部 署 文 档 v1.0

*** 文档中红色部分请根据实际情况替换**

• Docker 安装

1. 安装docker：

```
yum –y install docker–io
```

2. 启动docker服务：

```
systemctl start docker.service
```

3. 将docker服务加入到开机启动项：

```
systemctl enable docker.service
```

• FastDFS集群（先安装docker） – 安装脚本目录FastDFS

1. 修改文件mod_fastdfs.conf（修改为计划安装为tracker服务器的公网IP和端口）

```
tracker_server=47.94.56.181:22522
```

2. 修改文件nginx–tracker.conf（修改为计划安装为storage服务器的内网IP和端口）

```
upstream group1 {
        ip_hash;
        server 10.31.145.144:3001 weight=1;
        server 10.144.113.82:3001 weight=1;
}
```

和

```
location ^~ /group1/ {
        proxy_pass http://group1;
        client_max_body_size 1024m;
}
```

3. 修改文件storage.conf（修改为计划安装为tracker服务器的公网IP和端口）

```
tracker_server=47.94.56.181:22522
```

4. 上传FastDFS目录到服务器
5. 安装fastdfs镜像（大概需要10分钟）：

```
docker build -t fastdfs --rm=true .
```

6. **安装：** tracker

```
docker run -d --name tracker1 -v ~/tracker/data01:/fastdfs/tracker/data --net=host
-e TR_PORT=22522 -e TR_NGX_PORT=3001 55ceb2e98eb2 tracker
```

7. **安装：** storage

```
docker run -d --name storage1 -v ~/storage/data01:/fastdfs/storage/data -v ~/
storage/store_path01:/fastdfs/store_path --net=host -e ST_PORT=23001 -e
ST_NGX_PORT=3001 -e GROUP_NAME=group1 55ceb2e98eb2 storage
```

- **Redis & Sentinel集群（不需要docker）**

```
wget http://download.redis.io/redis-stable.tar.gz
tar -xvzf redis-stable.tar.gz
cd redis-stable
make
make install
```

- **Redis**

```
cp ~/redis-stable/redis.conf /etc/redis.conf
vim /etc/redis.conf
```

查找并修改

```
daemonize >>> yes
bind >>> 本机的内网IP + 空格 + 127.0.0.1
requirepass >>> redis密码
```

```
masterauth >>> redids密码
```

* 如果是slave机器，还需要配置

```
slaveof master–IP master–PASS
```

配置完成后启动redis（先启动master）

```
/usr/local/bin/redis–server /etc/redis.conf
```

- **Sentinel**

```
cp ~/redis–stable/sentinel.conf /etc/sentinel.conf

vim /etc/sentinel.conf
```

查找并修改或添加

```
protected–mode >>> no

daemonize >>> yes

sentinel monitor [Cluster Name] [Redis Master–IP] [Redis Master–PORT] 2

sentinel down–after–milliseconds [Cluster Name] 5000

sentinel failover–timeout [Cluster Name] 10000

sentinel auth–pass [Cluster Name] [Redis PASS]
```

配置完成后启动Sentinel（启动Sentinel之前先将Redis的所有节点启动）

```
/usr/local/bin/redis–sentinel /etc/sentinel.conf
```

* 建议一个Redis节点伴随一个Sentinel

- **MySQL集群（不需要docker） – 安装脚本目录MYSQL**

上传mysql–community–release–el6–5.noarch.rpm文件至服务器

执行命令安装：

```
yum localinstall mysql–community–release–el6–5.noarch.rpm

yum install mysql–community–server
```

- **基本设置（Master & Slave都需要）**

vim /etc/my.cnf

```
[mysqld]

character_set_server=utf8

default-time-zone='+8:00'

max_connections=1024


[mysql]

default-character-set=utf8
```

启动MySQL并加入开机启动项

```
systemctl start mysqld.service

systemctl enable mysqld.service
```

命令行连接MySQL：mysql -u root

```
select user,host,password from mysql.user;

-- 将user不为空的用户密码全部设置一遍

set password for root@'localhost'=password('Inspeeding123456');

...

-- 删除user为空的记录

delete from mysql.user where user='';

grant all privileges on *.* to root@"%" identified by 'Inspeeding123456' with grant
option;

flush privileges;

quit;
```

将两台MySQL单实例先启动后配置集群

- **Node Master配置**

vim /etc/my.cnf

```
[mysqld]

server_id=1（两个节点不能一样）
```

```
binlog-ignore-db=mysql

log-bin=sibosen-bin

binlog_cache_size=1M

binlog_format=mixed

expire_logs_days=7

slave_skip_errors=1062

relay_log=sibosen-relay-bin

log_slave_updates=1

auto_increment_increment=2

auto_increment_offset=1
```

