分布式资源管理系统YARN

范 颖 捷 | 2018年7月

TRANSWARP
星 环 科 技

TRANSWARP
星 环 科 技

目录》
CONTENTS

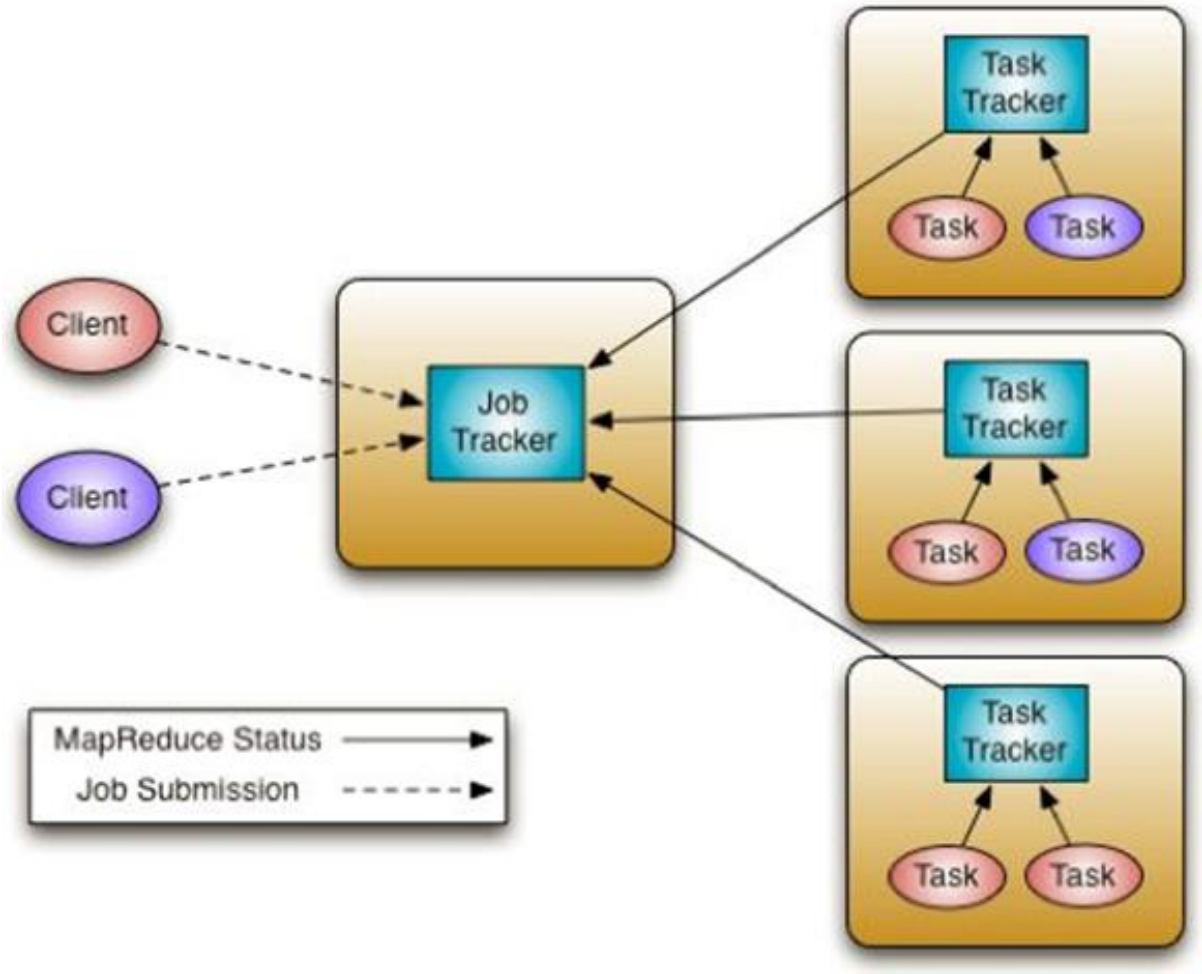
1
chapter

YARN简介

- ✓ MapReduce缺陷
- ✓ 设计目标

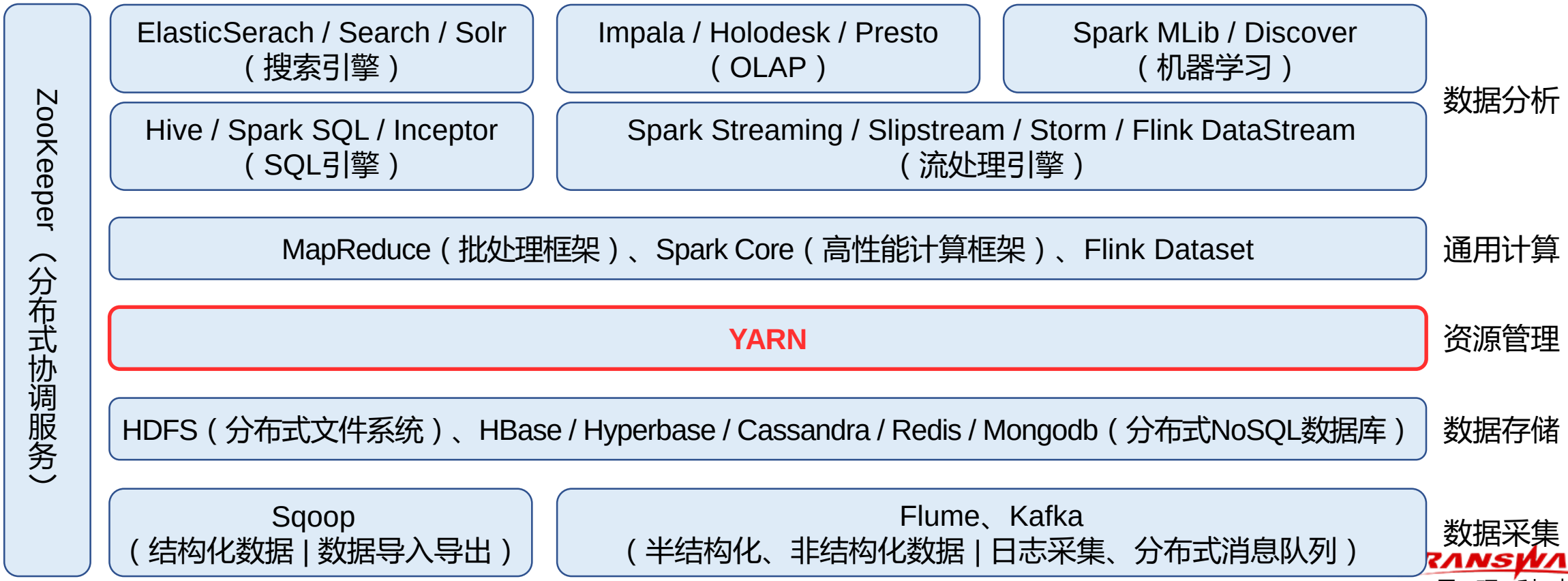
MapReduce存在先天缺陷（Hadoop 1.X）

- 身兼两职：计算框架 + 资源管理框架
- JobTracker
  - 既做资源管理，又做任务调度
  - 任务太重，开销过大
  - 存在单点故障
- 资源描述模型过于简单，资源利用率较低
  - 仅把Task数量看作资源，没有考虑CPU和内存
