**Spark on Kubernetes**

Spark能够在由Kubernetes管理的集群上运行，其中Spark已添加了调用Kubernetes本地计划表的特性（试验）。

**Security**

Spark的默认安全机制是关闭的，意味着容易遭受攻击

**User Identity**

项目提供的Dokerfile并不包含用户的目录，所以镜像会以root用户运行spark进程。在非安全的集群上，这可能会为权限提升和容器突破提供攻击媒介。所以部署时应该考虑提供带有声明了无特权UID和GID的用户目录。

Pod模板功能可用于将带有runAsUser的安全上下文添加到Spark提交的pod中。请记住，这需要您的用户进行合作，因此可能不适合共享环境。 如果管理员希望限制pod可能运行的用户，则应使用Pod安全策略。

**Volume Mounts**

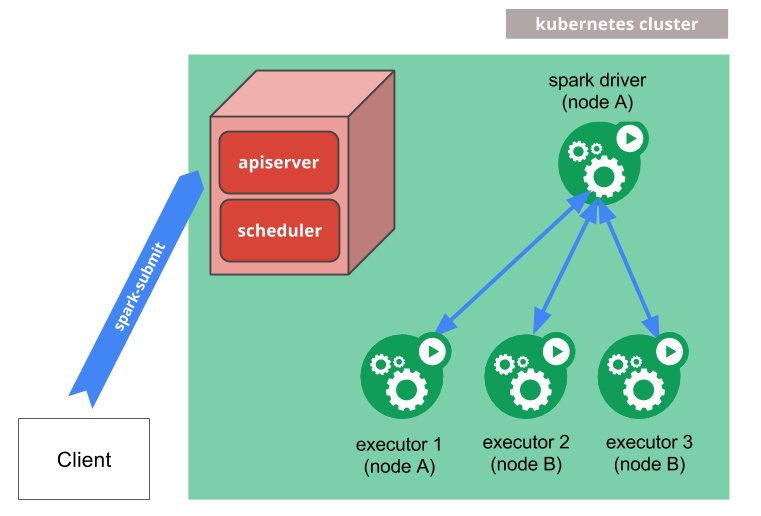
Spark on K8S提供了允许将某些卷类型安装到驱动程序和执行程序窗格中的配置选项。 特别是它允许使用Kubernetes文档中描述的hostPath卷具有已知的安全漏洞。

管理员应使用Pod安全策略来限制为其环境适当安装hostPath卷的功能。

**Prerequisite**

* Spark 2.3以上的可运行分区。
* 版本大于1.6的运行中的Kubernetes，可以通过kubectl访问配置，若没有工作的集群，可以在本地用minikube设置一个测试集群。
  + 推荐使用最新版的minikube，并激活了DNS插件。
  + 默认配置不足以运行Spark，使用3CPU和4g内存来启动一个基本的单执行Spark应用。
* 有足够的权限操作pods，可通过指令验证pods资源。
  + 被driver pod使用的服务账户证书需要允许创建pods，service和configmap。
* 必须配置Kubernetes DNS。

**How it Works**



Spark-submit能直接用于提交Spark应用，提交机制如下：

* Spark创建一个Spark Driver运行在pod上。
* Driver创建Executor，同样运行在pod上，且与之连接，并执行应用代码。
* 当应用完成时，Executor pod终止并清理，但driver的pod会保存日志并保持在完成状态，知道其最终被垃圾回收或手动清除。

在完成状态时，driver pod不会使用任何计算或内存资源。

Driver和Executor pod通过Kubernetes计划管理，可以通过节点选择器使用其配置属性在可用节点的子集上调度Driver和Executor pod。

**Submitting Applications to Kubernetes**

**Docker Images**

Kubernetes要求用户提供可部署到容器中的景象，该镜像构建在Kubernetes支持的容器运行时环境。Docker则与Kubernetes使用频繁。

Spark还附带了一个bin / docker-image-tool.sh脚本，可用于构建和发布Docker镜像以与Kubernetes后端一起使用。

**Cluster Mode**

运行Spark Pi样例

$ bin/spark-submit \

--master k8s://https://<k8s-apiserver-host>:<k8s-apiserver-port> \

--deploy-mode cluster \

--name spark-pi \

--class org.apache.spark.examples.SparkPi \

--conf spark.executor.instances=5 \

--conf spark.kubernetes.container.image=<spark-image> \

local:///path/to/examples.jar

Spark master，通过--master指令指定货设置spark.master在应用配置中，必须是k8s://<api\_server\_url>的URL形式。前缀是k8s使得Spark在Kubernetes集群上启动。若没有HTTP协议声明，默认是HTTPS。