重启MySQL

```
systemctl restart mysqld.service
```

配置Slave用户，命令行登录MySQL：mysql -u root -p

```
grant replication slave, replication client on *.* to 'repl'@'slave ip' identified by

'password';

flush privileges;

quit;
```

- **Node Slave配置**

命令行登录MySQL：mysql -u root -p

```
change master to master_host='master ip',master_user='user for slave(ext. repl)',

master_password='password', master_port=3306, master_log_file='sibosen-bin.

000004', master_log_pos=439, master_connect_retry=30;

quit;
```

其中master_log_file、master_log_pos通过以下命令在master机器上查看：

```
show master status;
```

启动slave，命令行登录MySQL：mysql -u root -p

```
start slave;
-- 查看状态
show slave status\G;
quit;
```

- **FTP Server（不需要docker）**

1. 软件安装

```
yum -y install vsftpd
```

2. 安装完成后，关闭匿名用户及添加限制端口范围

```
vim /etc/vsftpd/vsftpd.conf
> anonymous_enable=NO
> chroot_local_user=YES
>
> allow_writeable_chroot=YES
> pasv_enable=YES
> pasv_min_port=60000
> pasv_max_port=62000
```

3. 创建FTP虚拟宿主帐户

```
mkdir /opt/ftp
useradd -d /opt/ftp/ryxx -g ftp -s /sbin/nologin ryxx
passwd ryxx
chown -R ryxx /opt/ftp
chown -R 777 /opt/ftp
mkdir /opt/ftp/ryxx/out2in
mkdir /opt/ftp/ryxx/in2out
cd /opt/ftp
chmod -R 777 *
```

4. 启动FTP服务并添加开机自启动

```
systemctl start vsftpd.service

systemctl enable vsftpd.service
```

- **Zookeeper集群 – 至少需要三个节点（不需要docker）**

- **安装Java环境**

```
yum -y install java-1.8.0-openjdk*
```

- 安装Zookeeper

```
wget http://mirror.bit.edu.cn/apache/zookeeper/zookeeper-3.4.10/

zookeeper-3.4.10.tar.gz

tar -zxvf zookeeper-3.4.10.tar.gz -C /opt/

mkdir /opt/zoologs

mkdir /opt/zoostorage

cp /opt/zookeeper-3.4.10/conf/zoo_sample.cfg /opt/zookeeper-3.4.10/conf/zoo.cfg
```

- **配置Zookeeper集群**

设置severid

```
echo 1 > /opt/zoostorage/myid
```

修改配置文件：vim /opt/zookeeper-3.4.10/conf/zoo.cfg

```
dataDir=/opt/zoostorage

dataLogDir=/opt/zoologs

maxClientCnxns=1024

server.1=10.144.113.56:2888:3888

server.2=10.144.113.48:2888:3888

server.3=10.31.151.165:2888:3888
```

* 其中1、2、3对应echo的serverid（int类型）

- **依次启动Zookeeper**

```
/opt/zookeeper-3.4.10/bin/zkServer.sh start
```

- Kafka（不需要docker）

- **安装Java环境**

yum -y install java-1.8.0-openjdk*

安装Kafka

```
wget http://apache.fayea.com/kafka/0.11.0.0/kafka_2.11-0.11.0.0.tgz
tar -zxvf kafka_2.11-0.11.0.0.tgz -C /opt/
```

- **配置Kafka**

vim /opt/kafka_2.11-0.11.0.0/config/server.properties

```
delete.topic.enable=true
log.dirs=/opt/kaf-kalogs
zookeeper.connect=10.144.113.56:2181,10.144.113.48:2181,10.31.151.165:2181
```

- **启动Kafka（启动前需要先启动Zookeeper集群）**

```
/opt/kafka_2.11-0.11.0.0/bin/kafka-server-start.sh -daemon /opt/
kafka_2.11-0.11.0.0/config/server.properties
```

- **创建Topic（该Topic将用于TCP Server）**

```
/opt/kafka_2.11-0.11.0.0/bin/kafka-topics.sh --create --topic SibosenPushsTopic
--replication-factor 1 --partitions 1 --zookeeper
10.144.113.56:2181,10.144.113.48:2181,10.31.151.165:2181
```