  - 强制把资源分成Map Task Slot和Reduce Task Slot
- 扩展性较差，集群规模上限4K
- 源码难于理解，升级维护困难



1.2 设计目标

- YARN，Yet Another Resource Negotiator，另一种资源管理器
- 分布式通用资源管理系统
- 设计目标：聚焦资源管理、通用（适用各种计算框架）、高可用、高扩展



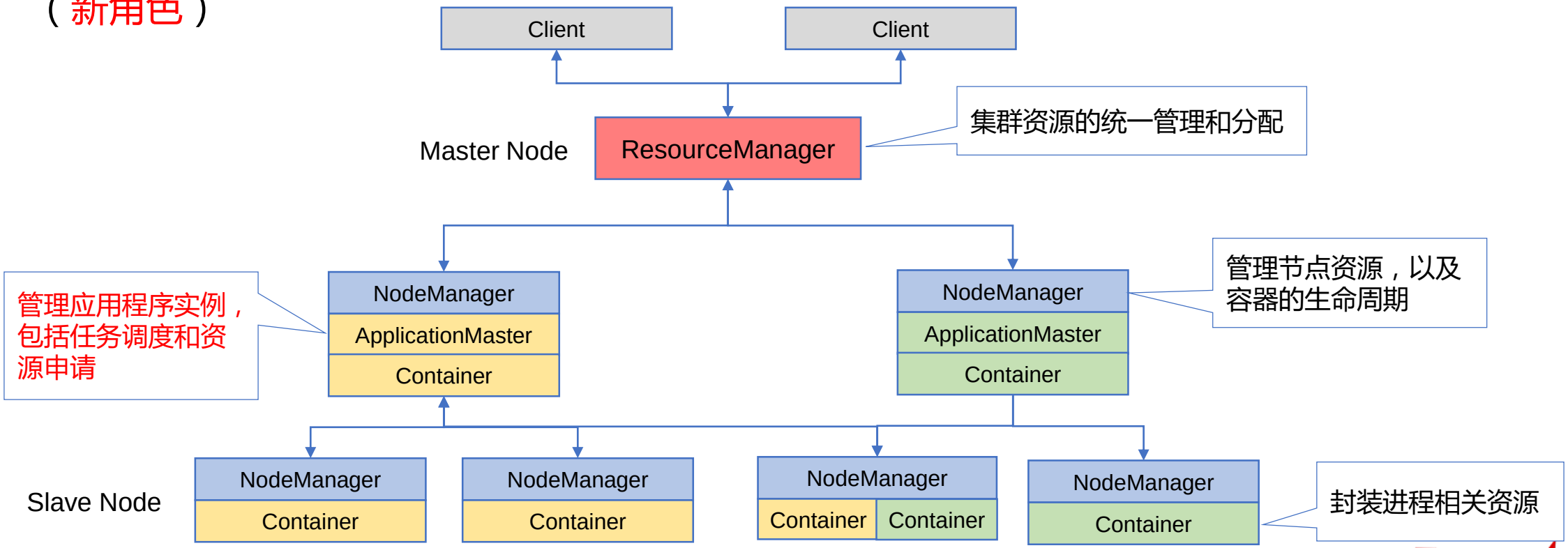
2
chapter

YARN原理

- ✓ 系统架构
- ✓ 高可用

2.1 系统架构：基本思想

- Master/Slave架构
- 将JobTracker的资源管理、任务调度功能分离
- 三种角色：ResourceManager（Master）、NodeManager（Slave）、ApplicationMaster（新角色）

