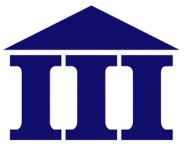
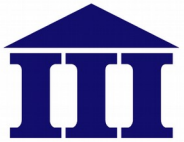
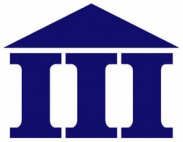


Data Flow in Hadoop – Apache NiFi



About Me

- Jacob
- Work for III
- NYUST dept. IM
- Hadoop, Python, R, etc..



About Me

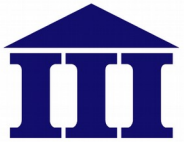
FB: 陳建融

Web: http://chenjr-jacob.idv.tw/

GitHub: https://github.com/chenjr0719

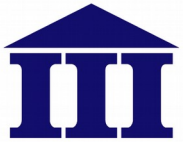
Mail: jacobchen@iii.org.tw

or: chenjr0719@gmail.com

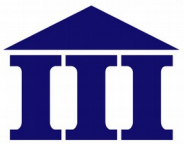


Contents

1. Backgrounds
2. Hortonworks DataFlow (HDF)
3. Hands On

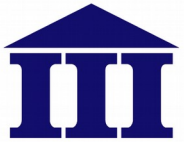


Backgrounds



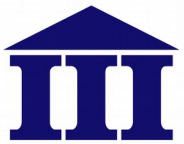
Situation

- Sensor -> Database -> Backend

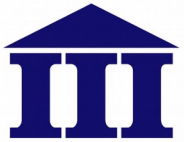


And More..

- App -> Database -> Backend
- Online Shop -> OLAP -> Backend

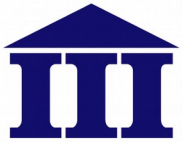


This is **FLOW**.



What is Data Flow

- Source
- Sink
- Between **Source** and **Sink** – **Pipeline**.

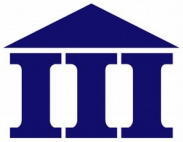


In real world

- The data from source is **Real Time**.
- And this also called **Streaming**.

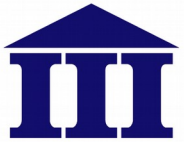
Common data source

- Facebook, Twitter (SNS)
- Factory machine (Sensor)
- PTT, Dcard, News (Web)
- App
- IoT



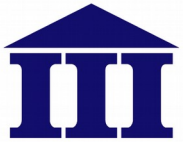
Role of Hadoop

- Data Warehouse
- Process Platform
- Analysis Platform
- Most of them is about **Batch** processing.



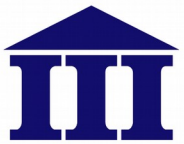
But..

Streaming is more important than before.

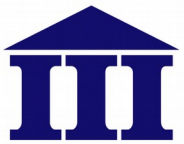


Now..

- Many projects aim to deal with **Streaming**.
  - NiFi
  - Spark Streaming
  - Flink
  - Storm
- And more..



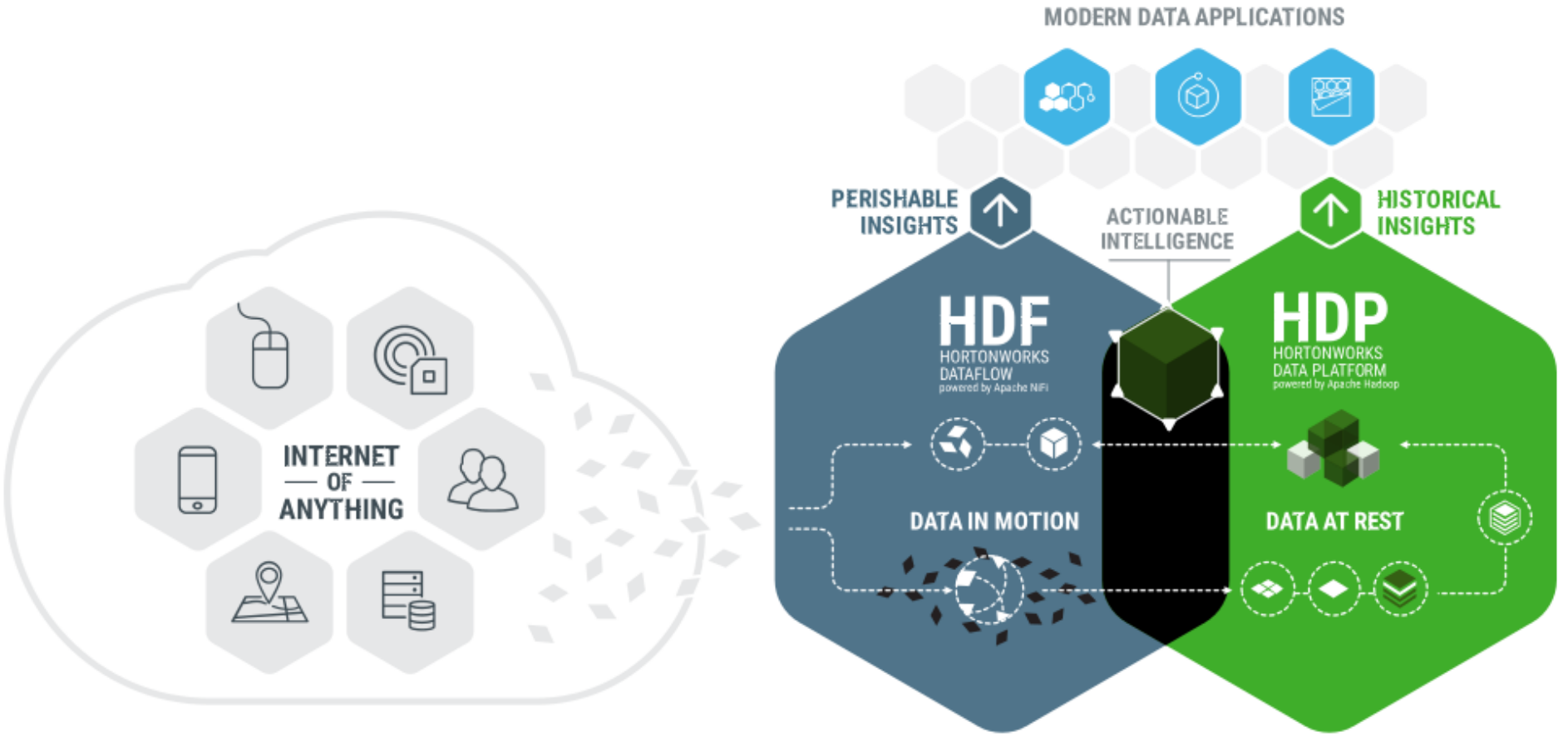
Hortonworks DataFlow (HDF)



