

部署Smartbi-UnionServer集群

- 1、系统环境准备
 - 1.1集群系统环境
 - 1.2.1关闭防火墙
 - 1.2.2开启防火墙
 - 1.2.3关闭selinux
- 2、开始安装
 - 2.1、Coordinator节点安装
 - 2.1.1、修改配置文件
 - 2.2、work节点安装
- 4、运维操作
 - 4.1、启动Smartbi-UnionServer
 - 4.2、停止SmartbiUnionServer
 - 4.3、设置开机启动
 - 4.3.1 方法一
 - 4.3.2 方法二
- 5、日志文件
- 6、版本更新

Smartbi UnionServer是一个分布式SQL查询引擎，处在Smartbi的SQL引擎层，为不同的数据源提供统一的SQL解析、跨库查询能力。

1、系统环境准备

1.1集群系统环境

文档集群地址：

IP地址	主机名	角色	
192.168.137.110	192-168-137-110	Smartbi-UnionServer	Coordinator节点
192.168.137.111	192-168-137-111	Smartbi-UnionServer	Worker节点



部署SmartbiUnionServer集群时，指定一台服务器为主节点（称为**Coordinator**），其余服务器为子节点（称为**Worker**）。建议每台服务器只部署Coordinator，或者只部署Worker，不要一台服务器安装多个节点。

Work节点可以横向扩展，部署方法相同。

1.2防火墙配置



所有集群内的主机均需进行防火墙配置。

为了便于安装，建议在安装前关闭防火墙。使用过程中，为了系统安全可以选择启用防火墙，但必须启用SmartbiUnionServer使用到的相关端口。

1.2.1关闭防火墙

临时关闭防火墙

```
systemctl stop firewalld
```

永久关闭防火墙

```
systemctl disable firewalld
```

查看防火墙状态

```
systemctl status firewalld
```

1.2.2开启防火墙

相关服务及端口对照表

服务名	开放端口
SmartbiUnionServer	48080

如果确实需要打开防火墙安装，需要给防火墙放开以下需要使用到的端口
开启端口：48080

```
firewall-cmd --permanent --add-port=48080/tcp
```

配置完以后重新加载firewalld，使配置生效

```
firewall-cmd --reload
```

查看防火墙的配置信息

```
firewall-cmd --list-all
```

1.2.3关闭selinux

临时关闭selinux，立即生效，不需要重启服务器。

```
setenforce 0
```

永久关闭selinux，修改完配置后需要重启服务器才能生效

```
sed -i 's/=enforcing/=disabled/g' /etc/selinux/config
```

2、开始安装

2.1、Coordinator节点安装

上传SmartbiUnionServer.tar.gz到Coordinator节点服务器，并解压到/opt目录。

```
tar -zxvf SmartbiUnionServer.tar.gz -C /opt
```

2.1.1、修改配置文件

1) 修改run.sh配置文件

通过修改启动文件可以设置JVM的最大内存、nodeID等参数。

```
cd /opt/SmartbiUnionServer
vi run.sh
```

-Xmx参数，默认的最大内存值为8G，可根据服务器实际配置进行情况填写；

nodeID：**集群中每个节点的nodeID是惟一的，不可重复**，nodeID值为十六进制，可随意修改成其他十六进制数值（十六进制数值包含：0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F）。

```
#!/bin/bash
PRG="$0"
PRGDIR=$(dirname "$PRG")
cd $PRGDIR
Current_Dir=$(cd `dirname $0`; pwd)
PRESTO_SERVER=$Current_Dir
#根据服务器实际内存设置最大JVM内存
JAVA_OPTS="-server -Xmx8G -XX:+UseG1GC -XX:G1HeapRegionSize=32M -XX:+UseGCOverheadLimit -XX:+ExplicitGCInvokesConcurrent -XX:+HeapDumpOnOutOfMemoryError -XX:+ExitOnOutOfMemoryError"
nodeID="ffffffff-ffff-ffff-ffff-ffffffffffff" #集群每个节点的nodeID不能相同
nodeEnvironment="production"

$PRESTO_SERVER/jdk_linux/bin/java $JAVA_OPTS \
  -cp "$PRESTO_SERVER/lib/*:/target/*" \
  -Dlog_output=file:$PRESTO_SERVER/var/log/server.log \
  -Dnode_data_dir=$PRESTO_SERVER/data \
  -Dnode_id=$nodeID \
  -Dnode_environment=$nodeEnvironment \
  -Dlog_enable_console=true \
  -Dlog_levels=file:$PRESTO_SERVER/etc/log.properties \
  -Dconfig=$PRESTO_SERVER/etc/config.properties \
  -Djava.security.egd=file:/dev/./urandom \
  smartbi.presto.Server
```