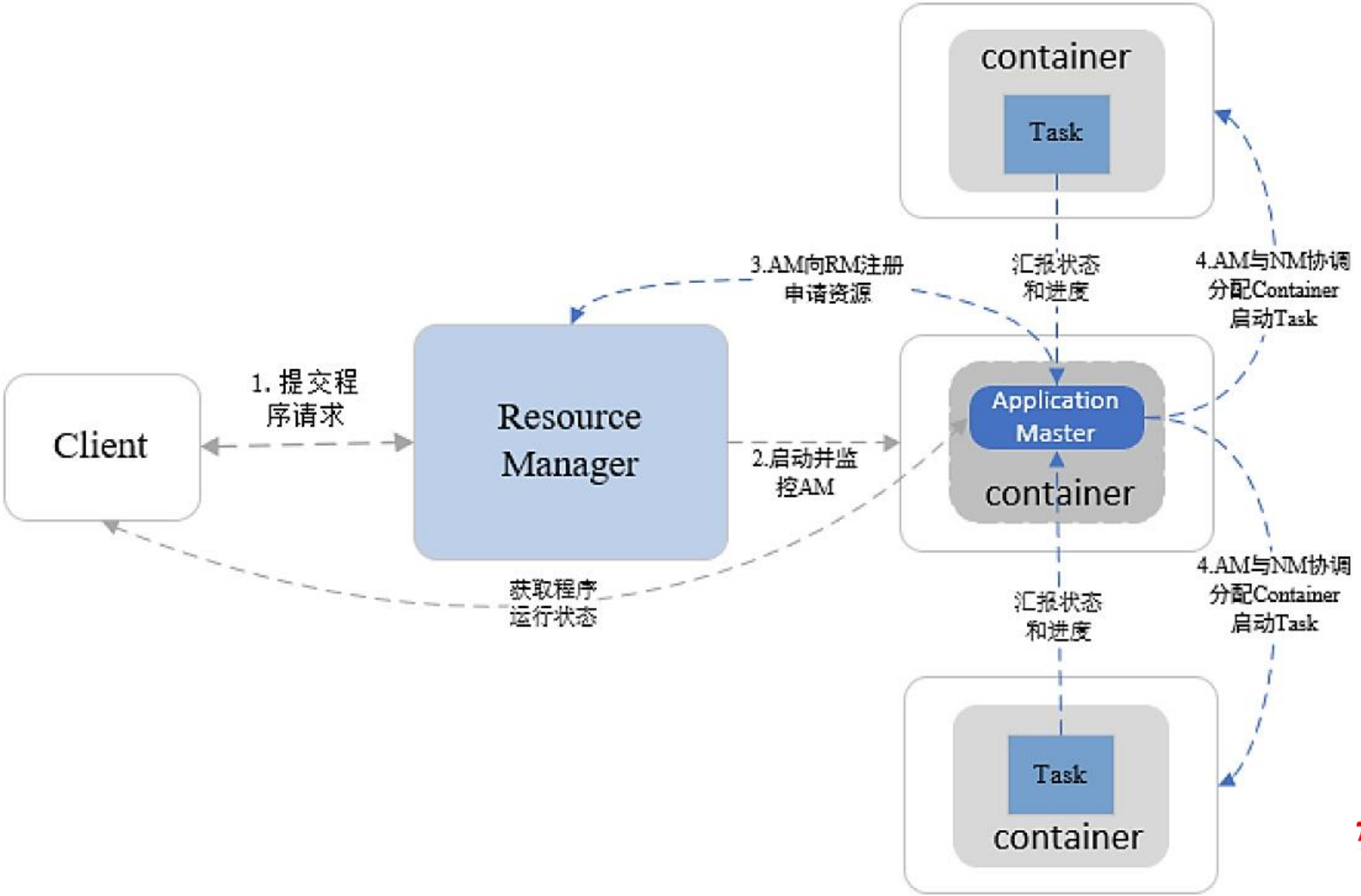


- ResourceManager（RM）
  - 主要功能
    - 统一管理集群的所有资源
    - 将资源按照一定策略分配给各个应用（ApplicationMaster）
    - 接收NodeManager的资源上报信息
  - 核心组件
    - 用户交互服务（User Service）
    - NodeManager管理
    - ApplicationMaster管理
    - Application管理
    - 安全管理
    - 资源管理

- NodeManager（NM）
  - 主要功能
    - 管理单个节点的资源
    - 向ResourceManager汇报节点资源使用情况
    - 管理Container的生命周期
  - 核心组件
    - NodeStatusUpdater
    - ContainerManager
    - ContainerExecutor
    - NodeHealthCheckerService
    - Security
    - WebServer

