今天我讲的主题是任务监控和任务调度，总体上涉及四个部分，先是在Yarn中定位任务监控和任务调度的位置，再介绍如何进行任务监控，主要是如何通过Web UI来进行任务监控，在后面是介绍什么是任务调度和常见的任务调度调度器。首先是定位，我们知道Yarn提供了通用的集群资源管理和任务监控功能，回顾Yarn的组成，其中一个是全局的资源管理器ResourceManager，包含了Yarn Scheduler与ApplicationMaster Service，对应了任务调度与任务监控，Scheduler会对每个作业队列进行资源的分配，ApplicationMaster Service将Application Master的资源请求传递给Scheduler。这个Application Master是每个作业一个的，它负责向ResourceManager申请资源，与NodeManager交互进行程序的运行和监控，监控申请的资源的使用情况，跟踪作业状态和进度，定时向ResourceManager发送心跳消息，报告资源的使用情况和应用的进度信息。这就是今天讲的两个部分的有关负责组件。

第二部分是WebUI的介绍，YARN提供了一个WebUI的内置服务，随着ResourceManager的启动而启动。通过浏览器输入ResourceManager对应host访问，默认情况下是8088端口就可以访问Yarn WebUI服务，也可以更改这条对应属性，重新指定端口。

登录到WebUI主界面后可以看到，在WebUI服务支持下，通过浏览器，可以监视集群状态，包括队列，应用程序，服务，节点信息，集群配置信息，应用程序与服务的日志。

这里我们只关注Cluster菜单下方两个功能，Applications和Scheduler，默认主界面就是这个Application选项，会显示所有的应用与相关的信息，像在这的应用ID，起始时间，状态，分配的CPU，也可以在左边cluster筛选，查看像新提交的，完成的，强制退出的，处于不同状态的应用。

Scheduler是任务调度相关的部分，一会再看。

再提交一个任务之后，可以看到主界面的应用数量进行了更新，提交的任务状态也会被实时监控着，当状态改变时会自动更新页面。如果要对某一个具体的任务进行更详细的监控，可以点击相应的ID查看，而且可以在底部日志里排查任务失败的报错原因。

效果大概是这样。

Yarn的WebUI还可以提供JobHistory和Timeline插件，用于查看过去的任务记录和时间轴，这里就不再展开了。

第三部分是任务调度。

理想情况下，客户端程序一旦提出资源申请，就应该得到响应，获取到请求的资源；但实际情况下，如果提交的任务数量较多，资源可能无法马上进行分配。有时是资源不够，有时是任务本身有着不同优先级，这时就需要进行等待资源分配，而在等待过程中yarn按一定规则对各个任务进行资源的分配，这种资源的分配就是job scheduling，负责这种调度的就是Scheduler，可以在这的Scheduler Type查看调度器类型。

调度器里面的核心部分就是这里工作队列Job Queue，里面存放从不同客户端收到的各种任务的集合，Scheduler就根据相应策略去分配资源给队列中的任务。默认情况下Yarn只有一个用于提交任务的default队列，用户可以自行配置队列树，指定应用使用的队列，但是根队列root是不能运行任务的。

现在简单介绍常见的三种任务调度器

第一种先进先出，先提交的应用先运行，优势是不需要额外配置，只需要将scheduler改为FIFO就可以使用，逻辑也非常简单，她也是上一代Hadoop里JobTracker的调度器实现，坏处在于这个调度器不考虑优先级问题，所以就算有高优先级的作业也需要等待前面低优先级任务，而且可能导致护航效应，前面一个长时间的计算任务可以把后续所有任务都阻塞住。还只有一个全局的队列default，会获取当前集群的所有资源作用于这个全局队列，所以他只适合负载较低的小规模集群，不适合大规模，需要共享的集群。

第二种容量调度，它是现在Apache Hadoop 3.x的默认调度策略，可以允许多个组织共享整个集群资源，通过每个组织分配专门的队列，再为每个队列分配一定的资源，就可以实现层次化的管理，实现资源共享。用户可以自行分配一个个资源队列，并且队列内部还可以继续划分队列，使得同一个队列的多个成员能够共享队列资源，且每个队列可设定一定比例的资源最低保证和使用上限，在同一队列内部采用FIFO策略。此外，每个队列有严格的访问控制，用户只能向自己的队列里提交任务，不能修改也不能访问其他队列的任务。但调度器本身具有一定的灵活性，空闲的资源可以被分配给其他任何队列，如果多个队列竞争，会按照权重比例分配。Yarn也支持动态修改队列容量，权限分配，运行时直接修改。也允许基于用户或者组将一个映射到特定队列。

最后一种是公平调度，他是一种基于队列层面的公平策略，也就是队列内的作业均分资源，默认公平定义在内存资源上，可以通过参数设置，改变公平定义的资源类型。公平调度在多个队列间工作，运行资源可以共享也允许抢占。可以设置队列最小配额，当不能满足时，可以从其他队列抢占。如果其他队列没有任务进行，也可以使用其他队列，当其他队列有应用时再释放。它也支持基于用户或组的队列映射。

以上就是介绍的全部内容。