# 课程介绍

课程名称：Storm集群部署及单词技术

课程目标：

通过本课程能够掌握Strom集群搭建、Storm配置文件、Storm源码管理、Storm编程模型。

课程大纲：

1. 集群部署的基本流程
2. 集群部署的基础环境准备
3. Storm集群部署
4. Storm集群的常用操作命令
5. Storm集群的进程及日志熟悉
6. Storm源码下载及目录熟悉
7. Storm 单词计数案列

# 课程内容

## 集群部署的基本流程

集群部署的流程：下载安装包、解压安装包、修改配置文件、分发安装包、启动集群

注意：

    所有的集群上都需要配置hosts

    vi  /etc/hosts

192.168.239.128 storm01 zk01 hadoop01 直接都用master就行

      192.168.239.129 storm02 zk02 hadoop02

  192.168.239.130 storm03 zk03 hadoop03

## 集群部署的基础环境准备

**安装前的准备工作（zk集群已经部署完毕）**

* 关闭防火墙

chkconfig iptables off && setenforce 0

* 创建用户

groupadd realtime &&　useradd realtime　&& usermod -a -G realtime realtime

* 创建工作目录并赋权

mkdir /export

mkdir /export/servers

chmod 755 -R /export

* 切换到realtime用户下

su realtime

## 3、Storm集群部署

### 3.1、下载安装包

wget    <http://124.202.164.6/files/1139000006794ECA/apache.fayea.com/storm/apache-storm-0.9.5/apache-storm-0.9.5.tar.gz>

### 3.2、解压安装包

tar -zxvf apache-storm-0.9.5.tar.gz -C /export/servers/

cd /export/servers/

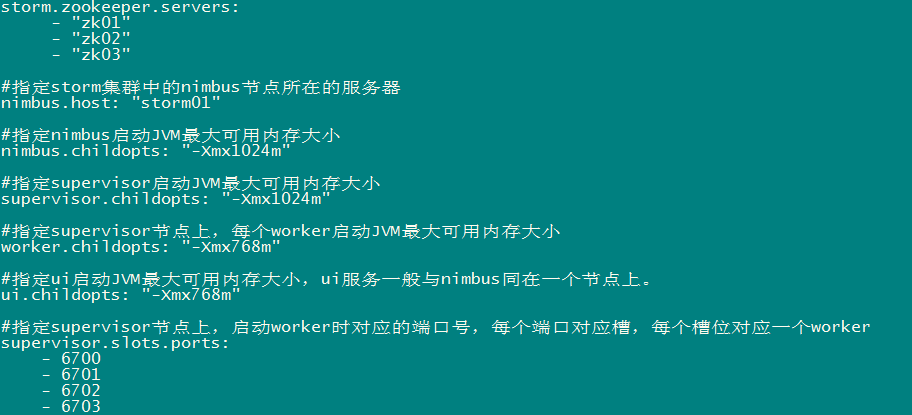
ln -s apache-storm-0.9.5 storm

### 3.3、修改配置文件

mv /export/servers/storm/conf/storm.yaml /export/servers/storm/conf/storm.yaml.bak

