车联网方案技术开发说明

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **修改人** | **修改日期** | **修改内容** |
| V0.9 | 王奇 | 20181222 | 建档 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[车联网方案技术开发说明 1](#_Toc536263432)

[1概述 3](#_Toc536263433)

[1.1架构设计图： 3](#_Toc536263434)

[1.2国内测试服服务器配置： 3](#_Toc536263435)

[2中间件技术实施 4](#_Toc536263436)

[2.1 Redis集群： 4](#_Toc536263437)

[2.1.1下载安装Redis 4](#_Toc536263438)

[2.1.2 测试Redis 4](#_Toc536263439)

[2.1.3 安装依赖包 5](#_Toc536263440)

[2.1.4 安装Ruby 5](#_Toc536263441)

[2.1.5安装Redis插件 6](#_Toc536263442)

[2.1.6 配置Redis 6](#_Toc536263443)

[3.1.7 创建集群 8](#_Toc536263444)

[3.1.8 给集群设置密码 9](#_Toc536263445)

[3.1.9 Redis使用方式 11](#_Toc536263446)

[2.2 ActiveMQ： 11](#_Toc536263447)

[2.2.1 ActiveMQ下载安装 11](#_Toc536263448)

[2.2.2 设置ActiveMQ服务开机启动 12](#_Toc536263449)

[2.2.3 设置账户密码 13](#_Toc536263450)

[2.3 ZooKeeper: 13](#_Toc536263451)

[2.4 Kafka： 13](#_Toc536263452)

[2.5消费端： 14](#_Toc536263453)

[2.6 Mysql: 14](#_Toc536263454)

[2.6.1 下载Mysql 14](#_Toc536263455)

[2.6.2 安装Mysql 14](#_Toc536263456)

[2.6.3 单机模式 14](#_Toc536263457)

[2.6.4 多实例模式 16](#_Toc536263458)

[2.6.5 Mysql主从： 19](#_Toc536263459)

[2.6.6 其它操作： 19](#_Toc536263460)

[2.7 Hadoop： 19](#_Toc536263461)

[2.8 HBase： 20](#_Toc536263462)

[下载 HBase： 20](#_Toc536263463)

[安装配置 HBase： 20](#_Toc536263464)

[3技术实施2 20](#_Toc536263465)

[3.1 Netty大并发网关 20](#_Toc536263466)

[3.2大并发测试系统 20](#_Toc536263467)

[4 技术实施3 20](#_Toc536263468)

[4.1 接入入口Nginx 20](#_Toc536263469)

[4.2 接入网关JavaAPI 20](#_Toc536263470)

[4.3 FTP服务器 20](#_Toc536263471)

[4.3.1 VSFTP服务器 20](#_Toc536263472)

[5支撑技术实施 21](#_Toc536263473)

[5.1操作系统优化 21](#_Toc536263474)

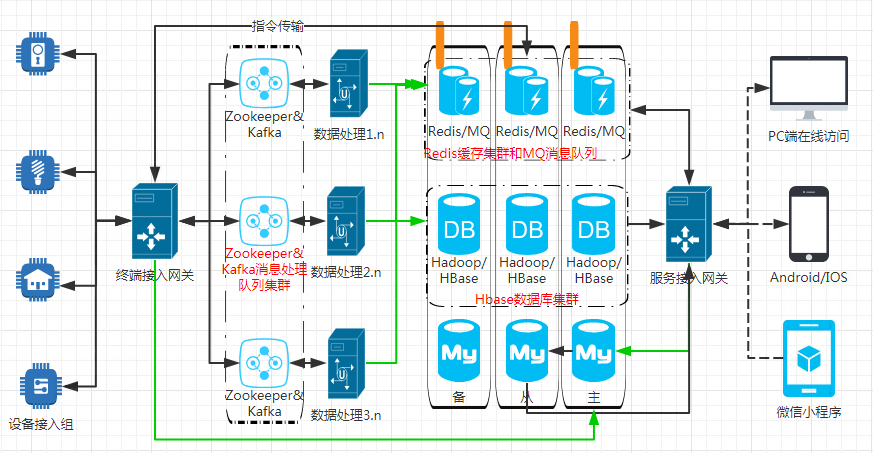
[5.2产品数据模拟系统 21](#_Toc536263475)

[5.3服务检测系统 21](#_Toc536263476)

[5.4运维脚本 21](#_Toc536263477)