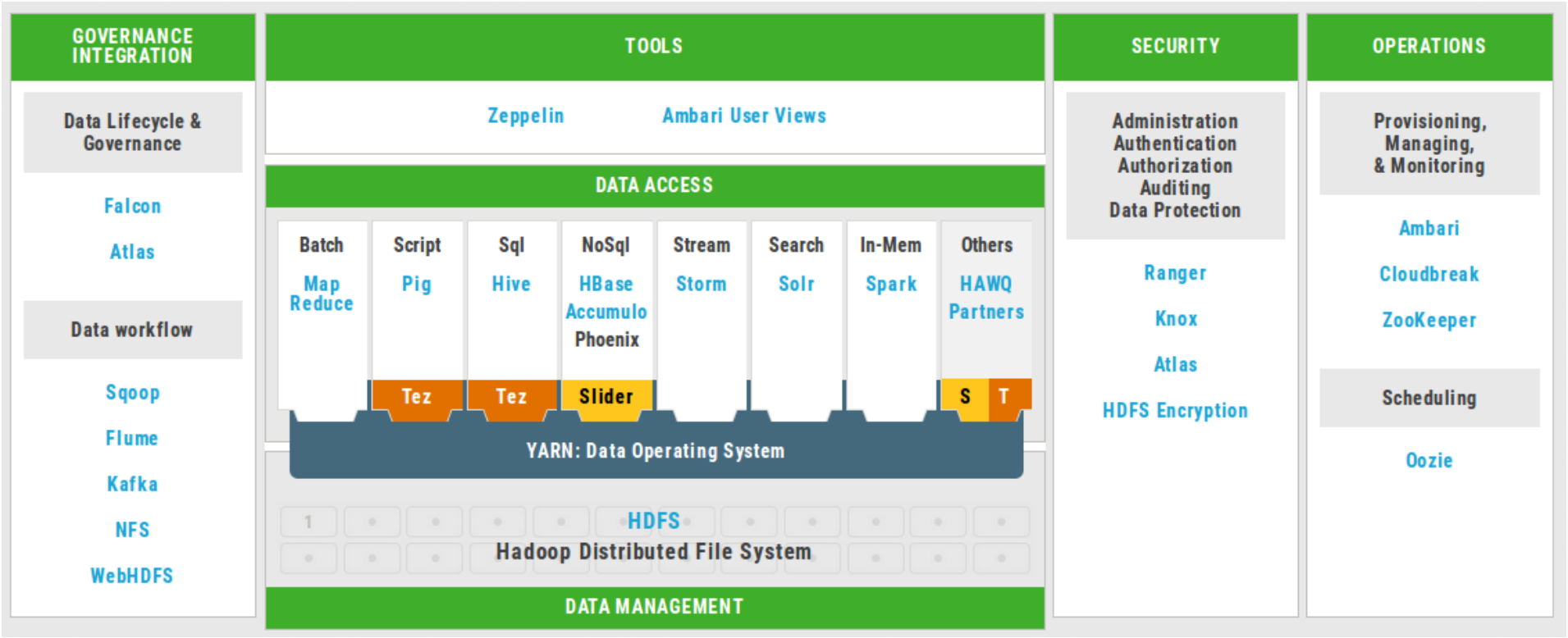
Hortonworks

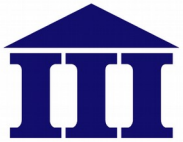
- The company focuses on the development and support of **Apache Hadoop**.
- Hortonworks Data Platform (HDP®)
- Hortonworks DataFlow (HDF™)

