：记录 master 进程的标准输出（即通道 1）的所有输出内容。如果运行作业的集群没有打开日志保存，不会有此查看功能。

stderr：记录 master 进程的诊断输出（即通道 2）的所有输出内容。如果运行作业的集群没有打开日志保存，不会有此查看功能。

查看作业实例：查看作业的所有 worker 的节点的日志。如果运行作业的集群没有打开日志保存，不会有此查看功能。

作业worker日志查看



云服务器实例 ID/IP：运行作业的 ECS 实例 ID，以及对应的内网 IP。

容器 ID：Yarn 运行的容器 ID。

类型：日志的不同类型。stdout 与 stderr，来自不同的输出。

操作

- 查看日志：单击对应的类型，查看对应的日志。

为了最大化利用集群的可用计算资源，目前可以将多个执行计划挂载到同一个集群来达到多个执行计划并行执行的效果。

总结为如下几点：

同一个执行计划内的作业是串行执行的，默认认为前序作业执行完毕，后序作业才能被提交执行。

在集群资源足够的情况下，如果想要让多个作业达到并行执行的效果，需要创建多个不同的执行计划，同时关联到同一个集群提交运行即可（默认一个集群最多支持 20 个执行计划同时执行）。

目前管控系统支持将关联到同一个集群的执行计划并行提交到 Yarn，但如果集群本身资源不足，还是可能阻塞在 Yarn 队列中等待调度。

创建执行计划并关联到集群的流程参见：执行计划创建流程。

您可以在报警标签下的页面，进行联系人和报警接收组的添加和修改等功能。每个报警接收组可关联一到多个联系人。

添加联系人

登录阿里云 E-MapReduce 控制台的报警管理联系人页面。

单击**添加联系人**，进行联系人的添加操作界面。

输入联系人姓名和手机号码，并单击发送验证码。获取对应的手机验证码，输入验证码。

单击**确定**，完成联系人的添加操作。

添加报警接收组

登录阿里云 E-MapReduce 控制台的报警接收组页面。

单击**添加报警接收组**，进入报警接收组的添加界面。

填写报警接收组名称。

添加联系人。

执行计划关联报警接收组

当完成了联系人以及报警接收组等记录的录入操作之后，可以在执行计划的管理页面，为执行计划关联对应的

报警接收组。

登录阿里云 E-MapReduce 控制台执行计划页面。

单击执行计划条目右侧操作中的管理按钮，进入执行计划的管理页面。

在管理页面的“报警信息”栏中，可以选择打开“报警通知”，并关联具体的报警接收组。

打开“报警通知”之后，当执行计划执行完成时，关联的报警接收组中的联系人，都将会接收到短信通知。短信内容包含执行计划名、作业的执行情况（成功多少、失败多少）、对应的执行集群名以及具体的执行时长信息。

软件配置的作用

Hadoop、Hive、Pig 等软件含有大量的配置，当需要对其软件配置进行修改时，就可以使用软件配置功能来实现。例如，HDFS 服务器的服务线程数目 dfs.namenode.handler.count 默认是 10，假设要加大到 50；HDFS 的文件块的大小 dfs.blocksize 默认是 128MB，假设系统都是小文件，想要改小到 64MB。

目前这个操作只能在集群启动的时候执行一次。

如何使用

登录阿里云 E-MapReduce 控制台集群列表。

在上方选择所在的地域（Region），所创建集群将会在对应的Region内。

单击**创建集群**，即会进入创建集群的操作界面。

在创建集群的软件配置这一步中可以看到所有包含的软件以及对应的版本。若想修改集群的配置，可以通过软件配置（可选）框选择相应的 json 格式配置文件，对集群的默认参数进行覆盖或添加。json 文件的样例内容如下

```
{
"configurations": [
{
"classification": "core-site",
"properties": {
"fs.trash.interval": "61"
}
},
{
```

```
"classification": "hadoop-log4j",
"properties": {
"hadoop.log.file": "hadoop1.log",
"hadoop.root.logger": "INFO",
"a.b.c": "ABC"
}
},
{
"classification": "hdfs-site",
"properties": {
"dfs.namenode.handler.count": "12"
}
},
{
"classification": "mapred-site",
"properties": {
"mapreduce.task.io.sort.mb": "201"
}
},
{
"classification": "yarn-site",
"properties": {
"hadoop.security.groups.cache.secs": "251",
"yarn.nodemanager.remote-app-log-dir": "/tmp/logs1"
}
},
{
"classification": "httpsfs-site",
"properties": {
"a.b.c.d": "200"
}
},
{
"classification": "capacity-scheduler",
"properties": {
"yarn.scheduler.capacity.maximum-am-resource-percent": "0.2"
}
},
{
"classification": "hadoop-env",
"properties": {
"BC":"CD"
},
"configurations":[
{
"classification":"export",
"properties": {
"AB":"${BC}",
"HADOOP_CLIENT_OPTS":"\"-Xmx512m -Xms512m $HADOOP_CLIENT_OPTS\""
}
}
]
},
{
"classification": "httpfs-env",
"properties": {
```

```
},
"configurations":[
{
"classification":"export",
"properties": {
"HTTPFS_SSL_KEYSTORE_PASS":"passwd"
}
}
]
},
{
"classification": "mapred-env",
"properties": {
},
"configurations":[
{
"classification":"export",
"properties": {
"HADOOP_JOB_HISTORYSERVER_HEAPSIZE":"1001"
}
}
]
},
{
"classification": "yarn-env",
"properties": {
},
"configurations":[
{
"classification":"export",
"properties": {
"HADOOP_YARN_USER":"${HADOOP_YARN_USER:-yarn1}"
}
}
]
},
{
"classification": "pig",
"properties": {
"pig.tez.auto.parallelism": "false"
}
},
{
"classification": "pig-log4j",
"properties": {
"log4j.logger.org.apache.pig": "error, A"
}
},
{
"classification": "hive-env",
"properties": {
"BC":"CD"
},
"configurations":[
{
"classification":"export",
```

```
"properties": {
"AB":"${BC}",
"HADOOP_CLIENT_OPTS1":"\"-Xmx512m -Xms512m $HADOOP_CLIENT_OPTS1\""
}
}
]
},
{
"classification": "hive-site",
"properties": {
"hive.tez.java.opts": "-Xmx3900m"
}
},
{
"classification": "hive-exec-log4j",
"properties": {
"log4j.logger.org.apache.zookeeper.ClientCnxnSocketNIO": "INFO,FA"
}
},
{
"classification": "hive-log4j",
"properties": {
"log4j.logger.org.apache.zookeeper.server.NIOServerCnxn": "INFO,DRFA"
}
}
]
}
```

classification 参数指定要修改的配置文件，properties 参数放置要修改的 key value 键值对，默认配置文件有对应的 key 有则只覆盖 value，没有则添加对应的 key value 键值对。

配置文件与 classification 的对应关系如下列表格所示：

Hadoop

Filename	classification
core-site.xml	core-site
log4j.properties	hadoop-log4j
hdfs-site.xml	hdfs-site
mapred-site.xml	mapred-site
yarn-site.xml	yarn-site
httpsfs-site.xml	httpsfs-site
capacity-scheduler.xml	capacity-scheduler
hadoop-env.sh	hadoop-env
httpfs-env.sh	httpfs-env
mapred-env.sh	mapred-env
yarn-env.sh	yarn-env

Pig

Filename	classification
pig.properties	pig
log4j.properties	pig-log4j

Hive

Filename	classification
hive-env.sh	hive-env
hive-site.xml	hive-site
hive-exec-log4j.properties	hive-exec-log4j
hive-log4j.properties	hive-log4j

core-site 这类扁平的 xml 文件只有一层，配置都放在 properties 里。而 hadoop-en v这类 sh 文件可能有两层结构，可以通过嵌套 configurations 的方式来设置，请参见示例里 hadoop-env 的部分，为 export 的 HADOOP_CLIENT_OPTS 属性添加了 -Xmx512m -Xms512m 的设置。

设置好后，确认后单击**下一步**。

引导操作的作用

引导操作的作用是在集群启动 Hadoop 前执行您自定义的脚本，以便安装您需要的第三方软件，或者修改集群运行环境。

通过引导操作，您可以安装很多目前集群尚未支持的东西到您的集群上，例如：

使用 yum 安装已经提供的软件。

直接下载公网上的一些公开的软件。

读取 OSS 中您的自有数据。

安装并运行一个服务，例如 Flink 或者 Impala，但需要编写的脚本会复杂些。

强烈建议您先用按量付费的集群来进行引导操作的测试，测试都正确以后再创建包年包月的集群。

如何使用

登录阿里云 E-MapReduce 控制台集群列表。

在上方选择所在的地域（Region），所创建集群将会在对应的 Region 内。

单击**创建集群**，即会进入创建集群的操作界面。

在创建集群的基本信息页面，单击**添加引导操作**，进入其操作界面。

填写添加引导操作界面上的配置项，完成添加。

您最多可以添加 16 个引导操作，它们会按照您指定的顺序在集群初始化时执行。默认会使用 root 账户执行您指定的脚本，您可以在脚本中使用 su hadoop 切换到 Hadoop 账户。

引导操作可能会执行失败。为方便您的使用，引导操作失败并不会影响集群的创建。集群创建成功后，您可以在集群详情页集群信息栏内的引导/软件配置查看是否有异常发生。如果有异常，您可以登录到各个节点上查看运行日志，运行日志在 /var/log/bootstrap-actions 目录下。

引导操作类型

引导操作有两种，一种是自定义引导操作，另一种是运行条件引导操作。两者的主要区别是运行条件引导操作只会在满足条件的节点上运行您的指定操作。

自定义引导操作

自定义引导操作需要指定引导操作名称和执行脚本在 OSS 中的位置，根据需要指定可选参数。集群初始化时各个节点会下载您指定的 OSS 脚本，直接执行或者附加上可选参数执行。

您可以在脚本中指定从 OSS 下载需要的文件，下面的例子会将 oss://yourbucket/myfile.tar.gz 这个文件下载到本地，并解压到 /yourdir 目录下：

```
#!/bin/bash
osscmd --id=<yourid> --key=<yourkey> --host=oss-cn-hangzhou-internal.aliyuncs.com get
oss://<yourbucket>/<myfile>.tar.gz ./<myfile>.tar.gz
mkdir -p /<yourdir>
tar -zxvf <myfile>.tar.gz -C /<yourdir>
```

osscmd 已预安装在节点上，可以直接调用来下载文件。

注意：OSS 地址 host 有内网地址、外网地址和 VPC 网络地址之分。如果用经典网络，需要指定内网地址，杭州是 oss-cn-hangzhou-internal.aliyuncs.com。如果用 VPC 网络，要指定 VPC 内网可访问的域名，杭州是 vpc100-oss-cn-hangzhou.aliyuncs.com。

引导操作也可以通过 yum 安装额外的系统软件包，下面的例子会安装 ld-linux.so.2：

```
#!/bin/bash
yum install -y ld-linux.so.2
```

运行条件引导操作

运行条件引导操作的执行脚本是预定义的不需要您额外指定，您只需要指定名称和可选参数。运行条件引导操作必须提供可选参数，可选参数需要包括运行条件和操作命令，以空格间隔。运行条件支持 instance.isMaster=true/false，指定只在 master 或者在非 master 节点上运行。以下示例为运行条件引导操作下面的可选参数指定只在 master 节点上创建目录：

```
instance.isMaster=true mkdir -p /tmp/abc
```

如果需要指定多个操作命令，您可以用分号“；”分割多个语句，例如：instance.isMaster=true mkdir -p /tmp/abc;mkdir -p /tmp/def

专有网络（Virtual Private Cloud，VPC）为用户创建一个隔离的网络环境，用户可以选择自有的 IP 地址范围、划分网络、配置路由表、网关等，详见**专有网络产品简介**；另外通过**高速通道** 可以实现跨地域或跨用户的 VPC 内网互通、VPC 与物理 IDC 机房互通。

创建专有网络集群

E-MapReduce 在创建集群的时候可以选择网络类型，即经典网络/专有网络，若选择专有网络，需要如下额外操作：

所属 VPC：选择将当前创建的 E-MapReduce 集群放在哪个 VPC 中，如果还没创建可以进入**VPC 控制台**进行创建，一般一个账号最多创建 2 个 VPC 网络，超过 2 个需要提工单。

子网（交换机）：E-MapReduce 集群内的 ECS 实例通过交换机进行通信，如果还没创建可以进入**VPC 控制台**进行创建，因为交换机有可用区的属性，所以在 E-MapReduce 创建集群时选定了可用区后，创建的交换机也必须属于该可用区。

新建安全组：如果打开，那么就需要输入新建的安全组的名字。

所属安全组：集群所属的安全组，经典网络的安全组不能在 VPC 中使用，VPC 的安全组只能在当前 VPC 中使用。这里只展示用户在 E-MapReduce 产品中创建的安全组。因为一些安全的原因目前尚不支持选择在 E-MapReduce 外创建的安全组。如果需要新建安全组，可以选择“新建安全组”选项，同时输入安全组的名字完成新建。

示例

不同 VPC 中的 EMR 集群通信（Hive 访问 HBase）

创建集群：

在 E-MapReduce 上面创建两个集群，Hive 集群 C1 处于 VPC1 中，HBase 集群 C2 处于 VPC2 中，两个集群都在杭州区域。

配置高速通道：

配置详见同地域下的 VPC 私网互通。

ssh 登录 HBase 集群，通过 HBase Shell 创建表。

```
hbase(main):001:0> create 'testfromHbase','cf'
```

ssh 登录 Hive

修改 hosts，增加如下一行:

```
$zk_ip emr-cluster //$zk_ip为Hbase集群的zk节点IP
```

通过 Hive Shell 访问 HBase。

```
hive> set hbase.zookeeper.quorum=172.16.126.111,172.16.126.112,172.16.126.113;
hive> CREATE EXTERNAL TABLE IF NOT EXISTS testfromHive (rowkey STRING, pageviews Int, bytes STRING) STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH SERDEPROPERTIES ('hbase.columns.mapping' = ':key,cf:c1,cf:c2') TBLPROPERTIES ('hbase.table.name' = 'testfromHbase');
```

此时命令会卡住，然后会报 java.net.SocketTimeoutException 的异常，原因是 HBase 集群的 ECS 所在的安全组限制了相关端口的访问（E-MapReduce 创建的安全组默认只开放 22 端口）E-MapReduce，所以需要给 HBase 集群的安全组增加安全组规则开放端口给 Hive 集群，如下图所示：

自有 IDC 专线接入（访问 VPC 中 EMR 集群）

详见自行专线接入访问 VPC。

从 E-MapReduce 的 2.0.0 版本开始，支持多个 Python 版本。

版本列表如下：

版本	包含库	安装位置
2.6		
2.7.11	numpy	/usr/local/Python-2.7.11/
3.4.4	numpy	/usr/local/Python-3.4.4/

如果要使用，请在对应的脚本中写全对应版本的 Python 命令所在路径即可。

开源组件介绍

目前 E-MapReduce 中支持了 Hue，选择支持 Hue 的镜像创建集群并且开启公网 IP 即可以在 E-MapReduce 访问和使用 Hue。

准备工作

在 E-MapReduce 上 Hue 的服务端口是 8888 ，由于Hue本身做了安全校验和用户管理，可以有两种方式来访问 E-MapReduce 的 Hue 服务。

打ssh安全隧道

在集群建立出来之后，需要打通 ssh 隧道来访问Hue的服务端口 8888 ，详细步骤请参考：
https://help.aliyun.com/document_detail/28187.html

这里以 Mac 环境为例，使用 Chrome 浏览器实现端口转发（假设集群 master 节点公网 IP 为xx.xx.xx.xx）：

登录到 master 节点。

```
ssh root@xx.xx.xx.xx
```

输入密码。

查看本机的 id_rsa.pub 内容（注意在本机执行，不要在远程的 master 节点上执行）。

```
cat ~/.ssh/id_rsa.pub
```

将本机的 id_rsa.pub 内容写入到远程 master 节点的 ~/.ssh/authorized_keys 中（在远端 master 节点上执行）。

```
mkdir ~/.ssh/
vim ~/.ssh/authorized_keys
```

将步骤 2 中看到的内容粘贴进来并保存。现在就可以直接使用 ssh root@xx.xx.xx.xx 免密登录 master 节点了。

在本机执行以下命令进行端口转发。

```
ssh -i ~/.ssh/id_rsa -ND 8157 root@xx.xx.xx.xx
```

启动 Chrome（在本机新开 terminal 执行）。

```
/Applications/Google\ Chrome.app/Contents/MacOS/Google\ Chrome --proxy-server="socks5://localhost:8157" --host-resolver-rules="MAP * 0.0.0.0 , EXCLUDE localhost" --user-data-dir=/tmp
```

开发集群所在安全组的 8888 端口

1. 在集群详情中找到集群所在的 ecs 安全组；
2. 在ecs控制台修改对应的安全组，在“公网入方向”添加一条规则，打开8888端口：**注意，为了安全原因，这里而设置的授权对象必须是您的一个有限的ip段范围，禁止使用0.0.0.0/0**

注意：打开安全组的8888端口之后，该安全组内的所有机器均会打开公网入方向的 8888 端口，包括非E-MapReduce的ecs机器。

访问 Hue

如果是使用打 ssh 安全隧道的方式，则在进行端口转发的 Chrome 浏览器中访问：xx.xx.xx.xx:8888；如果是

开发集群安全组8888端口的方式，直接访问公网 IP:8888

访问密码

由于Hue默认是第一次运行以后如果还没有管理员，第一个登陆的用户就默认设置为管理员。为了安全起见，EMR会默认生成一个管理员账号与密码，管理员账号是admin，密码通过如下途径查看：

1. 在集群列表上点击配置管理
2. 选择服务列表，选择Hue服务
3. 查看配置列表，其中有一个admin_pwd的参数，这个就是随机密码

版本信息

阿里云 E-MapReduce 在 2.0.0 及之后的版本中提供了对 Oozie 的支持，如果需要在集群中使用 Oozie，请确认集群的版本不低于 2.0.0。

