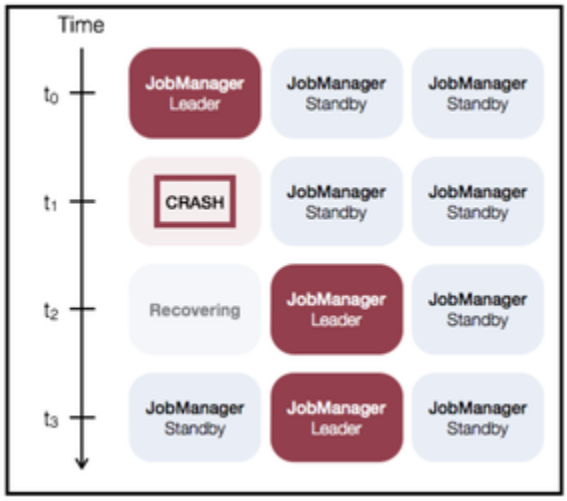
Install Apache Flink on Multi-node Cluster

Architecture

The general idea of JobManager high availability for standalone clusters is that there is a **single leading JobManager** at any time and **multiple standby JobManagers** to take over leadership in case the leader fails. This guarantees that there is **no single point of failure** and programs can make progress as soon as a standby JobManager has taken leadership. There is no explicit distinction between standby and master JobManager instances. Each JobManager can take the role of master or standby.

As an example, consider the following setup with three JobManager instances:



Platform Requirements

1. Operating system: Ubuntu 16.04 or later.
2. Java 8.x or higher.
3. Create an odd number of machines (eg: 3, 5, 7..).
4. System requirement is job specific.
    a. Minimum memory requirement is 4 GB/ machine.
    b. Select the number of CPU cores according to the number of flink jobs.
5. Need a shared filesystem. For example.
    a. Elastic File System (EFS, Managed service provided by AWS.)
    b. Gluster File System (GFS)
    c. Hadoop Distributed File System (HDFS)

Prerequisites

The below document creates a 3 node cluster

1. Install Java 8 (For each Flink machines).

```
apt-get update && apt-get -y upgrade
apt-get install openjdk-8-jdk-headless
```

2. Setup environment (For each Flink machines).

```
echo """
LANGUAGE=\"en_US.UTF-8\"
LANG=\"en_US.UTF-8\"
LC_ALL=\"en_US.UTF-8\"
JAVA_HOME=\"/usr/lib/jvm/java-8-openjdk-amd64\"
""" >> /etc/environment
```

3. Assign proper hostname for machines (For each Flink machines).

```
eg: flink-server1
hostnamectl set-hostname flink-server1
hostname -F /etc/hostname
echo """
<Ip of flink-server1>  flink-server1
<Ip of flink-server2>  flink-server2
<Ip of flink-server3>  flink-server3
""" > /etc/hosts
```

4. Configure passwordless root user ssh between each machine.

```
eg: flink-server1

Generate the ssh-keypair
    sudo su -
 apt-get install openssh-server openssh-client
 ssh-keygen -t rsa -P ""
 cat /root/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys

Copy the public key From "flink-server1" (cat
/root/.ssh/id_rsa.pub) and add that key in other servers
/root/.ssh/authorized_keys.

Test passwordless ssh, try root ssh from flink-server1 to
flink-server2 & flink-server3 and vice-versa
```

Installation and configuration

1. Installation => Take 1 machine (eg: flink-server1)

```
cd /opt/
wget
https://archive.apache.org/dist/flink/flink-1.7.0/flink-1.7.0-bin-scala_2.12.tgz
tar -zxvf flink-1.7.0-bin-scala_2.12.tgz
cd flink-1.7.0
```