修改完成后保存。

2) 参数配置

参数配置文件放在/opt/SmartbiUnionServer/etc目录下，需要修改config.properties。

config.properties的基本配置信息如下：

```
coordinator=true
node-scheduler.include-coordinator=false
http-server.http.port=48080
query.max-memory=2GB
query.max-memory-per-node=1GB
discovery-server.enabled=true
discovery.uri=http://Coordinator_IP:48080 #Coordinator_IPCoordinatorIP
```

参数说明如下：

配置项	说明
coordinator	是否允许此Presto实例作为coordinator（接受来自客户端的查询并管理查询执行），默认为true。
node-scheduler.include-coordinator	是否允许在coordinator服务中进行调度工作，配置跨库联合数据源集群，主节点建议配置为false。
http-server.http.port	设置presto的端口，默认为48080，启动时如果端口冲突，需要修改。
query.max-memory	设置单条查询语句最大使用内存，默认为2GB。
query.max-memory-per-node	设置单条查询语句在每个节点上的最大使用内存，默认为1GB。
discovery-server.enabled	Presto使用discovery服务来查找集群中的所有节点。每个Presto实例将在启动时向Discovery服务注册。Presto为了简化部署， Presto coordinator 可以运行一个内嵌在coordinator 里面的Discovery 服务。这个内嵌的Discovery 服务和Presto 共享HTTP server并且使用同样的端口
discovery.uri	设置Smartbi UnionServer的url，默认为http://0.0.0.0:48080，部署集群时需要修改为http://Coordinator_IP:48080 48080表示端口，必须与http-server.http.port保持一致。

3) 日志级别配置

日志级别配置文件放在/opt/SmartbiUnionServer/etc目录下log.properties，默认日志级别为INFO。

```
com.facebook.presto=INFO
```

日志级别可以选择：DEBUG、INFO、WARN和ERROR，其中DEBUG的日志级别最高，输出的日志最多，ERROR的日志级别最低，输出的日志最少。

2.2、work节点安装

上传SmartbiUnionServer.tar.gz到work节点服务器，并解压到/opt目录。

```
tar -zxvf SmartbiUnionServer.tar.gz -C /opt
```

进入work节点下的SmartbiUnionServer安装目录

```
cd /opt/SmartbiUnionServer
```

1) 参数配置文件放在/opt/SmartbiUnionServer/etc目录下，需要修改config.properties。

config.properties的基本配置信息如下：

```
coordinator=false                                #workcoordinatorfalse
node-scheduler.include-coordinator=true
http-server.http.port=48080
query.max-memory=2GB
query.max-memory-per-node=1GB
#discovery-server.enabled=true                  #
discovery.uri=http://Coordinator_IP:48080      #Coordinator_IPCoordinatorIP
```

