

实验二 Xen 平台的搭建

1.实验目的

学习并动手了解开源虚拟化系统 Xen 的编译、安装、启动、运行等全过程，通过在实验过程中解决遇到的各种问题，增加同学们对虚拟化平台的了解，为后续的实验和学习打下基础。

2.相关材料

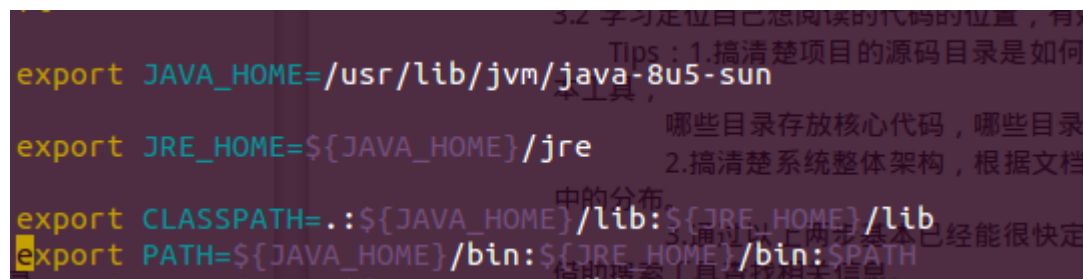
- ✧ hadoop 下载（示例是 hadoop-2.4.0 版）
- ✧ JDK 安装
- ✧ SSH 安装
- ✧ 实验平台为 ubuntu 14.04

3.实验步骤

1.安装 JDK

从官网下载 JDK 包，以 jdk-8u5-linux-x64.tar.gz 为例，解压到目录下
`/usr/lib/jvm/java-8u5-sun`

配置环境变量 `vim ~/.bashrc` ， 在文件末尾添加



```
export JAVA_HOME=/usr/lib/jvm/java-8u5-sun
export JRE_HOME=${JAVA_HOME}/jre
export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib
export PATH=${JAVA_HOME}/bin:${JRE_HOME}/bin:SPATH
```

再次输入 `source ~/.bashrc` 使之生效 。输入 `java -version` 测试设置是否成功

输出信息有误，请查看配置文件的设置。若输出的 JDK 版本不是你所安装的 JDK 版本，那可进行如下系统默认配置（较低版本的 JDK 或者系统自带的，可能会有问题，建议更新统一下）

配置系统默认 JDK（可选）

输入 `sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/java-8u5-sun/bin/java 300`

`sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/java-8u5-sun/bin/javac 300`

输入 `sudo update-alternatives --config java` 系统可能会出现多种 JDK，根据提示进行选择某一 JDK 为默认 JDK。本试验机仅有一个版本 JDK

2，安装 SSH 以及免密码登录

输入 `sudo apt-get install openssh-server openssh-client` 安装 SSH，若安装不了，用 `sudo apt-get update` 更新下再输入一次

在用户自己的主目录下创建.ssh 文件夹（若已有，先删除之）。

接下来进行免密码登录设置，否则在 yarn 后台服务开启的时候会要求进行 3 次 SSH 密码输入，比较麻烦

输入 `ssh-keygen -t dsa -P "" -f ~/.ssh/id_dsa`，接着添加公匙要公匙文件
`cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys`

输入 `ssh localhost` 进行测试，看看是否免密码。

To get a Hadoop distribution, download a recent stable release from one of the [Apache Download Mirrors](#).

Unpack the downloaded Hadoop distribution. In the distribution, edit the file `etc/hadoop/hadoop-env.sh` to define some parameters as follows:

```
# set to the root of your Java installation
export JAVA_HOME=/usr/java/latest

# Assuming your installation directory is /usr/local/hadoop
export HADOOP_PREFIX=/usr/local/hadoop
```

Try the following command:

```
$ bin/hadoop
```

This will display the usage documentation for the hadoop script.

Now you are ready to start your Hadoop cluster in one of the three supported modes:

- [Local \(Standalone\) Mode](#)
 - [Pseudo-Distributed Mode](#)
 - [Fully-Distributed Mode](#)
- By default, Hadoop is configured to run in a non-distributed mode, as a single Java process. This is useful for debugging.
 - The following example copies the unpacked conf directory to use as input and then finds and displays every match of the given regular expression. Output is written to the given output directory.
 - `$ mkdir input`
 - `$ cp etc/hadoop/*.xml input`
 - `$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.1.jar grep input output 'dfs[a-z.]+'`
 - `$ cat output/*`

Hadoop can also be run on a single-node in a pseudo-distributed mode where each Hadoop daemon runs in a separate Java process.