Hortonworks Solution



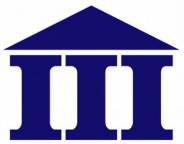
Hortonworks Overviews





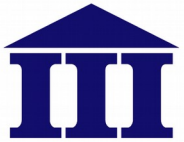
HDF

- ZooKeeper
- Storm
- Ambari Infra
- Kafka
- NiFi
- https://youtu.be/-2Zw3n1qQUA



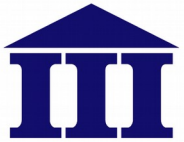
Apache NiFi

- An easy to use, powerful, and reliable system to process and distribute data.
- Web UI
- Highly configurable



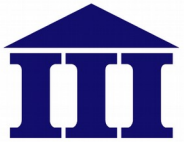
Apache Zookeeper

- A centralized service.
- Providing distributed synchronization
- Providing group services



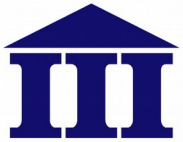
Storm

- Distributed realtime computation system.
- Streaming.
- Can be used with any programming language.



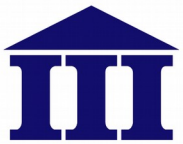
Ambari Infra

- Provide common shared services for stack components

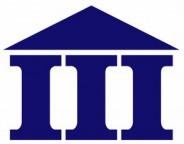


Kafka

- Publish-subscribe messaging system.
- Fast
- Scalable

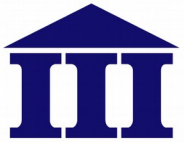


Hands On



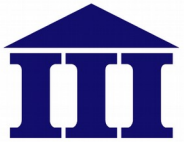
Hands On

1. Install HDF
2. NiFi with Kafka
3. HDF to HDP
4. Let’s Tweet
5. Pre-Process
6. ExecuteProcess
7. To Spark Streaming



Install HDF

```
sudo wget -nv http://public-repo-1.hortonworks.com/ambari/centos6/2.x/updates/2.4.0.1/ambari.repo -O /etc/yum.repos.d/ambari.repo

sudo yum -y install ambari-server
```