vi /export/servers/storm/conf/storm.yaml

输入以下内容：



### 3.4、分发安装包

scp -r /export/servers/apache-storm-0.9.5 storm02:/export/servers

然后分别在各机器上创建软连接

cd /export/servers/

ln -s apache-storm-0.9.5 storm

### 3.5、启动集群

* 在nimbus.host所属的机器上启动 nimbus服务

cd /export/servers/storm/bin/

nohup ./storm nimbus &

* 在nimbus.host所属的机器上启动ui服务

cd /export/servers/storm/bin/

nohup ./storm ui &

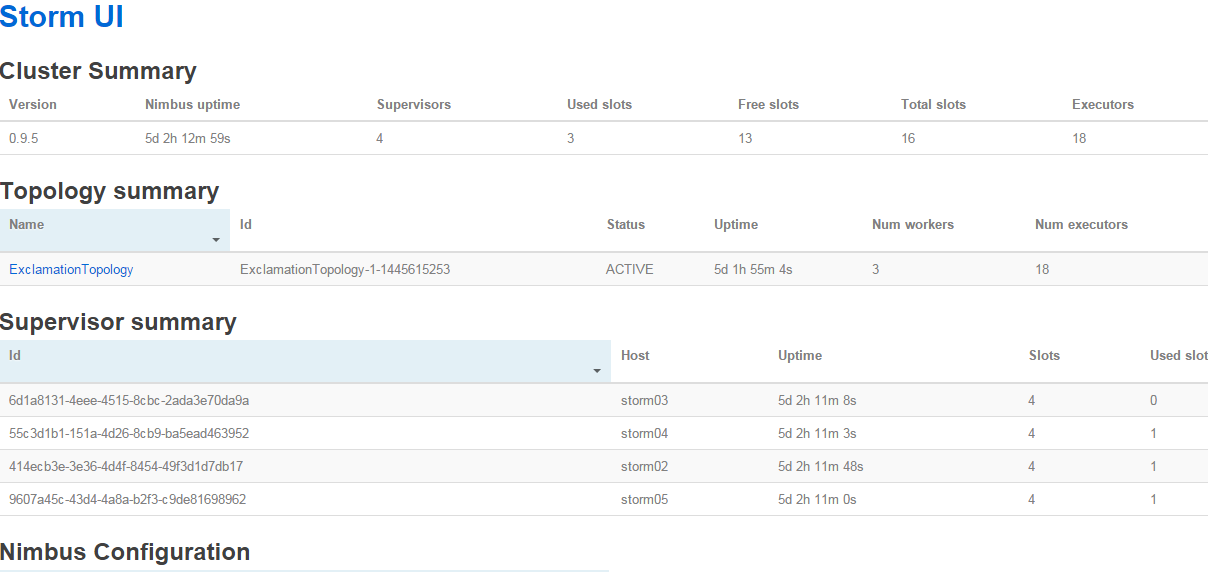
* 在其它个点击上启动supervisor服务

cd /export/servers/storm/bin/

nohup ./storm supervisor &

### 3.6、查看集群

访问nimbus.host:/8080，即可看到storm的ui界面。



# 4、Storm常用操作命令

有许多简单且有用的命令可以用来管理拓扑，它们可以提交、杀死、禁用、再平衡拓扑。

* 提交任务命令格式：storm jar 【jar路径】 【拓扑包名.拓扑类名】 【拓扑名称】

bin/storm jar examples/storm-starter/storm-starter-topologies-0.9.6.jar storm.starter.WordCountTopology wordcount

* 杀死任务命令格式：storm kill 【拓扑名称】 -w 10（执行kill命令时可以通过-w [等待秒数]指定拓扑停用以后的等待时间）

storm kill topology-name -w 10

* 停用任务命令格式：storm deactivte 【拓扑名称】

storm deactivte topology-name

我们能够挂起或停用运行中的拓扑。当停用拓扑时，所有已分发的元组都会得到处理，但是spouts的nextTuple方法不会被调用。销毁一个拓扑，可以使用kill命令。它会以一种安全的方式销毁一个拓扑，首先停用拓扑，在等待拓扑消息的时间段内允许拓扑完成当前的数据流。

* 启用任务命令格式：storm activate【拓扑名称】

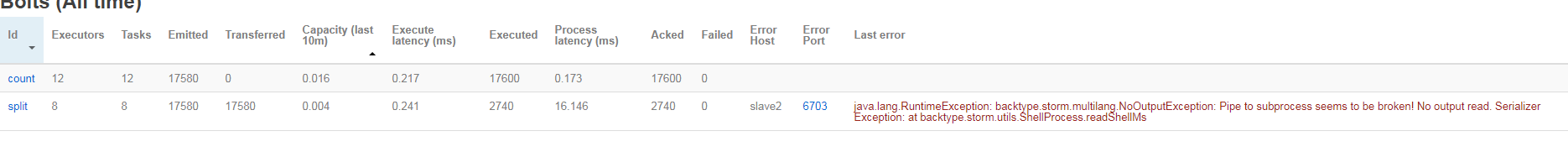
storm activate topology-name

* 重新部署任务命令格式：storm rebalance 【拓扑名称】

storm rebalance topology-name

再平衡使你重分配集群任务。这是个很强大的命令。比如，你向一个运行中的集群增加了节点。**再平衡命令将会停用拓扑**，然后在相应超时时间之后重分配工人，并重启拓扑。

**假如你提交一个wordcount的topology，它的spout（假如有五个task，需要三个worker）分配给了slave1（启动一个worker）和slave2（启动两个worker）的worker去执行，如果你加了一个supervisor节点在master上，你使用rebalance命令让它重新负载均衡，这时就会变成三台supervisor节点每台启动一个worker去运行spout，这样刚刚好。但是你kill掉master的supervisor进程后，master上的worker也随之消失，这时wordcount就会运行不正常，会报错：**

