ETL日志系统架构部署文档

全局说明：

以下环境只要没有特别说明，则操作系统一律为红帽子开源版CentOS7.X，JAVA虚拟机环境一律为JDK1.8+，应用框架环境一律为SpringBoot2.X，关系数据库一律为MySql5.7+，数据库反向代理一律为Atlas2.2.1，数据存储一律为MongoDB4.2.6，数据缓存一律为Redis-5.0.6，搜索引擎一律为Elasticsearch6.8.8，消息中间件一律为Emqx4.2.1，应用报表服务一律为Metabase0.37+，应用服务反向代理一律为Nginx1.17+，本篇文档运维级的部署过程全部是基于Lixiang编写的自动化运维脚本来讲述，若想基于原生部署和搭建方式请参考对应的官网文档，推荐使用自动化运维脚本构建整套ETL架构流程

一、开发组

服务端日志SDK安装

1. 下载SDK工具包
   wget https://github.com/lixiang2114/LogUtil/raw/main/target/LogUtil-1.0.jar

2. 安装SDK工具包
   mkdir -p $repository/com/bfw/log/LogUtil/1.0 && cp -a LogUtil-1.0.jar $repository/com/bfw/log/LogUtil/1.0/

3. 工程中引入SDK包依赖

```
<dependency>
    <groupId>com.bfw.log</groupId>
    <artifactId>LogUtil</artifactId>
    <version>1.0</version>
</dependency>
```

服务端日志SDK集成

1. 配置SDK组件日志参数

```
vi application.yml
logging:
  mode: file
  product.id: MY
  maxHistory: 30
  maxFileSize: 50MB
  eventLevel: info
  filePath: /opt/logs/my/my.log
```

备注：

上面仅仅作为一个配置样例，除了logging.mode参数为file以外，开发人员应根据应用产品和实际环境对其余参数进行相应调整。

2. 调用SDK组件输出日志

```
// 构建日志数据包
LogData logData=LogData.get(Event.Queried,LoggerType.ExternalCallee);
logData.ProductId=.....
logData.LoggerName=....
logData.Message=......
logData.Value=......
logData.ValueUnit=......
logData.NodeId=......
logData.ServiceName=......
logData.InstanceId=......
logData.CodePath=......
logData.HardwarePlatform=......
logData.RunningEnvironment=......
logData.Keyword=......

//打印日志数据到文件
LogUtil.info(logData);
```

备注：

上面仅仅作为一个伪代码样例，开发人员应根据应用产品和实际环境对其余的日志属性值做出相应的调整。

二、运维组

在应用服务器上搭建Flume环境

1. 下载JDK-1.8.271
   wget https://github.com/lixiang2114/Software/raw/main/jdk-8u271-linux-x64.tar.gz

2. 安装JDK-1.8.271
   tar -zxvf jdk-8u271-linux-x64.tar.gz -C /software/jdk1.8.0_271
   echo -e
   "JAVA_HOME=/software/jdk1.8.0_271\nPATH=$PATH:$JAVA_HOME/lib:$JAVA_HOME/bin\nexport PATH JAVA_HOME">>/etc/profile && source /etc/profile

3. 下载Flume-1.9.0
   wget https://github.com/lixiang2114/Software/raw/main/flume-all-1.9.0.zip

4. 安装Flume-1.9.0
   unzip flume-all-1.9.0.zip -d /software/

对接应用服务器日志到Flume服务

1. 编写Shell命令或脚本

```
vi /software/flume-1.9.0/process/script/getLogger.sh
#!/usr/bin/env bash
while true;do
    tailf /opt/logs/my/my.log 2>/dev/null
    sleep 1s
done

chmod a+x /software/flume-1.9.0/process/script/getLogger.sh
```

2. 配置Flume插件参数

```
vi /software/flume-1.9.0/process/conf/example13.conf
a1.sources=s1
a1.sinks=k1
a1.channels=c1

a1.sources.s1.type=exec
a1.sources.s1.command=/software/flume-1.9.0/process/script/getLogger.sh
a1.sources.s1.batchSize=20
a1.sources.s1.batchTimeout=3000
a1.sources.s1.restart=true
a1.sources.s1.restartThrottle=10000
a1.sources.s1.channels=c1

a1.sinks.k1.type=logger
a1.sinks.k1.channel=c1

a1.channels.c1.type = memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100
```

3. 测试Flume对接效果
   使用下面的命令启动Flume例程：

```
/software/flume-1.9.0/bin/flume-ng agent -c /software/flume-1.9.0/conf -f
/software/flume-1.9.0/process/conf/example13.conf -n a1 -
Dflume.root.logger=INFO,console
```

观察控制台是否有应用方的日志打印输出，如果没有日志打印输出，说明对接存在问题，跟着上述流程再仔细查验一遍以排查问题所在，如果有日志打印输出，则说明服务对接成功，按下键盘上的快捷键"Ctrl+C"中断Flume的控制台进程后，继续....