准备工作

在集群建立出来之后，需要打通 ssh 隧道，详细步骤请参考：
https://help.aliyun.com/document_detail/28187.html

这里以 MAC 环境为例，使用 Chrome 浏览器实现端口转发（假设集群 master 节点公网 IP 为 xx.xx.xx.xx）：

登录到 master 节点。

```
ssh root@xx.xx.xx.xx
```

输入密码。

查看本机的 id_rsa.pub 内容（注意在本机执行，不要在远程的 master 节点上执行）。

```
cat ~/.ssh/id_rsa.pub
```

将本机的 id_rsa.pub 内容写入到远程 master 节点的 ~/.ssh/authorized_keys 中（在远端 master 节点上执行）。

```
mkdir ~/.ssh/
vim ~/.ssh/authorized_keys
```

然后将步骤 2 中看到的内容粘贴进来，现在应该可以直接使用 ssh root@xx.xx.xx.xx 免密登录 master 节点了。

在本机执行以下命令进行端口转发。

```
ssh -i ~/.ssh/id_rsa -ND 8157 root@xx.xx.xx.xx
```

7. 启动 Chrome（在本机新开 terminal 执行）。

```
/Applications/Google\ Chrome.app/Contents/MacOS/Google\ Chrome --proxy-server="socks5://localhost:8157" --host-resolver-rules="MAP * 0.0.0.0 , EXCLUDE localhost" --user-data-dir=/tmp
```

访问 Oozie UI 页面

在进行端口转发的 Chrome 浏览器中访问：xx.xx.xx.xx:11000/oozie，localhost:11000/oozie 或者内网 ip:11000/oozie。

提交workflow作业

运行 Oozie 需要先安装 Oozie 的sharelib：
https://oozie.apache.org/docs/4.2.0/WorkflowFunctionalSpec.html#ShareLib

在 E-MapReduce 集群中，默认给 Oozie 用户安装了 sharelib，即如果使用 Oozie 用户来提交 workflow 作业，则不需要再进行 sharelib 的安装。

由于开启 HA的集群和没有开启 HA 的集群，访问 NameNode 和 ResourceManager 的方式不同，在提交 oozie workflow job 的时候，job.properties 文件中需要指定不同的 NameNode 和 JobTracker（ResourceManager）。具体如下：

非 HA 集群

```
nameNode=hdfs://emr-header-1:9000
jobTracker=emr-header-1:8032
```

HA 集群

```
nameNode=hdfs://emr-cluster
jobTracker=rm1,rm2
```

下面操作示例中，已经针对是否是 HA 集群配置好了，即样例代码不需要任何修改即可以直接运行。关于 workflow 文件的具体格式，请参考 Oozie 官方文档：https://oozie.apache.org/docs/4.2.0/。

在非 HA 集群上提交 workflow 作业

登录集群的主 master 节点。

```
ssh root@master公网Ip
```

下载示例代码。

```
[root@emr-header-1 ~]# su oozie
[oozie@emr-header-1 root]$ cd /tmp
[oozie@emr-header-1 tmp]$ wget http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/oozie-examples/oozie-examples.zip
[oozie@emr-header-1 tmp]$ unzip oozie-examples.zip
```

将 Oozie workflow 代码同步到 hdfs 上。

```
[oozie@emr-header-1 tmp]$ hadoop fs -copyFromLocal examples/ /user/oozie/examples
```

提交 Oozie workflow 样例作业。

```
[oozie@emr-header-1 tmp]$ $OOZIE_HOME/bin/oozie job -config examples/apps/map-reduce/job.properties -run
```

执行成功之后，会返回一个 jobId，类似：

```
job: 0000000-160627195651086-oozie-oozi-W
```

访问 Oozie UI 页面，可以看到刚刚提交的 Oozie workflow job。

在 HA 集群上提交 workflow 作业

登录 HA 集群的主 master 节点。

```
ssh root@主master公网Ip
```

可以通过是否能访问 Oozie UI 来判断哪个 master 节点是当前的主 master 节点， Oozie server 服务默认是启动在主 master 节点 xx.xx.xx.xx:11000/oozie。

下载 HA 集群的示例代码。

```
[root@emr-header-1 ~]# su oozie
[oozie@emr-header-1 root]$ cd /tmp
[oozie@emr-header-1 tmp]$ wget http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/oozie-examples/oozie-examples-ha.zip
[oozie@emr-header-1 tmp]$ unzip oozie-examples-ha.zip
```

将 Oozie workflow 代码同步到 hdfs 上。

```
[oozie@emr-header-1 tmp]$ hadoop fs -copyFromLocal examples/ /user/oozie/examples
```

提交 Oozie workflow 样例作业。

```
[oozie@emr-header-1 tmp]$ $OOZIE_HOME/bin/oozie job -config examples/apps/map-reduce/job.properties -run
```

执行成功之后，会返回一个 jobId，类似：

```
job: 0000000-160627195651086-oozie-oozi-W
```

访问 Oozie UI 页面，可以看到刚刚提交的 Oozie workflow job。

2.0.0 以上版本支持 presto，选择支持 presto 的镜像并勾选 presto 软件即可在 E-MapReduce 中使用 presto。

集群创建后，登录 master，presto 软件被安装在 /usr/lib/presto-current 目录，可以 jps 看到 PrestoServer 进程。

presto 服务进程分为 coordinator 和 worker，master 上（HA 集群为 hostname 以emr-header-1 开头的 master 节点）启动 coordinator，core 节点启动 worker 进程。服务进程的配置在 /usr/lib/presto-current/etc 目录下，其中 coordinator 使用 coordinator-config.properties，worker 使用 worker-config.preperties,其他配置文件公用。web 端口设置为 9090。

presto 服务默认设置了 Hive 的支持，连接集群 hive 的 metastore，可以读取 Hive 的表信息，进行查询。集群预装了 presto cli，可以直接执行

```
presto --server localhost: 9090 --catalog hive --schema default --user hadoop --execute "show tables"
```

查看 Hive 的表。需要注意同步 Hive 表会有几秒的延时。

目前 E-MapReduce 中支持了 Appache Zeppelin，选择支持 Zeppelin 的镜像创建集群并且开启公网 IP 即可以在 E-MapReduce 访问和使用 Zeppelin。

准备工作

在集群建立出来之后，需要打通 ssh 隧道，详细步骤请参考：
https://help.aliyun.com/document_detail/28187.html。

这里以 Mac 环境为例，使用 Chrome 浏览器实现端口转发（假设集群 master 节点公网 IP 为xx.xx.xx.xx）：

登录到 master 节点。

```
ssh root@xx.xx.xx.xx
```

输入密码。

查看本机的 id_rsa.pub 内容（注意在本机执行，不要在远程的 master 节点上执行）。

```
cat ~/.ssh/id_rsa.pub
```

将本机的 id_rsa.pub 内容写入到远程 master 节点的 ~/.ssh/authorized_keys 中（在远端 master 节点上执行）。

```
mkdir ~/.ssh/
vim ~/.ssh/authorized_keys
```

将步骤 2 中看到的内容粘贴进来。现在就可以直接使用 sshroot@xx.xx.xx.xx 免密登录 master 节点了。

在本机执行以下命令进行端口转发。

```
ssh -i ~/.ssh/id_rsa -ND 8157 root@xx.xx.xx.xx
```

启动 Chrome（在本机新开 terminal 执行）。

```
/Applications/Google\ Chrome.app/Contents/MacOS/Google\ Chrome --proxy-server="socks5://localhost:8157" --host-resolver-rules="MAP * 0.0.0.0 , EXCLUDE localhost" --user-data-dir=/tmp
```

访问 Zeppelin

方式一：设置好端口转发，然后在进行端口转发的 Chrome 浏览器中访问：xx.xx.xx.xx:8080。

方式二：针对有限的ip段范围，开放安全组，然后直接访问。禁止在配置的时候对0.0.0.0/0开放规则。

目前 E-MapReduce 集群中默认启动了 ZooKeeper 服务。

注意事项

目前无论集群内有多少台机器，ZooKeeper 只会有 3 个节点。目前还不支持更多的节点。

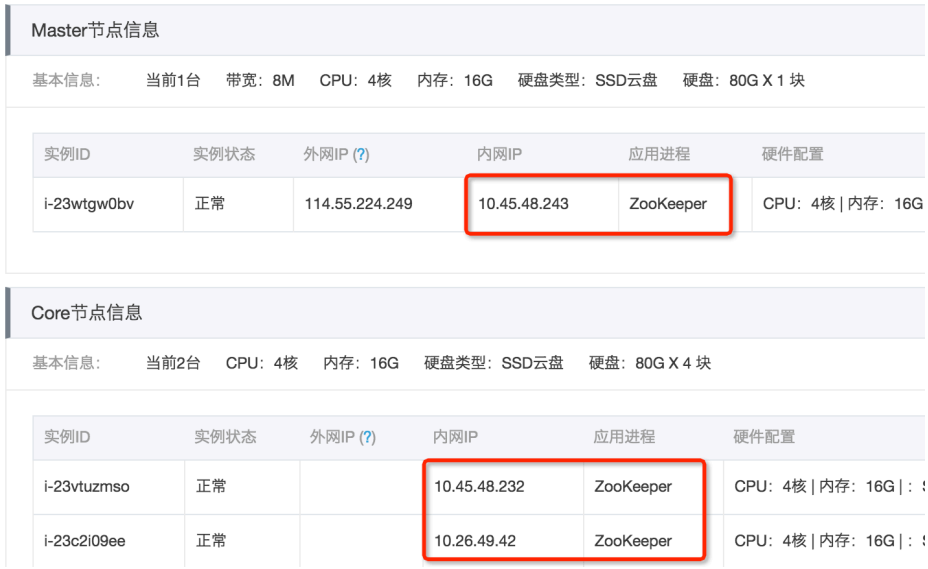
创建集群

E-MapReduce 创建集群的软件配置页面，会默认勾选 ZooKeeper，如下图所示：

节点信息

集群创建成功，状态空闲后，查看集群的详情页面，可以查到 ZooKeeper 的节点信息，E-MapReduce 会启

动 3 个 ZooKeeper 节点。如下图所示，应用进程一栏标有 ZooKeeper 节点对应的内网 IP (端口默认为 2181)，即可访问 ZooKeeper 服务。



从EMR-3.4.0版本开始将支持Kafka服务。

创建Kafka集群

在E-MapReduce控制台创建集群时，选择集群类型为Kafka，则会创建一个默认只包含Kafka组件的集群，除了基础组件外包括Zookeeper，Kafka和KafkaManager三个组件。每个节点将只部署一个Kafka broker。我们建议您的Kafka集群是一个专用集群，不要和Hadoop相关服务混部在一起。

跨集群访问Kakfa

通常，我们会单独部署一个Kafka集群来提供服务，所以经常需要跨集群访问Kafka服务。这时，我们需要在机器上配置Kafka集群节点的host信息。注意，这里我们需要在client端机器配置Kafka集群节点的**长域名**，否则会出现访问不到Kafka服务的问题。示例如下：

```
 / etc/hosts

# kafka cluster
10.0.1.23 emr-header-1.cluster-48742
10.0.1.24 emr-worker-1.cluster-48742
10.0.1.25 emr-worker-2.cluster-48742
10.0.1.26 emr-worker-3.cluster-48742
```

参数说明

您可以在E-MapReduce的集群配置管理中查看Kafka的软件配置，当前主要有：

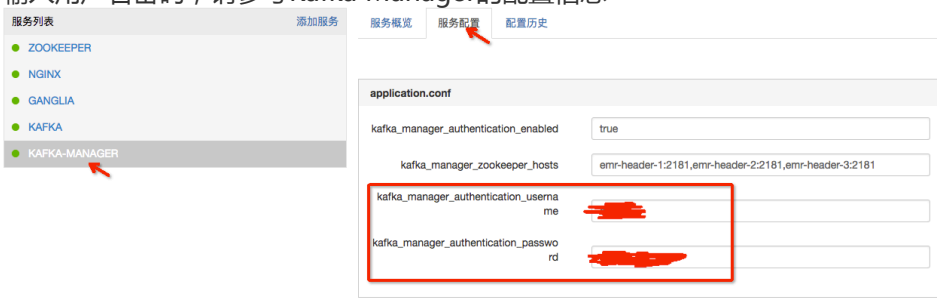
配置项	说明

zookeeper.connect	Kafka配置的Zookeeper连接地址
kafka.heap.opts	Kafka broker的堆内存大小
num.io.threads	Kafka broker的IO线程数，默认为机器CPU核数目的2倍
num.network.threads	Kafka broker的网络线程数，默认为机器的CPU核数目

从EMR-3.4.0版本开始将支持Kafka Manager服务进行Kafka集群管理。

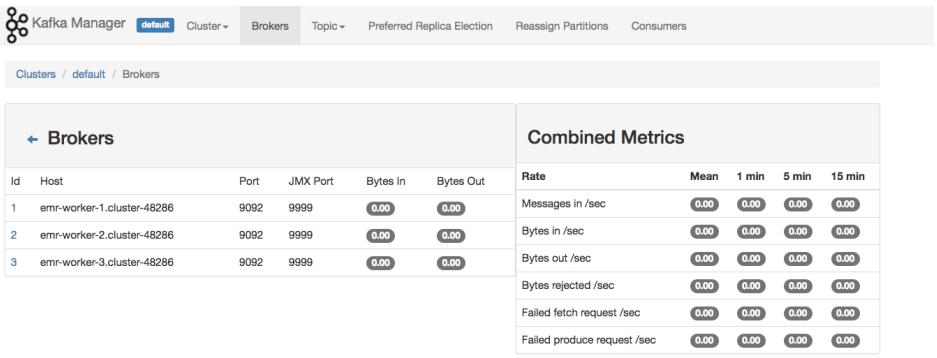
使用步骤

- 建议使用SSH隧道方式访问Web界面，参考“用户指南”-> “SSH登陆集群”一节
- 访问http://localhost:8085
- 输入用户名密码，请参考Kafka Manager的配置信息



- 添加一个创建好的Kafka集群，需要注意配置正确Kafka集群的Zookeeper地址，不清楚的可以看Kakfa的配置信息。选择对应的Kafka版本，另外建议打开JMX功能。

- 创建好之后即可使用一些常见的Kakfa功能



注意事项

- 创建Kafka集群时将默认安装Kafka Manager软件服务，并开启Kafka Manager的认证功能。我们**强烈建议您首次使用Kafka Manager时修改默认密码，且使用SSH隧道方式来访问。不建议您公网暴露8085端口，否则需要做好IP白名单保护，避免数据泄漏。**

目前E-MapReduce中支持了Apache Knox，选择支持Knox的镜像创建集群，完成以下准备工作后，即可在公网直接访问Yarn，HDFS，SparkHistory等服务的Web UI

准备工作

开启Knox公网IP访问

1. 在E-MapReduce上Knox的服务端口是8443，在集群详情中找到集群所在的ecs安全组；
2. 在ecs控制台修改对应的安全组，在“公网入方向”添加一条规则，打开8443端口。

注意 1：为了安全原因，这里设置的授权对象必须是您的一个有限的ip段范围，禁止使用0.0.0.0/0

注意 2：打开安全组的8443端口之后，该安全组内的所有机器均会打开公网入方向的 8443 端口，包括非E-MapReduce的ecs机器。

设置Knox用户

访问Knox时需要对身份进行验证，会要求您输入用户名和密码。Knox的用户身份验证基于LDAP，您可以使用自有LDAP服务，也可以使用集群中的Apache Directory Server的LDAP服务

使用集群中的LDAP服务

1. ssh登录到集群上，详细步骤请参考SSH登录集群
2. 准备您的用户数据，如：Tom。将文件中所有的“emr-guest”替换为“Tom”,将”cn:EMR GUEST”替换为“cn:Tom”，设置“userPassword”的值为您自己的密码

```
su knox
cd /usr/lib/knox-current/templates
```

```
vi users.ldif
```

3. 导入到LDAP。**注意，出于安全原因，导入前务必修改users.ldif的用户密码，即：设置”userPassword”的值为您自己的用户密码**

```
su knox
cd /usr/lib/knox-current/templates
sh ldap-sample-users.sh
```

使用自有LDAP服务的情况

1. 在集群配置管理中找到KNOX的配置管理，在cluster-topo配置中设置两个属性：main.ldapRealm.userDnTemplate与main.ldapRealm.contextFactory.url。main.ldapRealm.userDnTemplate设置为自己的用户DN模板，main.ldapRealm.contextFactory.url设置为自己的LDAP服务器域名和端口。设置完成后保存并重启Knox

一般自己的LDAP服务不在集群上运行，所以需要开启Knox访问公网LDAP服务的端口，如：10389。参考8443端口的开启步骤，选择“公网出方向”。

注意，为了安全原因，这里设置的授权对象必须是您的Knox所在集群的公网ip，禁止使用0.0.0.0/0

开始访问Knox

使用E-MapReduce快捷链接访问

1. 登录到E-MapReduce管理控制台
2. 在“集群与服务管理”页面点击相应的服务，如：HDFS，Yarn等
3. 点击右上角的“快捷链接”

使用集群公网ip地址访问

1. 通过集群详情查看公网ip

   在浏览器中访问相应服务的URL

   - HDFS UI：https://{集群公网ip}:8443/gateway/cluster-topo/hdfs/
   - Yarn UI: https://{集群公网ip}:8443/gateway/cluster-topo/yarn/

- SparkHistory UI : https://{集群公网ip}:8443/gateway/cluster-topo/sparkhistory/
- Ganglia UI: https://{集群公网ip}:8443/gateway/cluster-topo/ganglia/
- Storm UI: https://{集群公网ip}:8443/gateway/cluster-topo/storm/
- Oozie UI: https://{集群公网ip}:8443/gateway/cluster-topo/oozie/

浏览器显示“您的链接不是私密链接”，是因为Knox服务使用了自签名证书，请再次确认访问的是自己集群的ip，且端口为8443；点击“高级”->”继续前往”