# 1概述

## 1.1架构设计图：



## 1.2国内测试服服务器配置：

国内测试服为两台高性能双电源服务器。24核心48线程128G内存。服务器A采用RAID10磁盘阵列，增加稳定性的同时，提升硬盘的读写速度。为Kafka、HBase、Mysql提供高速IO读写能力。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | **服务器ServerA** | | | | **服务器ServerB** | | |
| **CPU /颗数/核数** | | E5-2680 v3 @ 2.50GHz/2/24(48\*) | | | | E5-2680 v3 @ 2.50GHz/2/24(48\*) | | |
| **内存** | | DDR4 128G | | | | DDR4 128G | | |
| **硬盘阵列/容量** | | RAID10 1.2T硬盘\*4/2.4T | | | | RAID1 1.2T硬盘\*2/1.2T | | |
| **操作系统** | | CentOS7.5 | | | | CentOS7.5 | | |
| **服务器IP** | | 113.106.93.247/106.3.226.247 | | | | 113.106.93.254/106.3.226.254 | | |
| **部署**  **软件** | Zookeeper1 | Kafka1 | 消息端1 | 消息端2 |  | Zookeeper3 | Kafka3 | 消息端5 |
| Zookeeper2 | Kafka2 | 消息端3 | 消息端4 |  |  |  | 消息端6 |
| HBase | Redis2000 | Redis2001 | Redis2002 |  | ifengstarIO | GT121IO |  |
|  | Redis2003 | Redis2004 | Redis2005 |  | HK-IO | GT52IO |  |
|  | Mysql8610 | Mysql3306 | Mysql3619 |  | Redis6379 | Nginx | Mysql8610 |
| Nginx | Tomcat | Tomcat | Tomcat | Tomcat | Tomcat | Tomcat | Tomcat |

# 2中间件技术实施

## 2.1 Redis集群：

|  |  |  |  |
| --- | --- | --- | --- |
| **端口** | **关系** | **部署位置** | |
| 2000 | 主  从 | ServerA: 113.106.93.247 | /home/deve/redis-cluster/2000 |
| 2003 | ServerA: 113.106.93.247 | /home/deve/redis-cluster/2001 |
| 2001 | 主  从 | ServerA: 113.106.93.247 | /home/deve/redis-cluster/2002 |
| 2004 | ServerA: 113.106.93.247 | /home/deve/redis-cluster/2003 |
| 2002 | 主  从 | ServerA: 113.106.93.247 | /home/deve/redis-cluster/2004 |
| 2005 | ServerA: 113.106.93.247 | /home/deve/redis-cluster/2005 |

### 2.1.1下载安装Redis

[deve@r730-104 ~]$ cd soft

[deve@r730-104 ~]$ wget http://download.redis.io/releases/redis-4.0.11.tar.gz

[deve@r730-104 ~]$ mkdir -p /home/deve/redis-cluster

[deve@r730-104 ~]$ tar -zxvf redis-4.0.11.tar.gz

[deve@r730-104 ~]$ cd redis-4.0.11.tar.gz

[deve@r730-104 ~]$ make PREFIX=/home/deve/redis-cluster/ install

[deve@r730-104 ~]$ cd /home/deve/redis-cluster

### 2.1.2 测试Redis

[deve@r730-104 ~]$ yum install gcc tcl

[deve@r730-104 ~]$ make test

### 2.1.3 安装依赖包

[deve@r730-104 ~]$ yum -y update

[deve@r730-104 ~]$ yum -y install wget

[deve@r730-104 ~]$ wget http://www.zlib.net/zlib-1.2.11.tar.gz

安装zlib

[deve@r730-104 ~]$ tar zxf zlib-1.2.11.tar.gz

[deve@r730-104 ~]$ cd zlib-1.2.11.tar.gz && ./configure --prefix=/usr/local/zlib

[deve@r730-104 ~]$ make && make install

更新动态连接库数据

[deve@r730-104 ~]$ echo "usr/local/zlib/lib" >> /etc/ld.so.conf

[deve@r730-104 ~]$ ldconfig -v

安装zlib-devel库

[deve@r730-104 ~]$ yum install zlib-devel

安装openssl

[deve@r730-104 ~]$ wget https://www.openssl.org/source/openssl-1.0.2q.tar.gz

[deve@r730-104 ~]$ tar -zxf openssl-1.0.2.q.tar.gz

[deve@r730-104 ~]$ cd openssl-1.0.2q

[deve@r730-104 ~]$ ./config -fPIC --prefix=/usr/local/openssl enable-shared

[deve@r730-104 ~]$ ./config -t

[deve@r730-104 ~]$ make && make install

[deve@r730-104 ~]$ ln -s /usr/local/openssl/bin/openssl /usr/bin/openssl

[deve@r730-104 ~]$ ln -s /usr/local/ssl/include/openssl /usr/include/openssl

[deve@r730-104 ~]$ echo "/usr/local/openssl/lib" >> /etc/ld.so.conf

[deve@r730-104 ~]$ openssl version

[deve@r730-104 ~]$ ldconfig -v

[deve@r730-104 ~]$ openssl version

### 2.1.4 安装Ruby

[deve@r730-104 ~]$ wget https://cache.ruby-lang.org/pub/ruby/2.5/ruby-2.5.3.tar.gz