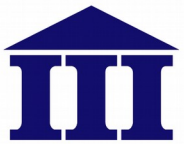


Install HDF

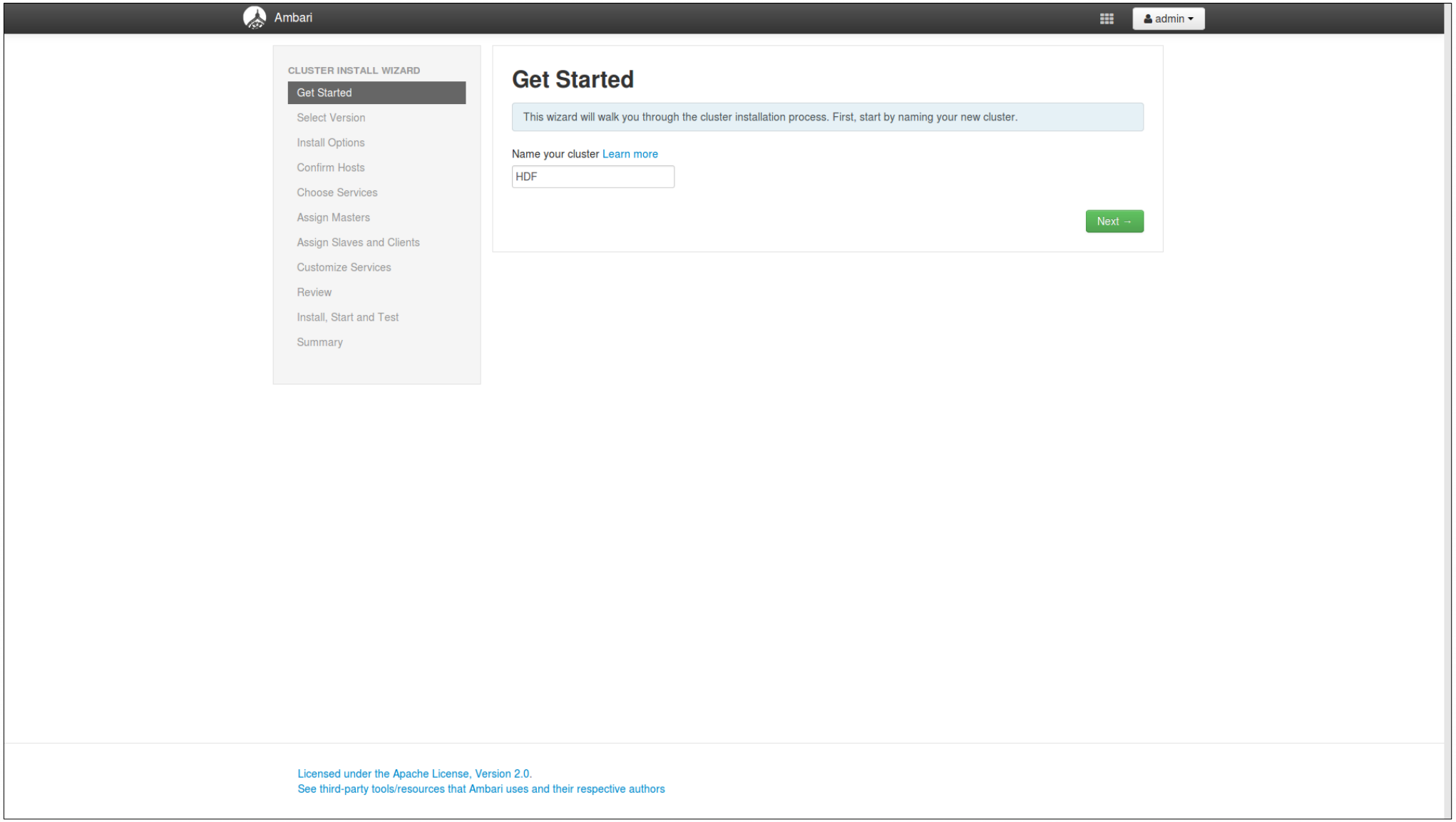
sudo ambari-server setup -s

sudo ambari-server install-mpack --mpack=http://public-repo-1.hortonworks.com/HDF/centos6/2.x/updates/2.0.0.0/tars/hdf_ambari_mp/hdf-ambari-mpack-2.0.0.0-579.tar.gz --purge --verbose