在登录框中输入您在LDAP中设置的用户名和密码

用户权限管理（ACLs）

Knox提供服务级别的权限管理，可以限制特定的用户，特定的用户组和特定的ip地址访问特定的服务，可以参考Apache Knox 授权

示例：

- 场景：Yarn UI只允许用户Tom访问
- 步骤：在集群配置管理中找到KNOX的配置管理，找到cluster-topo配置，在cluter-topo配置的<gateway>...</gateway>标签之间添加ACLs代码

```
<provider>
<role>authorization</role>
<name>AclsAuthz</name>
<enabled>true</enabled>
<param>
<name>YARNUI.acl</name>
<value>Tom;*;*</value>
</param>
</provider>
```

注意事项

Knox会开放相应服务的REST API，用户可以通过各服务的REST API操作服务，如：HDFS的文件添加，删除等。出于安全原因，请务必确保在ecs控制台开启安全组Knox端口8443时，授权对象必须是您的一个有限ip地址段，禁止使用0.0.0.0/0；请勿使用Knox安装目录下的LDAP用户名和密码作为Knox的访问用户

Presto

Presto是一款由FaceBook开源的一个分布式SQL-on—Hadoop分析引擎。Presto目前由开源社区和FaceBook内部工程师共同维护，并衍生出多个商业版本。

基本特性

Presto使用Java语言进行开发，具备易用、高性能、强扩展能力等特点，具体的：

- 完全支持ANSI SQL

支持丰富的数据源 Presto可接入丰富的数据来源，如下所示：

- 与Hive数仓互操作
- Cassandra
- Kafka
- MongoDB
- MySQL
- PostgreSQL
- SQL Server
- Redis
- Redshift
- 本地文件

支持高级数据结构

- 支持数组和Map数据
- 支持JSON数据
- 支持GIS数据
- 支持颜色数据

功能扩展能力强 Presto提供了多种扩展机制，包括：

- 扩展数据连接器
- 自定义数据类型
- 自定义SQL函数

用户可以根据自身业务特点扩展相应的模块，实现高效的业务处理。

基于Pipeline处理模型 数据在处理过程中实时返回给用户

监控接口完善

- 提供友好的WebUI，可视化的呈现查询任务执行过程
- 支持JMX协议

应用场景

Presto是定位在数据仓库和数据分析业务的分布式SQL引擎，比较适合如下几个应用场景：

- ETL
- Ad-Hoc查询
- 海量结构化数据/半结构化数据分析
- 海量多维数据聚合/报表

特别需要注意的是，Presto是一个数仓类产品，其设计目标并不是为了替代MySQL、PostgreSQL等传统的RDBMS数据库，对事务对支持有限，不适合在线业务场景。

产品优势

EMR Presto产品除了开源Presto本身具有的优点外，还具备如下优势：

- 即买即用 分分钟完成上百节点的Presto集群搭建
- 弹性扩容 简单操作即可完成集群的扩容和缩容
- 与EMR软件栈完美结合，支持处理存储在OSS的数据
- 免运维 724一站式服务

操作步骤

E-MapReduce 支持使用 RAM 来隔离不同子账号的数据。操作步骤如下所示：

登录阿里云 RAM 的管理控制台。

在RAM中创建子账号，具体流程请参见如何在 RAM 中创建子账号。

单击阿里云 RAM 的管理控制台页面左侧的**授权策略管理**，进入授权策略管理界面。

单击**自定义授权策略**。

单击页面右上方的**新建授权策略**按钮，即进入创建授权策略界面，然后按照提示步骤进行创建。您需要多少套不同的权限控制，就创建多少个策略。

假设您需要以下 2 套数据控制策略：

测试环境，bucketname：test-bucket。其所对应的完整策略如下：

```
{
"Version": "1",
"Statement": [
{
"Effect": "Allow",
"Action": [
"oss:ListBuckets"
],
"Resource": [
"acs:oss:*:*:*"
]
},
{
"Effect": "Allow",
"Action": [
"oss:Listobjects",
"oss:GetObject",
"oss:PutObject",
"oss:DeleteObject"
],
"Resource": [
"acs:oss:*:*:test-bucket",
"acs:oss:*:*:test-bucket/*"
]
}
]
}
```

生产环境， bucketname：prod-bucket。其所对应的完整策略如下：

```
{
"Version": "1",
"Statement": [
{
"Effect": "Allow",
"Action": [
"oss:ListBuckets"
],
"Resource": [
"acs:oss:*:*:*"
]
},
{
"Effect": "Allow",
"Action": [
"oss:Listobjects",
"oss:GetObject",
"oss:PutObject"
],
"Resource": [
"acs:oss:*:*:prod-bucket",
"acs:oss:*:*:prod-bucket/*"
]
}
]
```

```
}
```

单击阿里云 RAM 的管理控制台页面左侧的**用户管理**。

找到需要将策略赋给的子账号条目，单击其右侧的**管理**按钮，进入用户管理页面。

单击页面左侧的**用户授权策略**。

单击右上角的**编辑授权策略**按钮，进入策略授权页面。

选择并添加授权策略。

单击**确定**，完成对子账号的策略授权。

单击用户管理页面左侧的**用户详情**，进入子账号的用户详情页面。

在 Web 控制台登录管理栏中，单击**启用控制台登录**，以打开子账号的登录控制台的权限。

完成并使用

完成以上所有步骤以后，使用对应的子账号登录 E-MapReduce，会有以下限制：

在创建集群、创建作业和创建执行计划的 OSS 选择界面，可以看到所有的 bucket，但是只能进入被授权的 bucket。

只能看到被授权的 bucket 下的内容，无法看到其他 bucket 内的内容。

作业中只能读写被授权的 bucket，读写未被授权的 bucket 会报错。

登录集群内部的用途

若您觉得在网页上的作业和执行计划无法满足您更加复杂的应用需求，您可以登录到 E-MapReduce 集群的主机上，找到集群的详情页，其中就有集群 master 机器的公网 IP 地址，您可以直接 SSH 登录到这台机器上，查看各种设置与状态。

机器上已为您设置好相关的环境变量，其中包括以下常用的环境变量：

JAVA_HOME

HADOOP_HOME

HADOOP_CONF_DIR

HADOOP_LOG_DIR

YARN_LOG_DIR

HIVE_HOME

HIVE_CONF_DIR

PIG_HOME

PIG_CONF_DIR

您可以在脚本中直接引用这些变量，但请不要去修改这些变量的值，以免造成 E-MapReduce 的意外错误。

登录 master 主机步骤

使用如下命令 SSH 登录到 master 主机。请在**集群详情页**的硬件信息栏中获取集群 master 机器的公网 IP。

```
ssh root@ip.of.master
```

输入创建时设定的密码。

打通本地机器与集群 master 机器的 SSH 无密码登录

通常，您需要登录到集群上进行一些管理和操作。为便于登录集群 master 机器，您可打通与 master 机器的 SSH 无密码登录（集群 master 机器默认开通了公网 IP）。操作步骤如下：

通过上面提到的 root + 密码的方式登录到 master 主机。

切换到到Hadoop 用户或者 hdfs 用户。su hadoop

Linux 的 SSH 方式

复制私钥到本地。

```
sz ~/.ssh/id_rsa
```

回到您的本地机器，尝试重新登录 master 机器。

```
ssh -i 私钥存放路径/id_rsa hadoop@120.26.221.130
```

当然如果你只有这一个私钥，也可以直接放到你的 ~/.ssh/ 下，默认使用这个私钥，就不需要 -i 指定了。

Windows 的 SSH 方式

在 Windows 下你可以有多种方式来使用 SSH 免密码登录 master 机器。

方式一：使用 PuTTY

点击下载 PuTTY。

在同样的位置下载 PuTTYgen。

打开 PuTTYgen，并 Load 您的私钥。

注意：请妥善保管这个私钥，保证该私钥的安全。若私钥不幸泄漏了，请立刻重新生成一个新的取代。

使用默认的配置，并 Save private key。会保存出一个后缀为 ppk 的 PuTTY 使用的密钥文件。

运行 PuTTY，并在配置页面选择 Session。

输入您要连接的目标机器公网 IP 地址，要加上登录使用的用户名，类似 hadoop@MasterNodeIP。

在配置页面选择 Connetion 并展开 > 选择 SSH 并展开 > 选择 Auth。

选择之前生成好的 ppk 文件。

最后单击 Open，就会自动登录到 master 节点了。

方式二：使用 Cygwin | MinGW

这是在 Windows 上模拟 Linux 的非常方便的工具，使用起来也非常简单。

如果采用这种方式，连接过程就可以参考上面的 Linux 的 SSH 方式了。

推荐采用 MinGW 的方式，这个是最小巧的一种方式。如果官网打不开，可以下载 git 的客户端，默认带的 Git Bash 就可以满足。

查看 Hadoop、Spark、Ganglia 等系统的 webui

注意：在进行本步骤前，请确认您已经完成了上面的 SSH 无密码登录流程。

由于安全的缘故，E-MapReduce 集群的 Hadoop、Spark 和 Ganglia 等系统的 webui 监控系统的端口都没有对外开放。如果用户想要访问这些 webui，需要建立一个 SSH 隧道，通过端口转发的方式来达到目的。有如下两种方式：

注意：下面的操作是在您本地机器上完成的，不是集群内部机器。

方式一：端口动态转发

创建一个 SSH 隧道，该隧道可打通您本地机器跟 E-MapReduce 集群的 master 机器的某个动态端口的连接。

```
ssh -i /path/id_xxx  -ND 8157 hadoop@masterNodeIP
```

8157 是您本地机器没有被使用过的任何一个端口，用户可以自定定义。

完成动态转发以后，您可以选择如下两种方式来查看。

推荐方式

推荐使用 Chrome 浏览器，可以使用如下的方式来访问 Web UI：

```
chrome --proxy-server="socks5://localhost:8157" --host-resolver-rules="MAP * 0.0.0.0 , EXCLUDE localhost" --user-data-dir=/tmp/
```

若是 Windows 系统，这里的 tmppath 可以写成类似 d:/tmppath；若是 Linux 或者 OSX，可以直接写成 /tmp/。

在不同的操作系统中，Chrome 的位置不同，请参见下表：

操作系统	Chrome 位置

Mac OS X	/Applications/Google Chrome.app/Contents/MacOS/Google Chrome
Linux	/usr/bin/google-chrome
Windows	C:\Program Files (x86)\Google\Chrome\Application\chrome.exe

插件方式

此时，您本地机器跟 E-MapReduce 集群的 master 主机的 SSH 通道已经打通，要在浏览器中查看 Hadoop、Spark、Ganglia 的 webui，您还需要配置一个本地代理。操作步骤如下：

假设您使用的是 Chrome 或者 Firefox 浏览器，请点击下载 FoxyProxy Standard 代理软件。

安装完成并重启浏览器后，打开一个文本编辑器，编辑如下内容：

```
<?xml version="1.0" encoding="UTF-8"?>
<foxyproxy>
<proxies>
<proxy name="aliyun-emr-socks-proxy" id="2322596116" notes="" fromSubscription="false"
enabled="true" mode="manual" selectedTabIndex="2" lastresort="false"
animatedIcons="true" includeInCycle="true" color="#0055E5" proxyDNS="true"
noInternalIPs="false" autoconfMode="pac" clearCacheBeforeUse="false"
disableCache="false" clearCookiesBeforeUse="false" rejectCookies="false">
<matches>
<match enabled="true" name="120.*" pattern="http://120.*" isRegEx="false"
isBlackList="false" isMultiLine="false" caseSensitive="false" fromSubscription="false"
></match>
</matches>
<manualconf host="localhost" port="8157" socksversion="5" isSocks="true" username=""
password="" domain="" ></manualconf>
</proxy>
</proxies>
</foxyproxy>
```

其中：

Port 8157 是您本地用来建立与集群 master 机器 SSH 连接的端口，这个需要跟您之前执行的在终端中执行的 SSH 命令中使用的端口匹配。

120.* 这个匹配是用来匹配 master 主机的 IP 地址，请根据 master 的 IP 地址的情况来定。

在浏览器中点击 Foxyproxy 按钮，选择 Options。

选择 Import/Export。

选择刚才您编辑的 xml 文件，点击 Open。

在 Import FoxyProxy Setting 对话框中，点击 Add。

点击浏览器中的 Foxyproxy 按钮，选择“Use Proxy aliyun-emr-socks-proxy for all URLs”。

在浏览器中输入 localhost:8088，就可以打开远端的 Hadoop 界面了。

方式二：本地端口转发

注意：这个方式的缺陷是只能看到最外层的界面，一旦要看详细的作业信息，就会出错。

```
ssh -i /path/id_rsa -N -L 8157:masterNodeIP:8088 hadoop@masterNodeIP
```

参数说明：

path：私钥存放路径。

masterNodeIP：要连接的 master节点 IP。

8088：是 master 节点上 ResourceManager 的访问端口号。

Gateway

一些客户需要自主搭建Gateway向E-MapReduce集群提交作业，目前E-MapReduce在产品页面上不支持购买Gateway，后续可以在产品上直接购买Gateway，并把Hadoop环境准备好供用户使用。

网络

首先要保证Gateway机器在EMR对应集群的安全组中，Gateway节点可以顺利的访问EMR集群。设置机器的安全组请参考ECS的安全组设置说明。

环境

- Java环境

安装至少JDK 1.7及以上

- 复制所有的依赖的hadoop的包到gateway

```
scp -r root@masterip:/opt/apps/extra-jars  /opt/apps/

scp -r root@masterip:/usr/lib/hadoop-current /opt/apps/
scp -r root@masterip:/usr/lib/hive-current /opt/apps/
scp -r root@masterip:/usr/lib/spark-current /opt/apps/

ln -s /opt/apps/hadoop-current /usr/lib/hadoop-current
ln -s /opt/apps/hive-current /usr/lib/hive-current
ln -s /opt/apps/spark-current /usr/lib/spark-current
```

- 复制配置文件到gateway

EMR-3.2.0 及以上版本

```
mkdir /etc/ecm
scp -r root@masterip:/etc/ecm/hadoop-conf /etc/ecm/hadoop-conf
scp -r root@masterip:/etc/ecm/hive-conf /etc/ecm/hive-conf/
```

EMR-3.2.0 以下版本

```
mkdir /etc/emr
scp -r root@masterip:/etc/emr/hadoop-conf /etc/emr/hadoop-conf
scp -r root@masterip:/etc/emr/hive-conf /etc/emr/hive-conf/
```

复制环境变量到gateway,并执行

```
scp root@masterip:/etc/profile.d/hadoop.sh /etc/profile.d/
source /etc/profile.d/hadoop.sh
```

修改hosts配置将集群的master节点中的host内容复制到gateway的/etc/hosts中

```
#start add cluster host of cluster 22663,Mon May 30 19:21:51 CST 2016
xx.yy.zz.tt emr-header-1.cluster-1212 emr-header-1 xxxxxxxx1
xx.yy.zz.tt1 emr-worker-2.cluster-1212 emr-worker-2 emr-header-3 xxxxxxx2
xx.yy.zz.tt2 emr-worker-1.cluster-1212 emr-worker-1 emr-header-2 xxxxxxxx3
#end add cluster host
```

完成以上以后，配置就完成了。

测试

Hive

```
[hadoop@iZ23bc05hrvZ ~]$ hive
hive> show databases;
OK
default
Time taken: 1.124 seconds, Fetched: 1 row(s)
hive> create database school;
OK
Time taken: 0.362 seconds
hive>
```

运行Hadoop作业

```
 [hadoop@iZ23bc05hrvZ ~]$ hadoop  jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar pi 10 10
Number of Maps = 10
Samples per Map = 10
Wrote input for Map #0
Wrote input for Map #1
Wrote input for Map #2
Wrote input for Map #3
Wrote input for Map #4
Wrote input for Map #5
Wrote input for Map #6
Wrote input for Map #7
Wrote input for Map #8
Wrote input for Map #9

File Input Format Counters
Bytes Read=1180
File Output Format Counters
Bytes Written=97
Job Finished in 29.798 seconds
Estimated value of Pi is 3.20000000000000000000
```

E-MapReduce环境下提供MetaService服务。基于此服务，您可以在E-MapReduce集群中以免AK的方式访问阿里云资源。

默认应用角色

默认地，您在创建集群时将需要向E-MapReduce服务授权一个应用角色（AliyunEmrEcsDefaultRole）。授权之后，您在E-MapReduce上的作业将可以无需显式输入AK来访问阿里云资源。AliyunEmrEcsDefaultRole默

认授予以下权限策略：

```
{
"Version": "1",
"Statement": [
{
"Action": [
"oss:GetObject",
"oss:ListObjects",
"oss:PutObject",
"oss:DeleteObject",
"oss:ListBuckets",
"oss:AbortMultipartUpload"
],
"Resource": "*",
"Effect": "Allow"
}
]
}
```

所以默认情况下，基于MetaService的作业将**只能访问OSS数据**。如果您想基于MetaService访问其他阿里云资源，例如LogService等等，则需要给AliyunEmrEcsDefaultRole补充授予相应的权限。以上操作需要去RAM控制台完成。

注意: 当前metaservice服务只支持OSS，LogService和MNS数据的免AK操作。请**谨慎编辑，删除**默认角色，否则会造成集群创建失败或者作业运行失败。

自定义应用角色

大多数情况下，您只需要使用默认应用角色或者修改默认应用角色即可。E-MapReduce同时支持您使用自定义的应用角色。在创建集群时，您既可以使用默认应用角色，也可以选择自定义应用角色。如何创建角色并授权给服务，请参考RAM的相关文档。

访问MetaService

MetaService是一个HTTP服务，您可以直接访问这个HTTP服务来获取相关Meta信息: 例如 “ curl http://localhost:10011/cluster-region ” 可以获得当前集群所在Region。

当前MetaService支持以下几类信息：

- Region： “/cluster-region”
- 角色名： “/cluster-role-name”
- AccessKeyId：” /role-access-key-id”
- AccessKeySecret：” /role-access-key-secret”
- SecurityToken：” /role-security-token”
- 网络类型：” /cluster-network-type”

使用MetaService

基于MetaSerivce服务，我们可以在作业中免AK地访问阿里云资源，这样可以带来两个优势：

- 降低AK泄漏的风险。基于RAM的使用方式，可以将安全风险降到最低。需要什么权限就给角色授予什么权限，做到权限最小化。
- 提高用户体验。尤其在交互式访问OSS资源时，可以避免写一长串的OSS路径。

