NiFi Installation and Setup Guide

# Installing NiFi

(All paths below are on the OS and not HDFS unless otherwise noted)

1. Assuming install on an edge node
2. Make sure there is a keytab giving nifi user read/write permissions to HDFS data paths, located at /home/nifi/nifi.keytab. The nifi user also needs to be able to use the Hadoop CLI.
3. In /home/nifi, do:
   1. virtualenv env
   2. . env/bin/activate
   3. pip install requests xmltodict
4. mkdir /tmp/sfdc-csv
5. mkdir /tmp/jira-json
6. Make sure the files sfdc\_creds.txt and jira\_creds.txt are in nifi’s user directory in HDFS. These have one field per line: user, password, token (for SFDC). *To change the Jira or SFDC credentials, pull these from HDFS, edit them, and replace them.*
7. Download or copy nifi-0.4.0-bin.tar.gz to /opt or /home/nifi
8. Un-tar the binary and give ownership of the dir to nifi
9. Create /storage/nifi-repos and give ownership to nifi
10. Become the nifi user, eg; $sudo -u nifi /bin/bash
11. Clone the hadoop\_bi repo (using a deploy key) into /home/nifi
12. In the NiFi install directory, copy everything from hadoop\_bi/nifi in the git repo to conf/
13. Edit the conf/bootstrap.conf file so that the service will start at the NiFi user account
    1. Edit the line and add the ‘nifi’ user to the ‘run.as=’ line in the file as follows
    2. run.as=nifi
14. ./bin/nifi.sh lists service commands for NiFi
15. Install NiFi as a Service so that it will start on reboot. Run the following.
    1. cd /home/nifi/nifi-0.4.0
    2. ./bin/nifi.sh install
    3. chkconfig nifi on
16. Modify Logging Settings in conf/logback.xml (details below)
17. Start Nifi running the following:
    1. service nifi start
18. The activity logs are in logs/nifi-app.log
19. Once the log has indicated Jetty is started, it will give an IP address for the UI
20. You can also use DNS like so:  
    <http://cdh-edge1.staging.hq.YourCompanyDomain.com:8383/nifi>
21. Make sure SMTP is available on port 25

# NiFi Logging Settings

The file **conf/logback.xml** controls the format of the how the NiFi related log files are generated and the number of files retained, including compression options. The logs files are located in the NiFi home directory under the **logs** subdirectory. By default there are three types of log files generated:

* nifi-app.log
* nifi-bootstrap.log
* nifi-user.log

**Note:** All log files have had compression enabled for the rolled files. This is enabled by adding the **.gz** suffix to the log filename.

## Current Log Settings and Recommendations

### nifi-app.log

The nifi-app.log is currently set to cycle on an hourly basis with files being rolled to a new sequence number after it reaches 100MB.

<fileNamePattern>./logs/nifi-app\_%d{yyyy-MM-dd\_HH}.%i.log.gz</fileNamePattern>

<maxFileSize>100MB</maxFileSize>

<maxHistory>30</maxHistory>

Currently, the files are rolling as follows:

-rw-rw-r-- 1 nifi nifi 835 Jan 26 08:00 nifi-app\_2016-01-26\_06.0.log

-rw-rw-r-- 1 nifi nifi 4617149 Mar 13 15:59 nifi-app\_2016-03-13\_15.0.log

-rw-rw-r-- 1 nifi nifi 4617173 Mar 13 16:59 nifi-app\_2016-03-13\_16.0.log

This equates to only 30 hours of log files for the applications running.

**Recommendation**: Change the logging to daily and increasing the size to 250MB to prevent the logs from splitting over multiple files. The peak log size over the last 24 hours as just over 8MB with most just under 5MB.

With compression enabled, the last log file was reduced from just over 5MB to less than .5MB as shown below.

-rw-rw-r-- 1 nifi nifi 3460420 Mar 14 19:59 nifi-app\_2016-03-14\_19.0.log

-rw-r--r-- 1 nifi nifi 4884575 Mar 14 20:59 nifi-app\_2016-03-14\_20.0.log

-rw-r--r-- 1 nifi nifi 493499 Mar 14 22:00 nifi-app\_2016-03-14\_21.0.log.gz

-rw-r--r-- 1 nifi nifi 624 Mar 14 22:00 nifi-app.log

Estimating future growth in log sizes, with compression the daily log size should be less than 20MB/day. Keeping 30 days will consume only 600MB of space as well as the uncompressed daily file (up to 850MB in total).

### nifi-bootstrap.log

The nifi-bootstrap.log is currently set to cycle on a daily basis.

<fileNamePattern>./logs/nifi-bootstrap\_%d.log.gz</fileNamePattern>

<maxHistory>5</maxHistory>

The logs are running around 2 KB/day with 5 files being retained.

### nifi-user.log

The nifi-user.log is currently set to cycle on a daily basis with a maximum of 30 files.

<fileNamePattern>./logs/nifi-user\_%d.log.gz</fileNamePattern>

<maxHistory>30</maxHistory>

Each daily file is up to 2MB in size before compression.

## Log Changes to logback.xml

The following are the changes made to the **logback.xml** configuration file.

