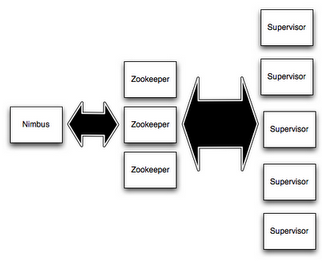
**1. Storm集群组件**

Storm集群中包含两类节点：主控节点（Master Node）和工作节点（Work Node）。其分别对应的角色如下：

* 主控节点（Master Node）上运行一个被称为Nimbus的后台程序，它负责在Storm集群内分发代码，分配任务给工作机器，并且负责监控集群运行状态。Nimbus的作用类似于Hadoop中JobTracker的角色。
* 每个工作节点（Work Node）上运行一个被称为Supervisor的后台程序。Supervisor负责监听从Nimbus分配给它执行的任务，据此启动或停止执行任务的工作进程。每一个工作进程执行一个Topology的子集；一个运行中的Topology由分布在不同工作节点上的多个工作进程组成。



**Storm集群组件**

Nimbus和Supervisor节点之间所有的协调工作是通过Zookeeper集群来实现的。此外，Nimbus和Supervisor进程都是快速失败（fail-fast)和无状态（stateless）的；Storm集群所有的状态要么在Zookeeper集群中，要么存储在本地磁盘上。这意味着你可以用kill -9来杀死Nimbus和Supervisor进程，它们在重启后可以继续工作。这个设计使得Storm集群拥有不可思议的稳定性。

**2. 安装Storm集群**

这一章节将详细描述如何搭建一个Storm集群。下面是接下来需要依次完成的安装步骤：

* 搭建Zookeeper集群；
* 安装Storm依赖库；
* 下载并解压Storm发布版本；
* 修改storm.yaml配置文件；
* 启动Storm各个后台进程。

**2.1 搭建Zookeeper集群**

Storm使用Zookeeper协调集群，由于Zookeeper并不用于消息传递，所以Storm给Zookeeper带来的压力相当低。大多数情况下，单个节点的Zookeeper集群足够胜任，不过为了确保故障恢复或者部署大规模Storm集群，可能需要更大规模节点的Zookeeper集群（对于Zookeeper集群的话，官方推荐的最小节点数为3个）。在Zookeeper集群的每台机器上完成以下安装部署步骤：

1）下载安装Java JDK，官方下载链接为<http://java.sun.com/javase/downloads/index.jsp>，JDK版本为JDK 6或以上。

2）根据Zookeeper集群的负载情况，合理设置Java堆大小，尽可能避免发生swap，导致Zookeeper性能下降。保守期间，4GB内存的机器可以为Zookeeper分配3GB最大堆空间。

3）下载后解压安装Zookeeper包，官方下载链接为<http://hadoop.apache.org/zookeeper/releases.html>。

4）根据Zookeeper集群节点情况，创建如下格式的Zookeeper配置文件zoo.cfg：

[复制代码](javascript:void(0);)

tickTime=2000

dataDir=/var/zookeeper/

clientPort=2181

initLimit=5

syncLimit=2

server.1=zoo1:2888:3888

server.2=zoo2:2888:3888

server.3=zoo3:2888:3888

[复制代码](javascript:void(0);)

其中，dataDir指定Zookeeper的数据文件目录；其中server.id=host:port:port，id是为每个Zookeeper节点的编号，保存在dataDir目录下的myid文件中，zoo1~zoo3表示各个Zookeeper节点的hostname，第一个port是用于连接leader的端口，第二个port是用于leader选举的端口。

5）在dataDir目录下创建myid文件，文件中只包含一行，且内容为该节点对应的server.id中的id编号。

6）启动Zookeeper服务：

java -cp zookeeper.jar:lib/log4j-1.2.15.jar:conf \ org.apache.zookeeper.server.quorum.QuorumPeerMain zoo.cfg

也可以通过*bin/zkServer.sh*脚本启动Zookeeper服务。

7）通过Zookeeper客户端测试服务是否可用：

* Java客户端下，执行如下命令：

java -cp zookeeper.jar:src/java/lib/log4j-1.2.15.jar:conf:src/java/lib/jline-0.9.94.jar \ org.apache.zookeeper.ZooKeeperMain -server 127.0.0.1:2181