搭建MongoDB分布式集群环境

说明：
下面以三节点为例来说明MongoDB集群环境搭建过程，实际节点数量主要取决于每天的日志生成量、单台节点的磁盘容量及计算负载，可以先按3~6个节点来部署（最低不少于3个节点），后续在做集群调试时再酌情增加物理节点

1. 下载MongoDB4.2.6
   wget https://github.com/lixiang2114/Software/raw/main/mongodb-4.2.6.zip
   说明：
   暂时不要使用官方最新版，目前官方最新版是MongoDB4.4.1，该版本尚未稳定，执行MpaReduce分布式运算时出现数据上的逻辑统计错误，本安装版本暂无此问题

2. 安装MongoDB
   unzip mongodb-4.2.6.zip -d /software/

3. 配置MongoDB集群

```
vi configScript.sh
#!/usr/bin/env bash
declare -a MDB_HOME=$(cd $1/../;pwd)
#check envirment
(! which expect &>/dev/null) && {
  rpm -ivh --force --nodeps $MDB_HOME/rpm/{expect-5.45-14.el7_1.x86_64.rpm,tcl-8.5.13-8.el7.x86_64.rpm}
}

#global params config
declare -a CLIENT_PORT=27017
declare -a CURRENT_HOST=$(hostname)
declare -a hosts=(CC8 CC9 CC10)
declare -a nodes=(192.168.162.129 192.168.162.130 192.168.162.131)

#system login config
declare -a sshport=22
declare -a sshuser=root
declare -a sshpass=123456

#shard cluster config
confSets=(192.168.162.129:27018 192.168.162.130:27018 192.168.162.131:27018)
mongosSets=(192.168.162.129:27017 192.168.162.130:27017 192.168.162.131:27017)
replSets=(192.168.162.129:27019,192.168.162.130:27019,192.168.162.131:27019
192.168.162.129:27020,192.168.162.130:27020,192.168.162.131:27020)
mongosConfs="sharding.configDB: conf/`echo ${confSets[@]}|tr ' ' ','`"
```

说明：

以上是三节点MongoDB集群配置，现解释一下上述参数含义：
1)、基础配置：

MDB_HOME：
MongoDB的安装目录，集群中每个节点上的安装目录必须一致
CLIENT_PORT：
MongoDB默认的客户端通信端口，默认值为27017，如果没有特别理由，无需修改
CURRENT_HOST：
MongoDB例程所在的当前主机名称，如果没有特别理由，无需修改
hosts：
MongoDB集群中所有物理节点的主机名列表
hosts：
MongoDB集群中所有物理节点的主机名列表
nodes：
MongoDB集群中所有物理节点的IP地址列表

2)、系统通信配置：

sshport：
MongoDB集群节点之间SSH协议通信端口，通常操作系统默认为22，如果没有特别理由，无需修改
sshuser：
MongoDB集群节点之间SSH协议登录用户名，指定的用户名必须在每个节点上存在，且存在非系统文件的读写和执行权限，即创建的用户至少具备755权限
sshpass
MongoDB集群节点之间SSH协议登录密码

3)、集群节点通信配置：

confSets：
Config元数据集群配置，该集群节点因不存在过大的存储负载，故可与Mongos代理集群共享物理节点，该集群节点数不得少于3个
mongosSets：
Mongos代理集群配置，该集群节点因不存在过大的存储负载，故可与Config元数据集群共享物理节点，该集群节点数不得少于2个
replSets：
MongoDB分片集群配置，各个分片配置之间使用英文空格分隔，分片内部各个副本例程节点配置之间使用英文逗号分隔，上述配置的所有集群都是始终共享3个物理节点的，这是一种最简化配置方案，在生产环境中应尽量避免分片集群之间以及分片集群与配置集群或代理集群共享物理节点，即分片集群应尽可能做到具备独立物理节点；分片数的多少取决于每天的日志生成量、磁盘的剩余可用容量和各个物理节点的计算负载，后续在集群调试期间增加节点主要就是修改该参数；分片数不得少于2个，分片内副本节点数不得少于3个
mongosConfs：
Mongos代理集群与元数据配置集群之间的对接配置，如果没有特别理由，无需修改

