

---

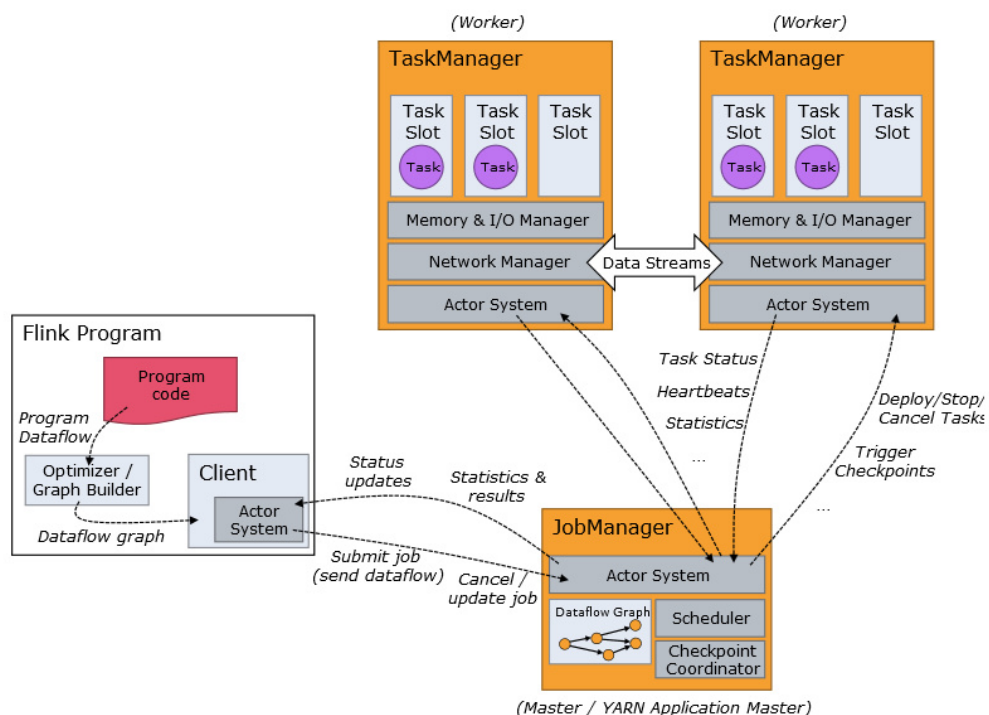
toc: true title: 《从0到1学习Flink》—— Flink JobManager 高可用性配置 date: 2019-01-13 tags:

- Flink
  - 大数据
  - 流式计算
- 

## 前言

之前在 [《从0到1学习Flink》—— Flink 配置文件详解](#) 讲过 Flink 的配置，但是后面陆续有人来问我一些配置相关的东西，在加上我现在对 Flink 也更熟悉了些，这里我就再写下 Flink JobManager 的配置相关信息。

在 [《从0到1学习Flink》—— Apache Flink 介绍](#) 一文中介绍过了 Flink Job 的运行架构图：



JobManager 协调每个 Flink 作业的部署。它负责调度和资源管理。

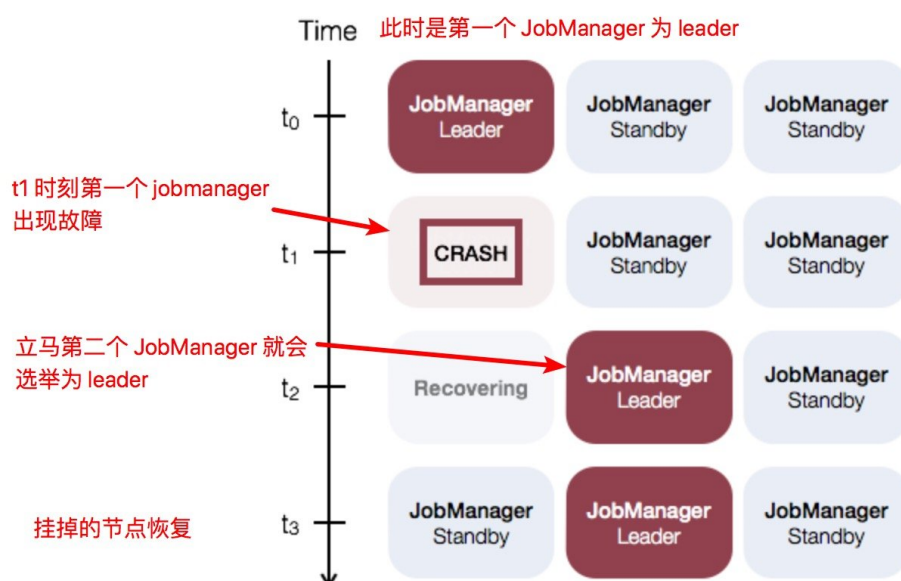
默认情况下，每个 Flink 集群都有一个 JobManager 实例。这会产生单点故障（SPOF）：如果 JobManager 崩溃，则无法提交新作业且运行中的作业也会失败。

