1.下载Flink

```
wget https://archive.apache.org/dist/flink/flink-1.14.4/flink-1.14.4-bin-scala_2.11.tgz
```

- 解压、创建软连接

```
tar -zxvf flink-1.14.4-bin-scala_2.11.tgz -C /export/server

cd /export/server
ln -s flink-1.14.4-bin-scala_2.11.tgz flink
```

- 配置文件(修改以下参数)

```
cd /export/server/flink/conf
vim flink-conf.yaml

# jobManager 的IP地址（配置HA需要把想要的master设置对应的本机名字，比如node2，下载则需要修改为node2）
jobmanager.rpc.address: node1
# JobManager 的端口号
jobmanager.rpc.port: 6123
# JobManager JVM heap 内存大小
jobmanager.memory.process.size: 1600m
# TaskManager JVM heap 内存大小
taskmanager.memory.process.size: 1728m
# 每个 TaskManager 提供的任务 slots 数量大小
taskmanager.numberOfTaskSlots: 4
#是否进行预分配内存，默认不进行预分配，这样在我们不使用flink集群时候不会占用集群资源
#taskmanager.memory.preallocate: false
# 程序默认并行计算的个数
parallelism.default: 1
#JobManager的Web界面的端口（默认：8081）
jobmanager.web.port: 8081
```

- 配置worker文件

```
vim worker

node1
node2
node3
```

- 配置环境变量

```
vim /etc/profile
FLINK_HOME=/export/server/flink
PATH=$PATH:$FLINK_HOME/bin
# 立即生效
source /etc/profile
scp /etc/profile node2:/etc
scp /etc/profile node2:/etc
#都需要source一下
```

- 分发

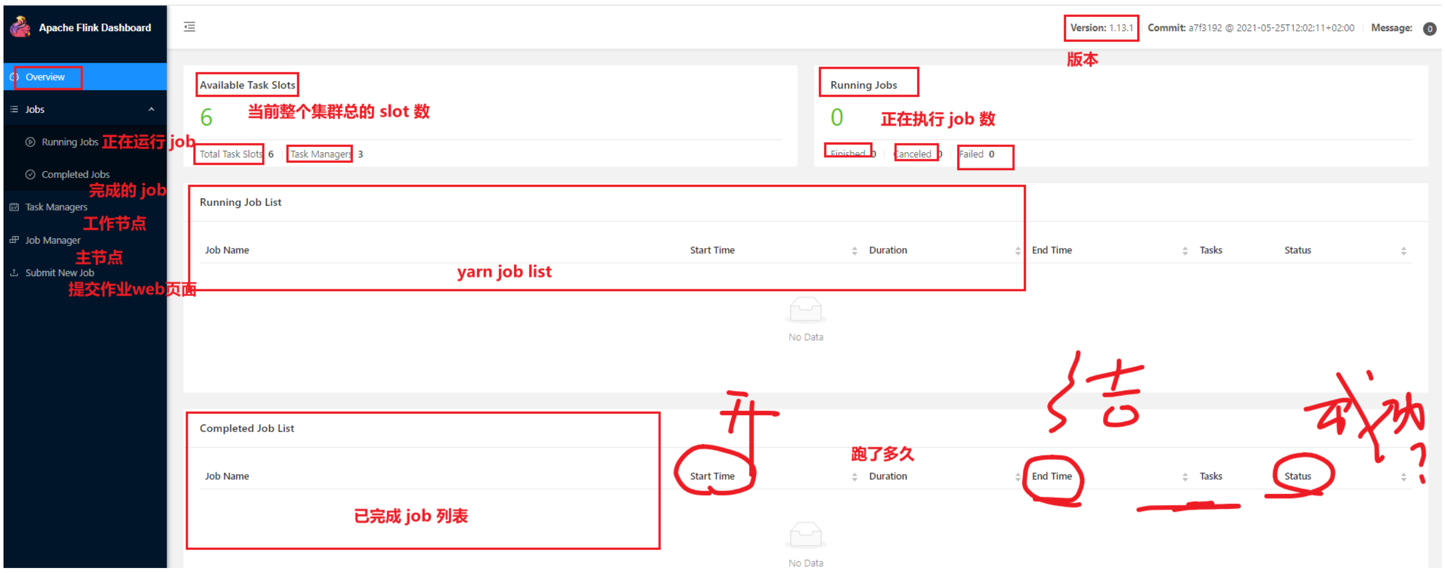
```
scp -r /export/server/flink  node2:/export/server
scp -r /export/server/flink  node3:/export/server
```