2. Configuration => Take 1 machine (eg: flink-server1)
    a. Edit /opt/flink-1.7.0/conf/flink-conf.yaml (Main configuration file)

```
Comment the below line
# jobmanager.rpc.address: localhost

Assign necessary heap memory as required
jobmanager.heap.size: 3072m
taskmanager.heap.size: 8192m

Assign necessary task slots as required (recommended assign
number_of_cpu_cores-1 slots)
taskmanager.numberOfTaskSlots: 7

Add the below entries at the end of this configuration file
high-availability: zookeeper
high-availability.zookeeper.quorum:
flink-server1:2181,flink-server1:2181,flink-server3:2181
high-availability.cluster-id: /cluster_one
high-availability.storageDir: file:///opt/flink-1.7.0/ha
zookeeper.sasl.disable: true

akka.watch.heartbeat.pause: 600s
akka.watch.threshold: 50

akka.ask.timeout: 600s

restart-strategy: failure-rate
restart-strategy.failure-rate.max-failures-per-interval: 10
restart-strategy.failure-rate.failure-rate-interval: 5 min
restart-strategy.failure-rate.delay: 10 s
```

    b. Edit /opt/flink-1.7.0/conf/zoo.cfg (Zookeeper configurtion file)

```
Add below ZooKeeper quorum peers by

Removing the entry
server.1=localhost:2888:3888

Adding the entry
server.1=flink-server1:2888:3888
server.2=flink-server2:2888:3888
server.3=flink-server3:2888:3888
```

c. Edit /opt/flink-1.7.0/conf/masters (Flink master configuration file)

```
Add the below entries
flink-server1:8081
flink-server2:8081
flink-server3:8081
```

d. Edit /opt/flink-1.7.0/conf/slaves (Flink slave configuration file)

```
Add the below entries
flink-server1
flink-server2
flink-server3
```

e. Edit /opt/flink-1.7.0/conf/log4j.properties (Flink log configuration file)

```
Comment the below lines
#log4j.appender.file=org.apache.log4j.FileAppender
#log4j.appender.file.file=${log.file}
#log4j.appender.file.append=false
#log4j.appender.file.layout=org.apache.log4j.PatternLayout
#log4j.appender.file.layout.ConversionPattern=%d{yyyy-MM-dd
HH:mm:ss,SSS} %-5p %-60c %x - %m%n

Adding the below lines at the end of configuration file.
log4j.appender.file=org.apache.log4j.RollingFileAppender
log4j.appender.file.file=${log.file}
log4j.appender.file.MaxFileSize=300MB
log4j.appender.file.MaxBackupIndex=10
log4j.appender.file.append=false
log4j.appender.file.layout=org.apache.log4j.PatternLayout
log4j.appender.file.layout.ConversionPattern=%d{yyyy-MM-dd
HH:mm:ss,SSS} %-5p %-60c %x - %m%n
```

f. Copy flink installation folder to other machines

```
scp -r /opt/flink-1.7.0 flink-server2:/opt
scp -r /opt/flink-1.7.0 flink-server3:/opt
```

3. Create and configure the shared file system
   a. Create EFS Link, more details.
   b. Mount the filesystem in each machine.

```
Mounting, execute the below commands.
 mkdir /opt/flink-1.7.0/ha
 echo """<file_system_id>.efs.us-east-1.amazonaws.com:/
/opt/flink-1.7.0/ha nfs defaults,_netdev 0 0""" >> /etc/fstab
 mount -a

Confirm the mount, execute the below command.
 df -Th
```

Flink cluster managing commands

1. Starting the cluster

```
Login to any flink server and change to flink installation folder.
cd /opt/flink-1.7.0/

Start zookeeper quorum
./bin/start-zookeeper-quorum.sh

Starting the flink cluster
./bin/start-cluster.sh
```

2. Stopping the cluster

```
Login to any flink server and change to flink installation folder.
cd /opt/flink-1.7.0/

Start zookeeper quorum
./bin/stop-cluster.sh

Starting the flink cluster
./bin/stop-zookeeper-quorum.sh
```

3. Starting JobManager and TaskManager

```
Login to any flink server and change to flink installation folder.
cd /opt/flink-1.7.0/

Starting job manager
./bin/jobmanager.sh start

Starting task manager
./bin/taskmanager.sh start
```

4. Stopping JobManager and TaskManager

```
Login to any flink server and change to the flink installation
folder.
cd /opt/flink-1.7.0/

Starting job manager
./bin/jobmanager.sh stop

Starting task manager
./bin/taskmanager.sh stop
```

Note: AutoScaling for both jobmanager and taskmanager is possible by adding a new server

- Copy the flink folder from one of the servers to the new server in the same path.
- EFS mount is not required for taskmanager autoscaling
- Run command to start jobmanager/ taskmanager.