2) 修改run.sh配置文件

```
vi run.sh
```

-Xmx参数，默认的最大内存值为8G，可根据服务器实际配置进行情况填写；

nodeID: **集群中每个节点的nodeID是唯一的，不可重复**，nodeID值为十六进制，可随意修改成其他十六进制数值(十六进制数值包含：0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)。

```
#!/bin/bash
PRG="$0"
PRGDIR= `dirname "$PRG"`
cd $PRGDIR
Current_Dir=$(cd `dirname $0`; pwd)
PRESTO_SERVER=$Current_Dir
#根据服务器配置设置最大JVM内存
JAVA_OPTS="-server -Xmx8G -XX:+UseG1GC -XX:G1HeapRegionSize=32M -XX:+UseGCOverheadLimit -XX:+ExplicitGCInvokesConcurrent -XX:+HeapDumpOnOutOfMemoryError -XX:+ExitOnOutOfMemoryError"
nodeID="ffffffff-ffff-ffff-ffff-ffffffffffff" #集群中每个节点的nodeID不能相同
nodeEnvironment="production"
$PRESTO_SERVER/jdk_linux/bin/java $JAVA_OPTS \
-cp "$PRESTO_SERVER/lib/*:../target/*" \
-Dlog_output_file=$PRESTO_SERVER/var/log/server.log \
-Dnode.data-dir=$PRESTO_SERVER/data \
-Dnode.id=$nodeID \
-Dnode.environment=$nodeEnvironment \
-Dlog.enable_console=true \
-Dlog.levels.file=$PRESTO_SERVER/etc/log.properties \
-Dconfig=$PRESTO_SERVER/etc/config.properties \
-Djava.security.egd=file:/dev/./urandom \
smartbix.presto.Server
```

修改完成后保存。



如果有多个work节点，work节点修改配置方法相同，注意nodeID不能重复。

4、运维操作

4.1、启动Smartbi-UnionServer



集群内所有节点SmartbiUnionServer的启动/停止/设置开机启动方法相同，并且需要在集群中的每个节点执行操作

赋予启动脚本可执行权限

```
cd /opt/SmartbiUnionServer
chmod +x run.sh
```

给jdk赋予可执行权限：

```
chmod +x -R jdk_linux/
```

启动SmartbiUnionServer服务有两种方式：

```
#
sh run.sh

# var/log/server.log
sh run.sh > /dev/null 2>&1 &
```

使用前端方式启动Smartbi UnionServer时，当看到屏幕打印信息：

===== SERVER STARTED =====，说明服务启动成功。

```
2020-05-29T13:53:42.096+0800 INFO main Bootstrap PROPERTY DEFAULT RUNTIME DESCRIPTION
2020-05-29T13:53:42.096+0800 INFO main Bootstrap resource-groups.config-file null etc/queue_config.json
2020-05-29T13:53:43.147+0800 INFO main io.airlift.bootstrap.LifeCycleManager Life cycle starting...
2020-05-29T13:53:43.148+0800 INFO main io.airlift.bootstrap.LifeCycleManager Life cycle startup complete. System ready.
2020-05-29T13:53:43.148+0800 INFO main com.facebook.presto.execution.resourceGroups.InternalResourceGroupManager -- Loaded resource group configuration manag
er file --
2020-05-29T13:53:43.148+0800 INFO main com.facebook.presto.security.AccessControlManager -- Loading system access control --
2020-05-29T13:53:43.149+0800 INFO main com.facebook.presto.security.AccessControlManager -- Loaded system access control allow-all --
2020-05-29T13:53:43.276+0800 INFO main com.facebook.presto.server.PrestoServer ===== SERVER STARTED =====
2020-05-29T13:53:46.499+0800 INFO CatalogMonitorThread smartbix.presto.CatalogMonitorHandler Start monitor catalog path /data/SmartbiUnionServer/etc/catalog/smar
tbix
```