在Kubernetes模式下，Spark应用的名称通过spark.app.name或--name指定，以此来命名Kubernetes中资源创建的driver和executor名称。所以，应用名称必须有小写，”-”，”.”组成，且必须小写字母开头。

若有Kubernetes 集群，可通过指令获取apiserver的URL

$ kubectl cluster-info

Kubernetes master is running at http://127.0.0.1:6443

也可以使用验证代理，代理会与Kubernetes API通信，本地代理启动方式：

$ kubectl proxy

若本地代理在8001运行，--master k8s://http://127.0.0.1:8001可以作为参数。

**Client Mode**

在2.4.0以后，可以在Kubernetes上以Client Mode运行Spark应用。当应用以Client mode运行时，driver能在pod内部运行，client mode推荐配置如下：

**Client Mode Networking**

Spark Executor必须能通过主机名和端口连接到Spark Driver，指定的网络配置必须是Spark Client Mode必要的，若在pod内运行driver，可使用独立服务让driver pod获得一个稳定的ip，当部署独立服务时，确保服务的标签选择器仅能匹配driver pod，通过spark.driver.host和spark.driver.port来指定。

**Client Mode Executor Pod Garbage Collection**

若将Spark Driver运行在pod上，推荐设置spark.kubernetes.driver.pod.name来设置pod名称。当其正确设置后，Spark计划表部署Executor pods时会带有其持有对象引用，来确保党driver pod被删除时，所有的应用Executor pods也会被删除。Driver会通过spark.kubernetes.namespace指定的命名空间来寻找pod，且持有引用会添加到每个Executor pod的持有引用列表中。注意要避免为pod设置未持有正确的driver pod的持有引用。

若应用不是在pod内运行，或spark.kubernetes.driver.pod.name没有设置时，Executor pods可能不会正确的删除，Spark计划表会尝试删除这些pods，但如果请求失败，则会继续保留在集群中。

**Authentication Parameters**

使用明确的前缀，spark.kubernetes.authenticate作为Kubernetes验证参数。

**Dependency Management**

如果你的应用依赖是远程地址比如HDFS或HTTP服务，可以通过合适的远程URI引用，同时，应用依赖可以提前安装到自定义的Docker镜像。这些依赖可以添加到类路径中，通过local://的URI和/或设置SPARK\_EXTRA\_CLASSPATH环境变量到你的Dockerfiles。在spark-submit中引用自定义构建的Docker镜像中的依赖项时，还需要local：//模式。使用本地文件系统目前是不支持的。

**Secret Management**

密钥用于提供证书，使Spark应用可以接入安全服务。安装指定用户的密钥到Driver容器，用户可以使用配置项spark.kubernetes.driver.secrets.[SecretName]=<mount path>。类似的，配置项spark.kubernetes.executor.secrets.[SecretName]=<mount path>可以安装指定用户的密钥到Executor容器。要注意的是，密钥是假设安装在与Driver Pod和Executor Pod同一个命名空间的。安装和使用的样例如下：

--conf spark.kubernetes.driver.secrets.spark-secret=/etc/secrets

--conf spark.kubernetes.executor.secrets.spark-secret=/etc/secrets

--conf spark.kubernetes.driver.secretKeyRef.ENV\_NAME=name:key

--conf spark.kubernetes.executor.secretKeyRef.ENV\_NAME=name:key

**Using Kubernetes Volumes**

从2.4.0开始，用户可以安装如下Kubernetes Volume类型到Driver和Executor Pods中：

hostPath：安装宿主节点文件系统中的文件或目录到pod中。

emptyDir：初始化的空卷，当pod分配给Node时创建。

persistentVolumeClaim：安装PersistentVolume到pod。

安装任意类型的Volume到Driver pod，使用如下配置项：

--conf spark.kubernetes.driver.volumes.[VolumeType].[VolumeName].mount.path=<mount path>

--conf spark.kubernetes.driver.volumes.[VolumeType].[VolumeName].mount.readOnly=<true|false>

其中，VolumeType是上述3个类型其中之一，VolumeName是你想要使用的Volume名称。

每一个支持的类型都有对应的选项，可以通过配置指定：

spark.kubernetes.driver.volumes.[VolumeType].[VolumeName].options.[OptionName]=<value>