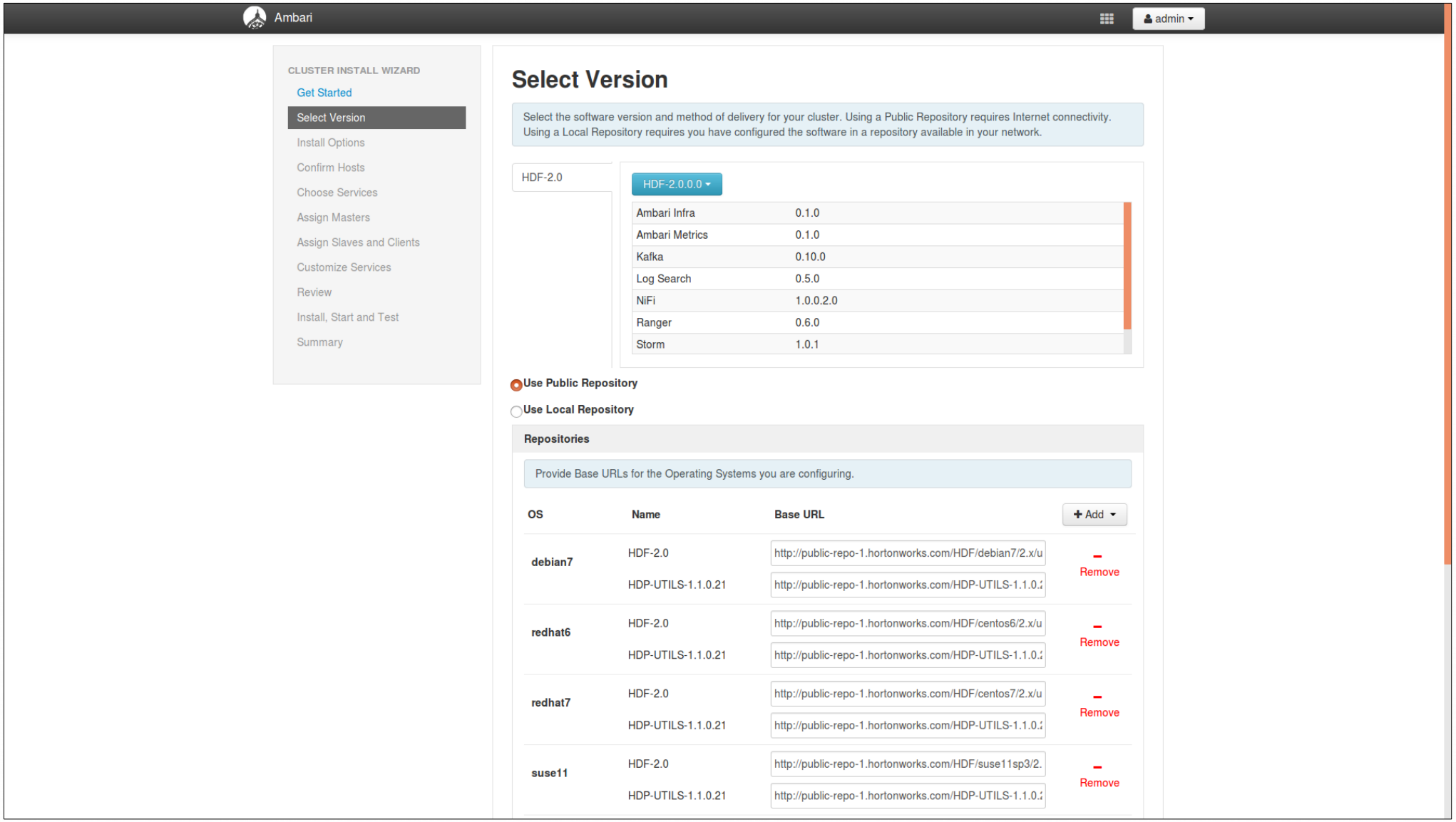
sudo ambari-server start