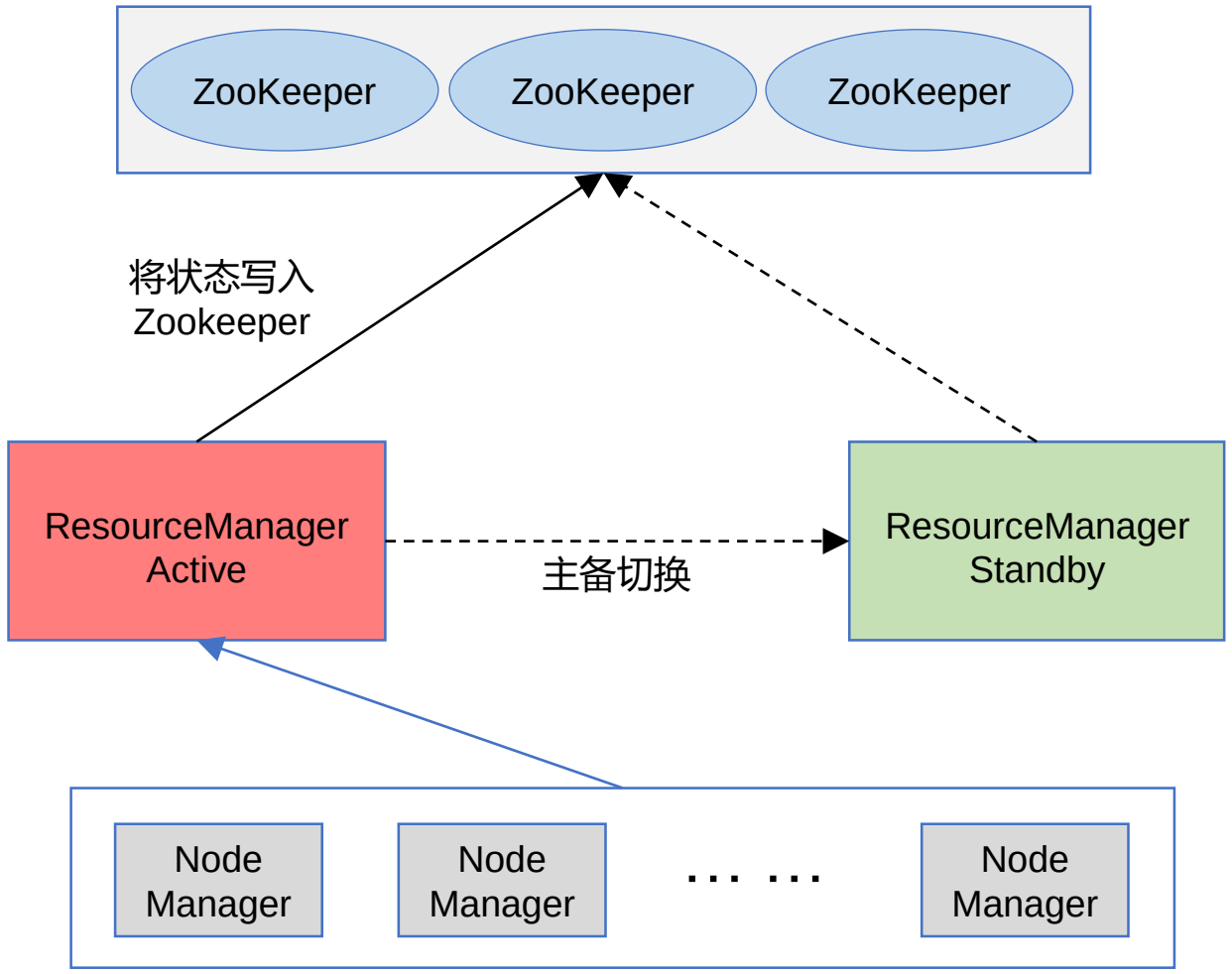
- ApplicationMaster（AM）
  - 主要功能
    - 管理应用程序实例
    - 向ResourceManager申请任务执行所需的资源
    - 任务调度和监管
  - 实现方式
    - 需要为每个应用开发一个AM组件
    - YARN提供MapReduce的ApplicationMaster实现
    - 采用基于事件驱动的异步编程模型，由中央事件调度器统一管理所有事件
    - 每种组件都是一种事件处理器，在中央事件调度器中注册
- Container
  - 概念：Container封装了节点上进程的相关资源，是YARN中资源的抽象
  - 分类：运行ApplicationMaster的Container 、运行应用任务的Container

2.1 系统架构：工作机制



ResourceManager高可用

- 1个Active RM、多个Standby RM
- 宕机后自动实现主备切换
- ZooKeeper的核心作用
  - Active节点选举
  - 恢复Active RM的原有状态信息
- 重启AM，杀死所有运行中的Container
- 切换方式：手动、自动



- RM管理命令

```
# yarn  rmadmin  [command_options]
```

Command Options	Description
-refreshQueues	Reload the queues' acls, states and scheduler specific properties.
-refreshNodes	Refresh the hosts information at the ResourceManager.
-failover [–forceactive] <serviceId1> <serviceId2>	Initiate a failover from serviceId1 to serviceId2.
-getServiceState <serviceId>	Returns the state of the service.