下面示例几种使用方式：

```
I. Hadoop命令行查看OSS数据

旧方式： hadoop fs -ls oss://ZaH******As1s:Ba23N**********sdaBj2@bucket.oss-cn-hangzhou-internal.aliyuncs.com/a/b/c
新方式： hadoop fs -ls oss://bucket/a/b/c

II. Hive建表

旧方式：
CREATE EXTERNAL TABLE test_table(id INT, name string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '/t'
LOCATION 'oss://ZaH******As1s:Ba23N**********sdaBj2@bucket.oss-cn-hangzhou-internal.aliyuncs.com/a/b/c';
新方式：
CREATE EXTERNAL TABLE test_table(id INT, name string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '/t'
LOCATION 'oss://bucket/a/b/c';

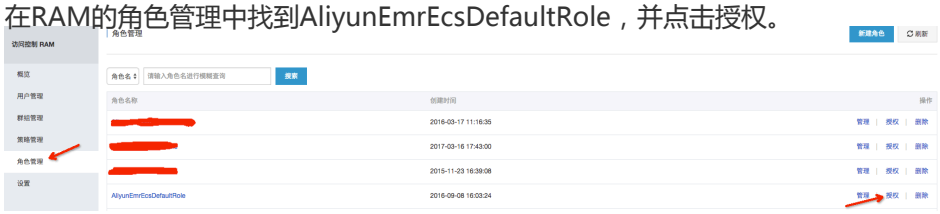
III. Spark

旧方式： val data = sc.textFile("oss://ZaH******As1s:Ba23N**********sdaBj2@bucket.oss-cn-hangzhou-internal.aliyuncs.com/a/b/c")
新方式： val data = sc.textFile("oss://bucket/a/b/c")
```

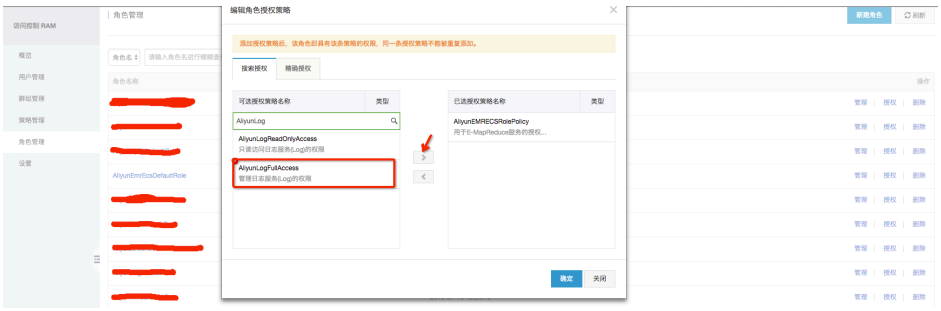
支持MetaService的数据源

目前在E-MapReduce上支持MetaService的有OSS，LogService和MNS。您可以在E-MapReduce集群上使用E-MapReduce SDK的接口免AK来读写上述数据源。需要强调的是，MetaService默认只有OSS的读写权限，如果您希望MetaService支持LogService或者MNS，请前往RAM控制台修改AliyunEmrEcsDefaultRole，增加LogService和MNS的权限。具体如何配置角色的权限，参考RAM的相关文档说明。下面将举例说明如何在给AliyunEmrEcsDefaultRole添加访问LogService的权限：

- 在RAM的角色管理中找到AliyunEmrEcsDefaultRole，并点击授权。



- 搜索找到AliyunLogFullAccess权限，并添加。



- 添加完之后，我们就可以在AliyunEmrEcsDefaultRole的角色权限策略中看到已经添加的AliyunLogFullAccess权限策略。

这样，我们就可以在E-MapReduce集群中免AK访问LogService数据了。**注意：这里演示了添加AliyunLogFullAccess权限策略，权限比较大，建议根据自己的实际需求，自定义一个权限策略，然后再授权给AliyunEmrEcsDefaultRole。**

交互式工作台

交互式工作台提供在E-MapReduce管理控制台直接编写并运行spark，sparksql，hivesql任务的能力，您可以在工作台直接看到运行结果。交互式工作台适合处理运行时间较短、想要直接看到数据结果、调试性质的任务，对于运行时间很长，需要定期执行的任务应使用作业和执行计划功能。

本节会介绍如何新建演示任务并运行，其他示例和操作说明请参考后面的章节。

创建演示任务

登录阿里云 E-MapReduce 控制台交互式工作。

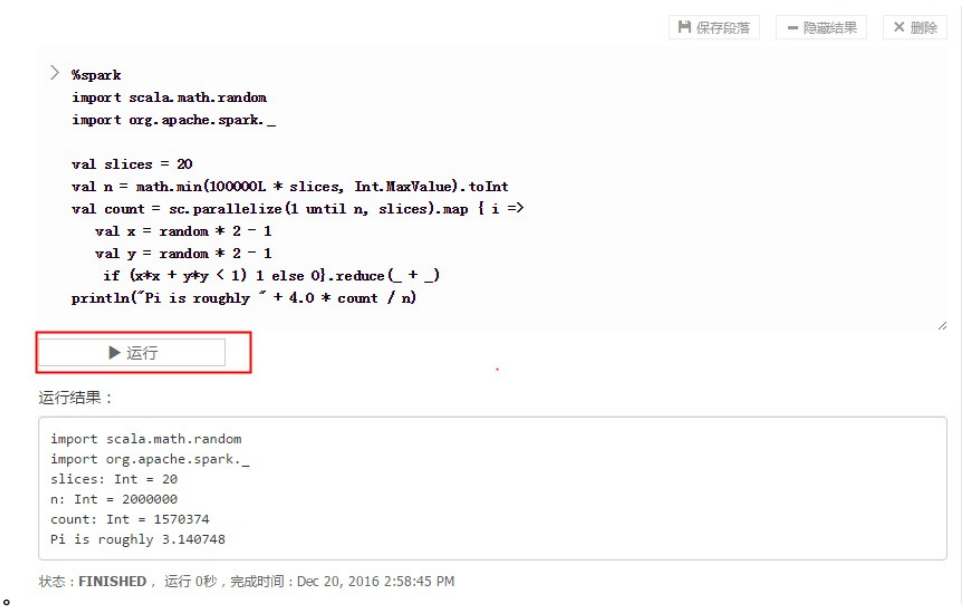
点击新建演示任务。

弹出确认框，提示运行需要的集群环境，点击确认创建演示任务。会新建三个示例的交互式任务。

运行Spark演示任务

点击EMR-Spark-Demo，显示Spark的交互式示例。运行之前首先要关联一个已经创建好的集群，点击在可用集群列表中选择一个。注意关联的集群必须是EMR-2.3以上版本，不小于三节点，4核8G即以上配置。

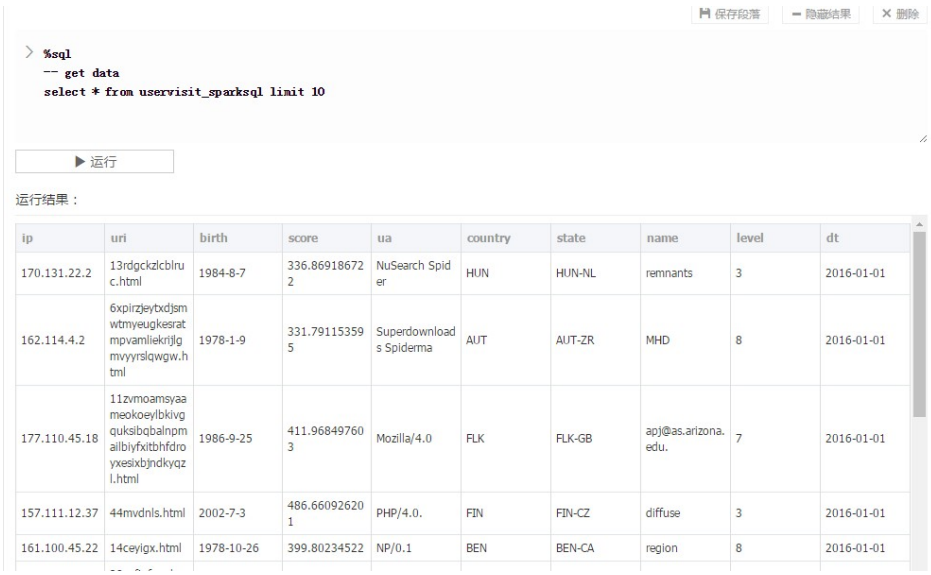
关联后，点击运行。关联的集群第一次执行Spark/SparkSQL交互式任务时会额外花费一些时间构建Spark上下文和运行环境，大概要1分钟，后续的执行就不需要再耗时构建了。运行结果显示在下方



运行SparkSQL演示任务

点击EMR-Spark-Demo，显示SparkSQL的交互式示例。运行之前依然要先关联一个已经创建好的集群， 点击右上角在可用集群列表中选择一个。

SparkSQL的演示任务有好几个演示段落，每个段落可以单独运行，也可以通过运行全部运行。运行后可以看到各段落返回的数据结果。注意创建表的段落如果运行多次会报错提示表已存在。

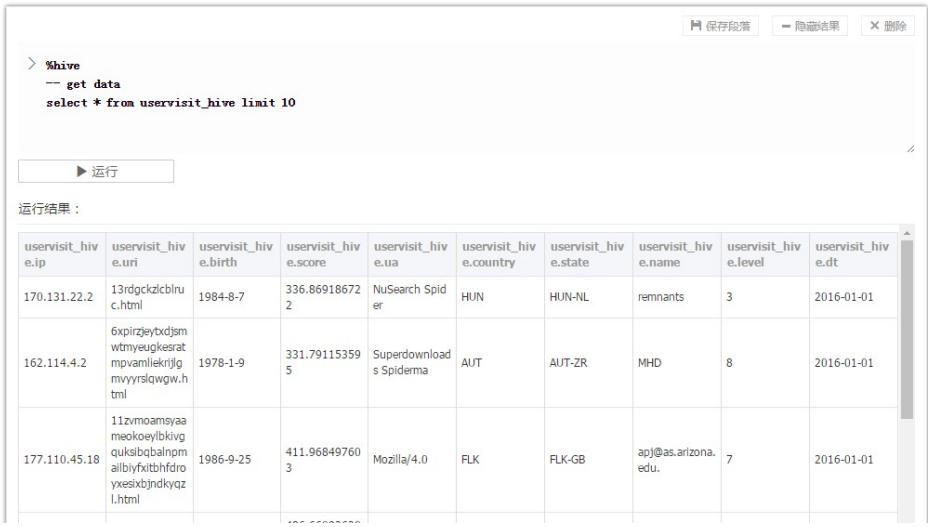


ip	uri	birth	score	ua	country	state	name	level	dt
170.131.22.2	13rdgckzlcblruc.html	1984-8-7	336.869186722	NuSearch Spider	HUN	HUN-NL	remnants	3	2016-01-01
162.114.4.2	6xpirzjeytxdjsmwtmyeugkesratmpvamliekrijlgmvyyrslqwgw.html	1978-1-9	331.791153595	Superdownloads Spiderma	AUT	AUT-ZR	MHD	8	2016-01-01
177.110.45.18	11zvmoamsyaameokoeylbkivgquksibqbalnpmailbiyfxitbhfdroyxesixbjndkyqzl.html	1986-9-25	411.968497603	Mozilla/4.0	FLK	FLK-GB	apj@as.arizona.edu.	7	2016-01-01
157.111.12.37	44mvdnls.html	2002-7-3	486.660926201	PHP/4.0.	FIN	FIN-CZ	diffuse	3	2016-01-01
161.100.45.22	14ceyigx.html	1978-10-26	399.80234522	NP/0.1	BEN	BEN-CA	region	8	2016-01-01

运行Hive演示任务

点击EMR-Hive-Demo，显示Hive的交互式示例。运行之前依然要先关联一个已经创建好的集群，点击右上角在可用集群列表中选择一个。

Hive的演示任务有好几个演示段落，每个段落可以单独运行，也可以通过运行全部运行。运行后可以看到各段落返回的数据结果。注意1.关联的集群第一次执行hive交互式任务时会额外花费一些时间构建hive客户端运行环境，大概要几十秒，后续的执行就不需要再耗时构建了。2.创建表的段落如果运行多次会报错提示表已存在。



uservisit_hive.ip	uservisit_hive.uri	uservisit_hive.birth	uservisit_hive.score	uservisit_hive.ua	uservisit_hive.country	uservisit_hive.state	uservisit_hive.name	uservisit_hive.level	uservisit_hive.dt
170.131.22.2	13rdgckzlcblruc.html	1984-8-7	336.869186722	NuSearch Spider	HUN	HUN-NL	remnants	3	2016-01-01
162.114.4.2	6xpirzjeytxdjsmwtmyeugkesratmpvamliekrijlgmvyyrslqwgw.html	1978-1-9	331.791153595	Superdownloads Spiderma	AUT	AUT-ZR	MHD	8	2016-01-01
177.110.45.18	11zvmoamsyaameokoeylbkivgquksibqbalnpmailbiyfxitbhfdroyxesixbjndkyqzl.html	1986-9-25	411.968497603	Mozilla/4.0	FLK	FLK-GB	apj@as.arizona.edu.	7	2016-01-01

取消关联集群

集群运行过交互式任务后，为了再次执行时能够快速响应，会创建进程缓存一些上下文运行环境。如果您暂时

不再执行交互式任务，想要释放缓存占用的集群资源，可以把运行过的交互式任务都取消关联，会释放掉原关联集群上占用的内存资源。

新建交互式任务

注意：要运行交互式任务的集群的配置，必须满足EMR-2.3及以上版本，不小于三节点，4核8G及以上配置。

1. 登录阿里云 E-MapReduce 控制台交互式工作。
2. 点击右侧新建交互式任务或文件-新建交互式任务

3. 填入名称，选择默认类型，关联集群可选，点击确认新建一个交互式任务。

类型目前支持三类，Spark可以编写scala spark代码，Spark SQL可以写spark支持的sql语句，Hive可以写Hive支持的sql语句。

4.关联集群,需要是一个创建好的集群，且必须是EMR-2.3及以上版本，不小于三节点，4核8G及以上配置。也可以先不关联，在运行前再关联。

目前一个账户最多创建20个交互式任务。

填写保存段落

段落是运行任务的最小单元，1个交互式任务可以填写多个段落。每个段落可以在内容开头写%spark,%sql,%hive表明该段落是scala spark代码段，spark sql，还是hive sql。类型前缀以空格或换行和实际内容分割，不写类型前缀则以交互式任务的默认类型作为该段落的运行类型。

一个创建spark临时表的示例如下：

将如下代码粘贴进段落内，会显示一个红*提醒有修改，通过保存段落按钮或运行按钮可以保存对段落内容的修改，点击段落下方的+可以新建一个段落。目前一个交互式任务最多可以创建30个段落。

```
%spark
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset

// load bank data
val bankText = sc.parallelize(
IOUtils.toString(
new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/bank.csv"),
Charset.forName("utf8")).split("\n"))
case class Bank(age: Integer, job: String, marital: String, education: String, balance: Integer)
val bank = bankText.map(s => s.split(";")).filter(s => s(0) != "\"age\"").map(
s => Bank(s(0).toInt,
s(1).replaceAll("\"", ""),
s(2).replaceAll("\"", ""),
```

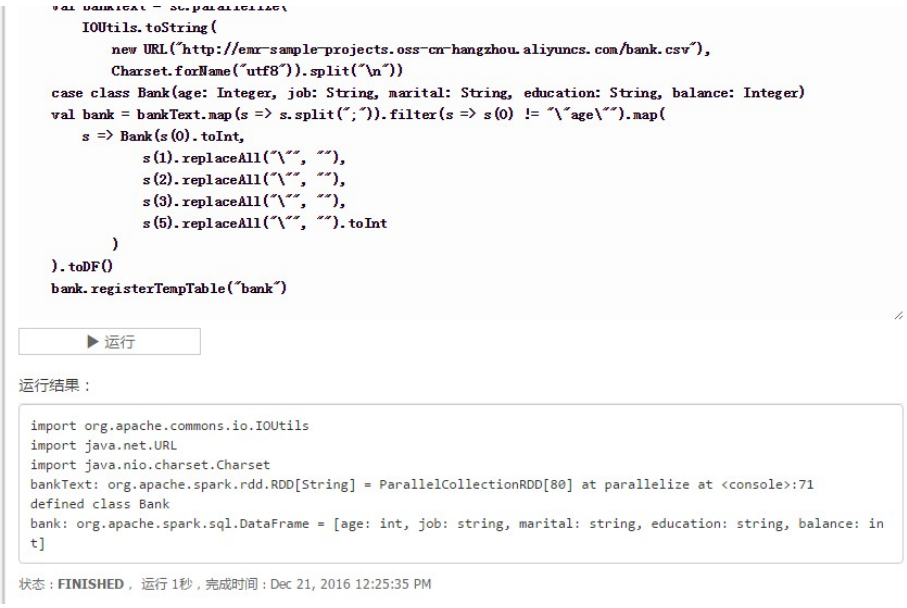
```
s(3).replaceAll("\"", ""),
s(5).replaceAll("\"", "").toInt
)
).toDF()
bank.registerTempTable("bank")
```

运行段落

运行之前首先要关联一个已经创建好的集群，如果创建交互式任务时未关联，右上角显示未关联，点击在可用集群列表中选择一个。注意关联的集群必须是EMR-2.3以上版本，不小于三节点，4核8G即以上配置。

点击运行按钮，会自动保存当前段落，运行内容，如果这是最后一个段落会自动新建一个段落。

运行后会显示当前的运行状态，还未实际运行的是PENDING，运行后是RUNNING。运行完成是FINISHED，如果有错误是ERROR,运行结果会显示在段落的运行按钮下方。运行时可以点运行按钮下方的取消按钮取消运行，取消的状态显示ABORT。