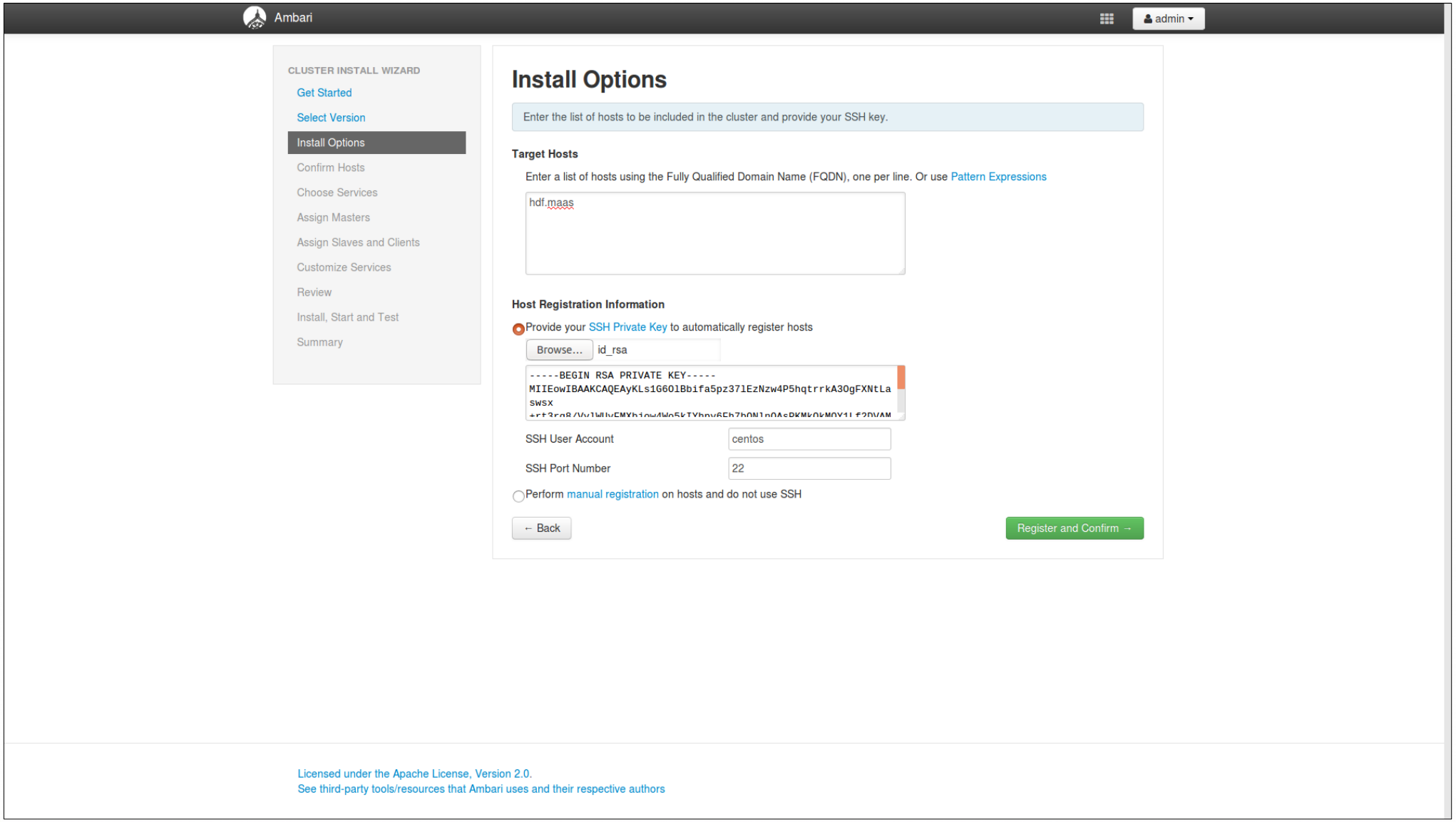
Install HDF



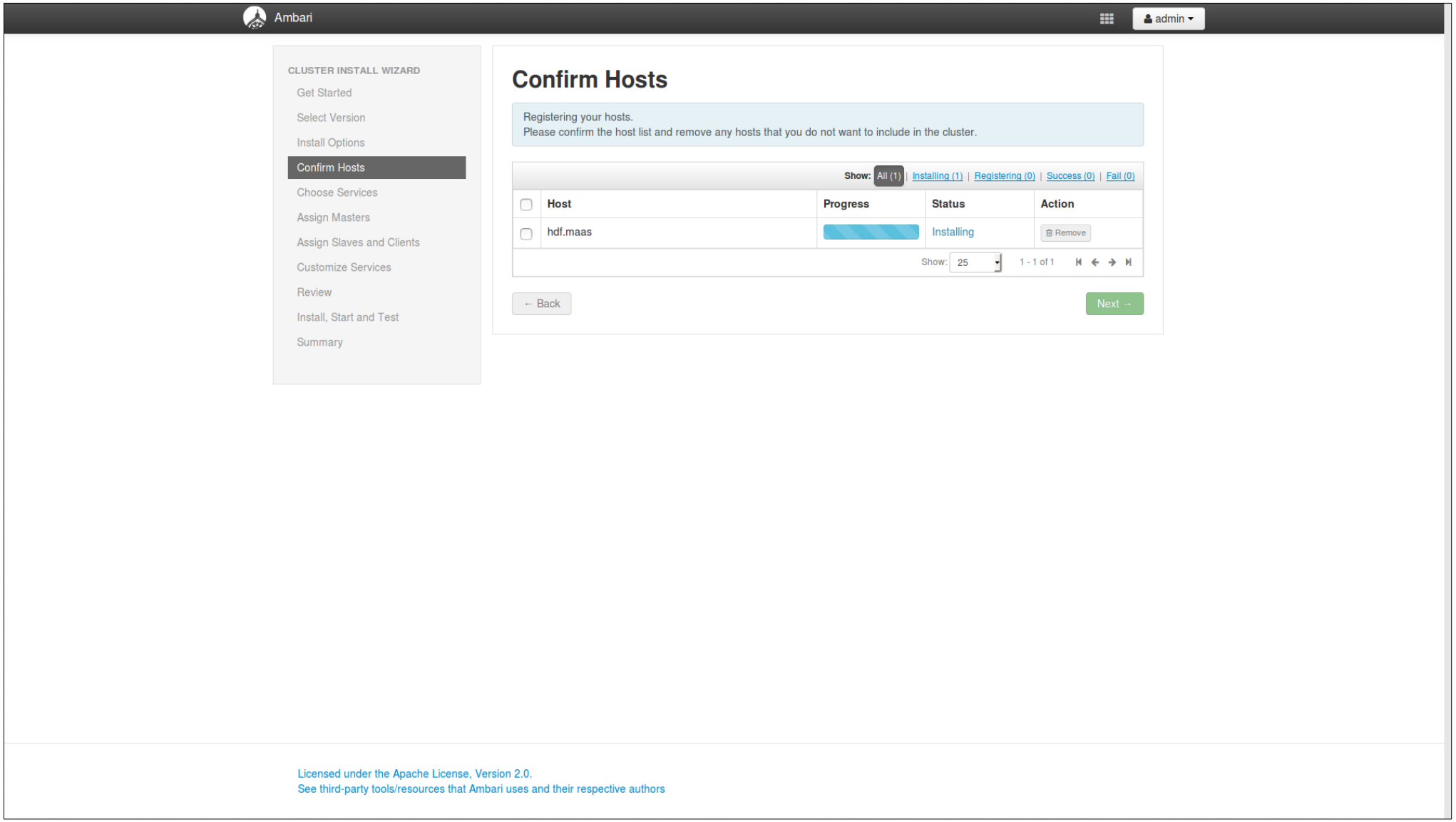
Install HDF