备注：

可以在集群节点之间通过RSA非对称秘钥认证实现免密登录认证，MongoDB应用进程之间以及访问MongoDB集群的客户端和MongoDB进程之间不要设置登录认证，这会增大内网通信阻力，降低数据传输效率

3. 分发MongoDB到物理节点
   请复制/粘贴并执行下面的脚本完成MongoDB安装包的分发：

```
source /software/mongodb-4.2.6/sbin/configScript.sh
for node in ${nodes[@]};do scp -r /software/mongodb-4.2.6 ${node}:/software/
done
```

4. 启动并初始化MongoDB集群

```
启动并初始化配置集群
/software/mongodb-4.2.6/sbin/ShardTools startc
/software/mongodb-4.2.6/sbin/ShardTools initc

启动并初始化分片集群
/software/mongodb-4.2.6/sbin/ShardTools startr
/software/mongodb-4.2.6/sbin/ShardTools initr

启动并初始化代理集群
/software/mongodb-4.2.6/sbin/ShardTools startm
/software/mongodb-4.2.6/sbin/ShardTools initm
```

温馨提示：

只有首次搭建MongoDB分布式集群时需要初始化，集群搭建完成之后，以后每次启停集群可以直接执行下面的命令来完成：

```
启动集群
/software/mongodb-4.2.6/sbin/ShardTools startAll
停止集群
/software/mongodb-4.2.6/sbin/ShardTools stopAll
重启集群
/software/mongodb-4.2.6/sbin/ShardTools restartAll
重置集群(会停止集群并清空所有集群数据，慎用!!!)
/software/mongodb-4.2.6/sbin/ShardTools resetAll
```

5. 配置MonoDB集群分片
   下面以产品"蜜柚"的服务端日志为例来说明分片集群的存储配置，其它产品类同：

```
/software/mongodb-4.2.6/sbin/ShardTools addShardDB -sd MY3
/software/mongodb-4.2.6/sbin/ShardTools addShardTab -sd MY3 -st ServerLog -sf
EventName
```

参数解释：

addShardDB：
用于添加分片数据库的子命令，-sd参数用于指定需要分片的数据库名称
addShardTab：
用于添加分片数据库表的子命令，-sd参数用于指定需要分片的数据库名称，-st参数用于指定需要分片的集合表名称，-sf参数用于指定分片的字段名称，即片键，默认系统采用Hash算法做分片

对接Flume流程到MongoDB集群环境

1. 配置Flume流程

```
vi /software/flume-1.9.0/process/conf/example09.conf
a1.sources=s1
a1.sinks=k1
a1.channels=c1
```

```
a1.sources.s1.type=exec
a1.sources.s1.command=/software/flume-1.9.0/process/script/getLogger.sh
a1.sources.s1.batchSize=20
a1.sources.s1.batchTimeout=3000
a1.sources.s1.restart=true
a1.sources.s1.restartThrottle=10000
a1.sources.s1.channels=c1

a1.sinks.k1.type=com.github.lixiang2114.flume.plugin.mdb.MongoSink
a1.sinks.k1.hostList=192.168.162.129:27017,192.168.162.130:27017,192.168.162.131
:27017
a1.sinks.k1.filterName=mdbFilter
a1.sinks.k1.channel=c1

a1.channels.c1.type = memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100
```

参数解释:

a1.sinks.k1.type:
固定写法，用于指定对接MongDB分布式集群的MogoSink插件
a1.sinks.k1.hostList:
用于指定对接MongDB分布式集群的Mongos代理节点，Mongos集群中的所有节点都需要配置到此处
a1.sinks.k1.filterName:
用于指定ETL流程中的数据转换过滤器配置文件名称，例如上面指定的配置文件名称为mdbFilter.properties，配置文件的后缀必须是"*.properties"

2. 配置Flume过滤器

```
vi /software/flume-1.9.0/filter/conf/mdbFilter.properties
type=ServerLogFilter
dataBaseName=MY3
collectionName=ServerLog
```

参数解释:

type：过滤器类全名（包名+类名）
dataBaseName:
对接的MongoDB数据库名称，该值通常为当前ETL流程对接的产品英文名缩写形式，如上面指定了"蜜柚"产品的英文名缩写为"MY"
collectionName:
对接MongoDB数据库中的集合表名称，该值通常为日志数据的Schema，如上面使用"ServerLog"指定为服务端的日志数据