如果提示Address already in use，说明端口冲突了，需要修改/opt/SmartbiUnionServer/etc/config.properties里的端口，然后重启SmartbiUnionServer。

```
presto.server.CoordinatorModule)
Caused by: java.io.UncheckedIOException (same stack trace as error #3)
5) Error injecting constructor, java.io.UncheckedIOException: java.net.BindException: 地址已在使用
at io.airlift.http.server.HttpServerInfo.<init>(HttpServerInfo.java:46)
at io.airlift.http.server.HttpServerModule.configure(HttpServerModule.java:66)
while locating io.airlift.http.server.HttpServerInfo
for the 1st parameter of io.airlift.http.server.LocalAnnouncementHttpServerInfo.<init>(LocalAnnouncementHttpServerInfo.java:31)
while locating io.airlift.http.server.LocalAnnouncementHttpServerInfo
at io.airlift.http.server.HttpServerModule.configure(HttpServerModule.java:78)
while locating io.airlift.discovery.client.AnnouncementHttpServerInfo
for the 1st parameter of io.airlift.discovery.client.DiscoveryBinder$HttpAnnouncementProvider.setAnnouncementHttpServerInfo(DiscoveryBinder.java:121)
at io.airlift.discovery.client.DiscoveryBinder.bindServiceAnnouncement(DiscoveryBinder.java:78) (via modules: com.facebook.presto.server.ServerMainModule -> io.airlift.co
nfiguration.ConditionalModule -> io.airlift.discovery.server.EmbeddedDiscoveryModule)
Caused by: java.io.UncheckedIOException (same stack trace as error #3)
6) Error injecting constructor, java.io.UncheckedIOException: java.net.BindException: 地址已在使用
at io.airlift.http.server.HttpServerInfo.<init>(HttpServerInfo.java:46)
at io.airlift.http.server.HttpServerModule.configure(HttpServerModule.java:66)
while locating io.airlift.http.server.HttpServerInfo
for the 1st parameter of io.airlift.http.server.LocalAnnouncementHttpServerInfo.<init>(LocalAnnouncementHttpServerInfo.java:31)
while locating io.airlift.http.server.LocalAnnouncementHttpServerInfo
at io.airlift.http.server.HttpServerModule.configure(HttpServerModule.java:78)
while locating io.airlift.discovery.client.AnnouncementHttpServerInfo
for the 1st parameter of io.airlift.discovery.client.DiscoveryBinder$HttpAnnouncementProvider.setAnnouncementHttpServerInfo(DiscoveryBinder.java:121)
at io.airlift.discovery.client.DiscoveryBinder.bindServiceAnnouncement(DiscoveryBinder.java:78)
Caused by: java.io.UncheckedIOException (same stack trace as error #3)
6 errors
at com.google.inject.internal.Errors.throwCreationExceptionIfErrorsExist(Errors.java:543)
at com.google.inject.internal.InternalInjectorCreator.injectDynamically(InternalInjectorCreator.java:178)
at com.google.inject.internal.InternalInjectorCreator.build(InternalInjectorCreator.java:109)
at com.google.inject.Guice.createInjector(Guice.java:87)
at io.airlift.bootstrap.Bootstrap.initialize(Bootstrap.java:241)
at com.facebook.presto.server.PrestoServer.run(PrestoServer.java:115)
at smartbix.presto.Server.main(Server.java:19)
```

使用后台方式启动SmartbiUnionServer时，如果使用后台启动，可以使用ps -ef | grep SmartbiUnionServer查看SmartbiUnionServer进程是否存在，如果存在，则启动成功。如下图所示。

```
[root@smartbi SmartbiUnionServer]# ps -ef | grep SmartbiUnionServer
root      19746   19741  52 13:55 pts/0    00:00:32 /data/SmartbiUnionServer/jdk_linux/bin/java -server -Xmx8G -XX:+UseG1GC -XX:G1HeapRegionSize=32M -XX:+UseGCOverheadLimit -XX:+ExplicitGCInvokesConcurrent -XX:+HeapDumpOnOutOfMemoryError -XX:+ExitOnOutOfMemoryError -cp /data/SmartbiUnionServer/lib/*:/target/* -Dlog.output-file=/data/SmartbiUnionServer/var/log/server.log -Dnode.data-dir=/data/SmartbiUnionServer/data -Dnode.id=ffffffff-ffff-ffff-ffff-ffffffffffff -Dnode.environment=production -Dlog.enable-console=true -Dlog.levels-file=/data/SmartbiUnionServer/etc/log.properties -Dconfig=/data/SmartbiUnionServer/etc/config.properties -Djava.security.egd=file:/dev/./urandom smartbix.presto.Server
root      19909   19481   0 13:56 pts/0    00:00:00 grep --color=auto SmartbiUnionServer
```

如果进程不存在，可以查看/opt/SmartbiUnionServer/var/log/server.log，查看报错信息。如果提示Address already in use，说明端口冲突了，需要修改/opt/SmartbiUnionServer/etc/config.properties里的端口，然后重启presto。

```
[root@smartbi log]# pwd
/data/SmartbiUnionServer/var/log
[root@smartbi log]# ll -h
总用量 580K
-rw-r--r--. 1 root root 0 5月 29 13:53 http-request.log
-rw-r--r--. 1 root root 20 5月 29 13:53 http-request.log-2019-03-15.0.log.gz
-rw-r--r--. 1 root root 494K 5月 29 14:02 server.log
[root@smartbi log]#
```