要安装Volume到Executor pods使用配置前缀spark.kubernetes.executor，不同于spark.kubernetes.driver。

**Introspection and Debugging**

有多个方式来调查Spark应用，监督进程和执行任务。

**Accessing Logs**

日志通过Kubernetes API和kubectl CLI使用，当Spark应用在运行时，可以通过指令收集日志：

$ kubectl -n=<namespace> logs -f <driver-pod-name>

若安装在集群上，还可以通过Kubernetes dashboard查看。

**Accessing Driver UI**

可以使用kubectl port-forward在本地访问与任何应用程序关联的UI。

$ kubectl port-forward <driver-pod-name> 4040:4040

这可以在http://localhost:4040访问到Spark Driver UI。

**Debugging**

有多种类型的失败，若Kubernetes API服务拒绝了从spark-submit的请求，或连接被拒绝，提交的逻辑应指出遭遇的错误，然而，如果运行期间遇到了错误，通常最好的方式是通过Kubernetes CLI。

获取Driver pod有关的计划表决策基本信息，可以执行：

$ kubectl describe pod <spark-driver-pod>

若pod遭遇运行时错误，可以在后续查看：

$ kubectl logs <spark-driver-pod>

状态和Executor pod的失败日志可以通过相似方式查看。最后，删除driver pod会清除整个Spark应用，包括所有的Executor，相关联的Service等等。Driver pod可认为是Spark应用在Kubernetes中的代表。