****

**所以你要重新rebalence一下，让它重新负载均衡。**

# 5、Storm集群的进程及日志熟悉

## 5.1、部署成功之后，启动storm集群。

依次启动集群的各种角色

## 5.2、查看nimbus的日志信息

在nimbus的服务器上

cd /export/servers/storm/logs

tail -100f /export/servers/storm/logs/nimbus.log

## 5.3、查看ui运行日志信息

在ui的服务器上，一般和nimbus一个服务器

cd /export/servers/storm/logs

tail -100f /export/servers/storm/logs/ui.log

## 5.4、查看supervisor运行日志信息

在supervisor服务上

cd /export/servers/storm/logs

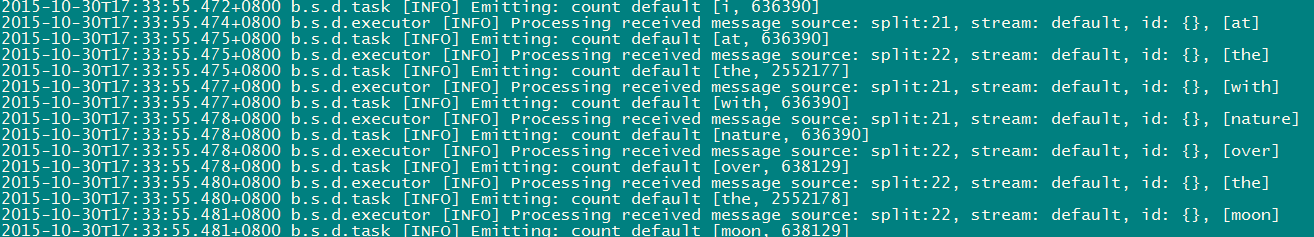
tail -100f /export/servers/storm/logs/supervisor.log

## 5.5、查看supervisor上worker运行日志信息

在supervisor服务上

cd /export/servers/storm/logs

tail -100f /export/servers/storm/logs/**worker-6702.log**

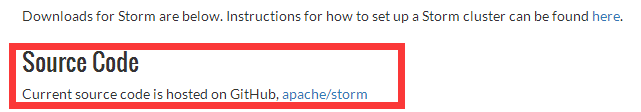


(该worker正在运行wordcount程序)

# 6、Storm源码下载及目录熟悉

## 6.1、在Storm官方网站上寻找源码地址

<http://storm.apache.org/downloads.html>



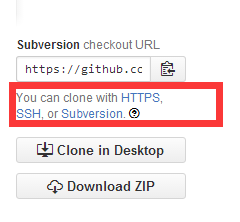
## 6.2、点击文字标签进入github

点击Apache/storm文字标签，进入github

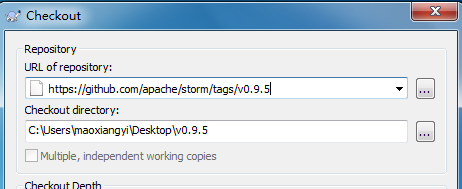
<https://github.com/apache/storm>

## 6.3、拷贝storm源码地址

在网页右侧，拷贝storm源码地址



## 6.4、使用Subversion客户端下载

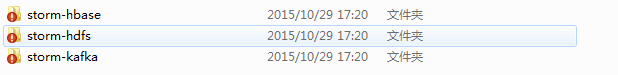


<https://github.com/apache/storm/tags/v0.9.5>

## 6.5、Storm源码目录分析（重要）



扩展包中的三个项目，使storm能与hbase、hdfs、kafka交互



# 7、Storm单词技术案例（重点掌握）

## 7.1、功能说明

设计一个topology，来实现对文档里面的单词出现的频率进行统计。

整个topology分为三个部分：

* RandomSentenceSpout：数据源，在已知的英文句子中，随机发送一条句子出去。
* SplitSentenceBolt：负责将单行文本记录（句子）切分成单词
* WordCountBolt：负责对单词的频率进行累加

## 7.2、项目主要流程



## 7.3、RandomSentenceSpout的实现及生命周期



## 7.4、SplitSentenceBolt的实现及生命周期



## 7.5、WordCountBolt的实现及生命周期



## 7.6、Stream Grouping详解

Storm里面有7种类型的stream grouping

* Shuffle Grouping: 随机分组， 随机派发stream里面的tuple，保证每个bolt接收到的tuple数目大致相同。**随机函数Random.nextInt（2）假如有两个blot**
* Fields Grouping：按字段分组，比如按userid来分组，具有同样userid的tuple会被分到相同的Bolts里的一个task，而不同的userid则会被分配到不同的bolts里的task。**Hashcode（userid）%blotnum**
* All Grouping：广播发送，对于每一个tuple，所有的bolts都会收到。
* Global Grouping：全局分组， 这个tuple被分配到storm中的一个bolt的其中一个task。再具体一点就是分配给id值最低的那个task。
* Non Grouping：不分组，这stream grouping个分组的意思是说stream不关心到底谁会收到它的tuple。**目前这种分组和Shuffle grouping是一样的效果，** 有一点不同的是storm会把这个bolt放到这个bolt的订阅者同一个线程里面去执行。
* Direct Grouping： 直接分组， 这是一种比较特别的分组方法，用这种分组意味着消息的发送者指定由消息接收者的哪个task处理这个消息。只有被声明为Direct Stream的消息流可以声明这种分组方法。而且这种消息tuple必须使用emitDirect方法来发射。消息处理者可以通过TopologyContext来获取处理它的消息的task的id （OutputCollector.emit方法也会返回task的id）。
* **Local or shuffle grouping：如果目标bolt有一个或者多个task在同一个工作进程中，tuple将会被随机发生给这些tasks。否则，和普通的Shuffle Grouping行为一致。**
* 