Configuration

Use the following:

`etc/hadoop/core-site.xml`:

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
</configuration>
```

`etc/hadoop/hdfs-site.xml`:

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

Setup passphraseless ssh

Now check that you can ssh to the localhost without a passphrase:

```
$ ssh localhost
```

If you cannot ssh to localhost without a passphrase, execute the following commands:

```
$ ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
$ cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys
```

Execution

The following instructions are to run a MapReduce job locally. If you want to execute a job on YARN, see [YARN on Single Node](#).

1. Format the filesystem:

```
$ bin/hdfs namenode -format
```

2. Start NameNode daemon and DataNode daemon:

```
$ sbin/start-dfs.sh
```

The hadoop daemon log output is written to the \$HADOOP_LOG_DIR directory (defaults to \$HADOOP_HOME/logs).

3. Browse the web interface for the NameNode; by default it is available at:

- o NameNode -
`http://localhost:50070/`

4. Make the HDFS directories required to execute MapReduce jobs:

```
$ bin/hdfs dfs -mkdir /user
$ bin/hdfs dfs -mkdir
/user/<username>
```

6. Copy the input files into the distributed filesystem:

```
$ bin/hdfs dfs -put etc/hadoop input
```

7. Run some of the examples provided:

```
$ bin/hadoop jar
share/hadoop/mapreduce/hadoop-
mapreduce-examples-2.5.1.jar grep
input output 'dfs[a-z.]+'
```

8. Examine the output files:

Copy the output files from the distributed filesystem to the local filesystem and examine them:

```
$ bin/hdfs dfs -get output output
$ cat output/*
```

or

View the output files on the distributed filesystem:

```
$ bin/hdfs dfs -cat output/*
```

9. When you're done, stop the daemons with:

```
$ sbin/stop-dfs.sh
```

YARN on Single Node

You can run a MapReduce job on YARN in a pseudo-distributed mode by setting a few parameters and running ResourceManager daemon and NodeManager daemon in addition.

The following instructions assume that 1. ~ 4. steps of [the above instructions](#) are already executed.

1. Configure parameters as follows:

etc/hadoop/mapred-site.xml:

```
<configuration>
  <property>

    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

etc/hadoop/yarn-site.xml:

```
<configuration>
  <property>
    <name>yarn.nodemanager.aux-
services</name>

    <value>mapreduce_shuffle</value>
  </property>
```

</configuration>

2. Start ResourceManager daemon and NodeManager daemon:

```
$ sbin/start-yarn.sh
```

3. Browse the web interface for the ResourceManager; by default it is available at:
 - o ResourceManager -
<http://localhost:8088/>
4. Run a MapReduce job.
5. When you're done, stop the daemons with:

```
$ sbin/stop-yarn.sh
```

3.单机运行 mapreduce 任务

(1) 使用及查看 hdfs 文件

首先在 hdfs 上创建目录 input

```
same@same-xx:~/hadoop/hadoop-2.4.1$ bin/hdfs dfs -mkdir /input
```

从本机上传文件至 input 目录，这里的 xxx.txt 为本机任意文件

```
same@same-xx:~/hadoop/hadoop-2.4.1$ bin/hdfs dfs -copyFromLocal /home/same/hadoop/hadoop-2.4.1/xxx.txt /input
```

查看 input 目录

```
same@same-xx:~/hadoop/hadoop-2.4.1$ bin/hdfs dfs -ls /input
```

(2) 采用拟蒙特卡罗法估算圆周率 pi 的大小

.jar 文件在 share 目录下

```
same@same-xx:~/hadoop/hadoop-2.4.1$ bin/hadoop jar hadoop-mapreduce-examples-2.4.1.jar pi 10 100
```