[deve@r730-104 ~]$ tar -zxvf ruby-2.5.3.tar.gz

[deve@r730-104 ~]$ mkdir -p /usr/local/ruby

[deve@r730-104 ~]$ ./configure --prefix=/usr/local/ruby

[deve@r730-104 ~]$ make && make install

默认情况下，Ruby安装到/usr/local目录。如果想使用其它目录，可以把--prefix=DIR选项给./configure脚本

##[root@r730-105 bin]# ln -s /usr/local/ruby/bin/ruby /usr/bin/ruby

##[root@r730-105 bin]# ln -s /usr/local/ruby/bin/gem /usr/bin/gem

增加环境变量

[deve@r730-104 ~]$ vim /etc/profile

增加**RUBY\_HOME=/usr/local/ruby**

增加export PATH=$PATH:$JAVA\_HOME/bin:$HADOOP\_HOME/bin**:$RUBY\_HOME/bin**

保存 退出

[deve@r730-104 ~]$ source /etc/profile

查看版本

[deve@r730-104 ~]$ ruby -v

安装自带的openssl

[deve@r730-104 ~]$ cd /home/deve/soft/ruby-2.3.0/ext/openssl

[deve@r730-104 ~]$ ruby extconf.rb --with-openssl-include=/usr/local/openssl/include/ --with-openssl-lib=/usr/local/openssl/lib

[deve@r730-104 ~]$ ln -s /usr/local/src/ruby-2.2.3/include /

[deve@r730-104 ~]$ make

[deve@r730-104 ~]$ make install

### 2.1.5安装Redis插件

[deve@r730-104 ~]$ gem install redis

ERROR:  Loading command: install (LoadError)

    no such file to load -- zlib

ERROR:  While executing gem ... (NameError)

    uninitialized constant Gem::Commands::InstallCommand

进入ruby源码文件夹安装ruby自身提供的zlib包

[deve@r730-104 ~]$ cd ext/zlib

[deve@r730-104 ~]$ ruby ./extconf.rb

[deve@r730-104 ~]$ make

[deve@r730-104 ~]$ make install

make: \*\*\* No rule to make target `/include/ruby.h', needed by `zlib.o'. Stop.

vim /tools/ruby-2.5.1/ext/zlib/Makefile

:/ruby.h

#zlib.o: $(top\_srcdir)/include/ruby.h #把这一行替换成下面一行

zlib.o: ../../include/ruby.h

[deve@r730-104 ~]$ gem install redis

ERROR: While executing gem ... (Gem::Exception)

Unable to require openssl, install OpenSSL and rebuild Ruby (preferred) or use non-HTTPS sources

[root@r730-104 openssl]# make

compiling openssl\_missing.c

make: \*\*\* No rule to make target `/include/ruby.h', needed by `ossl.o'. Stop.

\*\*\*提示找不到/include/ruby.h 需要将ruby目录下的include做软连接到/\*\*\*

如果出现openssl错误 ，先查看 openssl version

如果没有安装则下载进行安装；如果已经是安装的，那么可能是Ruby安装不正确

gem install redis 成功后开始配置Redis单机文件

### 2.1.6 配置Redis

新建6个文件夹用来存放每一个节点数据。为了方便移植将Redis可执行文件拷贝到每一个文件夹。

[deve@r730-104 ~] cd /home/deve/redis-cluster

[deve@r730-104 ~] mkdir -p /home/deve/redis-cluster/2000

[deve@r730-104 ~] cp bin/redis-check-aof 2000/

[deve@r730-104 ~] cp bin/redis-cli 2000/

[deve@r730-104 ~] cp bin/redis-server 2000/

[deve@r730-104 ~] cp bin/redis-benchmark 2000/

[deve@r730-104 ~] cp bin/redis-check-rdb 2000/

[deve@r730-104 ~] cp bin/redis-trib.rb 2000/

[deve@r730-104 ~] cp bin/redis.conf 2000/

[deve@r730-104 ~] vim 2000/redis.conf

#----redis cluster------------------------------------------------------------------

bind 127.0.0.1 106.3.226.247 113.106.93.247

daemonize yes //redis后台运行

pidfile /var/run/redis\_2000.pid //pidfile文件对应2000,2001,2002,2003,2004,2005

port 2000 //端口2000,2001,2002,2003,2004,2005

cluster-enabled yes //开启集群 把注释#去掉

cluster-config-file nodes\_2000.conf //集群的配置 配置文件首次启动自动生成

cluster-node-timeout 5000 //请求超时 设置5秒够了

一般来说不需要开启appendonly aof日志文件太大，GPS应用对Redis数据，并没有非常高要求

appendonly no

#appendonly yes //aof日志开启 有需要就开启，它会每次写操作都记录一条日志

dir /home/developer/redis-cluster/2000/

logfile "/home/developer/redis-cluster/2000/redis-2000.log"

#-----------------------------------------------------------------------------------