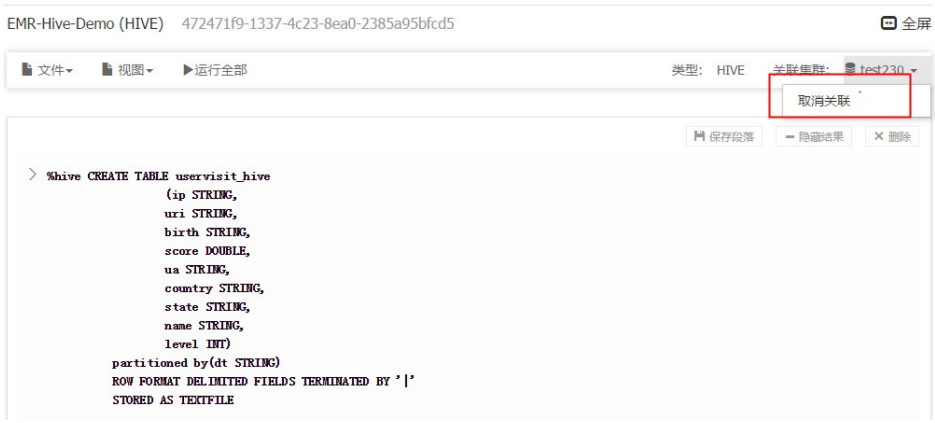
段落可以反复多次运行，只保留最后一次运行的结果。运行时不能修改段落的输入内容，运行后可以修改。

运行全部

交互式任务可以点击菜单栏上的运行全部运行所有的段落，段落会顺序提交运行。不同的类型有独立的执行队列,如果一个交互式任务包含多种段落类型，顺序提交运行后，实际在集群上的执行顺序是按照类型划分的。Spark和Spark SQL类型是顺序一个个的执行。HIVE支持并发执行，同一个集群交互式段落最大并发数是10。注意并发运行的作业同时受集群资源限制，集群规模小并发很多依然要在yarn上排队。

取消关联集群

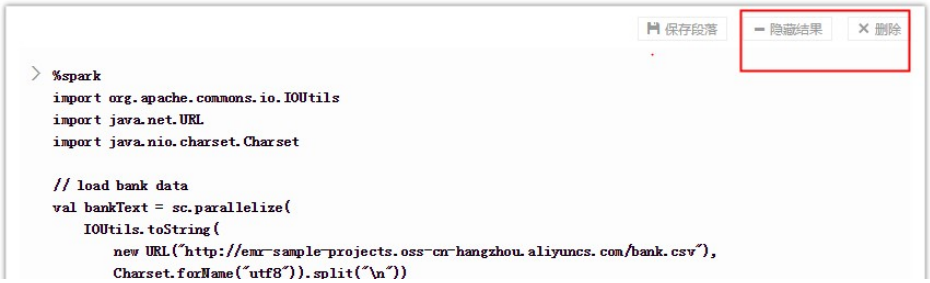
集群运行过交互式任务后，为了再次执行时能够快速响应，会创建进程缓存一些上下文运行环境。如果您暂时不再执行交互式任务，想要释放缓存占用的集群资源，可以把运行过的交互式任务都取消关联，会释放掉原关



联集群上占用的内存资源。

其他操作项

段落操作



隐藏结果/显示结果

可以将段落的结果隐藏掉，只显示段落的输入内容。

删除

删除当前段落，运行中的段落也可以删除。

文件菜单

新建交互式任务

新建一个交互式任务，并切换界面到新建的交互式任务上。

新建段落

在交互式任务的尾部添加一个新段落，一个交互式任务最多有30个段落。

保存所有段落

所有修改过的段落都会保存

删除交互式任务

删除掉当前的交互式任务。如果关联了集群会同时取消关联。

视图

只显示代码/显示代码和结果

所有段落只显示输入的代码，还是同时显示结果内容。

交互式工作台示例

段落1创建临时表

```
%spark
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset

// Zeppelin creates and injects sc (SparkContext) and sqlContext (HiveContext or SqlContext)
// So you don't need create them manually
// load bank data
val bankText = sc.parallelize(
IOUtils.toString(
new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/bank.csv"),
Charset.forName("utf8")).split("\n"))
case class Bank(age: Integer, job: String, marital: String, education: String, balance: Integer)
val bank = bankText.map(s => s.split(";")).filter(s => s(0) != "\"age\"").map(
s => Bank(s(0).toInt,
s(1).replaceAll("\"", ""),
s(2).replaceAll("\"", ""),
s(3).replaceAll("\"", ""),
```

```
s(5).replaceAll("\"", "").toInt
)
).toDF()
bank.registerTempTable("bank")
```

段落2查询表结构

```
%sql
desc bank
```

段落3查询年龄小于30各年龄段员工人数

```
%sql select age, count(1) value from bank where age < 30 group by age order by age
```

段落4 查询年龄小于等于20岁的员工信息

```
%sql select * from bank where age <= 20
```

数据准备

本示例需要您从oss上下载数据，并上传到您自己的oss bucket上。数据包含

- 用户表示例数据
- 视频表示例数据
- 播放表示例数据

分别上传到您oss bucket指定目录的userinfo子目录，videoinfo目录，playvideo目录。例如 bucket example 下的demo/userinfo目录。

将下面创建表的sql中[bucketname]替换成您的bucket名字例如example，[region]替换成您用的oss地域名如hangzhou,[bucketpath]替换成您oss的指定的路径前缀例如demo。

段落1创建用户表

```
%hive
CREATE EXTERNAL TABLE user_info(id int,sex int,age int, marital_status int) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION 'oss://[bucketname].oss-cn-[region]-internal.aliyuncs.com/[bucketpath]/userinfo'
```

段落2创建视频表

```
```

```
%hive
CREATE EXTERNAL TABLE video_info(id int,title string,type string) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION 'oss://[bucketname].oss-cn-[region]-internal.aliyuncs.com/[bucketpath]/videoinfo'
```

段落3创建播放表

```
%hive
CREATE EXTERNAL TABLE play_video(user_id int,video_id int, play_time bigint) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION 'oss://[bucketname].oss-cn-[region]-internal.aliyuncs.com/[bucketpath]/playvideo'
```

段落4用户表计数

```
%sql select count(*) from user_info
```

段落5 视频表计数

```
%sql select count(*) from video_info
```

段落6 播放表计数

```
%sql select count(*) from play_video
```

段落7统计各类型视频播放数

```
%sql select video.type, count(video.type) as count from play_video play join video_info video on (play.video_id = video.id) group by video.type order by count desc
```

段落8播放数top10的视频信息

```
%sql select video.id, video.title, video.type, video_count.count from (select video_id, count(video_id) as count from play_video group by video_id order by count desc limit 10) video_count join video_info video on (video_count.video_id = video.id) order by count desc
```

段落9播放数最高视频观看者的年龄分布

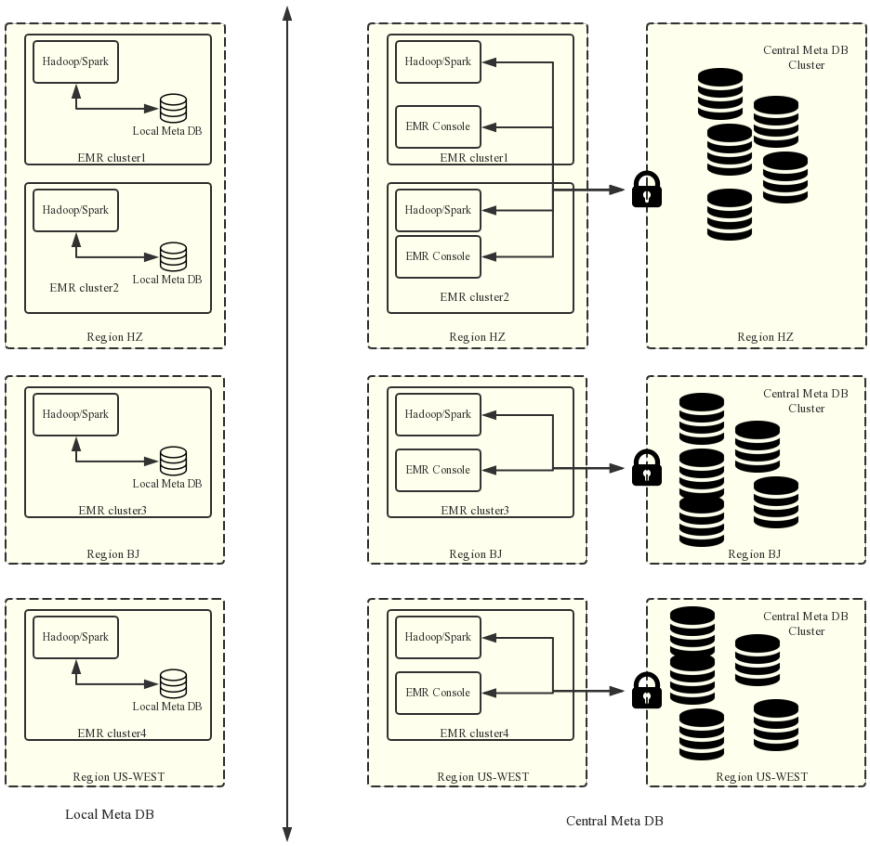
```
%sql select age , count(*) as count  from (select distinct(user_id) from play_video  where video_id =49 ) play join user_info userinfo on (play.user_id = userinfo.id) group by userinfo.age
```

段落10播放数最高视频观看者的性别，年龄段，婚姻状态分布汇总

```
%sql select if(sex=0,'女','男') as title, count(*) as count, '性别' as type from (select distinct(user_id) from play_video where video_id =49 ) play join user_info userinfo on (play.user_id = userinfo.id) group by userinfo.sex
union all
select case when userinfo.age<15 then '小于15' when age<25 then '15-25' when age<35 then '25-35' else '大于35' end , count(*) as count, '年龄段' as type from (select distinct(user_id) from play_video where video_id =49) play join user_info userinfo on (play.user_id = userinfo.id) group by case when userinfo.age<15 then '小于15' when age<25 then '15-25' when age<35 then '25-35' else '大于35' end
union all
select if(marital_status=0,'未婚','已婚') as title, count(*) as count, '婚否' as type from (select distinct(user_id) from play_video where video_id =49 ) play join user_info userinfo on (play.user_id = userinfo.id) group by marital_status
```

介绍

从EMR-2.4.0版本开始，EMR支持了统一元数据管理，在EMR-2.4.0版本之前，用户所有集群均采用的是集群本地的mysql数据库作为hive元数据库，在EMR-2.4.0版本以及之后的版本中，EMR会支持统一的高可靠的



hive元数据库: 用户可以选择
在创建集群的时候，打开我们的元数据库的开关，从而使用外部的元数据库。

注意 1：当前元数据库需要使用公网IP来连接，所以集群必须要有公网IP，同时请不要随意的切换公网IP地址，防止对应的数据库白名单失效。

注意 2：只有在创建集群的时候打开了统一元数据库这个开关才能使用表管理的功能，如果是本地的元数据库，目前还不支持进行管理。可以使用集群上的 Hue 工具来进行管理。

有了统一的元数据管理之后，就可以实现：

1. 提供持久化的元数据存储

之前元数据都是在集群内部的Mysql数据库，元数据会随着集群的释放而丢失，特别是EMR提供了灵活按量模式，集群可以按需创建用完就释放。如果用户需要保留现有的元数据信息，必须登录上集群手动将元数据信息导出。支持统一的元数据管理之后，不再存在该问题。

2. 能更方便地实现计算存储分离

EMR上可以支持将数据存放在阿里云OSS中，在大数据量的情况下将数据存储在OSS上会大大降低使用的成本，EMR集群主要用来作为计算资源，在计算完成之后机器可以随时释放，数据在OSS上，同时也不用再考虑元数据迁移的问题。

3. 更方便地实现数据共享

使用统一的元数据库，如果用户的所有数据都存放在OSS之上，则不需要做任何元数据的迁移和重建所有集群都是可以直接访问数据，这样每个EMR集群可以做不同的业务，但是可以很方便地实现数据的共享。

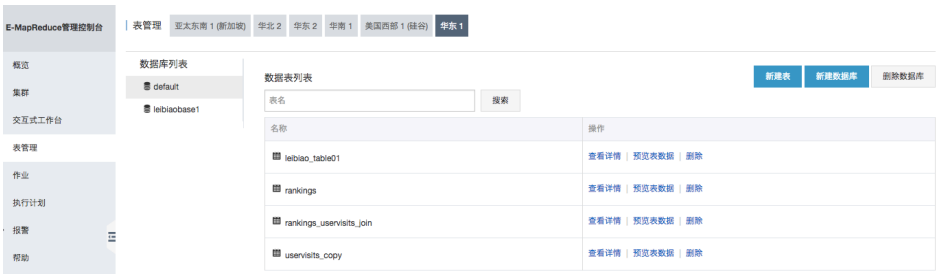
特别说明：

在支持统一元数据之前，元数据是存储在每个集群本地环境的Mysql数据库中，所以元数据会随着集群的释放消亡。在支持统一元数据之后，释放集群不会清理元数据信息。所以，在任何时候删除OSS上的数据或者集群HDFS上的数据（包括释放集群操作）的时候，需要先确认该数据对应的元数据已经删除（即要drop掉数据对应的表和数据库）。否则元数据库中可能出现一些脏的元数据信息。

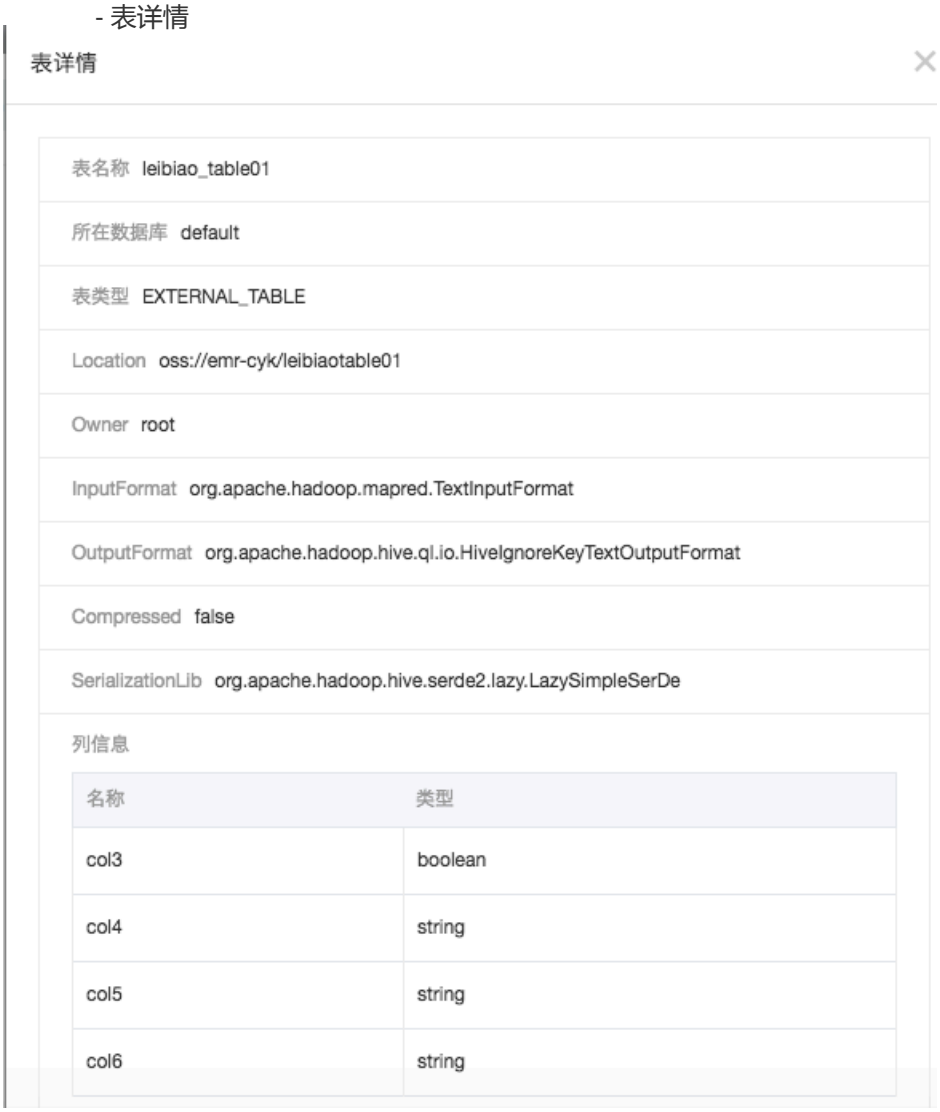
表管理操作

在EMR集群支持统一元数据支持之前，客户对集群上表的查看、增删操作必须要登录到集群内部环境操作，如果有多个集群，则需要每个集群上分别操作，非常不方便。在EMR支持统一元数据管理之后，为了用户更方便地对表进行管理，EMR在控制台上提供了表管理功能。包括对对数据库和表列表的查看、表详情的查看、新建数据库和表、删除数据库和表、数据预览等操作。

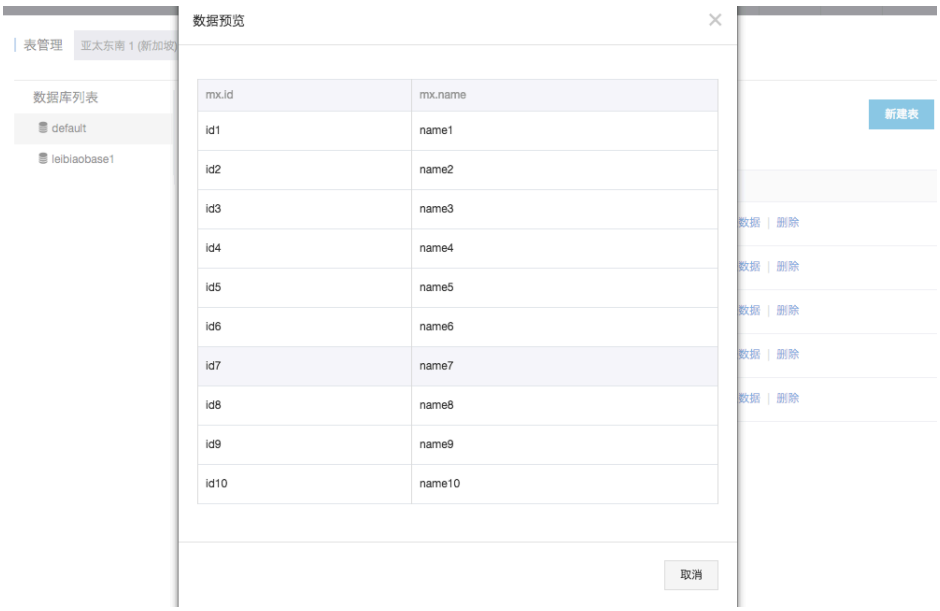
- 数据库和表列表



- 表详情


表详情
表名称 leibiao_table01
所在数据库 default
表类型 EXTERNAL_TABLE
Location oss://emr-cyk/leibiaotable01
Owner root
InputFormat org.apache.hadoop.mapred.TextInputFormat
OutputFormat org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Compressed false
SerializationLib org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
列信息
名称 类型
col3 boolean
col4 string
col5 string
col6 string