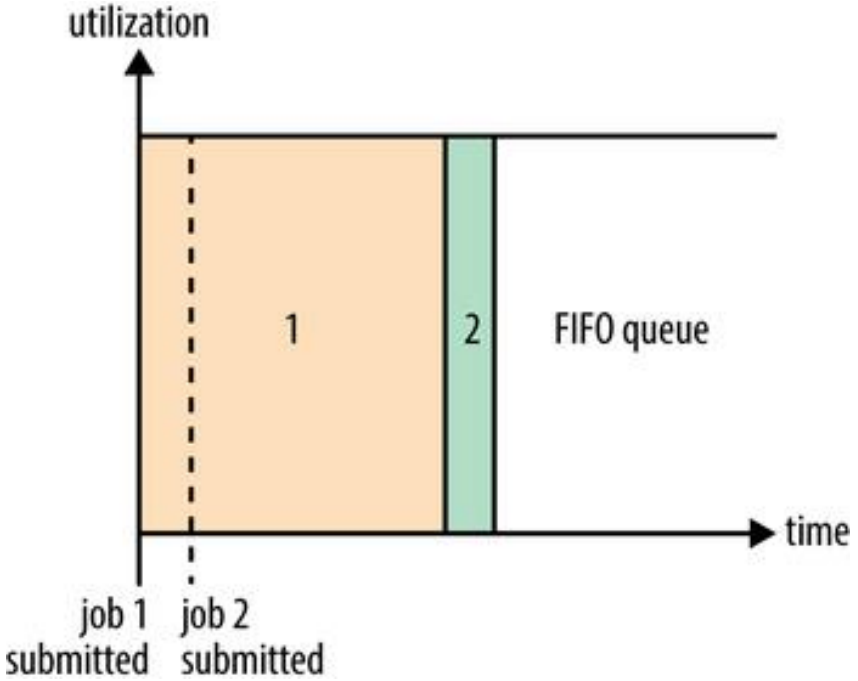
3
chapter

YARN资源调度策略

- ✓ FIFO调度器
- ✓ 容量调度器
- ✓ 公平调度器

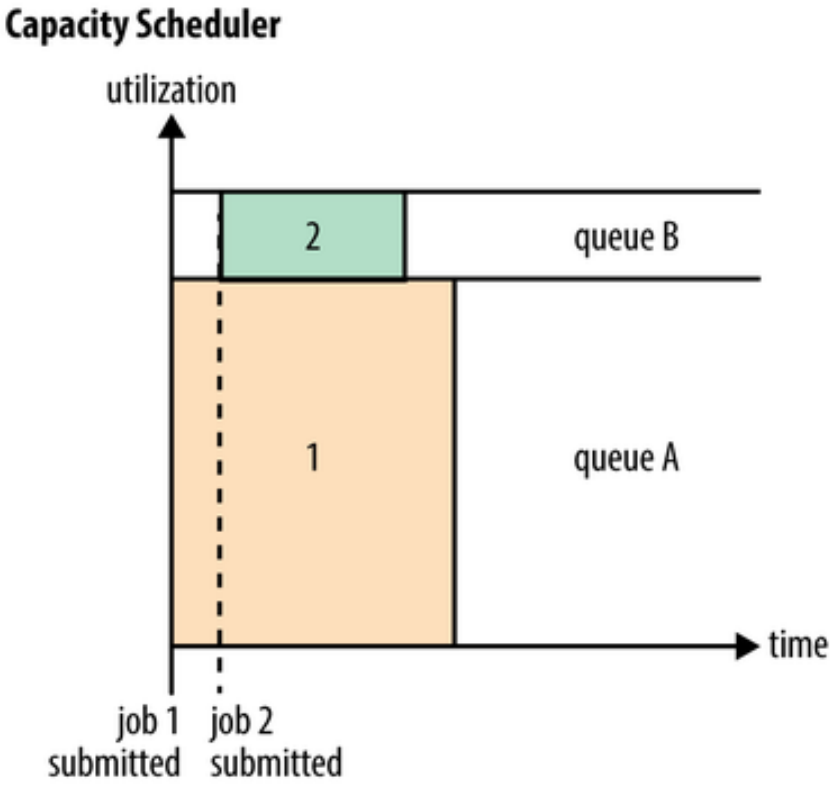
➢ FIFO Scheduler（先进先出调度器）

- 调度策略
  - 将所有任务放入一个队列，先进队列的先获得资源，排在后面的任务只有等待
- 缺点
  - 资源利用率低，无法交叉运行任务
  - 灵活性差，如：紧急任务无法插队，耗时长的任务拖慢耗时短的任务



➢ Capacity Scheduler（容量调度器）

- 核心思想：提前做预算，在预算指导下分享集群资源
- 调度策略
  - 集群资源由多个队列分享
  - 每个队列都要预设资源分配的比例（提前做预算）
  - 空闲资源优先分配给“实际资源/预算资源”比值最低的队列
  - 队列内部采用FIFO调度策略
- 特点
  - 层次化的队列设计：子队列可使用父队列资源
  - 容量保证：每个队列都要预设资源占比，防止资源独占
  - 弹性分配：空闲资源可以分配给任何队列，当多个队列争用时，会按比例进行平衡
  - 支持动态管理：可以动态调整队列的容量、权限等参数，也可动态增加、暂停队列
  - 访问控制：用户只能向自己的队列中提交任务，不能访问其他队列
  - 多租户：多用户共享集群资源



- 全局配置（yarn-site.xml）

Property	Value
yarn.resourcemanager.scheduler.class	org.apache.hadoop.yarn.server.resourcemanager.scheduler.capacity.CapacityScheduler

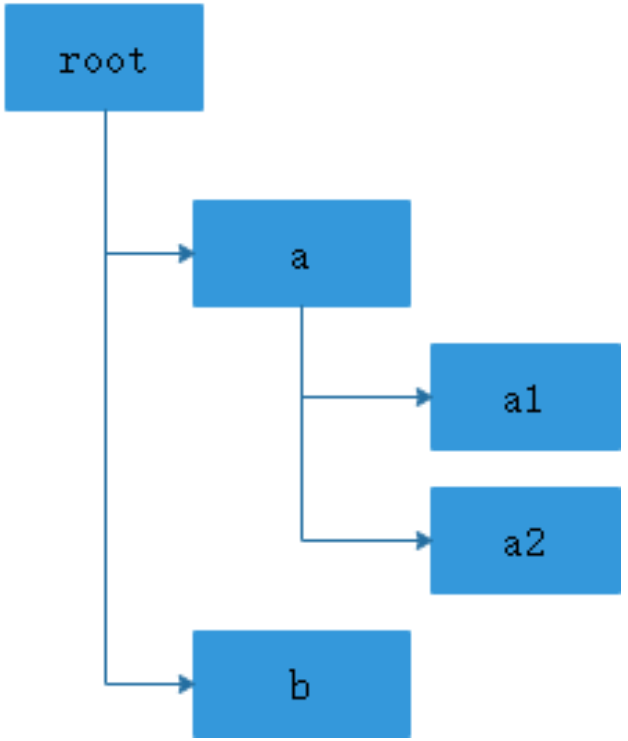
- 自定义配置（capacity-scheduler.xml）

Property	Value
yarn.scheduler.capacity.<queue-path>.capacity	Queue capacity in percentage (%) as a float (e.g. 12.5). The sum of capacities for all queues, at each level, must be equal to 100.
yarn.scheduler.capacity.<queue-path>.maximum-capacity	Maximum queue capacity in percentage (%) as a float. Defaults to -1 which disables it.
yarn.scheduler.capacity.<queue-path>.minimum-user-limit-percent	The user limit can vary between a minimum and maximum value. The the former (the minimum value) is set to this property value and the latter (the maximum value) depends on the number of users who have submitted applications. A value of 100 implies no user limits are imposed.