如果我们使用 JobManager 高可用模式，可以避免这个问题。您可以为 standalone 集群和 YARN 集群配置高可用模式。

## standalone 集群高可用性

standalone 集群的 JobManager 高可用性的概念是，任何时候都有一个主 JobManager 和多个备 JobManagers，以便在主节点失败时有新的 JobManager 接管集群。这样就保证了没有单点故障，一旦备 JobManager 接管集群，作业就可以依旧正常运行。主备 JobManager 实例之间没有明确的区分。每个 JobManager 都可以充当主备节点。

例如，请考虑以下三个 JobManager 实例的设置：



## 如何配置

要启用 JobManager 高可用性功能，您必须将高可用性模式设置为 zookeeper，配置 ZooKeeper quorum，将所有 JobManagers 主机及其 Web UI 端口写入配置文件。

Flink 利用 ZooKeeper 在所有正在运行的 JobManager 实例之间进行分布式协调。ZooKeeper 是独立于 Flink 的服务，通过 leader 选举和轻量级一致性状态存储提供高可靠的分布式协调服务。Flink 包含用于 Bootstrap ZooKeeper 安装的脚本。他在我们的 Flink 安装路径下面 /conf/zoo.cfg 。

```
-rw-r--r-- 1 zhisheng admin 2.2K 8 6 23:11 logback-console.xml
-rw-r--r-- 1 zhisheng admin 1.5K 8 6 23:11 logback-yarn.xml
-rw-r--r-- 1 zhisheng admin 2.3K 8 6 23:11 logback.xml
-rw-r--r-- 1 zhisheng admin 15B 8 6 23:11 masters
-rw-r--r-- 1 zhisheng admin 10B 8 6 23:11 slaves
-rw-r--r-- 1 zhisheng admin 3.2K 8 6 23:11 sql-client-defaults.yaml
-rw-r--r-- 1 zhisheng admin 1.4K 8 6 23:11 zoo.cfg
zhisheng@zhisheng > /usr/local/Cellar/apache-flink/1.6.0/libexec/conf > cat zoo.cfg
#####
# 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.
#####
# The number of milliseconds of each tick
tickTime=2000
# The number of ticks that the initial synchronization phase can take
initLimit=10
# The number of ticks that can pass between sending a request and getting an acknowledgement
syncLimit=5
# The directory where the snapshot is stored.
# dataDir=/tmp/zookeeper
# The port at which the clients will connect
clientPort=2181
# ZooKeeper quorum peers
server.1=localhost:2888:3888
# server.2=host:peer-port:leader-port
```

需要在这里配置zk

微信公众号: zhisheng

## Masters 文件

要启动 HA 集群，请在以下位置配置 Masters 文件 conf/masters：

```
localhost:8081
xxx.xxx.xxx.xxx:8081
```

masters 文件包含启动 JobManagers 的所有主机以及 Web 用户界面绑定的端口，上面一行写一个。

默认情况下，job manager 选一个随机端口作为进程通信端口。您可以通过 `high-availability.jobmanager.port` 更改此设置。此配置接受单个端口（例如 50010），范围（50000-50025）或两者的组合（50010,50011,50020-50025,50050-50075）。

## 配置文件 (flink-conf.yaml)

要启动 HA 集群，请将以下配置键添加到 `conf/flink-conf.yaml`：

高可用性模式（必需）：在 `conf/flink-conf.yaml` 中，必须将高可用性模式设置为 `zookeeper`，以打开高可用模式。

```
high-availability: zookeeper
```