- 数据预览


表管理
数据库列表
default
leibiaobase1
数据预览
mx.id
mx.name
id1
name1
id2
name2
id3
name3
id4
name4
id5
name5
id6
name6
id7
name7
id8
name8
id9
name9
id10
name10
取消
新建表
数据 | 删除

- 创建数据库


Database
* 名称:
1-128个字符，只允许包含大小写字母、数字和"_"，且首字母不能为数字
* 数据位置:
数据存放位置不能是bucket名称，需要指定到具体的文件夹，例如: oss://bucket-name/folder-name
确认
取消

- 创建表

创建表的时候，有两种选项：手动创建表和从文件创建表

- 手动创建表，表示还没有对应的业务数据，手动输入表结构建一个空的表；
- 从文件创建表，是指在已经有业务数据的情况下，可以直接将这些业务数据作为一个表，表接口是从文件内容中解析的。注意创建表时候设置的分隔符需要跟数据文件中的分隔符对应以保证表结构正确。

分隔符不仅支持普通的逗号、空格等字符，还支持了四种特殊字符：TAB、^A、^B和^C。

特别说明：

1. 如果没有任何EMR集群，不支持进行对数据库和表的创建和删除操作。
2. 由于HDFS是每个集群内部文件系统，在没有进行特殊的网络环境设置的情况下，不同集群之间的HDFS不无法相互访问的，所以EMR表管理功能对数据库和表的创建只支持基于OSS文件系统的。
3. 数据库和表的location都不能选择整个OSS bucket，需要选择到OSS bucket下面的目录。

常见问题

1. Wrong FS: oss://yourbucket/xxx/xxx/xxx

出现这个问题，是由于删除OSS上的表数据之前，没有删除数据表对应的元数据。导致表的schema还在，但实际的数据已不存在或已移动到别的路径。可以先修改表的location为一个存在的路径，然后再删除表。

alter table test set location 'oss://your_bucket/your_folder'

可以直接在EMR控制台的交互式控制台中完成：

注意，oss://your_bucket/your_folder必须是一个存在的oss路径。

2. Wrong FS: hdfs://yourhost:9000/xxx/xxx/xxx

出现这个问题是删除了表在HDFS上的数据，但是没有删除表对应的schema信息，解决方法同上。

3. 删除hive database的时候出现：java.lang.IllegalArgumentException: java.net.UnknownHostException: xxxxxxx

出现该问题的原因，是因为在之前的集群之上创建了hive的数据库，并且数据库的位置是落在之前集群的hdfs之上，但是在集群释放的时候，没有清理掉对应的hive database，导致新建集群之后，没法访问到之前已经释放集群的hdfs数据。所以如果是手动创建了在hdfs之上的数据库和表，在释放集群的时候请记得清理。

解决方案：

首先，通过命令行登录到集群master节点上，找到hive meta db的访问地址和用户名密码信息，在$HIVE_CONF_DIR/hive-site.xml中，找到对应属性。

```
javax.jdo.option.ConnectionUserName //对应数据库用户名;
javax.jdo.option.ConnectionPassword //对应数据库访问密码;
javax.jdo.option.ConnectionURL //对应数据库访问地址和库名;
```

```
<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>${ConnectionUserName}</value>
  <description>Username to use against metastore database</description>
</property>
<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>${ConnectionPassword}</value>
  <description>password to use against metastore database</description>
</property>
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://${DBConnectionURL}/${DBName}?createDatabaseIfNotExist=true&amp;amp;characterEncoding=UTF-8</value>
  <description>JDBC connect string for a JDBC metastore</description>
</property>
```

在集群master节点上登录hive meta db:

```
mysql -h ${DBConnectionURL} -u ${ConnectionUserName} -p [回车]
[输入密码]${ConnectionPassword}
```

登录上hive meta db之后，修改对应hive database的location为该region真实存在的oss路径即可：

```
mysql> show databases;
+--------------------------------------------------+
| Database                                         |
+--------------------------------------------------+
| information_schema                               |
| xxxxxxxxx77ac43c3bd0efae77e0bf1947d45fb4c896fb99 |
+--------------------------------------------------+

mysql> use xxxxxxxxx77ac43c3bd0efae77e0bf1947d45fb4c896fb99;

mysql> select * from dbs;
+-------+------------------------+-------------------------------+----------+------------+------------+
| DB_ID | DESC                   | DB_LOCATION_URI               | NAME     | OWNER_NAME | OWNER_TYPE |
+-------+------------------------+-------------------------------+----------+------------+------------+
|     1 | Default Hive database | oss://mybucket/hive/warehouse | default  | public     | ROLE       |
|     6 | NULL                   | hdfs://dirty-hostname/warehouse | dirty_db | NULL       | USER       |
+-------+------------------------+-------------------------------+----------+------------+------------+

mysql> update dbs set DB_LOCATION_URI = 'oss://your-bucket/your-db-folder' where DB_ID = 6;
```

4. 统一元数据库迁移到rds

1. 购买rds实例，保证rds可以和集群的master节点网络是通的；最好是跟EMR的ecs在同一个安全组，这样可以直接使用rds的内网地址；
2. 出于安全考虑，对rds的ip白名单进行设置，设置只允许对应的EMR机器可以访问；
3. 在rds中创建一个database，名称为hivemeta，同时创建一个用户，把hivemeta的读写权限赋给这个用户；

   登陆到EMR集群的master节点，将master中配置的mysql中的hive元数据库的数据导出来（只导出数据，不用导表结构），信息位置（3.2以及2.7以上版本可以在配置管理中找到，以下的可以在$HIVE_CONF_DIR/hive-site.xml 下找到）：

   ```
   javax.jdo.option.ConnectionUserName //对应数据库用户名;
   javax.jdo.option.ConnectionPassword //对应数据库访问密码;
   javax.jdo.option.ConnectionURL //对应数据库访问地址和库名;
   ```

   mysqldump -t hivemeta -h <数据库地址> -u <数据库用户名> -p > /tmp/metastore.sql密码就是上面提到的配置中的密码。

   为了保证数据的一致性，在执行这一步操作之前，最好将hive的metastore服务停掉，保证导出期间不会有新的meta数据变化

   修改集群master节点（如果是ha集群两个master都需要）上的/usr/local/emr/emr-agent/run/meta_db_info.json，把里面的use_local_meta_db设置为false，meta数据库信的链接地址、用户名和密码换成rds的信息；

6. 在EMR控制台页面，进入“配置管理”页面，修改hive的配置，把meta数据库信的链接地址、用户名和密码换成rds的信息；如果是老版本的集群就修改 $HIVE_CONF_DIR/hive-site.xml 中的对应的配置为需要连接的数据库。
7. 在一台master节点上，把hive-site.xml里面的meta数据库链接地址、用户名和密码换成rds的信息，然后执行init schema:

   ```
   cd /usr/lib/hive-current/bin
   ./schematool - initSchema -dbType mysql
   ```

把之前导出来的meta数据导入rds：命令行登陆mysqlmysql -h {rds的url} -u {rds的用户名} -p进入mysql的命令行之后，执行source /tmp/metastore.sql正常情况可以完全导入进去不会报错

在EMR控制台页面，“配置管理”页面，重启hive所有组件，“RESTART ALL COMPONENTS”

10. 验证hive功能是否正常，可以在master节点上，执行hive cli，看看数据是不是跟之前一样的

Kerberos认证

E-MapReduce从EMR-2.7.x/EMR-3.5.x版本开始支持创建安全类型的集群，即集群中的开源组件以Kerberos的安全模式启动,在这种安全环境下只有经过认证的客户端(Client)才能访问集群的服务(Service,如HDFS)。

1. 前置

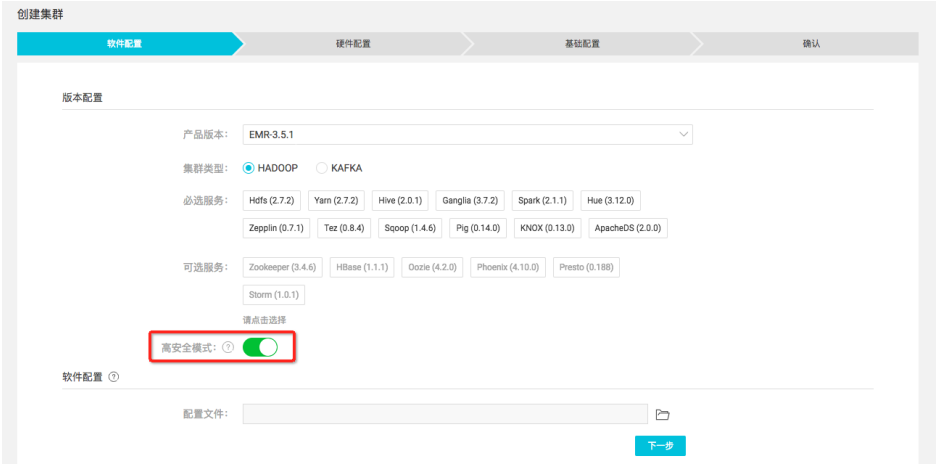
目前E-MapReduce版本中支持的Kerberos的组件列表如下所示：

组件名称	组件版本
HDFS	2.7.2
YARN	2.7.2
SPARK	2.1.1/1.6.3
HIVE	2.0.1
TEZ	0.8.4
ZOOKEEPER	3.4.6
HUE	3.12.0
ZEPPELIN	0.7.1
OOZIE	4.2.0
SQOOP	1.4.6
HBASE	1.1.1
PHOENIX	4.7.0

备注: Kafka/Presto/Storm目前版本不支持Kerberos

创建安全集群

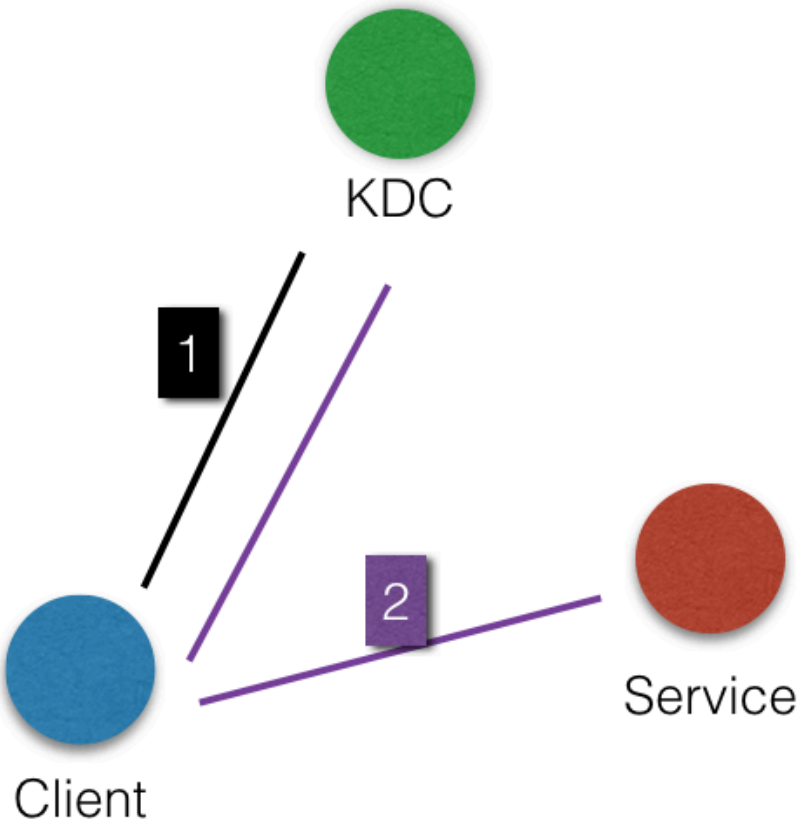
在集群创建页面的软件配置下打开安全按钮即可，如下所示:



2. Kerberos身份认证原理

Kerberos是一种基于对称密钥技术的身份认证协议，它作为一个独立的第三方的身份认证服务，可以为其它服务提供身份认证功能，且支持SSO(即客户端身份认证后，可以访问多个服务如HBase/HDFS等)。

Kerberos协议过程主要有两个阶段，第一个阶段是KDC对Client身份认证，第二个阶段是Service对Client身份认证。



KDC

Kerberos的服务端程序

Client

需要访问服务的用户(principal)，KDC和Service会对用户的身份进行认证

Service

集成了Kerberos的服务，如HDFS/YARN/HBase等

2.1 KDC对Client身份认证

当客户端用户(principal)访问一个集成了Kerberos的服务之前，需要先通过KDC的身份认证。

若身份认证通过则客户端会拿到一个TGT(Ticket Granting Ticket)，后续就可以拿该TGT去访问集成了Kerberos的服务。

2.2 Service对Client身份认证

当2.1中用户拿到TGT后，就可以继续访问Service服务。它会使用TGT以及需要访问的服务名称(如HDFS)去KDC获取SGT(Service Granting Ticket)，然后使用SGT去访问Service，Service会利用相关信息对Client进行身份认证，认证通过后就可以正常访问Service服务。

3. EMR实践

EMR的Kerberos安全集群中的服务在创建集群的时候会以Kerberos安全模式启动。

a) Kerberos服务端程序为HasServer

可以在控制台->集群配置管理->HAS查看/修改配置/重启等操作。

非HA集群部署在emr-header-1,HA集群部署在emr-header-1/emr-header-2两个节点

b) 支持四种身份认证方式

HasServer可同时支持以下4种身份认证方式，客户端可以通过配置相关参数来指定HasServer使用哪种方式进行身份认证。

兼容MIT Kerberos的身份认证方式

客户端配置:

如果在集群的某个节点上执行客户端命令，则需要将/etc/ecm/hadoop-conf/core-site.xml中hadoop.security.authentication.use.has设置为false.

如果有通过控制台的执行计划跑作业，则不能修改master节点上面/etc/ecm/hadoop-conf/core-site.xml中的值

，否则执行计划的作业认证就不通过而失败，可以使用下面的方式

export HADOOP_CONF_DIR=/etc/has/hadoop-conf临时export环境变量，该路径下的hadoop.security.authentication.use.has已经设置为false

访问方式:

Service的客户端包完全可使用开源的，如HDFS客户端等。详见

RAM身份认证

客户端配置:

如果在集群的某个节点上执行客户端命令，则需要将/etc/ecm/hadoop-conf/core-site.xml中hadoop.security.authentication.use.has设置为true，/etc/has/has-client.conf中auth_type设置为RAM.

如果有通过控制台的执行计划跑作业，则不能修改master节点上面/etc/ecm/hadoop-conf/core-site.xml以及/etc/has/has-client.conf中的值，否则执行计划的作业认证就不通过而失败，可以使用下面的方式

export HADOOP_CONF_DIR=/etc/has/hadoop-conf; export HAS_CONF_DIR=/path/to/has-client.conf临时export环境变量，其中HAS_CONF_DIR文件夹下的has-client.conf的auth_type设置为RAM

访问方式: 客户端需要使用集群中的软件包(如Hadoop/HBase等)，详见

LDAP身份认证

客户端配置:

如果在集群的某个节点上执行客户端命令，则需要将/etc/ecm/hadoop-conf/core-site.xml中hadoop.security.authentication.use.has设置为true，/etc/has/has-client.conf中auth_type设置为LDAP.