Configuration

专用于Spark on Kubernetes。

|  |  |  |
| --- | --- | --- |
| **Property Name** | **Default** | **Meaning** |
| spark.kubernetes.namespace | default | 运行Driver和Executor pod的命名空间 |
| spark.kubernetes.container.image | (none) | Spark应用使用的容器镜像，示例：example.com/repo/spark:v1.0.0  必要且必须由客户提供，除非是每个容器类型都提供有明确的镜像 |
| spark.kubernetes.driver.container.image | (value of spark.kubernetes.container.image) | Driver使用的自定义容器镜像 |
| spark.kubernetes.executor.container.image | (value of spark.kubernetes.container.image) | Executor使用的自定义容器镜像。 |
| spark.kubernetes.container.image.pullPolicy | IfNotPresent | 在Kubernetes中提取图像时使用的容器图像拉取策略。 |
| spark.kubernetes.container.image.pullSecrets |  | 逗号分隔的Kubernetes秘密列表，用于从私有图像注册表中提取图像。 |
| spark.kubernetes.allocation.batch.size | 5 | 每轮执行器pod分配中一次启动的pod数。 |
| spark.kubernetes.allocation.batch.delay | 1s | 在每轮执行器pod分配之间等待的时间。 指定小于1秒的值可能会导致火花驱动程序的CPU使用率过高。 |
| spark.kubernetes.authenticate.submission.caCertFile | (none) | 启动驱动程序时通过TLS连接到Kubernetes API服务器的CA证书文件的路径。 此文件必须位于提交计算机的磁盘上。 将其指定为路径而不是URI（即不提供方案）。 在客户端模式下，请改用spark.kubernetes.authenticate.caCertFile。 |
| spark.kubernetes.authenticate.submission.clientKeyFile | (none) | 客户端密钥文件的路径，用于在启动驱动程序时对Kubernetes API服务器进行身份验证。 此文件必须位于提交计算机的磁盘上。 将其指定为路径而不是URI（即不提供方案）。 在客户端模式下，请改用spark.kubernetes.authenticate.clientKeyFile。 |
| spark.kubernetes.authenticate.submission.clientCertFile | (none) | 客户端证书文件的路径，用于在启动驱动程序时对Kubernetes API服务器进行身份验证。 此文件必须位于提交计算机的磁盘上。 将其指定为路径而不是URI（即不提供方案）。 在客户端模式下，请改用spark.kubernetes.authenticate.clientCertFile。 |
| spark.kubernetes.authenticate.submission.oauthToken | (none) | 启动驱动程序时对Kubernetes API服务器进行身份验证时使用的OAuth令牌。 请注意，与其他身份验证选项不同，这应该是用于身份验证的令牌的确切字符串值。 在客户端模式下，请改用spark.kubernetes.authenticate.oauthToken。 |
| spark.kubernetes.authenticate.submission.oauthTokenFile | (none) | OAuth令牌文件的路径，包含在启动驱动程序时对Kubernetes API服务器进行身份验证时要使用的令牌。 此文件必须位于提交计算机的磁盘上。 将其指定为路径而不是URI（即不提供方案）。 在客户端模式下，请改用spark.kubernetes.authenticate.oauthTokenFile。 |
| spark.kubernetes.authenticate.driver.caCertFile | (none) | CA cert文件的路径，用于在请求执行程序时通过TLS从驱动程序窗口连接到Kubernetes API服务器。 此文件必须位于提交机器的磁盘上，并将上载到驱动程序窗格。 将其指定为路径而不是URI（即不提供方案）。 在客户端模式下，改为使用park.kubernetes.authenticate.caCertFile。 |
| spark.kubernetes.authenticate.driver.clientKeyFile | (none) | 客户端密钥文件的路径，用于在请求执行程序时从驱动程序窗格中对Kubernetes API服务器进行身份验证。 此文件必须位于提交计算机的磁盘上，并将作为Kubernetes密钥上载到驱动程序窗格。 将其指定为路径而不是URI（即不提供方案）。 在客户端模式下，请改用spark.kubernetes.authenticate.clientKeyFile。 |
| spark.kubernetes.authenticate.driver.clientCertFile | (none) | 客户端证书文件的路径，用于在请求执行程序时从驱动程序窗格中对Kubernetes API服务器进行身份验证。 此文件必须位于提交计算机的磁盘上，并将作为Kubernetes密钥上载到驱动程序窗格。 将其指定为路径而不是URI（即不提供方案）。 在客户端模式下，请改用spark.kubernetes.authenticate.clientCertFile。 |
| spark.kubernetes.authenticate.driver.oauthToken | (none) | 在请求执行程序时，从驱动程序窗格中对Kubernetes API服务器进行身份验证时使用的OAuth令牌。 请注意，与其他身份验证选项不同，这必须是用于身份验证的令牌的确切字符串值。 此标记值作为Kubernetes密钥上载到驱动程序窗格。 在客户端模式下，请改用spark.kubernetes.authenticate.oauthToken。 |
| spark.kubernetes.authenticate.driver.oauthTokenFile | (none) | OAuth令牌文件的路径，其中包含在请求执行程序时从驱动程序窗格中对Kubernetes API服务器进行身份验证时要使用的令牌。 请注意，与其他身份验证选项不同，此文件必须包含用于身份验证的令牌的确切字符串值。 此令牌值作为机密上传到驱动程序pod。 在客户端模式下，请改用spark.kubernetes.authenticate.oauthTokenFile。 |