### nifi-app.log

<fileNamePattern>./logs/nifi-app\_%d.%i.log.gz</fileNamePattern>

<maxFileSize>250MB</maxFileSize>

<maxHistory>30</maxHistory>

### nifi-bootstrap.log

<fileNamePattern>./logs/nifi-bootstrap\_%d.log.gz</fileNamePattern>

<maxHistory>5</maxHistory>

### nifi-user.log

<fileNamePattern>./logs/nifi-user\_%d.log.gz</fileNamePattern>

<maxHistory>30</maxHistory>

# Manually Starting and Stopping NiFi

1. Script location
   * Edge node only - folder location: /home/nifi/nifi-0.4.0
   * Commands: Usage ./bin/nifi.sh {start|stop|run|restart|status|dump|install}

After configuring NiFi to run as a service, the commands above will return failures saying the user ‘nifi’ is not in the sudoers file

* + **# service nifi status :** will return the port and PID for the running instance.
  + **# service nifi start :** starts NiFi in the background
  + **# service nifi stop :** stop: stops NiFi that is running in the background

1. Permissions and checking for interruptions
   * Nifi needs to be run as “nifi” user; to become nifi user, use this command:
   * $sudo -u nifi /bin/bash
2. /logs
   * Check for memory issues, failed jobs, or other errors.
   * Cloudera Manager may offer some insights into errors also. Check the related service management logs
3. Additional notes for users + developers
   * The configuration that is to be used when launching NiFi, such as Java heap size, the user to run as, which Java command to use, etc. are configurable via the conf/bootstrap.conf file.
   * Documentation can be found in the /home/nifi/nifi-0.4.0/docs directory.

# Updating configs on the existing Flow (including DNS records):

1. Use shift-click-drag or right-click to highlight the processors in question.
2. Click the red “Stop” button on the UI menu or right-click menu
3. Modify the processor config if needed. The DNS update for Mongo just needs the processor to be restarted.
4. Repeat steps 1 & 2 with the green “Play” button.

# Making changes to the NiFi Flow:

1. Make sure all ingest process are stopped. Use stop buttons in NiFi UI if necessary.
2. service nifi stop
3. If updating flow.xml.gz  
   rm -rf /storage/nifi-repos/\*
4. Copy config file(s) to conf/
5. service nifi start
6. Once the UI has come up, stop the “PutEmail” processor and change the “To:” property to the correct cluster (prod-cluster, staging-cluster, or r-d-cluster)@YourCompanyDomain.pagerduty.com

# How Alerting Works:

There are two categories of processor: built-in NiFi processors and ExecuteProcess.

1. Built-in NiFi processors are the simplest case. Errors in these are available from the NiFi bulletin rest API. There is a GetHTTP processor polling that endpoint every 2 minutes.
2. For ExecuteProcess nodes, there is an option to route stderr out of the processor.

The text emitted by 1 is a JSON blob, for 2 it is whatever the outside process logs on stderr. These sources are fed into a RouteText processor, that uses a regex to look for “ERROR”. If it finds an error, it is routed to a PutEmail processor which sends to pagerduty.

To add alerting for a processor of type 1, no action is needed. For type 2, make sure that either it emits error text containing ERROR on stderr, or add another rule to the RouteText processor to accommodate the log message you are looking for.

# How to Handle Failures:

There are two options here, once you are ready to retry.

1. Stop the offending processor, modify its cron schedule to run soon, then restart it. You will have to stop it again when finished and change the schedule back.
2. Except for the GetMongo processors, all other jobs have an associated shell script. Look at the configuration of the processor to get the location of the shell script and any arguments, then run the command manually (as nifi user). In the case of Mongo, you could run the mongoexport command manually.

# Job Schedule

|  |  |  |
| --- | --- | --- |
| **Ingest (I) or Analytic (A)** | **Start Time (ET)** | **Duration (Approx)** |
| Jira (I) | 0300 | < 1 min |
| SFDC (I) | 0300 | 1 hr (all objects) |
| blueprints (I) | 0000 | < 1 min |
| chartdata (I) | 0000 | 34 min |
| dual\_lift (I) | 0000 | < 1 min |
| eda (I) | 0000 | 1 hr 4 min |
| eda\_map (I) | 0100 | 2 min |
| events (I) | 0100 | < 1 min |
| instance\_log (I) | 0100 | < 1 min |
| instance\_requests (I) | 0100 | < 1 min |
| instances (I) | 0200 | < 1 min |
| job\_history (I) | 0200 | 3 min |
| leaderboard (I) | 0200 | 8 hr |
| metablueprint (I) | 0200 | 23 min |
| metadata (I) | 0300 | 9 min |
| metrics\_data (I) | 0300 | < 1 min |
| model\_access (I) | 0300 | < 1 min |
| model\_code (I) | 0300 | < 1 min |
| model\_xray (I) | 0400 | 5 min |
| open\_model\_code (I) | 0400 | < 1 min |
| organization (I) | 0400 | < 1 min |
| organization\_log (I) | 0400 | < 1 min |
| predict\_code (I) | 0500 | 8 min |
| prediction\_tabulation (I) | 0500 | < 1 min |
| predictions (I) | 0500 | 20 min ? |
| project (I) | 0500 | < 1 min |
| project\_clone (I) | 0500 | < 1 min |
| proxy\_auth (I) | 0600 | < 1 min |
| request\_tracker (I) | 0600 | 5 min |
| storage (I) | 0600 | 14 min |
| users (I) | 0600 | < 1 min |
| Upload Events (A) | 0800 | < 1 min |
| Project Report (A) | 0700 | 10 min |
| Opportunity (A) | 0400 | < 1 min |
| Case (A) | 0500 | < 1 min |