- Nginx（不需要docker）

```
rpm -Uvh http://nginx.org/packages/centos/7/noarch/RPMS/nginx-release-
centos-7-0.el7.ngx.noarch.rpm
yum install nginx
systemctl start nginx.service
systemctl enable nginx.service
```

Nginx需要配置以下负载均衡或反向代理：

```
http模块 – HTTP Server、FastDFS Storage、TCP Logic Server
steam模块 – TCP Comet Server
```

- **TCP Server（不需要docker） – 安装脚本目录TCP**

Comet集群

上传comet、comet-log.xml、comet.conf至服务器

修改配置后启动，启动命令（logs文件夹需要手动创建）：

```
nohup /opt/tcpserver/comet -c /opt/tcpserver/comet.conf 2>&1 > /opt/tcpserver/logs/comet.log &
```

Logic、Router、Job（这三个模块需要和Kafka安装在同一台服务器上）

同Comet模块，启动命令：

```
nohup /opt/tcpserver/router -c /opt/tcpserver/router.conf 2>&1 > /opt/tcpserver/logs/router.log &
nohup /opt/tcpserver/logic -c /opt/tcpserver/logic.conf 2>&1 > /opt/tcpserver/logs/logic.log &
nohup /opt/tcpserver/job -c /opt/tcpserver/job.conf 2>&1 > /opt/tcpserver/logs/job.log &
```

* 启动配置文件请见各个模块对应的.conf文件，模块启动顺序： router –> Logic –> Comet集群 –> Job

- **HTTP Server（先安装docker） – 安装脚本目录HTTP**

上传文件至服务器

```
cd Dockerfile目录
docker build -t web --rm=true .
docker run -d --name web1 -p 3001:8080 web
docker run -d --name web2 -p 3002:8080 web
…
```

- **Sync Service（先安装docker） – 安装脚本目录SYNC–OUTER**

上传文件至服务器

UploadFiletoIn

```
cd  Dockerfile目录
docker build –t sync–o2i ––rm=true .
docker run –d ––name sync–o2i1 sync–o2i
docker run –d ––name sync–o2i2 sync–o2i
…
```

ResultFromIn

```
cd  Dockerfile目录
docker build –t sync–i2o ––rm=true .
docker run –d ––name sync–i2o1 sync–i2o
docker run –d ––name sync–i2o2 sync–i2o
…
```

- **内网部署（先安装docker） – 安装脚本目录SYNC–INNER**

上传文件至服务器

Writer

```
cd  Dockerfile目录
docker build –t writer––rm=true .
docker run –d ––name writer1 writer
docker run –d ––name writer2 writer
…
```

Reader

```
cd  Dockerfile目录
docker build –t reader––rm=true .
docker run –d ––name reader1 reader
docker run –d ––name reader2 reader
…
```

- **Keepalived（非抢占模式）– 待续**

阿里云ECS经典网络不支持安装

- **测试环境服务器部署分布**

公网IP	内网IP	说明
47.94.56.181	10.31.185.235	FastDFS Tracker 1 - 5 TCP Comet
47.94.57.212	10.31.185.229	FastDFS Tracker 6 -10 TCP Comet
47.93.173.141	10.144.113.56	Zookeeper 1 UploadFileToIn 1 - 8 ResultFromIn 1 - 2 HTTP Server 1 - 10
47.93.174.101	10.144.113.48	Zookeeper 2 UploadFileToIn 9 - 16 ResultFromIn 3 - 4 HTTP Server 11 - 20
47.95.33.40	10.31.151.165	Zookeeper 3 UploadFileToIn 17 - 24 ResultFromIn 5 - 6 HTTP Server 21 - 30
47.93.174.100	10.144.113.52	VSFTP Kafka & TCP Job&Router&logic
47.95.33.212	10.80.49.222	Nginx Keepalived
47.94.37.195	10.31.145.144	FastDFS Storage Group 1 - 10 : 1 MySQL A + Keepalived A
47.93.174.96	10.144.113.82	FastDFS Storage Group 1 - 10 : 2 MySQL B + Keepalived B
47.93.174.130	10.29.131.179	FastDFS Storage Group 11 - 20 : 1 Redis A + Sentinel A
47.94.37.188	10.31.144.179	FastDFS Storage Group 11 - 20 : 2 Redis B + Sentinel B
47.93.80.55	10.144.112.69	经典网络不支持安装Keepalived 该服务器暂用于测试
47.95.33.153	10.80.50.22	Redis A + Sentinel A Reader 1 - 15 Writer 1 - 2
47.94.36.39	10.31.144.66	Redis B + Sentinel B Reader 16 - 30 Writer 3 - 4