也可以通过*bin/zkCli.sh*脚本启动Zookeeper Java客户端。

* C客户端下，进入*src/c*目录下，编译单线程或多线程客户端：

./configure

make cli\_st

make cli\_mt

运行进入C客户端：

cli\_st 127.0.0.1:2181

cli\_mt 127.0.0.1:2181

至此，完成了Zookeeper集群的部署与启动。

注意事项：

1. 由于Zookeeper是快速失败（fail-fast)的，且遇到任何错误情况，进程均会退出，因此，最好能通过监控程序将Zookeeper管理起来，保证Zookeeper退出后能被自动重启。详情参考[这里](http://zookeeper.apache.org/doc/r3.3.3/zookeeperAdmin.html" \l "sc_supervision)。
2. Zookeeper运行过程中会在dataDir目录下生成很多日志和快照文件，而Zookeeper运行进程并不负责定期清理合并这些文件，导致占用大量磁盘空间，因此，需要通过cron等方式定期清除没用的日志和快照文件。详情参考[这里](http://zookeeper.apache.org/doc/r3.3.3/zookeeperAdmin.html" \l "sc_maintenance)。具体命令格式如下：java -cp zookeeper.jar:log4j.jar:conf org.apache.zookeeper.server.PurgeTxnLog <dataDir> <snapDir> -n <count>

**2.2 安装Storm依赖库**

接下来，需要在Nimbus和Supervisor机器上安装Storm的依赖库，具体如下：

1. [ZeroMQ 2.1.7](http://www.zeromq.org/area:download) – 请勿使用2.1.10版本，因为该版本的一些严重bug会导致Storm集群运行时出现奇怪的问题。少数用户在2.1.7版本会遇到"IllegalArgumentException"的异常，此时降为2.1.4版本可修复这一问题。
2. [JZMQ](http://github.com/nathanmarz/jzmq)
3. Java 6
4. Python 2.6.6
5. unzip

以上依赖库的版本是经过Storm测试的，Storm并不能保证在其他版本的Java或Python库下可运行。

**2.2.1 安装ZMQ 2.1.7**

下载后编译安装ZMQ：

[复制代码](javascript:void(0);)

wget http://download.zeromq.org/zeromq-2.1.7.tar.gz

tar -xzf zeromq-2.1.7.tar.gz

cd zeromq-2.1.7

./configure

make

sudo make install

[复制代码](javascript:void(0);)

注意事项：

1. 如果安装过程报错uuid找不到，则通过如下的包安装uuid库：

sudo yum install e2fsprogsl -b current

sudo yum install e2fsprogs-devel -b current

**2.2.2 安装JZMQ**

下载后编译安装JZMQ：

[复制代码](javascript:void(0);)

git clone https://github.com/nathanmarz/jzmq.git

cd jzmq

./autogen.sh

./configure

make

sudo make install

[复制代码](javascript:void(0);)

为了保证JZMQ正常工作，可能需要完成以下配置：

1. 正确设置 JAVA\_HOME环境变量
2. 安装Java开发包
3. 升级autoconf
4. 如果你是Mac OSX，参考[这里](http://blog.pmorelli.com/getting-zeromq-and-jzmq-running-on-mac-os-x)

注意事项：

     1. 如果运行./configure命令出现问题，参考[这里](http://stackoverflow.com/questions/3522248/how-do-i-compile-jzmq-for-zeromq-on-osx)。

**2.2.3 安装Java 6**

1. 下载并安装JDK 6，参考[这里](http://www.oracle.com/technetwork/java/javase/index-137561.html" \l "linux)；

2. 配置JAVA\_HOME环境变量；

3. 运行java、javac命令，测试java正常安装。

**2.2.4 安装Python2.6.6**

1. 下载Python2.6.6：

wget http://www.python.org/ftp/python/2.6.6/Python-2.6.6.tar.bz2

2. 编译安装Python2.6.6：

tar –jxvf Python-2.6.6.tar.bz2

cd Python-2.6.6

./configure

make

make install

3. 测试Python2.6.6：

$ python -V

Python 2.6.6

**2.2.5 安装unzip**

1. 如果使用RedHat系列Linux系统，执行以下命令安装unzip：

apt-get install unzip

2. 如果使用Debian系列Linux系统，执行以下命令安装unzip：

yum install unzip

**2.3 下载并解压Storm发布版本**

下一步，需要在Nimbus和Supervisor机器上安装Storm发行版本。

1. 下载Storm发行版本，推荐使用Storm0.8.1：

wget https://github.com/downloads/nathanmarz/storm/storm-0.8.1.zip

2. 解压到安装目录下：

unzip storm-0.8.1.zip

**2.4 修改storm.yaml配置文件**

Storm发行版本解压目录下有一个conf/storm.yaml文件，用于配置Storm。默认配置在[这里](https://github.com/nathanmarz/storm/blob/master/conf/defaults.yaml)可以查看。conf/storm.yaml中的配置选项将覆盖defaults.yaml中的默认配置。以下配置选项是必须在conf/storm.yaml中进行配置的：