测试验证

使用smartbi连接跨库联合数据源验证。

跨库联合数据源

名称*

SmartbiUnionDB

别名

跨库联合数据源

驱动程序存放目录

产品内置

自定义

连接字符串*

jdbc:smartbi:uniondb://192.168.137.110:48080/填写Coordinator节点的IP和端口

验证类型

静态

动态

用户名

root用户名

密码

密码空

高级 >

测试连接(T)

保存(S)

关闭(C)

4.2、停止SmartbiUnionServer

通过命令：ps -ef | grep SmartbiUnionServer

查到SmartbiUnionserver的进程号：

```
[root@smartbi-SmartbiUnionServer]# ps -ef | grep SmartbiUnionServer
root      19746  19741  52 13:55 pts/0    00:00:32 /data/SmartbiUnionServer/jdk_linux/bin/java -server -Xmx8G -XX:+UseG1GC -XX:G1HeapRegionSize=32M -XX:+UseGCOverheadLimit -XX:+ExplicitGCInvokesConcurrent -XX:+HeapDumpOnOutOfMemoryError -XX:+ExitOnOutOfMemoryError -cp /data/SmartbiUnionServer/lib/*:../target/* -Dlog.output-file=/data/SmartbiUnionServer/var/log/server.log -Dnode.data-dir=/data/SmartbiUnionServer/data -Dnode.id=ffffffff-ffff-ffff-ffffffff -Dnode.environment=production -Dlog.enable-console=true -Dlog.levels-file=/data/SmartbiUnionServer/etc/log.properties -Dconfig=/data/SmartbiUnionServer/etc/config.properties -Djava.security.egd=file:/dev/./urandom smartbix.presto.Server
root      19909  19481  0 13:56 pts/0    00:00:00 grep --color=auto SmartbiUnionServer
```

然后使用kill -9 <进程号>命令杀掉SmartbiUnionserver进程。

4.3、设置开机启动

Linux部署SmartbiUnionServer开机启动设置方式：

4.3.1 方法一

Centos6.x

①赋予脚本可执行权限（/opt/SmartbiUnionServer/run.sh是SmartbiUnionServer的脚本路径）

```
chmod +x /opt/SmartbiUnionServer/run.sh
```

②在/etc/rc.d/rc.local文件末尾增加添加SmartbiUnionServer的脚本启动命令，保存退出

```
vi /etc/rc.d/rc.local
#
nohupsh /opt/SmartbiUnionServer/run.sh > /dev/null 2>&1 &
```

```
[root@redis ~]# cat /etc/rc.d/rc.local
#!/bin/bash
# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES
#
# It is highly advisable to create own systemd services or udev rules
# to run scripts during boot instead of using this file.
#
# In contrast to previous versions due to parallel execution during boot
# this script will NOT be run after all other services.
#
# Please note that you must run 'chmod +x /etc/rc.d/rc.local' to ensure
# that this script will be executed during boot.

touch /var/lock/subsys/local

nohup sh /opt/SmartbiUnionServer/run.sh > /dev/null 2>&1 &
[root@redis ~]#
```

③设置完成。

Centos7.x

①赋予脚本可执行权限（/opt/SmartbiUnionServer/run.sh是SmartbiUnionServer的脚本路径）

```
chmod +x /opt/SmartbiUnionServer/run.sh
```

②在/etc/rc.d/rc.local文件末尾增加添加SmartbiUnionServer的脚本启动命令，保存退出

```
vi /etc/rc.d/rc.local
#
nohupsh /opt/SmartbiUnionServer/run.sh > /dev/null 2>&1 &
```

```
[root@test ~]# cat /etc/rc.d/rc.local
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local
nohup sh /opt/SmartbiUnionServer/run.sh > /dev/null 2>&1 &
[root@test ~]#
```

③在centos7中，/etc/rc.d/rc.local的权限被降低了，所以需要执行如下命令赋予其可执行权限

```
chmod +x /etc/rc.d/rc.local
```