| spark.kubernetes.authenticate.driver.mounted.caCertFile | (none) | CA cert文件的路径，用于在请求执行程序时通过TLS从驱动程序窗口连接到Kubernetes API服务器。 必须可以从驱动程序窗格访问此路径。 将其指定为路径而不是URI（即不提供方案）。 在客户端模式下，使用spark.kubernetes.authenticate.caCertFileinstead。 |
| spark.kubernetes.authenticate.driver.mounted.clientKeyFile | (none) | 客户端密钥文件的路径，用于在请求执行程序时从驱动程序窗格中对Kubernetes API服务器进行身份验证。 必须可以从驱动程序窗格访问此路径。 将其指定为路径而不是URI（即不提供方案）。 在客户端模式下，使用spark.kubernetes.authenticate.clientKeyFileinstead。 |
| spark.kubernetes.authenticate.driver.mounted.clientCertFile | (none) | 客户端证书文件的路径，用于在请求执行程序时从驱动程序窗格中对Kubernetes API服务器进行身份验证。 必须可以从驱动程序窗格访问此路径。 将其指定为路径而不是URI（即不提供方案）。 在客户端模式下，使用spark.kubernetes.authenticate.clientCertFileinstead。 |
| spark.kubernetes.authenticate.driver.mounted.oauthTokenFile | (none) | 在请求执行程序时，从驱动程序窗格中对Kubernetes API服务器进行身份验证时，包含要使用的OAuth令牌的文件的路径。 必须可以从驱动程序窗格访问此路径。 请注意，与其他身份验证选项不同，此文件必须包含用于身份验证的令牌的确切字符串值。 在客户端模式下，请改用spark.kubernetes.authenticate.oauthTokenFile。 |
| spark.kubernetes.authenticate.driver.serviceAccountName | default | 运行驱动程序窗格时使用的服务帐户。 从API服务器请求执行程序窗格时，驱动程序窗格使用此服务帐户。 请注意，这不能与CA证书文件，客户端密钥文件，客户端证书文件和/或OAuth令牌一起指定。 在客户端模式下，请改用spark.kubernetes.authenticate.serviceAccountName。 |
| spark.kubernetes.authenticate.caCertFile | (none) | 在客户端模式下，CA cert文件的路径，用于在请求执行程序时通过TLS连接到Kubernetes API服务器。 将其指定为路径而不是URI（即不提供方案）。 |
| spark.kubernetes.authenticate.clientKeyFile | (none) | 在客户端模式下，客户端密钥文件的路径，用于在请求执行程序时针对Kubernetes API服务器进行身份验证。 将其指定为路径而不是URI（即不提供方案）。 |
| spark.kubernetes.authenticate.clientCertFile | (none) | 在客户端模式下，客户端证书文件的路径，用于在请求执行程序时针对Kubernetes API服务器进行身份验证。 将其指定为路径而不是URI（即不提供方案）。 |
| spark.kubernetes.authenticate.oauthToken | (none) | In client mode, the OAuth token to use when authenticating against the Kubernetes API server when requesting executors. Note that unlike the other authentication options, this must be the exact string value of the token to use for the authentication. |
| spark.kubernetes.authenticate.oauthTokenFile | (none) | 在客户端模式下，在请求执行程序时对Kubernetes API服务器进行身份验证时使用的OAuth令牌。 请注意，与其他身份验证选项不同，这必须是用于身份验证的令牌的确切字符串值。 |
| spark.kubernetes.driver.label.[LabelName] | (none) | 将LabelName指定的标签添加到驱动程序窗格。 例如，spark.kubernetes.driver.label.something = true。 请注意，Spark还会将自己的标签添加到驱动程序窗格以用于簿记目的。 |
| spark.kubernetes.driver.annotation.[AnnotationName] | (none) | 将AnnotationName指定的注释添加到驱动程序窗格。 例如，spark.kubernetes.driver.annotation.something = true。 |
| spark.kubernetes.executor.label.[LabelName] | (none) | 将LabelName指定的标签添加到执行程序窗格。 例如，spark.kubernetes.executor.label.something = true。 请注意，Spark还会将自己的标签添加到驱动程序窗格以用于簿记目的。 |
| spark.kubernetes.executor.annotation.[AnnotationName] | (none) | 将AnnotationName指定的注释添加到执行程序窗格。 例如，spark.kubernetes.executor.annotation.something = true。 |
| spark.kubernetes.driver.pod.name | (none) | 驱动程序窗格的名称。 在群集模式下，如果未设置此选项，则驱动程序窗格名称将设置为“spark.app.name”，并以当前时间戳为后缀，以避免名称冲突。 在客户端模式下，如果您的应用程序在pod中运行，强烈建议将其设置为运行驱动程序的pod的名称。在客户端模式下设置此值允许驱动程序成为其执行程序pod的所有者， 这反过来又允许执行器pod被集群垃圾收集。 |
| spark.kubernetes.executor.lostCheck.maxAttempts | 10 | 驱动程序尝试确定特定执行程序的丢失原因的次数。 丢失原因用于确定执行程序失败是由于框架还是应用程序错误，而错误又决定执行程序是被删除和替换，还是处于失败状态以进行调试。 |
| spark.kubernetes.submission.waitAppCompletion | true | 在群集模式下，是否在退出启动程序进程之前等待应用程序完成。 当更改为false时，启动Spark作业时启动器会出现“即发即忘”行为。 |
| spark.kubernetes.report.interval | 1s | 群集模式下当前Spark作业状态报告之间的间隔。 |