1) **storm.zookeeper.servers**: Storm集群使用的Zookeeper集群地址，其格式如下：

storm.zookeeper.servers:

- "111.222.333.444"

- "555.666.777.888"

如果Zookeeper集群使用的不是默认端口，那么还需要**storm.zookeeper.port选项**。

2) **storm.local.dir**: Nimbus和Supervisor进程用于存储少量状态，如jars、confs等的本地磁盘目录，需要提前创建该目录并给以足够的访问权限。然后在storm.yaml中配置该目录，如：

storm.local.dir: "/home/admin/storm/workdir"

3) **java.library.path**: Storm使用的本地库（ZMQ和JZMQ）加载路径，默认为"/usr/local/lib:/opt/local/lib:/usr/lib"，一般来说ZMQ和JZMQ默认安装在/usr/local/lib 下，因此不需要配置即可。

4) **nimbus.host**: Storm集群Nimbus机器地址，各个Supervisor工作节点需要知道哪个机器是Nimbus，以便下载Topologies的jars、confs等文件，如：

nimbus.host: "111.222.333.444"

5) **supervisor.slots.ports**: 对于每个Supervisor工作节点，需要配置该工作节点可以运行的worker数量。每个worker占用一个单独的端口用于接收消息，该配置选项即用于定义哪些端口是可被worker使用的。默认情况下，每个节点上可运行4个workers，分别在6700、6701、6702和6703端口，如：

supervisor.slots.ports:

- 6700

- 6701

- 6702

- 6703

**2.5 启动Storm各个后台进程**

最后一步，启动Storm的所有后台进程。和Zookeeper一样，Storm也是快速失败（fail-fast)的系统，这样Storm才能在任意时刻被停止，并且当进程重启后被正确地恢复执行。这也是为什么Storm不在进程内保存状态的原因，即使Nimbus或Supervisors被重启，运行中的Topologies不会受到影响。

以下是启动Storm各个后台进程的方式：

1. **Nimbus**: 在Storm主控节点上运行"bin/storm nimbus >/dev/null 2>&1 &"启动Nimbus后台程序，并放到后台执行；
2. **Supervisor**: 在Storm各个工作节点上运行"bin/storm supervisor >/dev/null 2>&1 &"启动Supervisor后台程序，并放到后台执行；
3. **UI**: 在Storm主控节点上运行"bin/storm ui >/dev/null 2>&1 &"启动UI后台程序，并放到后台执行，启动后可以通过http://{nimbus host}:8080观察集群的worker资源使用情况、Topologies的运行状态等信息。

注意事项：

1. Storm后台进程被启动后，将在Storm安装部署目录下的logs/子目录下生成各个进程的日志文件。
2. 经测试，Storm UI必须和Storm Nimbus部署在同一台机器上，否则UI无法正常工作，因为UI进程会检查本机是否存在Nimbus链接。
3. 为了方便使用，可以将bin/storm加入到系统环境变量中。

至此，Storm集群已经部署、配置完毕，可以向集群提交拓扑运行了。

**3. 向集群提交任务**

1）启动Storm Topology：

storm jar allmycode.jar org.me.MyTopology arg1 arg2 arg3

其中，allmycode.jar是包含Topology实现代码的jar包，org.me.MyTopology的main方法是Topology的入口，arg1、arg2和arg3为org.me.MyTopology执行时需要传入的参数。

2）停止Storm Topology：

storm kill {toponame}

其中，{toponame}为Topology提交到Storm集群时指定的Topology任务名称。

# CentOS 6.4单机环境下安装配置Storm

[2013-08-31 05:03:41](http://shiyanjun.cn/archives/241.html)    [Yanjun](http://shiyanjun.cn/archives/author/yanjun" \o "查看作者 Yanjun 的全部文章)

Storm是一个分布式的、高容错的实时计算系统，在实时性要求比较强的应用场景下，可以用它来处理海量数据。我们尝试着搭建Storm平台，来实现实时计算。下面，我们在CentOS 6.4上安装配置Storm系统。