④设置完成

Suse12

①赋予脚本可执行权限（/opt/SmartbiUnionServer/run.sh是SmartbiUnionServer的脚本路径）

```
chmod +x /opt/SmartbiUnionServer/run.sh
```

②在/etc/rc.d/after.local文件末尾增加添加SmartbiUnionServer的脚本启动命令，保存退出

```
vi /etc/rc.d/after.local
#
nohupsh /opt/SmartbiUnionServer/run.sh > /dev/null 2>&1 &
```

```
smartbi:~ # cat /etc/rc.d/after.local
#!/bin/sh
#
# Copyright (c) 2010 SuSE LINUX Products GmbH, Germany. All rights reserved.
#
# Author: Werner Fink, 2010
#
# /etc/init.d/after.local
#
# script with local commands to be executed from init after all scripts
# of a runlevel have been executed.
#
# Here you should add things, that should happen directly after
# runlevel has been reached.
#
nohup sh /opt/SmartbiUnionServer/run.sh > /dev/null 2>&1 &
smartbi:~ #
```

③给/etc/rc.d/after.local添加执行权限

```
chmod +x /etc/rc.d/after.local
```

④设置完成

4.3.2 方法二

进入/etc/init.d目录，创建unionserver启动配置文件

```
vi /etc/init.d/unionserver
```

配置参考如下：


```
#!/bin/bash
# chkconfig: 345 80 20
# description: start the unionserver daemon
#
# Source function library
. /etc/rc.d/init.d/functions

prog=unionserver
CATALANA_HOME=/home/smartbi/SmartbiUnionServer/ #smartbi unionserver
export CATALANA_HOME

case "$1" in
start)
    echo "Starting unionserver..."
    $CATALANA_HOME/run.sh &
    ;;

stop)
    echo "Stopping unionserver..."
    kill -9 $(ps -ef | grep SmartbiUnionServer | grep jdk_linux | awk '{print $2}')
    ;;

restart)
    echo "Stopping unionserver..."
    kill -9 $(ps -ef | grep SmartbiUnionServer | grep jdk_linux | awk '{print $2}')
    sleep 2
    echo
    echo "Starting unionserver..."
    $CATALANA_HOME/run.sh &
    ;;

*)
    echo "Usage: $prog {start|stop|restart}"
    ;;

esac
exit 0
```

设置开机启动

```
chmod +x /etc/init.d/unionserver #
chkconfig unionserver on #
chkconfig --list #
```

5、日志文件

SmartbiUnionServer 的日志路径: <SmartbiUnionServer >/var/log/server.log。

如果出现启动失败时,可以通过分析日志来判断问题点。

6、版本更新



集群内所有节点均需执行更新操作,确保集群内所有节点版本一致

1) 停止现有的SmartbiUnionServer服务:

```
# ps -ef | grep SmartbiUnionServer
# kill -9 id
```

2) 升级

备份原来的SmartbiUnionServer/plugin目录和SmartbiUnionServer/lib目录

```
# mv plugin pluin_back
# mv lib lib_back
```

复制第一步解压出来的SmartbiUnionServer/plugin和SmartbiUnionServer/lib到原来的目录文件

```
# cp -r <SmartbiUnionServer>/plugin <SmartbiUnionServer>/plugin
# cp -r <SmartbiUnionServer>/lib <SmartbiUnionServer>/lib
```

复制SmartbiUnionServer/etc/queue_config.json 到etc目录

```
# cp -r <SmartbiUnionServer>/etc/queue_config.json <SmartbiUnionServer>/etc/
```

复制SmartbiUnionServer/etc/resource-groups.properties 到etc目录

```
# cp -r <SmartbiUnionServer>/etc/resource-groups.properties <SmartbiUnionServer>/etc/
```

3) 启动

```
# nohup ./run.sh &
```

4) 测试验证

使用smartbi连接跨库联合数据源验证。

跨库联合数据源

名称*

SmartbiUnionDB

别名

跨库联合数据源

驱动程序存放目录

产品内置

自定义

连接字符串*

jdbc:smartbiuniondb//192.168.137.110:48080/

填写Coordinator节点的IP和端口

验证类型

静态

动态

用户名

root

用户名

密码

密码空

高级 >

测试连接(T)

保存(S)

关闭(C)