语法: jar 表示后边的.jar 包，pi 后边的两个整型参数表示 Map Task 数目和拷贝样本数目

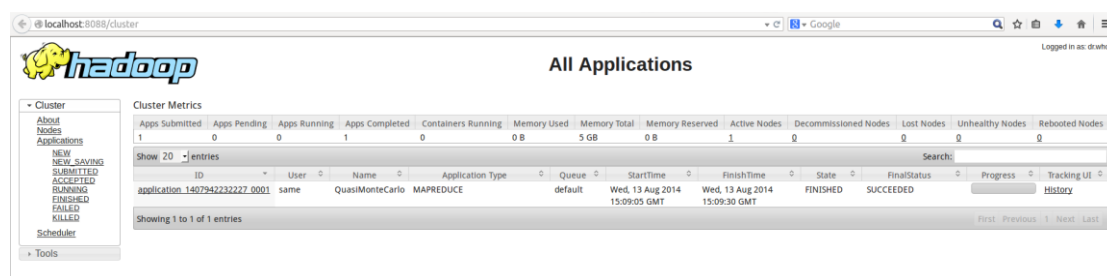
结果部分截图如下:

```
Reduce output records=0
Spilled Records=40
Shuffled Maps =10
Failed Shuffles=0
Merged Map outputs=10
GC time elapsed (ms)=715
CPU time spent (ms)=5910
Physical memory (bytes) snapshot=2478768128
Virtual memory (bytes) snapshot=21077688320
Total committed heap usage (bytes)=1662517248

Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0

File Input Format Counters
Bytes Read=1180
File Output Format Counters
Bytes Written=97
Job Finished in 27.6 seconds
Estimated value of Pi is 3.148000000000000000000000
same@same-xx:~/hadoop/hadoop-2.4.1$
```

相应的应用情况截图：



(3) 统计词频

应用上边创建的 input 文件夹，将 hadoop 的配置文件放其目录下

通过命令 `bin/hadoop fs -put etc/hadoop/* input` 将文件拷贝到分布式文件系统中，运行 `share` 目录下的示例程序进行单词统计

```
same@same-xx:~/hadoop/hadoop-2.4.1$ bin/hadoop fs -put etc/hadoop/* /input/
14/08/14 12:14:16 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform: using builtin-java classes where applicable
same@same-xx:~/hadoop/hadoop-2.4.1$
same@same-xx:~/hadoop/hadoop-2.4.1$ bin/hadoop jar hadoop-mapreduce-examples-2.4.1.jar grep /input /output 'dfs[a-z.]+'
14/08/14 12:17:20 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform: using builtin-java classes where applicable
```

```

14/08/14 12:17:33 INFO mapreduce.Job: map 7% reduce 0%
14/08/14 12:17:34 INFO mapreduce.Job: map 10% reduce 0%
14/08/14 12:17:35 INFO mapreduce.Job: map 14% reduce 0%
14/08/14 12:17:38 INFO mapreduce.Job: map 21% reduce 0%
14/08/14 12:17:39 INFO mapreduce.Job: map 24% reduce 0%
14/08/14 12:17:42 INFO mapreduce.Job: map 31% reduce 0%
14/08/14 12:17:43 INFO mapreduce.Job: map 34% reduce 0%
14/08/14 12:17:46 INFO mapreduce.Job: map 41% reduce 0%
14/08/14 12:17:48 INFO mapreduce.Job: map 45% reduce 0%
14/08/14 12:17:50 INFO mapreduce.Job: map 48% reduce 0%
14/08/14 12:17:52 INFO mapreduce.Job: map 52% reduce 0%
14/08/14 12:17:53 INFO mapreduce.Job: map 52% reduce 16%
14/08/14 12:17:54 INFO mapreduce.Job: map 55% reduce 16%
14/08/14 12:17:56 INFO mapreduce.Job: map 59% reduce 18%
14/08/14 12:17:58 INFO mapreduce.Job: map 62% reduce 18%
14/08/14 12:17:59 INFO mapreduce.Job: map 62% reduce 21%
14/08/14 12:18:00 INFO mapreduce.Job: map 66% reduce 21%
14/08/14 12:18:02 INFO mapreduce.Job: map 69% reduce 22%
14/08/14 12:18:04 INFO mapreduce.Job: map 72% reduce 22%
14/08/14 12:18:05 INFO mapreduce.Job: map 72% reduce 24%
14/08/14 12:18:06 INFO mapreduce.Job: map 76% reduce 24%
14/08/14 12:18:08 INFO mapreduce.Job: map 79% reduce 25%
14/08/14 12:18:10 INFO mapreduce.Job: map 83% reduce 25%
14/08/14 12:18:11 INFO mapreduce.Job: map 83% reduce 28%