[deve@r730-104 ~]$ cp -R 2000 2001

[deve@r730-104 ~]$ cp -R 2000 2002

[deve@r730-104 ~]$ cp -R 2000 2003

[deve@r730-104 ~]$ cp -R 2000 2004

[deve@r730-104 ~]$ cp -R 2000 2005

修改相应的redis.conf

[deve@r730-104 ~]$ cp bin/redis-trib.rb

编写启动脚本

[deve@r730-104 ~]$ vim start-all.sh

cd 2000

./redis-server redis.conf

cd ../2001

./redis-server redis.conf

cd ../2002

./redis-server redis.conf

cd ../2003

./redis-server redis.conf

cd ../2004

./redis-server redis.conf

cd ../2005

./redis-server redis.conf

关闭脚本

[deve@r730-104 ~]$ vim stop-all.sh

2000/redis-cli -h 172.0.0.1 -p 2000 shutdown

2001/redis-cli -h 172.0.0.1 -p 2001 shutdown

2002/redis-cli -h 172. 0.0.1 -p 2002 shutdown

2003/redis-cli -h 172. 0.0.1 -p 2003 shutdown

2004/redis-cli -h 172. 0.0.1 -p 2004 shutdown

2005/redis-cli -h 172. 0.0.1 -p 2005 shutdown

如果有需要需要使用redis-cli -h 172.0.0.1 -p 2000 -a youpassword shutdown

[deve@r730-104 ~]$ chmod 777 start-all.sh

[deve@r730-104 ~]$ chmod 777 stop-all.sh

启动所有Redis

[deve@r730-104 ~]$ ./ start-all.sh

[deve@r730-104 ~]$ ps -ef | grep redis

观察子目录下的redis-XXXX.log

### 3.1.7 创建集群

[deve@r730-104 ~]$ ./redis-trib.rb create --replicas 1 172.16.18.104:2000 172.16.18.104:2001 172.16.18.104:2002 172.16.18.104:2003 172.16.18.104:2004 172.16.18.104:2005

./redis-trib.rb create --replicas 1 106.3.226.247:2000 106.3.226.247:2001 106.3.226.247:2002 106.3.226.247:2003 106.3.226.247:2004 106.3.226.247:2005

>>> Creating cluster

>>> Performing hash slots allocation on 6 nodes...

Using 3 masters:

127.0.0.1:2000

127.0.0.1:2001

127.0.0.1:2002

Adding replica 127.0.0.1:2004 to 127.0.0.1:2000

Adding replica 127.0.0.1:2005 to 127.0.0.1:2001

Adding replica 127.0.0.1:2003 to 127.0.0.1:2002

>>> Trying to optimize slaves allocation for anti-affinity

[WARNING] Some slaves are in the same host as their master

M: d268ffb7f284ff22ad3d9d9877d89489f4f082a4 127.0.0.1:2000

slots:0-5460 (5461 slots) master

M: c576b1175b4035787cd087e8efa2ec8c33fe3352 127.0.0.1:2001

slots:5461-10922 (5462 slots) master

M: 9a0ef38514b5e9114687501e3fe84f7ac183c021 127.0.0.1:2002

slots:10923-16383 (5461 slots) master

S: 4bbee759b49d68ce7dc104a61bdbb05776c718b2 127.0.0.1:2003

replicates d268ffb7f284ff22ad3d9d9877d89489f4f082a4

S: a7fa12ff5dbeb8cabedfda8621b3a4708e119557 127.0.0.1:2004

replicates c576b1175b4035787cd087e8efa2ec8c33fe3352

S: 38300c40240cd80e9eb5e2f1d6cc556a6722f324 127.0.0.1:2005

replicates 9a0ef38514b5e9114687501e3fe84f7ac183c021

Can I set the above configuration? (type 'yes' to accept): yes

>>> Nodes configuration updated

>>> Assign a different config epoch to each node

>>> Sending CLUSTER MEET messages to join the cluster

Waiting for the cluster to join....

>>> Performing Cluster Check (using node 127.0.0.1:2000)

M: d268ffb7f284ff22ad3d9d9877d89489f4f082a4 127.0.0.1:2000

slots:0-5460 (5461 slots) master

1 additional replica(s)

S: a7fa12ff5dbeb8cabedfda8621b3a4708e119557 127.0.0.1:2004

slots: (0 slots) slave

replicates c576b1175b4035787cd087e8efa2ec8c33fe3352

S: 38300c40240cd80e9eb5e2f1d6cc556a6722f324 127.0.0.1:2005

slots: (0 slots) slave

replicates 9a0ef38514b5e9114687501e3fe84f7ac183c021

S: 4bbee759b49d68ce7dc104a61bdbb05776c718b2 127.0.0.1:2003

slots: (0 slots) slave

replicates d268ffb7f284ff22ad3d9d9877d89489f4f082a4

M: c576b1175b4035787cd087e8efa2ec8c33fe3352 127.0.0.1:2001

slots:5461-10922 (5462 slots) master

1 additional replica(s)

M: 9a0ef38514b5e9114687501e3fe84f7ac183c021 127.0.0.1:2002

slots:10923-16383 (5461 slots) master

1 additional replica(s)

[OK] All nodes agree about slots configuration.

>>> Check for open slots...

>>> Check slots coverage...

[OK] All 16384 slots covered.