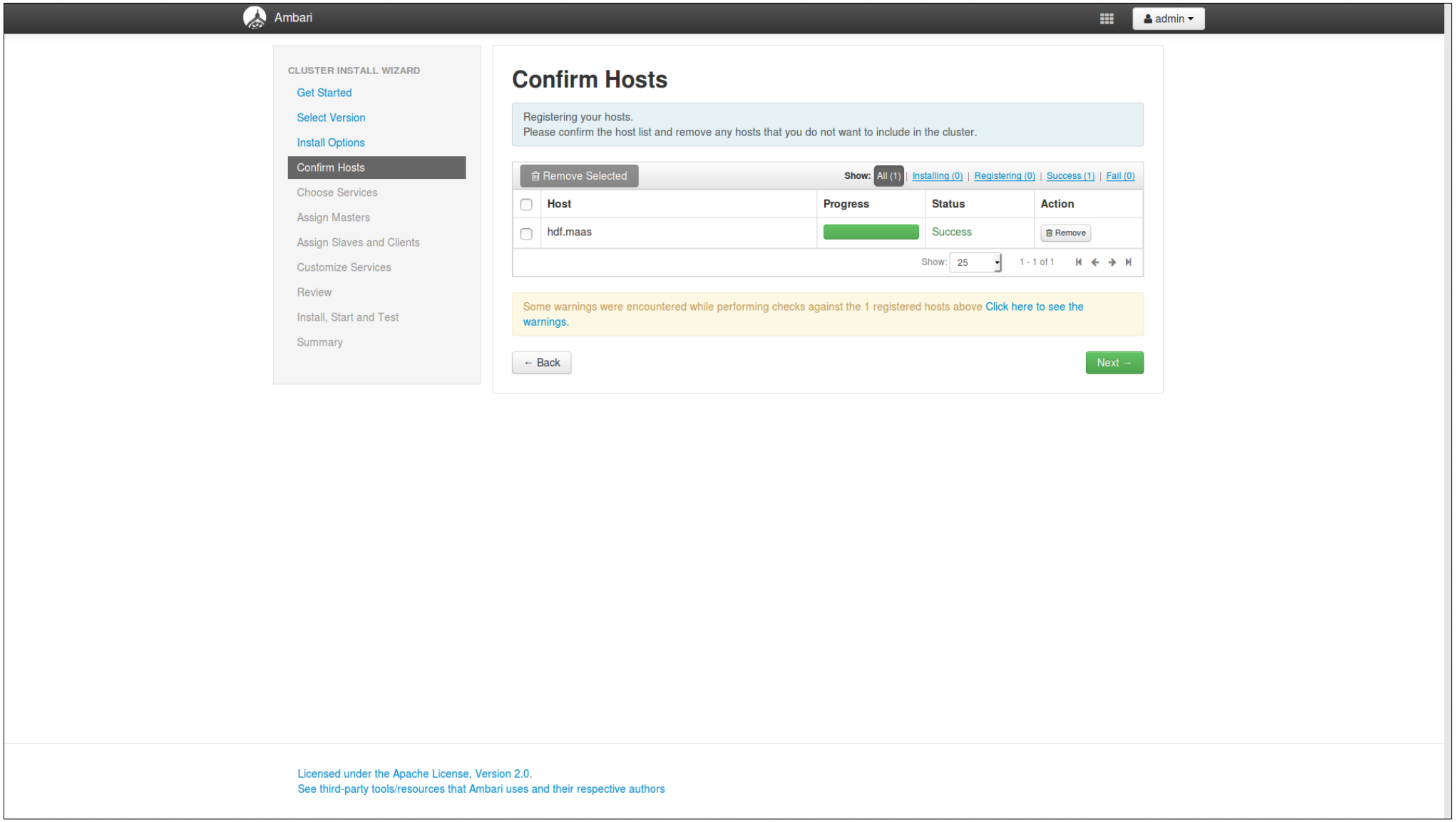
Install HDF



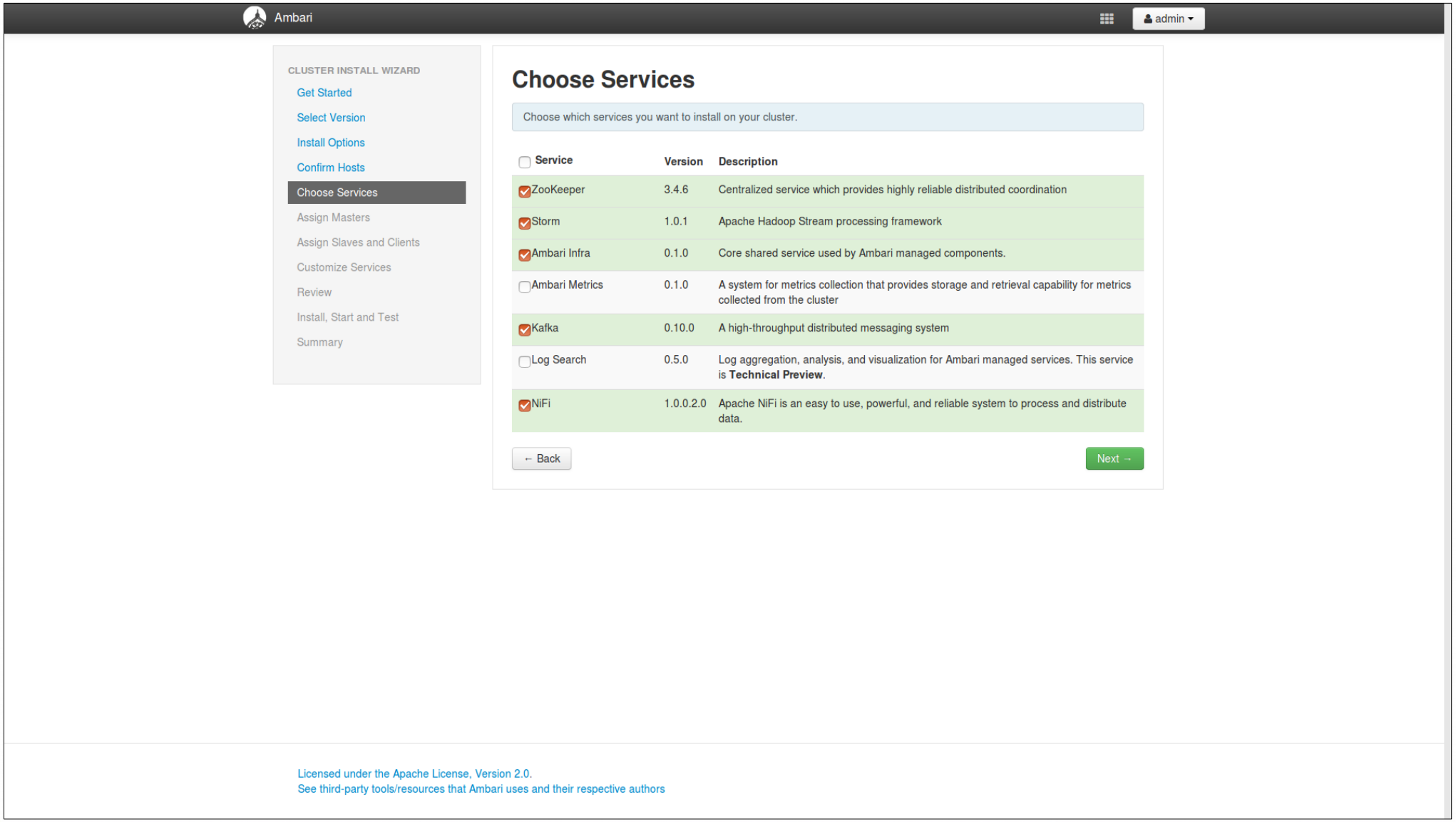