```

```

14/08/14 12:18:40 INFO mapreduce.Job: Counters: 49
File System Counters
  FILE: Number of bytes read=428
  FILE: Number of bytes written=185651
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
HDFS: Number of bytes read=736
HDFS: Number of bytes written=294
HDFS: Number of read operations=7
HDFS: Number of large read operations=0
HDFS: Number of write operations=2
Job Counters
  Launched map tasks=1
  Launched reduce tasks=1
  Rack-local map tasks=1
  Total time spent by all maps in occupied slots (ms)=1609
  Total time spent by all reduces in occupied slots (ms)=1752
  Total time spent by all map tasks (ms)=1609
  Total time spent by all reduce tasks (ms)=1752
  Total vcore-seconds taken by all map tasks=1609
  Total vcore-seconds taken by all reduce tasks=1752
  Total megabyte-seconds taken by all map tasks=1647616
  Total megabyte-seconds taken by all reduce tasks=1794048

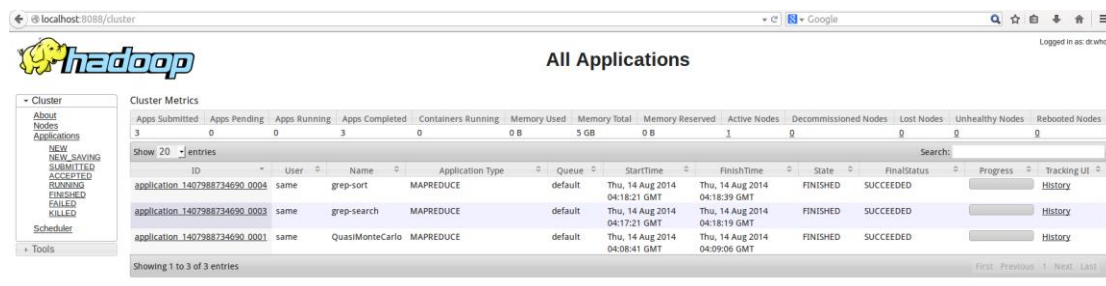
```

每个 shuffle 阶段统计

最后在 output 文件夹下可看到两个输出文件，分别是 log 和单词统计结果

```
same@same-xx:~/hadoop/hadoop-2.4.1$ bin/hdfs dfs -ls /output
14/08/14 12:21:53 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r-- 1 same supergroup 计结果 0 2014-08-14 12:18 /output/_SUCCESS
-rw-r--r-- 1 same supergroup 294 2014-08-14 12:18 /output/part-r-00000
same@same-xx:~/hadoop/hadoop-2.4.1$
```

任务运行情况：



The screenshot shows the Hadoop web interface at localhost:8088. The 'All Applications' page displays cluster metrics and a table of applications. The cluster metrics show 3 apps submitted, 0 pending, 3 running, and 0 completed. The applications table lists three tasks: 'grep-sort', 'grep-search', and 'QuasiMonteCarlo', all of which are in a 'FINISHED' state with a 'SUCCEEDED' final status.

Cluster Metrics													
Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes	
3	0	3	0	0	0 B	5 GB	0 B	1	0	0	0	0	

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI
application_1407988734690_0004	same	grep-sort	MAPREDUCE	default	Thu, 14 Aug 2014 04:18:21 GMT	Thu, 14 Aug 2014 04:18:39 GMT	FINISHED	SUCCEEDED		History
application_1407988734690_0003	same	grep-search	MAPREDUCE	default	Thu, 14 Aug 2014 04:17:21 GMT	Thu, 14 Aug 2014 04:18:19 GMT	FINISHED	SUCCEEDED		History
application_1407988734690_0001	same	QuasiMonteCarlo	MAPREDUCE	default	Thu, 14 Aug 2014 04:08:41 GMT	Thu, 14 Aug 2014 04:09:06 GMT	FINISHED	SUCCEEDED		History