- 队列设计（capacity-scheduler.xml）

```
<property>
  <name>yarn.scheduler.capacity.root.queues</name>
  <value>a,b</value>
  <description>The queues at the this level (root is the root queue).</description>
</property>

<property>
  <name>yarn.scheduler.capacity.root.a.queues</name>
  <value>a1,a2</value>
  <description>The queues at the this level (root is the root queue).</description>
</property>
```



根队列为root，其他队列必须定义为root的子队列，队列的继承关系以“.”号分隔

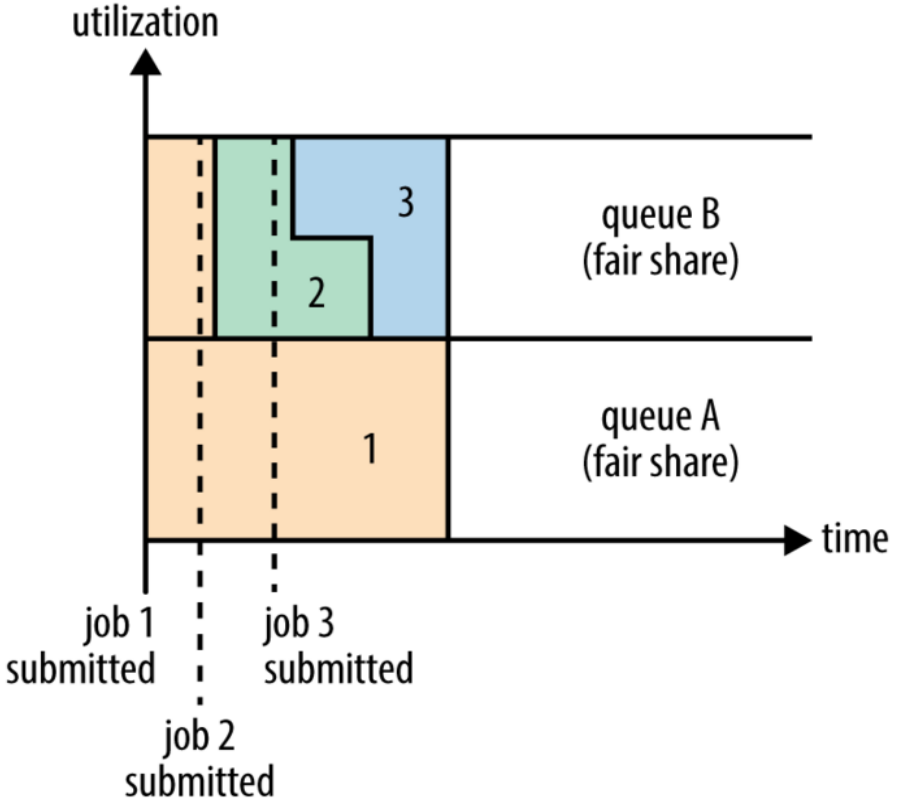
- 全局配置和队列设计（Web）

- 自定义配置（Web）


Queue配置
Queue 名称:
test
最大容量:
-1
用户限制因子:
1
最大Applications Master资源:
0.1
提交Application访问控制列表(ACL):
*
Queue容量:
40
最小用户限制比:
100
最大Application数量:
10000
状态:
Running
管理Application访问控制列表(ACL):
*

➢ Fair Scheduler（公平调度器）

- 调度策略
  - 多队列公平共享集群资源
  - 通过平分的方式，动态分配资源，无需预先设定资源分配比例
  - 队列内部可配置调度策略：FIFO、Fair（默认）
- 资源抢占
  - 终止其他队列的任务，使其让出所占资源，然后将资源分配给占用资源量少于最小资源量限制的队列
- 队列权重
  - 当队列中有任务等待，并且集群中有空闲资源时，每个队列可以根据权重获得不同比例的空闲资源



- 全局配置（yarn-site.xml）

Property	Value
yarn.resourcemanager.scheduler.class	org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.FairScheduler（配置调度器类型）
yarn.scheduler.fair.allocation.file	公平调度器自定义配置文件路径，该文件每10秒加载一次。
yarn.scheduler.fair.user-as-default-queue	当应用程序未指定队列名时，是否指定用户名作为应用程序所在队列的名字。如果设置为false或未设置，所有未知队列的应用程序将被提交到default队列中，默认值为true。
yarn.scheduler.fair.preemption	如果一个队列占用的资源量少于最小资源量，是否启用资源抢占机制，默认值是false。

➢ 全局配置（Web）

配置 YARN service Scheduler

Use Scheduler: ○ Capacity Scheduler ◉ Fair Scheduler

全局配置

用户名作为Queue:	True	允许抢占资源:	False
基于规模的权重:	False	分配多个Containers:	False
忽略其他Node调度因子:	-1	忽略其他Rack调度因子:	-1

Queue配置

Queue是对集群中资源的抽象,Fair Scheduler维护一组Queue,每个Queue拥有一定的资源.使用Fair Scheduler调度策略,调度器会监控每个Queue使用的资源,并且会尽量保证每个队列占用相同的资源..