ZooKeeper quorum（必需）：ZooKeeper quorum 是一组 ZooKeeper 服务器，它提供分布式协调服务。

```
high-availability.zookeeper.quorum: ip1:2181 [...],ip2:2181
```

每个 `ip:port` 都是一个 ZooKeeper 服务器的 ip 及其端口，Flink 可以通过指定的地址和端口访问 zookeeper。

```
#####
# High Availability
#####

# The high-availability mode. Possible options are 'NONE' or 'zookeeper'.
#
# 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, ...)
#
# high-availability.storageDir: hdfs:///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)
#
# high-availability.zookeeper.quorum: localhost:2181

# ACL options are based on https://zookeeper.apache.org/doc/r3.1.2/zookeeperProgrammers.html#sc_BuiltinACLSchema
# 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
#####
```

对应这两个配置

微信公众号: zhisheng

另外就是高可用存储目录，JobManager 元数据保存在文件系统 storageDir 中，在 ZooKeeper 中仅保存了指向此状态的指针，推荐这个目录是 HDFS, S3, Ceph, nfs 等，该 storageDir 中保存了 JobManager 恢复状态需要的所有元数据。

```
high-availability.storageDir: hdfs:///flink/ha/
```

配置 master 文件和 ZooKeeper 配置后，您可以使用提供的集群启动脚本。他们将启动 HA 集群。请注意，启动 Flink HA 集群前，必须启动 Zookeeper 集群，并确保为要启动的每个 HA 集群配置单独的 ZooKeeper 根路径。

## 示例

具有 2 个 JobManagers 的 Standalone 集群：

1、在 conf/flink-conf.yaml 中配置高可用模式和 Zookeeper：

```
high-availability: zookeeper
high-availability.zookeeper.quorum: localhost:2181
high-availability.storageDir: hdfs:///flink/recovery
```

2、在 conf/masters 中配置 masters：

```
localhost:8081
localhost:8082
```

3、在 conf/zoo.cfg 中配置 Zookeeper 服务：

```
server.0=localhost:2888:3888
```

4、启动 ZooKeeper 集群：

```
$ bin/start-zookeeper-quorum.sh
Starting zookeeper daemon on host localhost.
```

5、启动一个 Flink HA 集群：

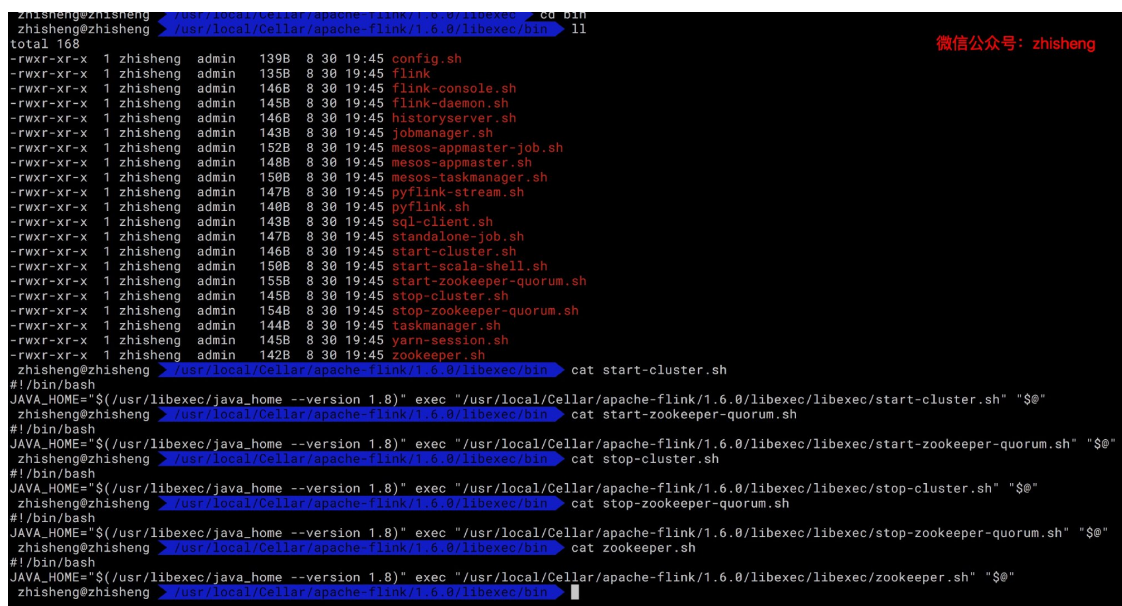
```
$ bin/start-cluster.sh
Starting HA cluster with 2 masters and 1 peers in ZooKeeper quorum.
Starting jobmanager daemon on host localhost.
Starting jobmanager daemon on host localhost.
Starting taskmanager daemon on host localhost.
```