- 开启集群

```
start-cluster.sh
```

- 查看当前Flink集群的状态webul

```
node1:8081
```



- 执行wordcount任务执行，run word-count案例

```
flink run -p 2  /export/server/flink/examples/batch/WordCount.jar
```

- 参数解释

```
 flink run 提交执行任务 类似于 spark-submit
 -p 1  并行度设置为1
 --input  当前输入的参数
/export/server/flink/examples/batch/WordCount.jar   jar包位置
```

HA高可用部署方式

- 修改配置文件(node1)

```
cd /export/server/flink/conf
vim flink-conf.yaml

################################################################################
#  Licensed to the Apache Software Foundation (ASF) under one
#  or more contributor license agreements.  See the NOTICE file
#  distributed with this work for additional information
#  regarding copyright ownership.  The ASF licenses this file
#  to you under the Apache License, Version 2.0 (the
#  "License"); you may not use this file except in compliance
#  with the License.  You may obtain a copy of the License at
#
#      http://www.apache.org/licenses/LICENSE-2.0
#
#  Unless required by applicable law or agreed to in writing, software
#  distributed under the License is distributed on an "AS IS" BASIS,
#  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#  See the License for the specific language governing permissions and
# limitations under the License.
################################################################################


#==============================================================================
# Common
#==============================================================================

# The external address of the host on which the JobManager runs and can be
# reached by the TaskManagers and any clients which want to connect. This setting
# is only used in Standalone mode and may be overwritten on the JobManager side
# by specifying the --host <hostname> parameter of the bin/jobmanager.sh executable.
# In high availability mode, if you use the bin/start-cluster.sh script and setup
# the conf/masters file, this will be taken care of automatically. Yarn
# automatically configure the host name based on the hostname of the node where the
# JobManager runs.

# jobManager 的IP地址（node2需要改为node2，有几个mater就改几个）
jobmanager.rpc.address: node1

# The RPC port where the JobManager is reachable.

# JobManager 的端口号
jobmanager.rpc.port: 6123


# The total process memory size for the JobManager.
#
# Note this accounts for all memory usage within the JobManager process, including JVM metaspace and other overhead.

# JobManager JVM heap 内存大小
jobmanager.memory.process.size: 1600m


# The total process memory size for the TaskManager.
#
# Note this accounts for all memory usage within the TaskManager process, including JVM metaspace and other overhead.

# TaskManager JVM heap 内存大小
taskmanager.memory.process.size: 1728m

# To exclude JVM metaspace and overhead, please, use total Flink memory size instead of 'taskmanager.memory.process.size'.
# It is not recommended to set both 'taskmanager.memory.process.size' and Flink memory.
#
# taskmanager.memory.flink.size: 1280m

# The number of task slots that each TaskManager offers. Each slot runs one parallel pipeline.

# 每个 TaskManager 提供的任务 slots 数量大小
taskmanager.numberOfTaskSlots: 4