共享资源抢占时间: 9223372036854774

+ 添加子队列 编辑 × 删除

Queue	最少资源	最大资源	最大Application数量	权重	调度策略
root	1024mb,1vcores	2048mb,8vcores	50	1	FAIR
default	1024mb,1vcores	2048mb,8vcores	50	1	FAIR

➢ 自定义配置（fair-scheduler.xml）

```
<allocations>
  <queue name="sample_queue">
    <minResources>10000mb,5vcores</minResources>
    <maxResources>90000mb,45vcores</maxResources>
    <maxRunningApps>50</maxRunningApps>
    <maxAMShare>0.1</maxAMShare>
    <weight>2.0</weight>
    <schedulingPolicy>fair</schedulingPolicy>
    <queue name="sample_sub_queue">
      <aclSubmitApps>charlie</aclSubmitApps>
      <minResources>5000 mb,3vcores</minResources>
    </queue>
  </queue>
</allocations>
```

minResources：分配给队列的最小资源量

maxResources：分配给队列的最大资源量

maxRunningApps：队列内同时运行的最大任务数

maxAMShare：AM使用资源占队列资源的最大比例

weight：队列的权重，默认值是1

schedulingPolicy：队列内部的调度策略，fifo或fair

aclSubmitApps：有权提交任务的用户列表

- 自定义配置（Web）

Queue配置

Queue 名称:	default	最小内存:	1024
最小VCores:	1	最大内存:	2048
最大VCores:	8	最大Application数量:	50
权重:	1	最小资源抢占时间:	92233720368547
调度策略:	FAIR	提交Application访问控制列表(ACL):	*
管理Application访问控制列表(ACL):	*		

确定

4
chapter

YARN运维与监控

- ✓ YARN运维
- ✓ YARN监控

Shell命令

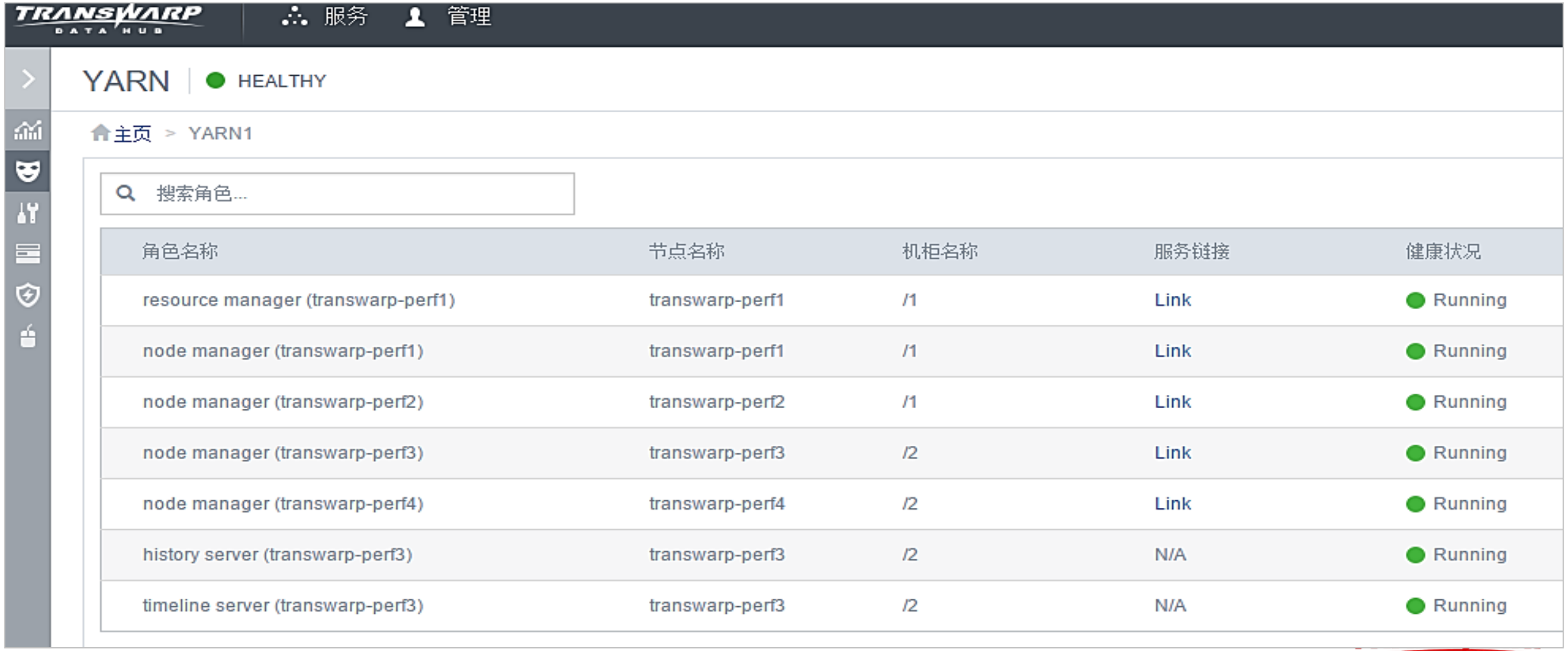
```
# yarn  application  [command_options]
```

Command Options	Description
-list	Lists applications from the RM
-kill <ApplicationId>	Kills the application
-status <ApplicationId>	Prints the status of the application