[deve@r730-104 ~]$

### 3.1.8 给集群设置密码

使用 ./stop-all.sh关闭集群，修改每个节点的配置文件

#----redis cluster------------------------------------------------------------------

bind 127.0.0.1 106.3.226.247 113.106.93.247

daemonize yes //redis后台运行

pidfile /var/run/redis\_2000.pid //pidfile文件对应2000,2001,2002,2003,2004,2005

port 2000 //端口2000,2001,2002,2003,2004,2005

cluster-enabled yes //开启集群 把注释#去掉

cluster-config-file nodes\_2000.conf //集群的配置 配置文件首次启动自动生成

cluster-node-timeout 5000 //请求超时 设置5秒够了

一般来说不需要开启appendonly aof日志文件太大，GPS应用对Redis数据，并没有非常高要求

appendonly no

#appendonly yes //aof日志开启 有需要就开启，它会每次写操作都记录一条日志

dir /home/developer/redis-cluster/2000/

masterauth youpassword //如果需要密码访问Redis请设置密码

requirepass hkredispassword1test

logfile "/home/developer/redis-cluster/2000/redis-2000.log"

#-----------------------------------------------------------------------------------

设置密码之后如果需要使用redis-trib.rb的各种命令

如：./redis-trib.rb check 127.0.0.1:2000，则会报错ERR] Sorry, can’t connect to node 127.0.0.1:2000

解决办法：/usr/local/ruby/lib/ruby/gems/2.5.0/gems/redis-4.0.3/lib/redis/client.rb，然后修改passord

find / -name client.rb

vim /usr/local/ruby/lib/ruby/gems/2.5.0/gems/redis-4.0.3/lib/redis/client.rb

class Client

DEFAULTS = {

:url => lambda { ENV["REDIS\_URL"] },

:scheme => "redis",

:host => "127.0.0.1",

:port => 6379,

:path => nil,

:timeout => 5.0,

:password => "hkredispassword1test",

:db => 0,

:driver => nil,

:id => nil,

:tcp\_keepalive => 0,

:reconnect\_attempts => 1,

:inherit\_socket => false

}

带密码访问集群

./redis-cli -h 127.0.0.1 -c -p 2000

2000/redis-cli -c -p 2000

auth hkredispassword1test

./redis-cli -h 127.0.0.1 -c -p 2000 -a hkredispassword1test

./redis-cli -h 172.16.18.104 -c -p 2000

auth hkredispassword1test

去除密码后出现：

[developer@r730-104 redis-cluster]$ ./redis-trib.rb create --replicas 1 172.16.18.104:2000 172.16.18.104:2001 172.16.18.104:2002 172.16.18.104:2003 172.16.18.104:2004 172.16.18.104:2005

>>> Creating cluster

[ERR] Sorry, can't connect to node 172.16.18.104:2000

vim /usr/local/ruby/lib/ruby/gems/2.5.0/gems/redis-4.0.3/lib/redis/client.rb

class Client

DEFAULTS = {

:url => lambda { ENV["REDIS\_URL"] },

:scheme => "redis",

:host => "127.16.18.104",

:port => 6379,

:path => nil,

:timeout => 5.0,

:password => nil,

#:password => "hkredispassword1test",

:db => 0,

:driver => nil,

:id => nil,

:tcp\_keepalive => 0,

:reconnect\_attempts => 1,

:inherit\_socket => false

}

屏蔽password

备注：集群配置时最好只使用一个IP地址。

### 3.1.9 Redis使用方式

## 2.2 ActiveMQ：

|  |  |  |  |
| --- | --- | --- | --- |
| **端口** | **部署位置** | **服务对象** | **Topic路径** |
| ServerA:61616 | /home/deve/module/apache-activemq-5.15.3 |  |  |
| ServerA:8161 |  |  |
| bs.ifengstar.com:61616 | /home/deve/module/apache-activemq-5.15.3 |  |  |
| bs.ifengstar.com: 8161 |  |  |

ActiveMQ是开源组织apache.org旗的一款开源消息中间件。访问网址为：

http://activemq.apache.org/activemq-5158-release.html

### 2.2.1 ActiveMQ下载安装

[deve@r730-104 soft]$ wget http://mirror.bit.edu.cn/apache//activemq/5.15.8/apache-activemq-5.15.8-bin.tar.gz

[deve@r730-104 soft]$ tar -zvxf a apache-activemq-5.15.8-bin.tar.gz -C /home/deve/module/activemq

**启动ActiveMQ**

[root@r730-104 activemq]# /home/deve/module/activemq/bin/activemq start

INFO: Loading '/home/deve/module/activemq//bin/env'

INFO: Using java '/usr/local/jdk1.8.0\_191/bin/java'

INFO: Starting - inspect logfiles specified in logging.properties and log4j.properties to get details

INFO: pidfile created : '/home/deve/module/activemq//data/activemq.pid' (pid '17206')

**查看ActiveMQ状态**

[root@r730-104 activemq]# /home/deve/module/activemq/bin/activemq status

INFO: Loading '/home/deve/module/activemq//bin/env'

INFO: Using java '/usr/local/jdk1.8.0\_191/bin/java'

ActiveMQ not running

**停止ActiveMQ运行**

[root@r730-104 activemq]# /home/deve/module/activemq/bin/activemq stop

INFO: Loading '/home/deve/module/activemq//bin/env'