```

```
#是否进行预分配内存，默认不进行预分配，这样在我们不使用flink集群时候不会占用集群资源
#taskmanager.memory.preallocate: false

# The parallelism used for programs that did not specify and other parallelism.

# # 程序默认并行计算的个数
parallelism.default: 1

# The default file system scheme and authority.
#
# By default file paths without scheme are interpreted relative to the local
# root file system 'file:///'. Use this to override the default and interpret
# relative paths relative to a different file system,
# for example 'hdfs://mynamenode:12345'
#
# fs.default-scheme

#==============================================================================
# High Availability
#==============================================================================

# The high-availability mode. Possible options are 'NONE' or 'zookeeper'.
#
#开启
high-availability: zookeeper

# The path where metadata for master recovery is persisted. While ZooKeeper stores
# the small ground truth for checkpoint and leader election, this location stores
# the larger objects, like persisted dataflow graphs.
#
# Must be a durable file system that is accessible from all nodes
# (like HDFS, S3, Ceph, nfs, ...)
#
# 存储JobManager的元数据到HDFS,用来恢复JobManager 所需的所有元数据
high-availability.storageDir: hdfs://node1:8020/flink/ha/

# The list of ZooKeeper quorum peers that coordinate the high-availability
# setup. This must be a list of the form:
# "host1:clientPort,host2:clientPort,..." (default clientPort: 2181)
```

```
#
#zookeeper集群地址
high-availability.zookeeper.quorum: node1:2181,node2:2181,node3:2181


# ACL options are based on https://zookeeper.apache.org/doc/r3.1.2/zookeeperProgrammers.html#sc_BuiltinACLSchemes
# It can be either "creator" (ZOO_CREATE_ALL_ACL) or "open" (ZOO_OPEN_ACL_UNSAFE)
# The default value is "open" and it can be changed to "creator" if ZK security is enabled
#
# high-availability.zookeeper.client.acl: open

#==============================================================================
# Fault tolerance and checkpointing
#==============================================================================

# The backend that will be used to store operator state checkpoints if
# checkpointing is enabled. Checkpointing is enabled when execution.checkpointing.interval > 0.
#
# Execution checkpointing related parameters. Please refer to CheckpointConfig and ExecutionCheckpointingOptions for more details.
#
# execution.checkpointing.interval: 3min
# execution.checkpointing.externalized-checkpoint-retention: [DELETE_ON_CANCELLATION, RETAIN_ON_CANCELLATION]
# execution.checkpointing.max-concurrent-checkpoints: 1
# execution.checkpointing.min-pause: 0
# execution.checkpointing.mode: [EXACTLY_ONCE, AT_LEAST_ONCE]
# execution.checkpointing.timeout: 10min
# execution.checkpointing.tolerable-failed-checkpoints: 0
# execution.checkpointing.unaligned: false
#
# Supported backends are 'jobmanager', 'filesystem', 'rocksdb', or the
# <class-name-of-factory>.
#
#开启HA，使用文件系统作为快照存储
state.backend: filesystem

# Directory for checkpoints filesystem, when using any of the default bundled
# state backends.
#
#默认为none，用于指定checkpoint的data files和meta data存储的目录
state.checkpoints.dir: hdfs://node1:8020/flink-checkpoints

# Default target directory for savepoints, optional.
#
#默认为none，用于指定savepoints的默认目录
state.savepoints.dir: hdfs://node1:8020/flink-savepoints

# Flag to enable/disable incremental checkpoints for backends that
# support incremental checkpoints (like the RocksDB state backend).
#
# state.backend.incremental: false

# The failover strategy, i.e., how the job computation recovers from task failures.
# Only restart tasks that may have been affected by the task failure, which typically includes
# downstream tasks and potentially upstream tasks if their produced data is no longer available for consumption.

jobmanager.execution.failover-strategy: region

#==============================================================================
# Rest & web frontend
#==============================================================================

# The port to which the REST client connects to. If rest.bind-port has
# not been specified, then the server will bind to this port as well.
#
#JobManager的Web界面的端口（默认：8081）
rest.port: 8081

# The address to which the REST client will connect to
#
#rest.address: 0.0.0.0

# Port range for the REST and web server to bind to.
#
#rest.bind-port: 8080-8090
```

```

# The address that the REST & web server binds to
#
#rest.bind-address: 0.0.0.0

# Flag to specify whether job submission is enabled from the web-based
# runtime monitor. Uncomment to disable.

#web.submit.enable: false

# Flag to specify whether job cancellation is enabled from the web-based
# runtime monitor. Uncomment to disable.

#web.cancel.enable: false

#==============================================================================
# Advanced
#==============================================================================

# Override the directories for temporary files. If not specified, the
# system-specific Java temporary directory (java.io.tmpdir property) is taken.
#
# For framework setups on Yarn, Flink will automatically pick up the
# containers' temp directories without any need for configuration.
#
# Add a delimited list for multiple directories, using the system directory
# delimiter (colon ':' on unix) or a comma, e.g.:
#     /data1/tmp:/data2/tmp:/data3/tmp
#
# Note: Each directory entry is read from and written to by a different I/O
# thread. You can include the same directory multiple times in order to create
# multiple I/O threads against that directory. This is for example relevant for
# high-throughput RAIDs.
#
#临时文件
io.tmp.dirs: /export/server/flink/tmp

# The classloading resolve order. Possible values are 'child-first' (Flink's default)
# and 'parent-first' (Java's default).
```

```
#
# Child first classloading allows users to use different dependency/library
# versions in their application than those in the classpath. Switching back
# to 'parent-first' may help with debugging dependency issues.
#
# classloader.resolve-order: child-first

# The amount of memory going to the network stack. These numbers usually need
# no tuning. Adjusting them may be necessary in case of an "Insufficient number
# of network buffers" error. The default min is 64MB, the default max is 1GB.
#
# taskmanager.memory.network.fraction: 0.1
# taskmanager.memory.network.min: 64mb
# taskmanager.memory.network.max: 1gb

#==============================================================================
# Flink Cluster Security Configuration
#==============================================================================

# Kerberos authentication for various components - Hadoop, ZooKeeper, and connectors -
# may be enabled in four steps:
# 1. configure the local krb5.conf file
# 2. provide Kerberos credentials (either a keytab or a ticket cache w/ kinit)
# 3. make the credentials available to various JAAS login contexts
# 4. configure the connector to use JAAS/SASL

# The below configure how Kerberos credentials are provided. A keytab will be used instead of
# a ticket cache if the keytab path and principal are set.

# security.kerberos.login.use-ticket-cache: true
# security.kerberos.login.keytab: /path/to/kerberos/keytab
# security.kerberos.login.principal: flink-user

# The configuration below defines which JAAS login contexts

# security.kerberos.login.contexts: Client,KafkaClient

#==============================================================================
# ZK Security Configuration
```

```
#==============================================================================

# Below configurations are applicable if ZK ensemble is configured for security

# Override below configuration to provide custom ZK service name if configured
# zookeeper.sasl.service-name: zookeeper

# The configuration below must match one of the values set in
"security.kerberos.login.contexts"
# zookeeper.sasl.login-context-name: Client

#==============================================================================
# HistoryServer
#==============================================================================

# The HistoryServer is started and stopped via bin/historyserver.sh (start|stop)

# Directory to upload completed jobs to. Add this directory to the list of
# monitored directories of the HistoryServer as well (see below).
#jobmanager.archive.fs.dir: hdfs:///completed-jobs/

# The address under which the web-based HistoryServer listens.
#historyserver.web.address: 0.0.0.0

# The port under which the web-based HistoryServer listens.
#historyserver.web.port: 8082

# Comma separated list of directories to monitor for completed jobs.
#historyserver.archive.fs.dir: hdfs:///completed-jobs/

# Interval in milliseconds for refreshing the monitored directories.
#historyserver.archive.fs.refresh-interval: 10000

#blob存储文件是在群集中分发Flink作业所必需的
blob.storage.directory: /export/server/flink/tmp