**安装配置**

安装配置过程，按照如下步骤进行：

1、安装配置sunjdk  
下载sunjdk，并安装Java运行环境：

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | wget <http://download.oracle.com/otn/java/jdk/6u45-b06/jdk-6u45-linux-x64.bin> | | |
| 2 | | chmod +x jdk-6u45-linux-x64-rpm.bin |

|  |  |
| --- | --- |
| 3 | ./jdk-6u45-linux-x64.bin |

配置Java运行时环境：

|  |  |
| --- | --- |
| 1 | vi ~/.bashrc |
| 2 | export JAVA\_HOME=/usr/java/jdk1.6.0\_45/ | |

|  |  |
| --- | --- |
| 3 | export PATH=$PATH:$JAVA\_HOME/bin |
| 4 | export CLASSPATH=$JAVA\_HOME/lib/\*.jar:$JAVA\_HOME/jre/lib/\*.jar | |

|  |  |
| --- | --- |
| 5 | . ~/.bashrc |
| 6 | java -version | |

2、安装zeromq  
执行如下命令，进行下载配置安装：

|  |  |  |
| --- | --- | --- |
| 1 | wget <http://download.zeromq.org/zeromq-2.2.0.>tar.gz | |
| 2 | tar -zvxf zeromq-2.2.0.tar.gz |

|  |  |  |
| --- | --- | --- |
| 3 | cd zeromq-2.2.0/ | |
| 4 | ./configure |

|  |  |
| --- | --- |
| 5 | sudo make |
| 6 | sudo make install | |

3、安装jzmq  
安装jzmq需要使用Git下载源码，从源代码编译安装：

|  |  |
| --- | --- |
| 1 | sudo yum install git |
| 2 | git clone <git://github.com/nathanmarz/jzmq.git> | |

|  |  |
| --- | --- |
| 3 | cd jzmq/ |
| 4 | sudo make | |

|  |  |
| --- | --- |
| 5 | sudo make install |

4、安装Storm  
下载解压缩Storm软件包即可：

|  |  |  |
| --- | --- | --- |
| 1 | wget <http://cloud.github.com/downloads/nathanmarz/storm/storm-0.8.1.zip> | |
| 2 | tar -xvzf storm-0.8.1.zip |

然后配置环境变量：

|  |  |  |
| --- | --- | --- |
| 1 | cd storm-0.8.1/ | |
| 2 | vi ~/.bashrc |

|  |  |  |
| --- | --- | --- |
| 3 | export STORM\_HOME=/home/shirdrn/programs/storm-0.8.1 | |
| 4 | export PATH=$PATH:$STORM\_HOME/bin |

|  |  |
| --- | --- |
| 5 | . ~/.bashrc |

5、安装构建storm-starter  
首先需要下载代码，并使用Maven构建：

|  |  |  |
| --- | --- | --- |
| 1 | git clone <https://github.com/nathanmarz/storm-starter.git> | |
| 2 | cd storm-starter/ |

|  |  |
| --- | --- |
| 3 | mvn -f m2-pom.xml package |

如果需要把Storm的示例代码导入到Eclipse环境中，需要执行如下命令：

|  |  |
| --- | --- |
| 1 | cd storm-starter/ |
| 2 | cp m2-pom.xml pom.xml | |

|  |  |
| --- | --- |
| 3 | mvn eclipse:eclipse |

6、配置Storm  
修改配置文件conf/storm.yaml内容如下所示：

|  |  |  |
| --- | --- | --- |
| 01 | storm.zookeeper.servers: | |
| 02 | - "nn" |

|  |  |  |
| --- | --- | --- |
| 03 | storm.zookeeper.port: 2181 | |
| 04 | nimbus.host: "nn" |

|  |  |  |
| --- | --- | --- |
| 05 | storm.local.dir: "/home/shirdrn/programs/storm-0.8.1/tmp" | |
| 06 | supervisor.slots.ports: |

|  |  |
| --- | --- |
| 07 | - 6700 |
| 08 | - 6701 |

|  |  |
| --- | --- |
| 09 | - 6702 |
| 10 | - 6703 |

**启动运行**

1、启动Storm相关服务

* 启动ZooKeeper

|  |  |
| --- | --- |
| 1 | bin/zkServer.sh start |

* 启动Nimbus

|  |  |
| --- | --- |
| 1 | bin/storm nimbus |

查看日志，确定Nimbus启动是否成功：

|  |  |
| --- | --- |
| 1 | [shirdrn@nn storm-0.8.1]$ tail -100f logs/nimbus.log |

* 启动Supervisor

|  |  |
| --- | --- |
| 1 | bin/storm supervisor |

查看日志，确定Supervisor启动是否成功：

|  |  |
| --- | --- |
| 1 | [shirdrn@nn storm-0.8.1]$ tail -100f logs/supervisor.log |