INFO: Using java '/usr/local/jdk1.8.0\_191/bin/java'

ERROR: No or outdated process id in '/home/deve/module/activemq//data/activemq.pid'

INFO: Removing /home/deve/module/activemq//data/activemq.pid

### 2.2.2 设置ActiveMQ服务开机启动

* 建立软连接：

[root@r730-104 activemq]# ln -s /home/deve/module/activemq/bin/activemq /etc/init.d/activemqd

* 注册为系统服务

[root@r730-104 activemq]# vim /etc/init.d/activemqd

- 添加下面内容到/etc/init.d/activemq脚本

# chkconfig: 345 63 37

# description: Auto start ActiveMQ

JAVA\_HOME=/usr/local/jdk1.8.0\_191

JAVA\_CMD=java1234

\*\*/usr/local/jdk1.8.0\_191/bin/env增加ACTIVEMQ\_HOME,JAVA\_HOME

\*\*echo $PATH

\*\* 增加PATH=

​ export PATH 代替JAVA\_HOME

\*\*\*重复第一步

---------------------------------------示例-----------------------------------------------------------------------------------

!/bin/sh

description: Auto start ActiveMQ

BEGIN INIT INFO

Provides: activemq

Required-Start: remote\_fs network $syslog

Required-Stop: remote\_fs network $syslog

Default-Start: 2 3 4 5

Default-Stop: 0 6

chkconfig: 2345 64 36

Short-Description: Starts ActiveMQ

Description: Starts ActiveMQ Message Broker Server

END INIT INFO

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/usr/java/jdk1.8.0\_191/bin:/usr/java/jdk1.8.0\_191/jre/bin:/root/bin

export PATH

JAVA\_CMD=java1234

ACTICVEMQ\_HOME=/usr/local/activemq-5.15.3

------------------------------------------------------------------------

* 开启开机自启

[root@r730-104 activemq]# chkconfig activemq on

[root@r730-104 activemq]# reboot

* 以系统服务的方式启动、查看状态和停止服务

[root@r730-104 activemq]# service activemq start

[root@r730-104 activemq]# service activemq status

[root@r730-104 activemq]# service activemq stop

### 2.2.3 设置账户密码

修改activemq.xml

<!-- destroy the spring context on shutdown to stop jetty -->

<shutdownHooks>

<bean xmlns="http://www.springframework.org/schema/beans" class="org.apache.activemq.hooks.SpringContextHook" />

</shutdownHooks>

<!-- 添加访问ActiveMQ的账号密码 -->

​ <plugins>

​ <simpleAuthenticationPlugin>

​ <users>

​ <authenticationUser username="hkadmin" password="hk667" groups="users,admins"/>

​ </users>

​ </simpleAuthenticationPlugin>

​ </plugins>

## 2.3 ZooKeeper:

|  |  |  |
| --- | --- | --- |
| **IP及端口** | **部署位置** |  |
| ServerA:2884:3884 | A:/home/deve/module/zookeeper1 |  |
| ServerA:2883:3883 | A:/home/deve/module/zookeeper2 |  |
| 106.3.226.254:2882:3882 | B:/home/deve/module/zookeeper3 |  |
|  |  |  |

## 2.4 Kafka：

ServerA: 106.3.226.247

|  |  |  |
| --- | --- | --- |
| **IP及端口** | **部署位置** |  |
| ServerA:2181 | A:/home/deve/module/kafka1 |  |
| ServerA:2182 | A:/home/deve/module/kafka2 |  |
| 106.3.226.254:2183 | B: /home/deve/module/kafka3 |  |
|  |  |  |

## 2.5消费端：

|  |  |  |
| --- | --- | --- |
| **IP及端口** | **部署位置** |  |
| ServerA:2181 | /home/deve/app/ifengConsumer |  |
| ServerA:2182 | /home/deve/app/ifengConsumer |  |
|  | /home/deve/app/ifengConsumer |  |
|  | /home/deve/app/ifengConsumer |  |
|  | /home/deve/app/ifengConsumer |  |
|  | /home/deve/app/ifengConsumer |  |

## 2.6 Mysql:

|  |  |  |  |
| --- | --- | --- | --- |
| **端口** | **类型** | **部署地址** |  |
| ServerA:3619 | 单 | /home/mysql/3619 |  |
| ServerA:8610 | 主 | /home/mysql/8610 |  |
| ServerA:3306 | 从 | /home/mysql/3306 |  |
| ServerB:8610 | 单 | /home/mysql/8610 |  |

### 2.6.1 下载Mysql

wget https://cdn.mysql.com//Downloads/MySQL-5.7/mysql-5.7.24-linux-glibc2.12-x86\_64.tar.gz