```

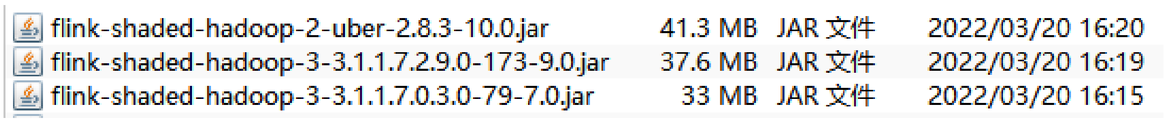
- 配置worker

```
node1
node2
node3
```

- 配置mater文件

```
node1:8081
node2:8081
```

- 下载jar包到flink\lib



```
flink-shaded-hadoop-2-uber-2.8.3-10.0.jar          41.3 MB  JAR 文件    2022/03/20 16:20
flink-shaded-hadoop-3-3.1.1.7.2.9.0-173-9.0.jar    37.6 MB  JAR 文件    2022/03/20 16:19
flink-shaded-hadoop-3-3.1.1.7.0.3.0-79-7.0.jar       33 MB  JAR 文件    2022/03/20 16:15
```

- 分发

```
scp -r /export/server/flink  node2:/export/server
scp -r /export/server/flink  node3:/export/server

```

- HA高可用，设置HDFS上的路径用于保存ha的数据，防止出现当前集群jobmanager挂掉恢复最新状态
- 需要先开启zookeeper，再启动flink集群，通过start-cluster.sh
- 切换jobmanager实现HA高可用
  - 关闭node1上的 jobmanager 进程
  - 查看 node2 上 8081 web的log日志，查看是否 granted leadership

YARN部署

- 配置
  - yarn-site.xml 中修改一下memcheck 置为 false ， 不让检查内存是否可用。

```
<property>
    <name>yarn.nodemanager.vmem-check-enabled</name>
    <value>false</value>
</property>
```

- 然后分发到各个节点上
- yarn-session + flink run
  - 应用场景：大量的小任务，当小任务执行完毕之后并不会关闭session，小任务之间共享session（内存和CPU cores）不隔离资源。

```
# 开启 yarn-session 会话
```

```
yarn-session.sh -tm 1024 -s 2 -d
# -tm taskmanager 的内存大小
# -s slot 数
# -d daemon 后台执行
flink run -p 2 /export/server/flink/examples/batch/WordCount.jart
```

- kill掉一直运行的session

```
yarn application -kill application_1638083192874_0001
```

- 每个任务都是直接 flink run 执行 per-job
  - 应用场景： 适合于大多数生产环境的，任务的执行，每个任务一个session，程序执行完毕关闭会话。

```
flink run \
-m yarn-cluster \
-yjm 1024m \
-ytm 1024m \
/export/server/flink/examples/batch/WordCount.jar \
--input hdfs://node1:8020/words.txt
```