如果有通过控制台的执行计划跑作业，则不能修改master节点上面/etc/ecm/hadoop-conf/core-site.xml以及/etc/has/has-client.conf中的值，否则执行计划的作业认证就不通过而失败，可以使用下面的方式

export HADOOP_CONF_DIR=/etc/has/hadoop-conf; export HAS_CONF_DIR=/path/to/has-client.conf临时export环境变量，其中HAS_CONF_DIR文件夹下的has-client.conf的auth_type设置为LDAP

访问方式: 客户端需要使用集群中的软件包(如Hadoop/HBase等)，详见

执行计划认证

如果用户有使用EMR控制台的执行计划提交作业，则emr-header-1节点的配置必须不能被修改(默认配置)。

客户端配置:

```
emr-header-1上面的/etc/ecm/hadoop-conf/core-site.xml中hadoop.security.authentication.use.has设置为true，/etc/has/has-client.conf中auth_type设置为EMR.
```

访问方式:跟非Kerberos安全集群使用方式一致。详见

c) 其他

登陆master节点访问集群

集群管理员也可以登陆master节点访问集群服务，登陆master节点切换到has账号(默认使用兼容MIT Kerberos的方式)即可访问集群服务，方便做一些排查问题或者运维等。

```
>sudo su has
>hadoop fs -ls /
```

备注:也可以登录其他账号操作集群，前提是该账号可以通过Kerberos认证。另外，如果在master节点上需要使用 兼容MITKerberos的方式，需要在该账号下先export一个环境变量

```
export HADOOP_CONF_DIR=/etc/has/hadoop-conf/
```

兼容MIT Kerberos的身份认证方式

EMR集群中Kerberos服务端启动在master节点，涉及一些管理操作需在master节点(emr-header-1)的root账号执行。

下面以test用户访问HDFS服务为例介绍相关流程。

a) Gateway上执行hadoop fs -ls /

配置krb5.conf

```
Gateway上面使用root账号
scp root@emr-header-1:/etc/krb5.conf /etc/
```

添加principal

->登录集群emr-header-1节点，切到root账号

->进入Kerberos的admin工具

```
sh /usr/lib/has-current/bin/hadmin-local.sh /etc/ecm/has-conf -k /etc/ecm/has-conf/admin.keytab
HadminLocalTool.local: #直接按回车可以看到一些命令的用法
HadminLocalTool.local: addprinc #输入命令按回车可以看到具体命令的用法
```

```
HadminLocalTool.local: addprinc -pw 123456 test #添加test的principal,密码设置为123456
```

导出keytab文件

使用Kerberos的admin工具可以导出principal对应的keytab文件

```
HadminLocalTool.local: ktadd -k /root/test.keytab test #导出keytab文件,后续可使用该文件
```

kinit获取Ticket

在执行hdfs命令的客户端机器上面，如Gateway

-> 添加linux账号test

```
useradd test
```

-> 安装MITKerberos 客户端工具

可以使用MITKerberos tools进行相关操作(如kinit/klist等)，详细使用方式参考MITKerberos文档

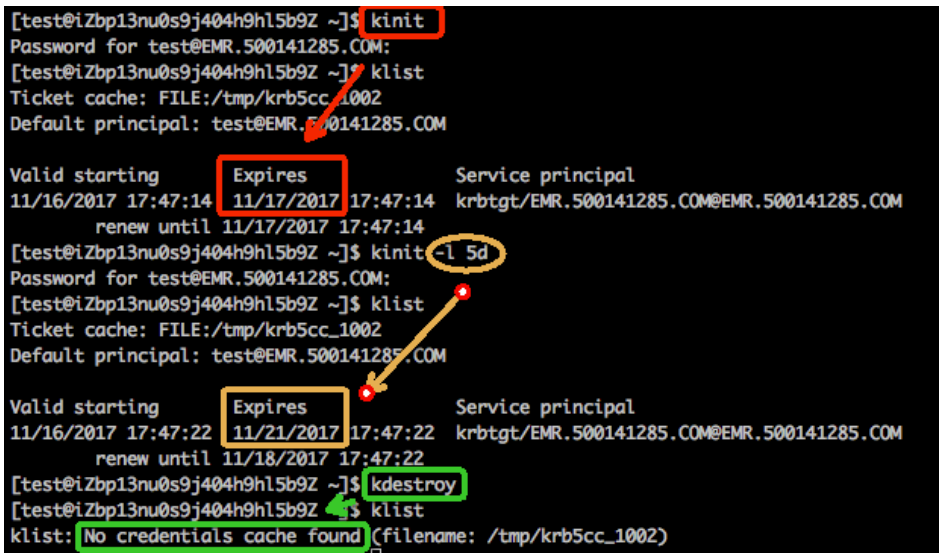
```
yum install krb5-libs krb5-workstation -y
```

-> 切到test账号执行kinit

```
su test
#如果没有keytab文件,则执行
kinit #直接回车
Password for test: 123456 #即可
#如有有keytab文件,也可执行
kinit -kt test.keytab test

#查看ticket
klist
```

备注: MITKerberos工具使用实例



执行hdfs命令

获取到Ticket后，就可以正常执行hdfs命令了

```
    hadoop fs -ls /
Found 5 items
drwxr-xr-x - hadoop hadoop 0 2017-11-12 14:23 /apps
drwx------ - hbase hadoop 0 2017-11-15 19:40 /hbase
drwxrwx--t+ - hadoop hadoop 0 2017-11-15 17:51 /spark-history
drwxrwxrwt - hadoop hadoop 0 2017-11-13 23:25 /tmp
drwxr-x--t - hadoop hadoop 0 2017-11-13 16:12 /user
```

备注: 跑yarn作业，需要提前在集群中所有节点添加对应的linux账号(详见下文中[EMR集群添加test账号].

b) java代码访问HDFS

使用本地ticket cache

备注:需要提前执行kinit获取ticket,且ticket过期后程序会访问异常。

```
public static void main(String[] args) throws IOException {
Configuration conf = new Configuration();

//加载hdfs的配置,配置从emr集群上复制一份
conf.addResource(new Path("/etc/ecm/hadoop-conf/hdfs-site.xml"));
conf.addResource(new Path("/etc/ecm/hadoop-conf/core-site.xml"));

//需要在程序所在linux账号下,提前kinit获取ticket
UserGroupInformation.setConfiguration(conf);
UserGroupInformation.loginUserFromSubject(null);
```

```
FileSystem fs = FileSystem.get(conf);
FileStatus[] fsStatus = fs.listStatus(new Path("/"));
for(int i = 0; i < fsStatus.length; i++){
System.out.println(fsStatus[i].getPath().toString());
}
}
```

使用keytab文件(推荐)

备注:keytab长期有效，跟本地ticket无关

```
public static void main(String[] args) throws IOException {
String keytab = args[0];
String principal = args[1];

Configuration conf = new Configuration();

//加载hdfs的配置,配置从emr集群上复制一份
conf.addResource(new Path("/etc/ecm/hadoop-conf/hdfs-site.xml"));
conf.addResource(new Path("/etc/ecm/hadoop-conf/core-site.xml"));

//直接使用keytab文件,该文件从emr集群master-1上面执行相关命令获取[文档前面有介绍命令]
UserGroupInformation.setConfiguration(conf);
UserGroupInformation.loginUserFromKeytab(principal, keytab);

FileSystem fs = FileSystem.get(conf);
FileStatus[] fsStatus = fs.listStatus(new Path("/"));
for(int i = 0; i < fsStatus.length; i++){
System.out.println(fsStatus[i].getPath().toString());
}
}
```

附pom依赖:

```
    <dependencies>
<dependency>
<groupId>org.apache.hadoop</groupId>
<artifactId>hadoop-common</artifactId>
<version>2.7.2</version>
</dependency>
<dependency>
<groupId>org.apache.hadoop</groupId>
<artifactId>hadoop-hdfs</artifactId>
<version>2.7.2</version>
</dependency>
</dependencies>
```

RAM身份认证

EMR集群中的Kerberos服务端除了可以支持第一种MIT Kerberos兼容的使用方式，也可以支持Kerberos客户端使用RAM作为身份信息进行身份认证。

RAM产品可以创建/管理子账号，通过子账号实现对云上各个资源的访问控制。

主账号的**管理员**可以在RAM的用户管理界面创建一个子账号(子账户名称必须符合linux用户的规范)，然后将子账号的AccessKey下载下来提供给该子账号对应的开发人员，后续开发人员可以通过配置AccessKey，从而通过Kerberos认证访问集群服务。

使用RAM身份认证不需要像第一部分MIT Kerberos使用方式一样，提前在Kerberos服务端添加principle等操作

下面以已经创建的子账号test在Gateway访问为例:

EMR集群添加test账号

EMR的安全集群的yarn使用了LinuxContainerExecutor，在集群上跑yarn作业必须要在集群所有节点上面添加跑作业的用户账号，LinuxContainerExecutor执行程序过程中会根据用户账号进行相关的权限校验。

EMR集群管理员在EMR集群的master节点上执行:

```
sudo su hadoop
sh adduser.sh test 1 2
```

附:adduser.sh代码

```
 #添加的账户名称
user_name=$1
#集群master节点个数,如HA集群有2个master
master_cnt=$2
#集群worker节点个数
worker_cnt=$3

for((i=1;i<=$master_cnt;i++))
do
ssh -o StrictHostKeyChecking=no emr-header-$i sudo useradd $user_name
done

for((i=1;i<=$worker_cnt;i++))
do
ssh -o StrictHostKeyChecking=no emr-worker-$i sudo useradd $user_name
done
```

Gateway管理员在Gateway机器上添加test用户

```
useradd test
```

Gateway管理员配置Kerberos基础环境

```
 sudo su root
sh config_gateway_kerberos.sh 10.27.230.10 /pathto/emrheader1_pwd_file

#确保Gateway上面/etc/ecm/hadoop-conf/core-site.xml中值为true
<property>
<name>hadoop.security.authentication.use.has</name>
<value>true</value>
</property>
```

附: config_gateway_kerberos.sh脚本代码

```
 #EMR集群的emr-header-1的ip
masterip=$1
#保存了masterip对应的root登录密码文件
masterpwdfile=$2

if ! type sshpass >/dev/null 2>&1; then
yum install -y sshpass
fi

## Kerberos conf
sshpass -f $masterpwdfile scp root@$masterip:/etc/krb5.conf /etc/
mkdir /etc/has
sshpass -f $masterpwdfile scp root@$masterip:/etc/has/has-client.conf /etc/has
sshpass -f $masterpwdfile scp root@$masterip:/etc/has/truststore /etc/has/
sshpass -f $masterpwdfile scp root@$masterip:/etc/has/ssl-client.conf /etc/has/

#修改Kerberos客户端配置,将默认的auth_type从EMR改为RAM
#也可以手工修改该文件
sed -i 's/EMR/RAM/g' /etc/has/has-client.conf
```

test用户登录Gateway配置AccessKey

```
 登录Gateway的test账号
#执行脚本
sh add_accesskey.sh test
```

附: add_accesskey.sh脚本(修改一下AccessKey)

```
 user=$1
if [[ `cat /home/$user/.bashrc | grep 'export AccessKey'` == "" ]];then
echo "
#修改为test用户的AccessKeyId/AccessKeySecret
export AccessKeyId=YOUR_AccessKeyId
export AccessKeySecret=YOUR_AccessKeySecret
" >>~/.bashrc
else
echo $user AccessKey has been added to .bashrc
```

```
fi
```

test用户执行命令

经过以上步骤，test用户可以执行相关命令访问集群服务了。

执行hdfs命令

```
[test@gateway ~]$ hadoop fs -ls /
17/11/19 12:32:15 INFO client.HasClient: The plugin type is: RAM
Found 4 items
drwxr-x--- - has hadoop 0 2017-11-18 21:12 /apps
drwxrwxrwt - hadoop hadoop 0 2017-11-19 12:32 /spark-history
drwxrwxrwt - hadoop hadoop 0 2017-11-18 21:16 /tmp
drwxrwxrwt - hadoop hadoop 0 2017-11-18 21:16 /user
```

跑hadoop作业

```
[test@gateway ~]$ hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.2.jar pi 10 1
```

跑spark作业

```
[test@gateway ~]$ spark-submit --conf spark.ui.view.acls=* --class org.apache.spark.examples.SparkPi --master yarn-client --driver-memory 512m --num-executors 1 --executor-memory 1g --executor-cores 2 /usr/lib/spark-current/examples/jars/spark-examples_2.11-2.1.1.jar 10
```

LDAP身份认证

EMR集群还支持基于LDAP的身份认证，通过LDAP来管理账号体系，Kerberos客户端使用LDAP中的账号信息作为身份信息进行身份认证。

LDAP账号可以是和其它服务共用，比如Hue等，只需要在Kerberos服务端进行相关配置即可。用户可以使用EMR集群中已经配置好的LDAP服务(ApacheDS)，也可以使用已经存在的LDAP服务，只需要在Kerberos服务端进行相关配置即可。

下面以集群中已经默认启动的LDAP服务(ApacheDS)为例:

Gateway管理对基础环境进行配置(跟第二部分RAM中的一致，如果已经配置可以跳过)

区别的地方只是/etc/has/has-client.conf中的auth_type需要改为LDAP

也可以不修改/etc/has/has-client.conf,用户test在自己的账号下拷贝一份该文件进行修改

auth_type，然后通过环境变量指定路径，如:

export HAS_CONF_DIR=/home/test/has-conf

EMR控制台配置LDAP管理员用户名/密码到Kerberos服务端(HAS)

进入EMR控制台集群的配置管理-HAS软件下，将LDAP的管理员用户名和密码配置到对应的bind_dn和bind_password字段，然后重启HAS服务。

此例中，LDAP服务即EMR集群中的ApacheDS服务，相关字段可以从ApacheDS中获取。

EMR集群管理员在LDAP中添加用户信息

- 获取ApacheDS的LDAP服务的管理员用户名和密码在EMR控制台集群的配置管理/ApacheDS的配置中可以查看manager_dn和manager_password

  在ApacheDS中添加test用户名和密码

```
 登录集群emr-header-1节点root账号
新建test.ldif文件,内容如下：
dn: cn=test,ou=people,o=emr
objectclass: inetOrgPerson
objectclass: organizationalPerson
objectclass: person
objectclass: top
cn: test
sn: test
mail: test@example.com
userpassword: test1234

#添加到LDAP，其中-w 对应修改成密码manager_password
ldapmodify -x -h localhost -p 10389 -D "uid=admin,ou=system" -w "Ns1aSe" -a -f test.ldif

#删除test.ldif
rm test.ldif
```

  将添加的用户名/密码提供给test使用

用户test配置LDAP信息

```
 登录Gateway的test账号
#执行脚本
sh add_ldap.sh test
```

附: add_ldap.sh脚本(修改一下LDAP账号信息)

```
 user=$1
```

```
if [[ `cat /home/$user/.bashrc | grep 'export LDAP_'` == "" ]];then
echo "
#修改为test用户的LDAP_USER/LDAP_PWD
export LDAP_USER=YOUR_LDAP_USER
export LDAP_PWD=YOUR_LDAP_USER
" >>~/.bashrc
else
echo $user LDAP user info has been added to .bashrc
fi
```

用户test访问集群服务

执行hdfs命令

```
[test@iZbp1cyio18s5ymggr7yhrZ ~]$ hadoop fs -ls /
17/11/19 13:33:33 INFO client.HasClient: The plugin type is: LDAP
Found 4 items
drwxr-x--- - has hadoop 0 2017-11-18 21:12 /apps
drwxrwxrwt - hadoop hadoop 0 2017-11-19 13:33 /spark-history
drwxrwxrwt - hadoop hadoop 0 2017-11-19 12:41 /tmp
drwxrwxrwt - hadoop hadoop 0 2017-11-19 12:41 /user
```

跑Hadoop/Spark作业等

E-MapReduce控制台-执行计划

1 主账号访问

在主账号登陆E-MapReduce控制台的情况下，在执行计划页面运行相关执行计划，将作业提交到安全集群上面执行，以hadoop用户访问作业中涉及的相关开源组件服务.

2 子账号访问

在RAM子账号登陆E-MapReduce控制台的情况下，在执行计划页面运行相关执行计划，将作业提交到安全集群上面执行，以RAM子账号对应的用户名访问作业中涉及的相关开源组件服务.

3 示例

- 主账号管理员根据需求创建多个子账号(如A/B/C),在RAM控制台页面给子账号授予AliyunEMRFullAccess的权限后，子账号就能正常登陆并使用E-MapReduce控制台上的相关功能。
- 主账号管理员将子账号提供给相关开发人员

  开发人员完成作业的创建/执行计划的创建,然后启动运行执行计划提交作业到集群运行，最终在集群上面以该子账号对应的用户名(A/B/C)去访问相关的组件服务

  备注:周期调度的执行计划目前统一以hadoop账号执行

- 组件服务使用子账户的用户名进行相关的权限控制，如子账户A是否有权限访问hdfs中的某个文件等。

组件授权

HDFS授权

HDFS开启了权限控制后，用户访问HDFS需要有合法的权限才能正常操作HDFS，如读取数据/创建文件夹等。

1. 添加配置

HDFS权限相关的配置如下:

dfs.permissions.enabled

开启权限检查，即使该值为false， chmod/chgrp/chown/setfacl操作还是会进行权限检查

dfs.datanode.data.dir.perm

datanode使用的本地文件夹路径的权限，默认755

fs.permissions.umask-mode

权限掩码，在新建文件/文件夹的时候的默认权限设置

新建文件: 0666 & ^umask

新建文件夹: 0777 & ^umask

默认umask值为022，即新建文件权限为644(666&^022=644)，新建文件夹权限为755(777&^022=755)

EMR的Kerberos安全集群默认设置为027，对应新建文件权限为640，新建文件夹权限为750

dfs.namenode.acls.enabled

打开ACL控制，打开后除了可以对owner/group进行权限控制外，还可以对其它用户进行设置。

设置ACL相关命令:

```
hadoop fs -getfacl [-R] <path>
```

```
hadoop fs -setfacl [-R] [-b |-k -m |-x <acl_spec> <path>] |[--set <acl_spec> <path>]
```

如:

```
 su test
#test用户创建文件夹
hadoop fs -mkdir /tmp/test

#查看创建的文件夹的权限
hadoop fs -ls /tmp
drwxr-x--- - test hadoop 0 2017-11-26 21:18 /tmp/test

#设置acl,授权给foo用户rwx
hadoop fs -setfacl -m user:foo:rwx /tmp/test

#查看文件权限(+号表示设置了ACL)
hadoop fs -ls /tmp/
drwxrwx---+ - test hadoop 0 2017-11-26 21:18 /tmp/test

#查看acl
hadoop fs -getfacl /tmp/test
# file: /tmp/test
# owner: test
# group: hadoop
user::rwx
user:foo:rwx
group::r-x
mask::rwx
other::---
```

dfs.permissions.superusergroup

超级用户组，属于该组的用户都具有超级用户的权限

2. 重启HDFS服务

对于Kerberos安全集群，已经默认设置了HDFS的权限(umask设置为027)，无需配置和重启服务。

对于非Kerberos安全集群需要添加配置并重启服务

3.其它

- umask值可以根据需求自行修改
- HDFS是一个基础的服务，Hive/HBase等都是基于HDFS，所以在配置其它上层服务时，需要提前配置好HDFS的权限控制。
- 在HDFS开启权限后，需要设置好服务的(如spark的/spark-history、yarn的/tmp/$user/等)

sticky bit:

针对文件夹可设置sticky bit，可以防止除了superuser/file owner/dir owner之外的其它用户删除该

文件夹中的文件/文件夹(即使其它用户对该文件夹有rwx权限)。如:

```
#即在第一位添加数字1
hadoop fs -chmod 1777 /tmp
hadoop fs -chmod 1777 /spark-history
hadoop fs -chmod 1777 /user/hive/warehouse
```