Kill任务

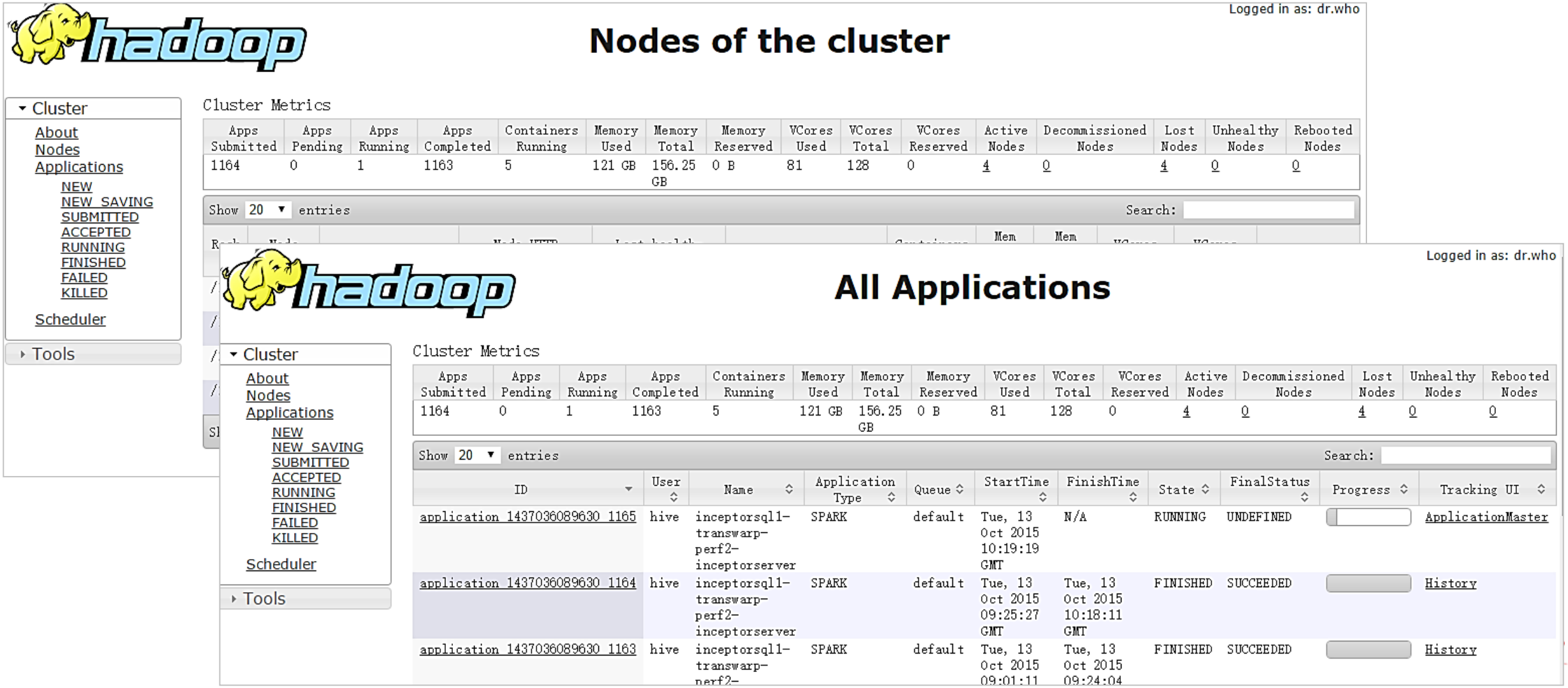
- CTRL^C不能终止任务，只是停止其在控制台的信息输出，任务仍在集群中运行
- 正确方法：先使用yarn application -list获取进程号，再使用-kill终止任务

➢ 访问Transwarp Manager → YARN → 角色 → Link，进入Resource Manager监控页面



角色名称	节点名称	机柜名称	服务链接	健康状况
resource manager (transwarp-perf1)	transwarp-perf1	/1	Link	Running
node manager (transwarp-perf1)	transwarp-perf1	/1	Link	Running
node manager (transwarp-perf2)	transwarp-perf2	/1	Link	Running
node manager (transwarp-perf3)	transwarp-perf3	/2	Link	Running
node manager (transwarp-perf4)	transwarp-perf4	/2	Link	Running
history server (transwarp-perf3)	transwarp-perf3	/2	N/A	Running
timeline server (transwarp-perf3)	transwarp-perf3	/2	N/A	Running

- 访问Resource Manager的8088端口，进入监控页面，如：http://172.16.140.204:8088



Logged in as: dr.who

Nodes of the cluster

Cluster
- About
- Nodes
- Applications
  - NEW
  - NEW_SAVING
  - SUBMITTED
  - ACCEPTED
  - RUNNING
  - FINISHED
  - FAILED
  - KILLED
- Scheduler

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
1164	0	1	1163	5	121 GB	156.25 GB	0 B	81	128	0	4	0	4	0	0

Show 20 entries Search:

Logged in as: dr.who

All Applications

Cluster
- About
- Nodes
- Applications
  - NEW
  - NEW_SAVING
  - SUBMITTED
  - ACCEPTED
  - RUNNING
  - FINISHED
  - FAILED
  - KILLED
- Scheduler

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
1164	0	1	1163	5	121 GB	156.25 GB	0 B	81	128	0	4	0	4	0	0

Show 20 entries Search:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI
application_1437036089630_1165	hive	inceptorsql1-transwarp-perf2-inceptorserver	SPARK	default	Tue, 13 Oct 2015 10:19:19 GMT	N/A	RUNNING	UNDEFINED		ApplicationMaster
application_1437036089630_1164	hive	inceptorsql1-transwarp-perf2-inceptorserver	SPARK	default	Tue, 13 Oct 2015 09:25:27 GMT	Tue, 13 Oct 2015 10:18:11 GMT	FINISHED	SUCCEEDED		History
application_1437036089630_1163	hive	inceptorsql1-transwarp-perf2-	SPARK	default	Tue, 13 Oct 2015 09:01:11	Tue, 13 Oct 2015 09:24:04	FINISHED	SUCCEEDED		History

Q&A

TRANSWARP
星 环 科 技