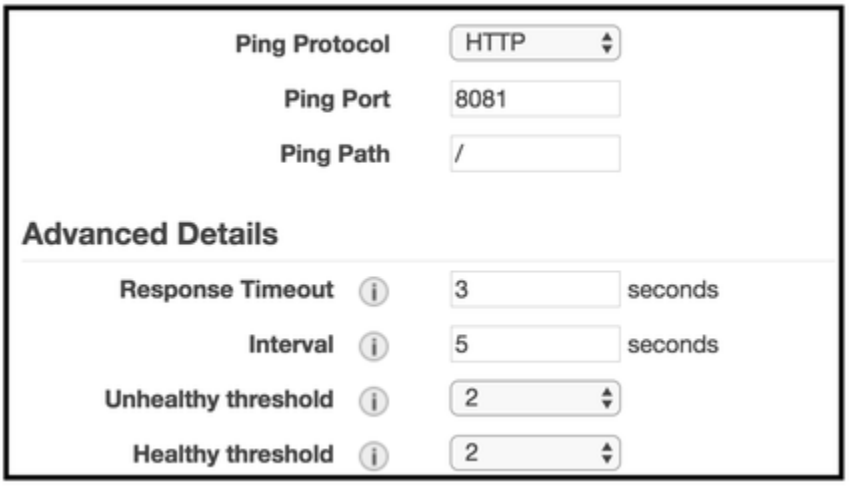
Accessing Flink Servers

Flink master provides a UI listening on **TCP port 8081** to manage the resources through a REST interface. Since we have multiple masters, it is required to create a LoadBalancers with proper health check for accessing Flink UI.

In the case of AWS,

1. It is recommended to use an internal Classic Load Balancer for internal routing.
2. For external access use either VPN (Secure access for prod and stage) or a public Classic Load Balancer.

Flink AWS CLB health check configuration is shown below.



Monitoring and Recovery

1. Cronjob monitoring and Recovery.

```
Assign the cronjob for each machine.

Commands
cd /opt/flink-1.7.0/
mkdir custom
vim /opt/flink-1.7.0/custom/autorestart.sh
    Add the code =>
https://github.com/xblockchainlabs/csfx-cicd/blob/master/sources/Fl
ink-Job-Monitoring/flink-service-restart-cron.sh
chmod +x /opt/flink-1.7.0/custom/autorestart.sh

crontab -e
Add the below lines
* * * * * bash /opt/flink-1.7.0/custom/autorestart.sh
* * * * * ( sleep 30 ; bash /opt/flink-1.7.0/custom/autorestart.sh)

Verify cronjobs
crontab -l
```

2. AWS lambda monitoring for flink jobs.

```
a) The lambda function is using SES for sending emails, verify the
sender and receiver emai addresses before creating the Lambda
function
b) Create a lambda function with python3.7 run time and assign
proper IAM roles to access NIC and cloudwatch.
c) Assign proper subnets in the vpc so that the lambda can access
flink servers
d) Add the below environment variables.
ENVIRONMENT => eg: Test/ DEV
EXPECTEDJOBS => eg:
ohlcv24h,ohlcv8h,ohlcv30m,rollup1m,ohlcv1m,ohlcv5m,ohlcv15m,rollupm
ins,ohlcv72h,ohlcv1h,rolluphrs,ohlcv48h
FLINKSERVER => Internal LoadBalancer URL
RECIPIENTS => Recipients email addresses
SENDER => senders email address
e) Use python virtual env to package the below script. Zip the code
block into one file and upload it to lambda function.
f) Use the below handler for the lambda function.
flinkJobMonitoring.flinkJobMonitoring
```

3. Install Logstash and push logs into ElasticSearch for enabling centralized log monitoring.

```
Execute the below bash script in each servers.
https://github.com/xblockchainlabs/csfx-cicd/blob/master/sources/Fl
ink-Job-Monitoring/flink-logstash.sh
```