| spark.kubernetes.driver.limit.cores | (none) | 为驱动程序窗格指定硬CPU限制。 |
| spark.kubernetes.executor.request.cores | (none) | 为每个执行程序窗格指定cpu请求。 值符合Kubernetes惯例。 示例值包括0.1,500m，1.5,5等，其中CPU单元中记录了cpu单元的定义。 这与spark.executor.cores不同：它仅用于并优先于spark.executor.cores，用于指定执行程序pod cpu请求（如果已设置）。 任务并行性，例如，执行器可以并发运行的任务数量不受此影响。 |
| spark.kubernetes.executor.limit.cores | (none) | 为Spark应用程序启动的每个执行程序窗体指定硬CPU限制。 |
| spark.kubernetes.node.selector.[labelKey] | (none) | 添加到驱动程序窗格和执行程序窗格的节点选择器，键标签为Key，值为配置值。 例如，将spark.kubernetes.node.selector.identifier设置为myIdentifier将导致驱动程序窗格和执行程序具有带有密钥标识符和valuemyIdentifier的节点选择器。 通过使用此前缀设置多个配置，可以添加多个节点选择器键。 |
| spark.kubernetes.driverEnv.[EnvironmentVariableName] | (none) | 将EnvironmentVariableName指定的环境变量添加到Driver进程。 用户可以指定其中的多个来设置多个环境变量。 |
| spark.kubernetes.driver.secrets.[SecretName] | (none) | 将名为SecretName的Kubernetes Secret添加到值中指定的路径上的驱动程序窗格中。 例如，spark.kubernetes.driver.secrets.spark-secret = / etc / secrets。 |
| spark.kubernetes.executor.secrets.[SecretName] | (none) | 将名为SecretName的Kubernetes Secret添加到值中指定的路径上的executor pod中。 例如，spark.kubernetes.executor.secrets.spark-secret = / etc / secrets。 |
| spark.kubernetes.driver.secretKeyRef.[EnvName] | (none) | 将环境变量添加到名为EnvName（区分大小写）的驱动程序容器中，该值由引用的Kubernetes Secret的数据中的键键引用。 例如，spark.kubernetes.driver.secretKeyRef.ENV\_VAR = spark-secret：key。 |
| spark.kubernetes.executor.secretKeyRef.[EnvName] | (none) | 将环境变量添加到名为EnvName（区分大小写）的执行程序容器中，该值由引用的Kubernetes Secret的数据中的键键引用。 例如，spark.kubernetes.executor.secrets.ENV\_VAR = spark-secret：key。 |
| spark.kubernetes.driver.volumes.[VolumeType].[VolumeName].mount.path | (none) | 将名为VolumeType类型的Kubernetes卷添加到值中指定的路径上的驱动程序窗格中。 例如，spark.kubernetes.driver.volumes.persistentVolumeClaim.checkpointpvc.mount.path = / checkpoint。 |
| spark.kubernetes.driver.volumes.[VolumeType].[VolumeName].mount.readOnly | (none) | 指定装入的卷是否为只读。 例如，spark.kubernetes.driver.volumes.persistentVolumeClaim.checkpointpvc.mount.readOnly = false。 |
| spark.kubernetes.driver.volumes.[VolumeType].[VolumeName].options.[OptionName] | (none) | 配置Kubernetes使用OptionName作为具有指定值的键传递给Kubernetes的卷选项必须符合Kubernetes选项格式。 例如，spark.kubernetes.driver.volumes.persistentVolumeClaim.checkpointpvc.options.claimName = spark-pvc-claim。 |
| spark.kubernetes.executor.volumes.[VolumeType].[VolumeName].mount.path | (none) | 将名为VolumeType类型的Kubernetes卷添加到值中指定的路径上的executor pod中。 例如，spark.kubernetes.executor.volumes.persistentVolumeClaim.checkpointpvc.mount.path = / checkpoint。 |
| spark.kubernetes.executor.volumes.[VolumeType].[VolumeName].mount.readOnly | false | 指定装入的卷是否为只读。 例如，spark.kubernetes.executor.volumes.persistentVolumeClaim.checkpointpvc.mount.readOnly = false。 |
| spark.kubernetes.executor.volumes.[VolumeType].[VolumeName].options.[OptionName] | (none) | 配置传递给Kubernetes的Kubernetes Volume选项，其中OptionName为具有指定值的键。 例如，spark.kubernetes.executor.volumes.persistentVolumeClaim.checkpointpvc.options.claimName = spark-pvc-claim。 |
| spark.kubernetes.memoryOverheadFactor | 0.1 | 这将设置内存开销因子，该内存开销因子将为非JVM内存分配内存，其中包括堆外内存分配，非JVM任务和各种系统进程。 对于基于JVM的作业，对于非JVM作业，此值将默认为0.10和0.40。 这是因为非JVM任务需要更多的非JVM堆空间，并且此类任务通常会因“超出内存开销”错误而失败。 这会以更高的默认值预先设置此错误。 |
| spark.kubernetes.pyspark.pythonVersion | "2" | 这将设置用于运行驱动程序和执行程序容器的docker镜像的主要Python版本。 可以是2或3。 |