## 6、停止 ZooKeeper 和集群:

```
$ bin/stop-cluster.sh
Stopping taskmanager daemon (pid: 7647) on localhost.
Stopping jobmanager daemon (pid: 7495) on host localhost.
Stopping jobmanager daemon (pid: 7349) on host localhost.

$ bin/stop-zookeeper-quorum.sh
Stopping zookeeper daemon (pid: 7101) on host localhost.
```

上面的执行脚本如下图可见:



```
zhisheng@zhisheng:~/flink-1.6.0/libexec$ cd bin
zhisheng@zhisheng:~/flink-1.6.0/libexec/bin$ ls -la
total 168
-rwxr-xr-x 1 zhisheng admin 139B 8 30 19:45 config.sh
-rwxr-xr-x 1 zhisheng admin 135B 8 30 19:45 flink
-rwxr-xr-x 1 zhisheng admin 146B 8 30 19:45 flink-console.sh
-rwxr-xr-x 1 zhisheng admin 145B 8 30 19:45 flink-daemon.sh
-rwxr-xr-x 1 zhisheng admin 146B 8 30 19:45 historyserver.sh
-rwxr-xr-x 1 zhisheng admin 143B 8 30 19:45 jobmanager.sh
-rwxr-xr-x 1 zhisheng admin 152B 8 30 19:45 mesos-appmaster-job.sh
-rwxr-xr-x 1 zhisheng admin 148B 8 30 19:45 mesos-appmaster.sh
-rwxr-xr-x 1 zhisheng admin 150B 8 30 19:45 mesos-taskmanager.sh
-rwxr-xr-x 1 zhisheng admin 147B 8 30 19:45 pyflink-stream.sh
-rwxr-xr-x 1 zhisheng admin 140B 8 30 19:45 pyflink.sh
-rwxr-xr-x 1 zhisheng admin 143B 8 30 19:45 sql-client.sh
-rwxr-xr-x 1 zhisheng admin 147B 8 30 19:45 standalone-job.sh
-rwxr-xr-x 1 zhisheng admin 146B 8 30 19:45 start-cluster.sh
-rwxr-xr-x 1 zhisheng admin 150B 8 30 19:45 start-scala-shell.sh
-rwxr-xr-x 1 zhisheng admin 155B 8 30 19:45 start-zookeeper-quorum.sh
-rwxr-xr-x 1 zhisheng admin 145B 8 30 19:45 stop-cluster.sh
-rwxr-xr-x 1 zhisheng admin 154B 8 30 19:45 stop-zookeeper-quorum.sh
-rwxr-xr-x 1 zhisheng admin 144B 8 30 19:45 taskmanager.sh
-rwxr-xr-x 1 zhisheng admin 145B 8 30 19:45 yarn-session.sh
-rwxr-xr-x 1 zhisheng admin 142B 8 30 19:45 zookeeper.sh
zhisheng@zhisheng:~/flink-1.6.0/libexec/bin$ cat start-cluster.sh
#!/bin/bash
JAVA_HOME=$(/usr/libexec/java_home --version 1.8) exec "/usr/local/Cellar/apache-flink/1.6.0/libexec/libexec/start-cluster.sh" "$@"
zhisheng@zhisheng:~/flink-1.6.0/libexec/bin$ cat start-zookeeper-quorum.sh
#!/bin/bash
JAVA_HOME=$(/usr/libexec/java_home --version 1.8) exec "/usr/local/Cellar/apache-flink/1.6.0/libexec/libexec/start-zookeeper-quorum.sh" "$@"
zhisheng@zhisheng:~/flink-1.6.0/libexec/bin$ cat stop-cluster.sh
#!/bin/bash
JAVA_HOME=$(/usr/libexec/java_home --version 1.8) exec "/usr/local/Cellar/apache-flink/1.6.0/libexec/libexec/stop-cluster.sh" "$@"
zhisheng@zhisheng:~/flink-1.6.0/libexec/bin$ cat stop-zookeeper-quorum.sh
#!/bin/bash
JAVA_HOME=$(/usr/libexec/java_home --version 1.8) exec "/usr/local/Cellar/apache-flink/1.6.0/libexec/libexec/stop-zookeeper-quorum.sh" "$@"
zhisheng@zhisheng:~/flink-1.6.0/libexec/bin$ cat zookeeper.sh
#!/bin/bash
JAVA_HOME=$(/usr/libexec/java_home --version 1.8) exec "/usr/local/Cellar/apache-flink/1.6.0/libexec/libexec/zookeeper.sh" "$@"
zhisheng@zhisheng:~/flink-1.6.0/libexec/bin$
```