2、提交一个Topology  
上面，已经使用Maven构建storm-starter工程，在target目录下生成一个jar文件，然后将该storm-starter工程中的WordCountTopology提交到Nimbus，执行如下命令：

|  |  |
| --- | --- |
| 1 | bin/storm jar ../storm-starter/target/storm-starter-0.0.1-SNAPSHOT-jar-with-dependencies.jar storm.starter.WordCountTopology myFirstStormApp |

上面myFirstStormApp是提交的Topology的名称，可以看到提交Topology的日志信息：

|  |  |  |
| --- | --- | --- |
| 1 | | 0    [main] INFO  backtype.storm.StormSubmitter  - Jar not uploaded to master yet. Submitting jar... |
| 2 | 12   [main] INFO  backtype.storm.StormSubmitter  - Uploading topology jar ../storm-starter/target/storm-starter-0.0.1-SNAPSHOT-jar-with-dependencies.jar to assigned location: /home/shirdrn/programs/storm-0.8.1/tmp/nimbus/inbox/stormjar-0ae68c15-130d-46f9-a46a-69dd4de29a99.jar | |

|  |  |  |
| --- | --- | --- |
| 3 | 119  [main] INFO  backtype.storm.StormSubmitter  - Successfully uploaded topology jar to assigned location: /home/shirdrn/programs/storm-0.8.1/tmp/nimbus/inbox/stormjar-0ae68c15-130d-46f9-a46a-69dd4de29a99.jar | |
| 4 | | 119  [main] INFO  backtype.storm.StormSubmitter  - Submitting topology myFirstStormApp in distributed mode with conf {"topology.workers":3,"topology.debug":true} |

|  |  |
| --- | --- |
| 5 | 423  [main] INFO  backtype.storm.StormSubmitter  - Finished submitting topology: myFirstStormApp |

这时，可以通过查看worker的日志，来确定我们提交的Topology的执行情况：

|  |  |
| --- | --- |
| 1 | [shirdrn@nn storm-0.8.1]$ tail -100f logs/worker-6700.log |
| 2 | [shirdrn@nn storm-0.8.1]$ tail -100f logs/worker-6701.log |

|  |  |
| --- | --- |
| 3 | [shirdrn@nn storm-0.8.1]$ tail -100f logs/worker-6702.log |
| 4 | [shirdrn@nn storm-0.8.1]$ tail -100f logs/worker-6703.log |

worker日志文件名称的后缀正好对应于我们在配置文件conf/storm.yaml中配置supervisor.slots.ports中的端口号。  
如果只是上面的命令不带参数，表示虚拟测试该程序，如下所示：

|  |  |
| --- | --- |
| 1 | bin/storm jar ../storm-starter/target/storm-starter-0.0.1-SNAPSHOT-jar-with-dependencies.jar storm.starter.WordCountTopology |

可以看到具体模拟执行的情况，它并不将这个Topology提交给Nimbus。

3、Storm管理命令  
可以通过如下命令查看Storm的管理操作命令：

|  |  |
| --- | --- |
| 1 | bin/storm help |

下面，给出一些常用的命令：

* 查询当前运行的Topology

|  |  |
| --- | --- |
| 1 | bin/storm list |

* 杀掉运行中的Topology

|  |  |
| --- | --- |
| 1 | bin/storm kill myFirstStormApp |

**问题说明**

在安装Storm的过程中，有关一些依赖安装包的问题，不像在Ubuntu系统下，可以模糊指定软件包名称，然后会给出一些提示信息，CentOS需要明确地指定软件包名称，记录下一下在安装过程中遇到的问题及其解决办法。

1、出现错误：configure: error: Unable to find a working C++ compiler  
需要安装g++编译器：

|  |  |
| --- | --- |
| 1 | sudo yum install gcc-c++ |

2、出现错误：configure: error: cannot link with -luuid, install uuid-dev.  
需要安装软件包uuid-devel和libuuid-devel：

|  |  |
| --- | --- |
| 1 | sudo yum install uuid-devel |
| 2 | sudo yum install libuuid-devel | |

3、出现错误：autogen.sh: error: could not find libtool. libtool is required to run autogen.sh.  
需要安装libtool：

|  |  |
| --- | --- |
| 1 | sudo yum install libtool |

**参考链接**

* <http://github.com/nathanmarz/storm-starter>
* <http://blog.sina.com.cn/s/blog_546abd9f0101cce8.html>
* <http://www.cnblogs.com/zeutrap/archive/2012/10/11/2720528.html>
* <http://in.sdo.com/?p=542>
* <http://cloud.github.com/downloads/nathanmarz/storm/storm-0.8.1.zip>