3. 调试ETL整套数据流程
   启动应用服务端的Flume例程

```
/software/flume-1.9.0/bin/flume-ng agent -c /software/flume-1.9.0/conf -f
/software/flume-1.9.0/process/conf/example09.conf -n a1 -
Dflume.root.logger=INFO,console
```

观察数据是否已经抵达MongoDB存储系统

```
cd /software/mongodb-4.2.6/bin
./mongo --quiet
> show databases;
MY3     0.000GB
admin   0.000GB
config  0.000GB
local   0.000GB
test    0.000GB
> use MY3;
switched to db MY3
> show tables;
Event
Product
ServerLog
> db.ServerLog.count()
228
> db.ServerLog.find()
{ "_id" : ObjectId("5fcbc6d1b8b4757c718b9fb3"), "ProductId" : 3, "SchemaName" :
"ServerLog", "NodeId" : "Product.PHYz", "ServiceName" : "PHYz", "InstanceId" :
"PHYz46", "LoggerType" : "Other", "Message" : "this is PHYz46 logger message",
"CodeLanguage" : "Java", "CodePath" : "com.bfw.test.PHYz.java", "StructedData" :
null, "ValueUnit" : "Millisecond", "Value" : 636.19, "HardwarePlatform" :
"X86/64", "OSType" : "IOS", "OSCoreVersion" : 6.1, "OSDistributionName" : "IOS-
6.1", "Keyword" : "API:UxXL", "Tags" : null, "RunningEnvironment" :
"ServiceFabric", "CreateTime" : ISODate("2020-12-05T17:43:34.462Z"), "Event" : {
"_id" : ObjectId("5fcbc596c067a269c4bc79b0"), "Value" : 41236, "Name" : "站内通
知", "EventType" : 12, "EventLevel" : 4, "LevelName" : "ERROR" }, "LoggerName" :
"KXav", "Product" : { "_id" : ObjectId("5fcba559196bab7141f96323"), "Id" : 3,
"Name" : "蜜柚", "ParentId" : 0, "Comment" : "一个牛逼的直播平台" } }
{ "_id" : ObjectId("5fcbc6d1b8b4757c718b9fb4"), "ProductId" : 3, "SchemaName" :
"ServerLog", "NodeId" : "Product.EkBQ", "ServiceName" : "EkBQ", "InstanceId" :
"EkBQ12", "LoggerType" : "InternalCallee", "Message" : "this is EkBQ12 logger
message", "CodeLanguage" : "Objective-C", "CodePath" : "com.bfw.test.EkBQ.java",
"StructedData" : null, "ValueUnit" : "CountPerSecond", "Value" : 641.83,
"HardwarePlatform" : "RISC-V", "OSType" : "Linux", "OSCoreVersion" : 6.1,
"OSDistributionName" : "Linux-6.1", "Keyword" : "API:KgjJ", "Tags" : null,
"RunningEnvironment" : "Kubernetes", "CreateTime" : ISODate("2020-12-
05T17:43:34.568Z"), "Event" : { "_id" : ObjectId("5fcbc51bc067a269c4bc79ad"),
"Value" : 11236, "Name" : "消息存储", "EventType" : 12, "EventLevel" : 1,
"LevelName" : "DEBUG" }, "LoggerName" : "Qonc", "Product" : { "_id" :
ObjectId("5fcba559196bab7141f96323"), "Id" : 3, "Name" : "蜜柚", "ParentId" : 0,
"Comment" : "一个牛逼的直播平台" } }
> exit
bye
```