## YARN 集群高可用性

当运行高可用的 YARN 集群时, 我们不会运行多个 JobManager 实例, 而只会运行一个, 该 JobManager 实例失败时, YARN 会将其重新启动。Yarn 的具体行为取决于您使用的 YARN 版本。

## 如何配置？

### Application Master 最大重试次数 (yarn-site.xml)

在 YARN 配置文件 yarn-site.xml 中，需要配置 application master 的最大重试次数：

```
<property>
  <name>yarn.resourcemanager.am.max-attempts</name>
  <value>4</value>
  <description>
    The maximum number of application master execution attempts.
  </description>
</property>
```

当前 YARN 版本的默认值为 2（表示允许单个 JobManager 失败两次）。

### Application Attempts (flink-conf.yaml)

除了上面可以配置最大重试次数外，你还可以在 flink-conf.yaml 配置如下：

```
yarn.application-attempts: 10
```

这意味着在如果程序启动失败，YARN 会再重试 9 次（9 次重试 + 1 次启动），如果启动 10 次作业还失败，yarn 才会将该任务的状态置为失败。如果因为节点硬件故障或重启，NodeManager 重新同步等操作，需要 YARN 继续尝试启动应用。这些重启尝试不计入 yarn.application-attempts 个数中。

## 容器关闭行为



- YARN 2.3.0 < 版本 < 2.4.0. 如果 application master 进程失败，则所有的 container 都会重启。
- YARN 2.4.0 < 版本 < 2.6.0. TaskManager container 在 application master 故障期间，会继续工作。这具有以下优点：作业恢复时间更快，且缩短所有 task manager 启动时申请资源的时间。
- YARN 2.6.0 <= version: 将尝试失败有效性间隔设置为 Flink 的 Akka 超时值。尝试失败有效性间隔表示只有在系统在一个间隔期间看到最大应用程序尝试次数后才会终止应用程序。这避免了持久的工作会耗尽它的应用程序尝试。

## 示例：高可用的 YARN Session

1、配置 HA 模式和 Zookeeper 集群 在 conf/flink-conf.yaml:

```
high-availability: zookeeper
high-availability.zookeeper.quorum: localhost:2181
yarn.application-attempts: 10
```

2、配置 ZooKeeper 服务 在 conf/zoo.cfg:

```
server.0=localhost:2888:3888
```

3、启动 Zookeeper 集群:

```
$ bin/start-zookeeper-quorum.sh
Starting zookeeper daemon on host localhost.
```

4、启动 HA 集群:

```
$ bin/yarn-session.sh -n 2
```

## 总结

本篇文章再次写了下 Flink JobManager 的高可用配置，如何在 standalone 集群和 YARN 集群中配置高可用。



本文原创地址是: <http://www.54tianzhisheng.cn/2019/01/13/Flink-JobManager-High-availability/> , 未经允许禁止转载。

## Github 代码仓库

<https://github.com/zhisheng17/flink-learning/>

以后这个项目的所有代码都将放在这个仓库里, 包含了自己学习 flink 的一些 demo 和博客