### 2.6.2 安装Mysql

* 解压：tar -zxvf mysql-5.7.24-linux-glibc2.12-x86\_64.tar.gz -C /usr/local
* 增加环境变量：

echo 'export PATH=$PATH:/usr/local/mysql/bin' >> /etc/profile

source /etc/profile

* 为centos添加mysql用户组和mysql用户(-s /bin/false参数指定mysql用户仅拥有所有权，而没有登录权限):
* groupadd mysql
* useradd -r -g mysql -s /bin/false mysql2.6.3 配置Mysql
* 进入安装mysql软件的目录： cd /usr/local/mysql
* 修改当前目录拥有者为新建的mysql用户： chown -R mysql:mysql ./

### 2.6.3 单机模式

* 1、修改my.cnf

|  |
| --- |
| vim /etc/my.conf  #-------my.cnf-----------#  # For advice on how to change settings please see  # http://dev.mysql.com/doc/refman/5.7/en/server-configuration-defaults.html  [mysqld]  #  # Remove leading # and set to the amount of RAM for the most important data  # cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.  # innodb\_buffer\_pool\_size = 128M  #  # Remove leading # to turn on a very important data integrity option: logging  # changes to the binary log between backups.  # log\_bin  #  # Remove leading # to set options mainly useful for reporting servers.  # The server defaults are faster for transactions and fast SELECTs.  # Adjust sizes as needed, experiment to find the optimal values.  # join\_buffer\_size = 128M  # sort\_buffer\_size = 2M  # read\_rnd\_buffer\_size = 2M  #datadir=/var/lib/mysql  user=mysql  datadir=/home/mysql/8610/mysql  port=8610  #socket=/var/lib/mysql/mysql.sock  socket=/home/mysql/8610/mysql.sock  # Disabling symbolic-links is recommended to prevent assorted security risks  symbolic-links=0  #log-error=/var/log/mysqld.log  log-error=/home/mysql/8610/mysqld.log  sql\_mode=STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_AUTO\_CREATE\_USER,NO\_ENGINE\_SUBSTITUTION  pid-file=/home/mysql/8610/mysqld.pid  [client]  socket=/home/mysql/8610/mysql.sock  #--------end my.cnf----------------# |

* 2、在mysql用户目录下建立数据文件夹

cd /home/mysql/ && mkdir -p 8610

chown mysql:mysql 8610

chmod 777 8610

cd /home/mysql/8610/ && mkdir mysql

chown mysql:mysql mysql

chmod 777 mysql

touch mysqld.log

chown mysql:mysql mysqld.log

chmod 777 mysqld.log

* 3、初始化数据库

cd /usr/local/mysql

bin/mysqld --initialize --user=mysql --basedir=/usr/local/mysql --datadir=/home/mysql/8610/mysql/

在初始化后需要记录root用户的随机密码。如果命令行没有给出，请查阅日志文件。

* 4、开启mysql服务，命令如下：

./support-files/mysql.server start

* 5、将mysql进程放入系统进程中，命令如下：

cp support-files/mysql.server /etc/init.d/mysqld

* 6、重新启动mysql服务，命令如下：

service mysqld restart

* 7、使用随机密码登录mysql数据库，命令如下：

mysql -u root -p

等待系统提示，输入随机密码，即可登录

* 8、进入mysql操作行，为root用户设置新密码（fhxt&clw715#）：

alter user 'root'@'localhost' identified by 'fhxt&clw715#';

* 9、设置允许远程连接数据库，命令如下：

update user set user.Host='%' where user.User='root';

* 10、新建用户并设置远程访问权限

#grant all privileges on \*.\* to gpsadmin@localhost identified by "1qaz&619" ;

grant all privileges on \*.\* to gpsadmin@"%" identified by "1qaz&619" ;

select host,user from user; 　　//查询mysql中所有用户权限

* 11、刷新权限，命令如下：

flush privileges;

### 2.6.4 多实例模式

* 1、修改my.cnf

|  |
| --- |
| vim /etc/my.conf  #-------my.cnf-----------#  vim /etc/my.conf  [client]  port=3306  socket=/tmp/mysql.sock  [mysqld\_multi]  mysqld = /usr/local/mysql/bin/mysqld\_safe  mysqladmin = /usr/local/mysql/bin/mysqladmin  log = /home/mysql/mysqld\_multi.log  [mysqld]  user=mysql  basedir = /usr/local/mysql  sql\_mode=NO\_ENGINE\_SUBSTITUTION,STRICT\_TRANS\_TABLES  [mysqld3306]  mysqld=mysqld  mysqladmin=mysqladmin  datadir=/home/mysql/mysql-3306/mysql  port=3306  server\_id=3306  socket=/tmp/mysql-3306.sock  log-output=file  slow\_query\_log = 1  long\_query\_time = 1  slow\_query\_log\_file = /home/mysql/mysql-3306/log/slow.log  log-error = /home/mysql/mysql-3306/log/error.log  log-bin = /home/mysql/mysql-3306/log/mysql3306-bin  binlog-ignore-db = mysql  [mysqld8610]  mysqld=mysqld  mysqladmin=mysqladmin  datadir=/home/mysql/mysql-8610/mysql  port=8610  server\_id=8610  socket=/tmp/mysql-8610.sock  log-output=file  slow\_query\_log = 1  long\_query\_time = 1  slow\_query\_log\_file = /home/mysql/mysql-8610/log/slow.log  log-error = /home/mysql/mysql-8610/log/error.log  log-bin = /home/mysql/mysql-8610/log/mysql8610-bin  replicate-ignore-db=mysql  relay-log = slave-relay-bin  relay-log-index = slave-relay-bin.index  read\_only  #--------end my.cnf----------------# |