备注

如你所看到的，如果在MongoDB中能否插叙出数据就证明ETL流程架构已经搭建成功了，如果没有数据则需要排查ETL架构流程中的各个环节，找出问题所在之后重新启动ETL流程；如果ETL流程搭建成功则注意观察日志数据的生成速度，如果日志数据生成速度过快，以至于不能满足单节点剩余磁盘空间存储30天的日志量，那么我们应该停止集群，重新配置configScript.sh中的replSets参数以增加物理节点数：

```
停止Flume例程，重置MongoDB集群以便于重新配置分片数量,集群调试期间的日志数据可以放弃
/software/mongodb-4.2.6/sbin/ShardTools resetAll
```

```
修改配置,增加一组分片集群
(192.168.162.132:27019,192.168.162.133:27019,192.168.162.134:27019)
vi /software/mongodb-4.2.6/sbin/configScript.sh
.............................................
replSets=(192.168.162.129:27019,192.168.162.130:27019,192.168.162.131:27019
192.168.162.129:27020,192.168.162.130:27020,192.168.162.131:27020
192.168.162.132:27019,192.168.162.133:27019,192.168.162.134:27019)

重新启动并初始化所有集群组
/software/mongodb-4.2.6/sbin/ShardTools startc
/software/mongodb-4.2.6/sbin/ShardTools initc
/software/mongodb-4.2.6/sbin/ShardTools startr
/software/mongodb-4.2.6/sbin/ShardTools initr
/software/mongodb-4.2.6/sbin/ShardTools startm
/software/mongodb-4.2.6/sbin/ShardTools initm
/software/mongodb-4.2.6/sbin/ShardTools addShardDB -sd MY3
/software/mongodb-4.2.6/sbin/ShardTools addShardTab -sd MY3 -st ServerLog -sf
EventName
.............................................
```

再次启动Flume例程，观察数据生成速度和磁盘空间耗用速度，重复以上集群调试流程，直到满足单节点能否存储30天的日志量为止，待MongoDB分布式集群调试完成并稳定后，后续如果需要通过增加物理节点来横向扩展MongoDB集群，则可以执行下面的脚本：

```
连接Mongos代理
cd /software/mongodb-4.2.6/sbin
./mongo --quiet 192.168.162.129:27017

执行addshard子命令添加分片
mongos>
db.runCommand({addshard:'rs2/192.168.162.132:27019,192.168.162.133:27019,192.168
.162.134:27019'});

执行一次数据平衡器以均衡数据节点存储负载
mongos> sh.setBalancerState(true);
mongos> exit;
```

安装并对接Metabase报表系统

1. 下载JDK-1.8.271
   wget https://github.com/lixiang2114/Software/raw/main/jdk-8u271-linux-x64.tar.gz

2. 安装JDK-1.8.271
   tar -zxvf jdk-8u271-linux-x64.tar.gz -C /software/jdk1.8.0_271
   echo -e
   "JAVA_HOME=/software/jdk1.8.0_271\nPATH=$PATH:$JAVA_HOME/lib:$JAVA_HOME/bin\nexp
   ort PATH JAVA_HOME">>/etc/profile && source /etc/profile

3. 下载MySql5.7
   wget https://github.com/lixiang2114/Software/raw/main/mysql-5.7.23.zip

4. 安装MySql5.7
   unzip mysql-5.7.23.zip -d /software/

5. 启动MySql服务
   /software/mysql-5.7.23/sbin/MysqlTools start
   lsof -i tcp:3306
   COMMAND PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
   mysqld 1559 mysql   29u IPv6 23600   0t0 TCP *:mysql (LISTEN)

6. 下载Metabase0.37
   wget https://github.com/lixiang2114/Software/raw/main/metabase0.37.2.zip

7. 安装Metabase0.37
   unzip metabase0.37.2.zip -d /software/

8. 启动Metabase服务
   /software/metabase0.37.2/bin/Startup.sh
   META_HOME: /software/metabase0.37.2
   lsof -i tcp:3000
   COMMAND PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
   java   1617 root   24u IPv6 24808   0t0 TCP *:hbci (LISTEN)

9. 测试Metabase服务是否正常
   curl -Ik -X GET http://192.168.162.127:3000/

备注:

Metabase的默认端口是3000，若Curl命令返回HTTP状态码为200则说明Metabase服务已经安装成功了
若第一次使用浏览器访问Metabase服务则需要首先创建管理员用户才能进入Metabase主界面，进入主界面之后便可设置，进入主界面之后点击右上角的"设置"—>"管理员"—>"数据库"—>"添加数据库"，填写连接参数表单:
数据库类型：MongoDB
名字：MongoDB-MY3
连接字符串:
mongodb://192.168.162.129:27017,192.168.162.130:27017,192.168.162.131:27017/MY3
最后点击保存即可创建好数据库连接，接下来可以点击右上角的"设置"—>"退出管理员"—>"浏览数据"—>"MongoDB-MY3"—>"ServerLog"即可查看服务端集合表中的日志数据

10. 下载安装Nginx并对接Metabase报表服务（略）

备注:

当Nginx安装好并对接完Metabase报表服务之后，我们就可以在外网通过浏览器访问Nginx代理服务来访问Metabase服务，由于Metabase是无状态的服务，所以我们很容易通过Nginx代理同时实现Metabase服务的高可用和负载均衡