YARN授权

YARN的授权根据授权实体，可以分为服务级别的授权、队列级别的授权。

1. 服务级别的授权

详见Hadoop官方文档

- 控制特定用户访问集群服务，如提交作业
- 配置在hadoop-policy.xml
- 服务级别的权限校验在其他权限校验之前(如HDFS的permission检查/yarn提交作业到队列控制)

备注：

一般设置了HDFS permission检查/yarn队列资源控制，可以不设置服务级别的授权控制，用户可以根据自己需求进行相关配置。

2. 队列级别的授权

YARN可以通过队列对资源进行授权管理，有两种队列调度 Capacity Scheduler和Fair Scheduler

这里以Capacity Scheduler为例。

2.1 添加配置

队列也有两个级别的授权，一个是提交作业到队列的授权，一个是管理队列的授权

备注：

a) 队列的ACL的控制对象为user/group，设置相关参数时，user和group可同时设置，中间用空格分开，user/group内部可用逗号分开，只有一个空格表示任何人都没有权限。

b) 队列ACL继承: 如果一个user/group可以向某个队列中提交应用程序，则它可以向它的所有子队列中提交应用程序，同理管理队列的ACL也具有继承性。所以如果要防止某个user/group提交作业到某个队列，则需要设置该队列以及该队列的所有父队列的ACL来限制该user/group的提交作业的权限。

yarn.acl.enable

ACL开关，设置为true

yarn.admin.acl

yarn的管理员设置，如可执行yarn rmadmin/yarn kill等命令，该值必须配置，否则后续的队列相关的acl管理员设置无法生效。

如上备注，配置值时可以设置user/group：

```
 user1,user2 group1,group2 #user和group用空格隔开
group1,group2 #只有group情况下，必须在最前面加上空格
```

EMR集群中需将has配置为admin的acl权限

yarn.scheduler.capacity.${queue-name}.acl_submit_applications

设置能够向该队列提交的user/group

其中${queue-name}为队列的名称，可以是多级队列，，注意多级情况下的ACL继承机制。如:

```
 #queue-name=root
<property>
<name>yarn.scheduler.capacity.root.acl_submit_applications</name>
<value> </value> #空格表示任何人都无法往root队列提交作业
</property>

#queue-name=root.testqueue
<property>
<name>yarn.scheduler.capacity.root.testqueue.acl_submit_applications</name>
<value>test testgrp</value> #testqueue只允许test用户/testgrp组提交作业
</property>
```

yarn.scheduler.capacity.${queue-name}.acl_administer_queue

设置某些user/group管理队列，比如kill队列中作业等

queue-name可以是多级，注意多级情况下的ACL继承机制。

```
 #queue-name=root
<property>
<name>yarn.scheduler.capacity.root.acl_administer_queue</name>
<value> </value>
</property>

#queue-name=root.testqueue
<property>
<name>yarn.scheduler.capacity.root.testqueue.acl_administer_queue</name>
<value>test testgrp</value>
</property>
```

2.2. 重启YARN服务

对于Kerberos安全集群已经默认开启ACL，用户可以根据自己需求配置队列的相关ACL权限控制。

对于非Kerberos安全集群根据上述开启ACL并配置好队列的权限控制，重启YARN服务

2.3 配置示例

yarn-site.xml

```
 <property>
<name>yarn.acl.enable</name>
<value>true</value>
</property>

<property>
<name>yarn.admin.acl</name>
<value>has</value>
</property>
```

capacity-scheduler.xml

default队列: 禁用default队列，不允许任何用户提交或管理

q1队列: 只允许test用户提交作业以及管理队列(如kill)

q2队列: 只允许foo用户提交作业以及管理队列

```
<configuration>
<property>
<name>yarn.scheduler.capacity.maximum-applications</name>
<value>10000</value>
<description>Maximum number of applications that can be pending and running.</description>
</property>
<property>
<name>yarn.scheduler.capacity.maximum-am-resource-percent</name>
<value>0.25</value>
<description>Maximum percent of resources in the cluster which can be used to run application masters i.e.
controls number of concurrent running applications.
</description>
</property>
<property>
<name>yarn.scheduler.capacity.resource-calculator</name>
<value>org.apache.hadoop.yarn.util.resource.DefaultResourceCalculator</value>
</property>
<property>
<name>yarn.scheduler.capacity.root.queues</name>
<value>default,q1,q2</value>
<!-- 3个队列-->
<description>The queues at the this level (root is the root queue).</description>
</property>
<property>
<name>yarn.scheduler.capacity.root.default.capacity</name>
```

```
<value>0</value>
<description>Default queue target capacity.</description>
</property>
<property>
<name>yarn.scheduler.capacity.root.default.user-limit-factor</name>
<value>1</value>
<description>Default queue user limit a percentage from 0.0 to 1.0.</description>
</property>
<property>
<name>yarn.scheduler.capacity.root.default.maximum-capacity</name>
<value>100</value>
<description>The maximum capacity of the default queue.</description>
</property>
<property>
<name>yarn.scheduler.capacity.root.default.state</name>
<value>STOPPED</value>
<!-- default队列状态设置为STOPPED-->
<description>The state of the default queue. State can be one of RUNNING or STOPPED.</description>
</property>
<property>
<name>yarn.scheduler.capacity.root.default.acl_submit_applications</name>
<value> </value>
<!-- default队列禁止提交作业-->
<description>The ACL of who can submit jobs to the default queue.</description>
</property>
<property>
<name>yarn.scheduler.capacity.root.default.acl_administer_queue</name>
<value> </value>
<!-- 禁止管理default队列-->
<description>The ACL of who can administer jobs on the default queue.</description>
</property>
<property>
<name>yarn.scheduler.capacity.node-locality-delay</name>
<value>40</value>
</property>
<property>
<name>yarn.scheduler.capacity.queue-mappings</name>
<value>u:test:q1,u:foo:q2</value>
<!-- 队列映射，test用户自动映射到q1队列-->
<description>A list of mappings that will be used to assign jobs to queues. The syntax for this list is
[u|g]:[name]:[queue_name][,next mapping]* Typically this list will be used to map users to queues,for
example, u:%user:%user maps all users to queues with the same name as the user.
</description>
</property>
<property>
<name>yarn.scheduler.capacity.queue-mappings-override.enable</name>
<value>true</value>
<!-- 上述queue-mappings设置的映射，是否覆盖客户端设置的队列参数-->
<description>If a queue mapping is present, will it override the value specified by the user? This can be used
by administrators to place jobs in queues that are different than the one specified by the user. The default
is false.
</description>
</property>

<property>
<name>yarn.scheduler.capacity.root.acl_submit_applications</name>
```

```
<value> </value>
<!-- ACL继承性，父队列需控制住权限-->
<description>
The ACL of who can submit jobs to the root queue.
</description>
</property>

<property>
<name>yarn.scheduler.capacity.root.q1.acl_submit_applications</name>
<value>test</value>
<!-- q1只允许test用户提交作业-->
</property>

<property>
<name>yarn.scheduler.capacity.root.q2.acl_submit_applications</name>
<value>foo</value>
<!-- q2只允许foo用户提交作业-->
</property>

<property>
<name>yarn.scheduler.capacity.root.q1.maximum-capacity</name>
<value>100</value>
</property>
<property>
<name>yarn.scheduler.capacity.root.q2.maximum-capacity</name>
<value>100</value>
</property>

<property>
<name>yarn.scheduler.capacity.root.q1.capacity</name>
<value>50</value>
</property>

<property>
<name>yarn.scheduler.capacity.root.q2.capacity</name>
<value>50</value>
</property>

<property>
<name>yarn.scheduler.capacity.root.acl_administer_queue</name>
<value> </value>
<!-- ACL继承性，父队列需控制住权限-->
</property>

<property>
<name>yarn.scheduler.capacity.root.q1.acl_administer_queue</name>
<value>test</value>
<!-- q1队列只允许test用户管理，如kill作业-->
</property>

<property>
<name>yarn.scheduler.capacity.root.q2.acl_administer_queue</name>
<value>foo</value>
<!-- q2队列只允许foo用户管理，如kill作业-->
</property>
```

```
<property>
<name>yarn.scheduler.capacity.root.q1.state</name>
<value>RUNNING</value>
</property>

<property>
<name>yarn.scheduler.capacity.root.q2.state</name>
<value>RUNNING</value>
</property>

</configuration>
```

Hive授权

Hive内置有两种授权机制，

基于底层HDFS的权限(Storage Based Authorization)

基于标准SQL的grant等命令(SQL Standards Based Authorization)。

详见Hive官方文档

备注

两种授权机制可以同时配置，不冲突。

Storage Based Authorization(针对HiveMetaStore)

场景:

如果集群的用户可以直接通过HDFS/Hive Client访问Hive的数据，需要对Hive在HDFS中的数据进行相关的权限控制，通过HDFS权限控制，进而可以控制Hive SQL相关的操作权限。

详见Hive文档

1. 添加配置

在集群的配置管理页面->Hive->配置->hive-site.xml->添加自定义配置

```
<property>
<name>hive.metastore.pre.event.listeners</name>
<value>org.apache.hadoop.hive.ql.security.authorization.AuthorizationPreEventListener</value>
</property>
<property>
<name>hive.security.metastore.authorization.manager</name>
<value>org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationProvider</value>
</property>
<property>
<name>hive.security.metastore.authenticator.manager</name>
<value>org.apache.hadoop.hive.ql.security.HadoopDefaultMetastoreAuthenticator</value>
```

```
</property>
```

2. 重启HiveMetaStore

在集群配置管理页面重启HiveMetaStore

3. HDFS权限控制

EMR的Kerberos安全集群已经设置了Hive的warehouse的HDFS相关权限；

对于非Kerberos安全集群，用户需要做如下步骤设置hive基本的HDFS权限:

- 打开HDFS的权限

配置Hive的warehouse权限

```
hadoop fs -chmod 1771 /user/hive/warehouse

也可以设置成，1表示stick bit(不能删除别人创建的文件/文件夹)
hadoop fs -chmod 1777 /user/hive/warehouse
```

有了上述设置基础权限后，可以通过对warehouse文件夹授权，让相关用户/用户组能够正常创建表/读写表等

```
    sudo su has

#授予test对warehouse文件夹rwx权限
hadoop fs -setfacl -m user:test:rwx /user/hive/warehouse

#授予hivegrp对warehouse文件夹rwx权限
hadoo fs -setfacl -m group:hivegrp:rwx /user/hive/warehouse
```

经过HDFS的授权后，让相关用户/用户组能够正常创建表/读写表等，而且 不同的账号创建的hive表在HDFS中的数据只能自己的账号才能访问。

4. 验证

test用户建表testtbl

```
hive> create table testtbl(a string);
FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTask.
MetaException(message:Got exception: org.apache.hadoop.security.AccessControlException Permission
denied: user=test, access=WRITE, inode="/user/hive/warehouse/testtbl":hadoop:hadoop:drwxrwx--t
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(FSPermissionChecker.java:320)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(FSPermissionChecker.java:292)
```

上面显示错误没有权限，需要给test用户添加权限

```
```

```
#从root账号切到has账号
su has

#给test账号添加acl，对warehouse目录的rwx权限
hadoop fs -setfacl -m user:test:rwx /user/hive/warehouse
```

test账号再创建database，成功

```
hive> create table testtbl(a string);
OK
Time taken: 1.371 seconds

#查看hdfs中testtbl的目录，从权限可以看出test用户创建的表数据只有test和hadoop组可以读取，其他用户没有任何权限
hadoop fs -ls /user/hive/warehouse
drwxr-x--- - test hadoop 0 2017-11-25 14:51 /user/hive/warehouse/testtbl

#插入一条数据
hive>insert into table testtbl select "hz"
```

- foo用户访问testtbl

```
#drop table
hive> drop table testtbl;
FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTask.
MetaException(message:Permission denied: user=foo, access=READ,
inode="/user/hive/warehouse/testtbl":test:hadoop:drwxr-x---
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(FSPermissionChecker.java:320)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:219)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:190)

#alter table
hive> alter table testtbl add columns(b string);
FAILED: SemanticException Unable to fetch table testtbl. java.security.AccessControlException: Permission denied:
user=foo, access=READ, inode="/user/hive/warehouse/testtbl":test:hadoop:drwxr-x---
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(FSPermissionChecker.java:320)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:219)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:190)
at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPermission(FSDirectory.java:1720)

#select
hive> select * from testtbl;
FAILED: SemanticException Unable to fetch table testtbl. java.security.AccessControlException: Permission denied:
user=foo, access=READ, inode="/user/hive/warehouse/testtbl":test:hadoop:drwxr-x---
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(FSPermissionChecker.java:320)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:219)
```

可见foo用户不能对test用户创建的表做任何的操作，如果想要授权给foo，需要通过HDFS的授权来实现

```
su has
#只授权读的权限，也可以根据情况授权写权限(比如alter)
```

```
#备注: -R 将testtbl文件夹下的文件也设置可读
hadoop fs -setfacl -R -m user:foo:r-x /user/hive/warehouse/testtbl

#可以select成功
hive> select * from testtbl;
OK
hz
Time taken: 2.134 seconds, Fetched: 1 row(s)
```

备注:

一般可以根据需求新建一个hive用户的group，然后通过给group授权，后续将新用户添加到group中，同一个group的数据权限都可以访问。

SQL Standards Based Authorization

场景

如果集群的用户不能直接通过hdfs/hive client访问，只能通过HiveServer(beeline/jdbc等)来执行hive相关的命令，可以使用 SQL Standards Based Authorization的授权方式。

如果用户能够使用hive shell等方式，即使做下面的一些设置操作，只要用户客户端的hive-site.xml没有相关配置，都可以正常访问hive。

详见Hive文档

1. 添加配置

配置是提供给HiveServer

在集群的配置管理页面->Hive->配置->hive-site.xml->添加自定义配置

```
<property>
<name>hive.security.authorization.enabled</name>
<value>true</value>
</property>
<property>
<name>hive.users.in.admin.role</name>
<value>hive</value>
</property>
<property>
<name>hive.security.authorization.createtable.owner.grants</name>
<value>ALL</value>
</property>
```

2. 重启HiveServer2

在集群配置管理页面重启HiveServer2

3. 权限操作命令

具体命令操作详见

4. 验证

用户foo通过beeline访问test用户的testtbl表

```
2: jdbc:hive2://emr-header-1.cluster-xxx:10> select * from testtbl;
Error: Error while compiling statement: FAILED: HiveAccessControlException Permission denied: Principal
[name=foo, type=USER] does not have following privileges for operation QUERY [[SELECT] on Object
[type=TABLE_OR_VIEW, name=default.testtbl]] (state=42000,code=40000)
```

grant权限

```
切到test账号执行grant给foo授权select操作
hive> grant select on table testtbl to user foo;
OK
Time taken: 1.205 seconds
```

foo可以正常select

```
0: jdbc:hive2://emr-header-1.cluster-xxxxx:10> select * from testtbl;
INFO : OK
+------------+--+
| testtbl.a |
+------------+--+
| hz |
+------------+--+
1 row selected (0.787 seconds)
```

- 回收权限

```
 切换到test账号，回收权限foo的select权限
hive> revoke select from user foo;
OK
Time taken: 1.094 seconds
```

- foo无法select testtbl的数据

```
0: jdbc:hive2://emr-header-1.cluster-xxxxx:10> select * from testtbl;
Error: Error while compiling statement: FAILED: HiveAccessControlException Permission denied: Principal
[name=foo, type=USER] does not have following privileges for operation QUERY [[SELECT] on Object
[type=TABLE_OR_VIEW, name=default.testtbl]] (state=42000,code=40000)
```

HBase授权

HBase在不开启授权的情况下，任何账号对HBase集群可以进行任何操作，比如disable table/drop table/major compact等等。

备注:

对于没有Kerberos认证的集群，即使开启了HBase授权，用户也可以伪造身份访问集群服务。所以建议创建高安全模式(即支持Kerberos)的集群，详见Kerberos安全文档。

1. 添加配置

在HBase集群的配置管理->HBase->配置->hbase-site->自定义配置

添加如下几个参数:

```
<property>
<name>hbase.security.authorization</name>
<value>true</value>
</property>
<property>
<name>hbase.coprocessor.master.classes</name>
<value>org.apache.hadoop.hbase.security.access.AccessController</value>
</property>
<property>
<name>hbase.coprocessor.region.classes</name>
<value>org.apache.hadoop.hbase.security.token.TokenProvider,org.apache.hadoop.hbase.security.access.AccessController</value>
</property>
<property>
<name>hbase.coprocessor.regionserver.classes</name>
<value>org.apache.hadoop.hbase.security.access.AccessController</value>
</property>
```

2. 重启HBase集群

在HBase集群的配置管理->HBase->操作->RESTART All Components

3. 授权(ACL)

3.1 基本概念

授权就是将对 [某个范围的资源] 的 [操作权限] 授予[某个实体]

在HBase中，上述对应的三个概念分别为:

某个范围(Scope)的资源

Superuser

超级账号可以进行任何操作，运行HBase服务的账号默认是Superuser。也可以通过在hbase-site.xml中配置hbase.superuser的值可以添加超级账号

Global

Global Scope拥有集群所有table的Admin权限

Namespace

在Namespace Scope进行相关权限控制

Table

在Table Scope进行相关权限控制

ColumnFamily

在ColumnFamily Scope进行相关权限控制

Cell

在Cell Scope进行相关权限控制

操作权限

Read (R)

读取某个Scope资源的数据

Write (W)

写数据到某个Scope的资源

Execute (X)

在某个Scope执行协处理器

Create (C)

在某个Scope创建/删除表等操作

- Admin (A)

在某个Scope进行集群相关操作，如balance/assign等

某个实体

User

对某个用户授权

Group

对某个用户组授权

3.2 授权命令

grant 授权

```
grant <user> <permissions> [<@namespace> [<table> [<column family> [<column qualifier>]]]
```

备注:

user/group的授权方式一样，group需要加一个前缀@

```
 grant 'test','R','tbl1'   #给用户test授予表tbl1的读权限
grant '@testgrp','R','tbl1' #给用户组testgrp授予表tbl1的读权限
```

namespace需要加一个前缀@

```
grant 'test 'C','@ns_1'  #给用户test授予namespace ns_1的CREATE权限
```

revoke 回收

user_permissions 查看权限