* 2、在mysql用户目录下建立数据文件夹

|  |
| --- |
| cd /home/mysql/ && mkdir -p 8610  chown mysql:mysql 8610  chmod 777 8610  cd /home/mysql/8610/ && mkdir mysql  chown mysql:mysql mysql  chmod 777 mysql  touch mysqld.log  chown mysql:mysql mysqld.log  chmod 777 mysqld.log |
| cd /home/mysql/ && mkdir -p 3619  chown mysql:mysql 3619  chmod 777 3619  cd /home/mysql/3619/ && mkdir mysql  chown mysql:mysql mysql  chmod 777 mysql  touch mysqld.log  chown mysql:mysql mysqld.log  chmod 777 mysqld.log |
| cd /home/mysql/ && mkdir -p 3306  chown mysql:mysql 3306  chmod 777 3306  cd /home/mysql/3306/ && mkdir mysql  chown mysql:mysql mysql  chmod 777 mysql  touch mysqld.log  chown mysql:mysql mysqld.log  chmod 777 mysqld.log |

* 3、将mysql进程放入系统进程中，命令如下：

cp support-files/mysql.server /etc/init.d/mysqld

* 4、初始化数据库

cd /usr/local/mysql

|  |
| --- |
| mysqld --initialize --user=mysql --basedir=/usr/local/mysql --datadir=/home/mysql/3306/mysql/ |
| mysqld --initialize --user=mysql --basedir=/usr/local/mysql --datadir=/home/mysql/3916/mysql/ |
| mysqld --initialize --user=mysql --basedir=/usr/local/mysql --datadir=/home/mysql/8610/mysql/ |

*在初始化后需要记录root用户的随机密码。如果命令行没有给出，请查阅日志文件。*

* 5、启动mysql集群，命令如下：

/usr/local/mysql/bin/mysqld\_multi start

* 6、查看集群：

/usr/local/mysql/bin/mysqld\_multi report

Reporting MySQL servers

MySQL server from group: mysqld3306 is running

MySQL server from group: mysqld8610 is running

[root@r430-102 mysql]# ss -tupln | grep mysqld

tcp LISTEN 0 80 :::8610 :::\* users:(("mysqld",pid=93909,fd=23))

tcp LISTEN 0 80 :::3306 :::\* users:(("mysqld",pid=93906,fd=23))

* 7、进入mysql操作行，为root用户设置新密码（fhxt&clw715#）：

查看单实例：/usr/local/mysql/bin/mysqld\_multi report 3306

查看监听端口：ss -tulpn | grep mysqld

修改密码：

mysql -S /tmp/mysql-3306.sock -p

set password=password('123456');

set password=password('fhxt&clw715#');

* 9、设置允许远程连接数据库，命令如下：

update user set user.Host='%' where user.User='root';

* 10、新建用户并设置远程访问权限

#grant all privileges on \*.\* to gpsadmin@localhost identified by "1qaz&619" ;

grant all privileges on \*.\* to gpsadmin@"%" identified by "1qaz&619" ;

select host,user from user; 　　//查询mysql中所有用户权限

* 11、刷新权限，命令如下：

flush privileges;

### 2.6.5 Mysql主从：

### 2.6.6 其它操作：

查询数据库中的存储过程

方法一：select `name` from mysql.proc where db = 'your\_db\_name' and `type` = 'PROCEDURE'

方法二：show procedure status;

查看存储过程或函数的创建代码

show create procedure proc\_name;

show create function func\_name;

## 2.7 Hadoop：

目前暂时单接点HBase，没有架设Hadoop

|  |  |  |  |
| --- | --- | --- | --- |
| **端口** | **类型** | **部署地址** |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## 2.8 HBase：

|  |  |  |  |
| --- | --- | --- | --- |
| **端口** | **类型** | **部署地址** |  |
| ServerA:启动时随机 | 单点 | /home/deve/hbase |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

### 下载 HBase：

wget http://mirrors.hust.edu.cn/apache/hbase/2.1.2/hbase-2.1.2-bin.tar.gz

### 安装配置 HBase：

# 3技术实施2

## 3.1 Netty大并发网关

## 3.2大并发测试系统

# 4 技术实施3

## 4.1 接入入口Nginx

## 4.2 接入网关JavaAPI

## 4.3 FTP服务器

### 4.3.1 VSFTP服务器

查看是否已经安装VSFTPD

rpm -qa | grep vsftpd

安装Vsftpd

yum -y install vsftpd

设置开机自动启动

chkconfig vsftpd on

service vsftpd on

# 5支撑技术实施

## 5.1操作系统优化

## 5.2产品数据模拟系统

## 5.3服务检测系统

## 5.4运维脚本