Install HDF



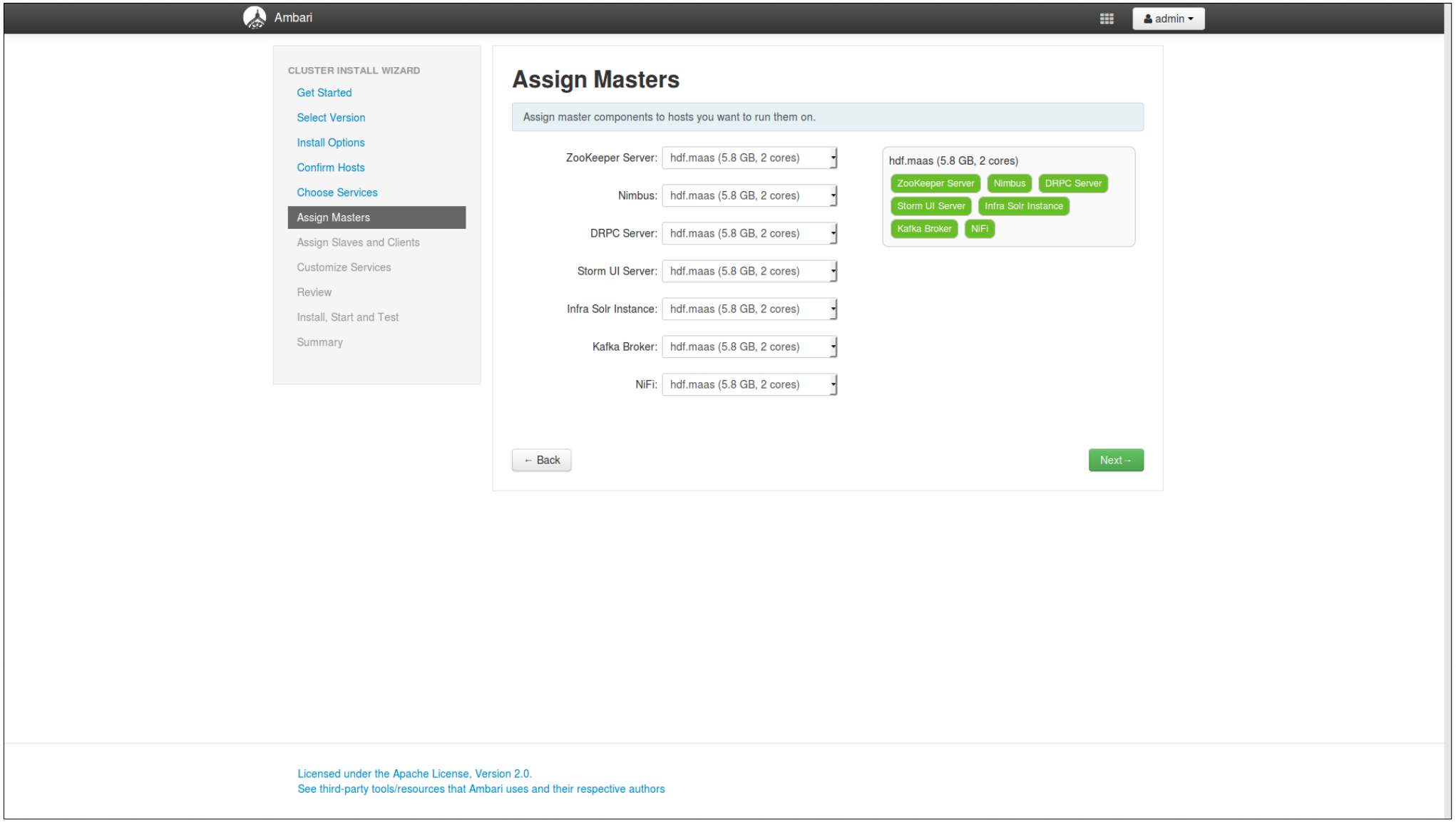
Install HDF



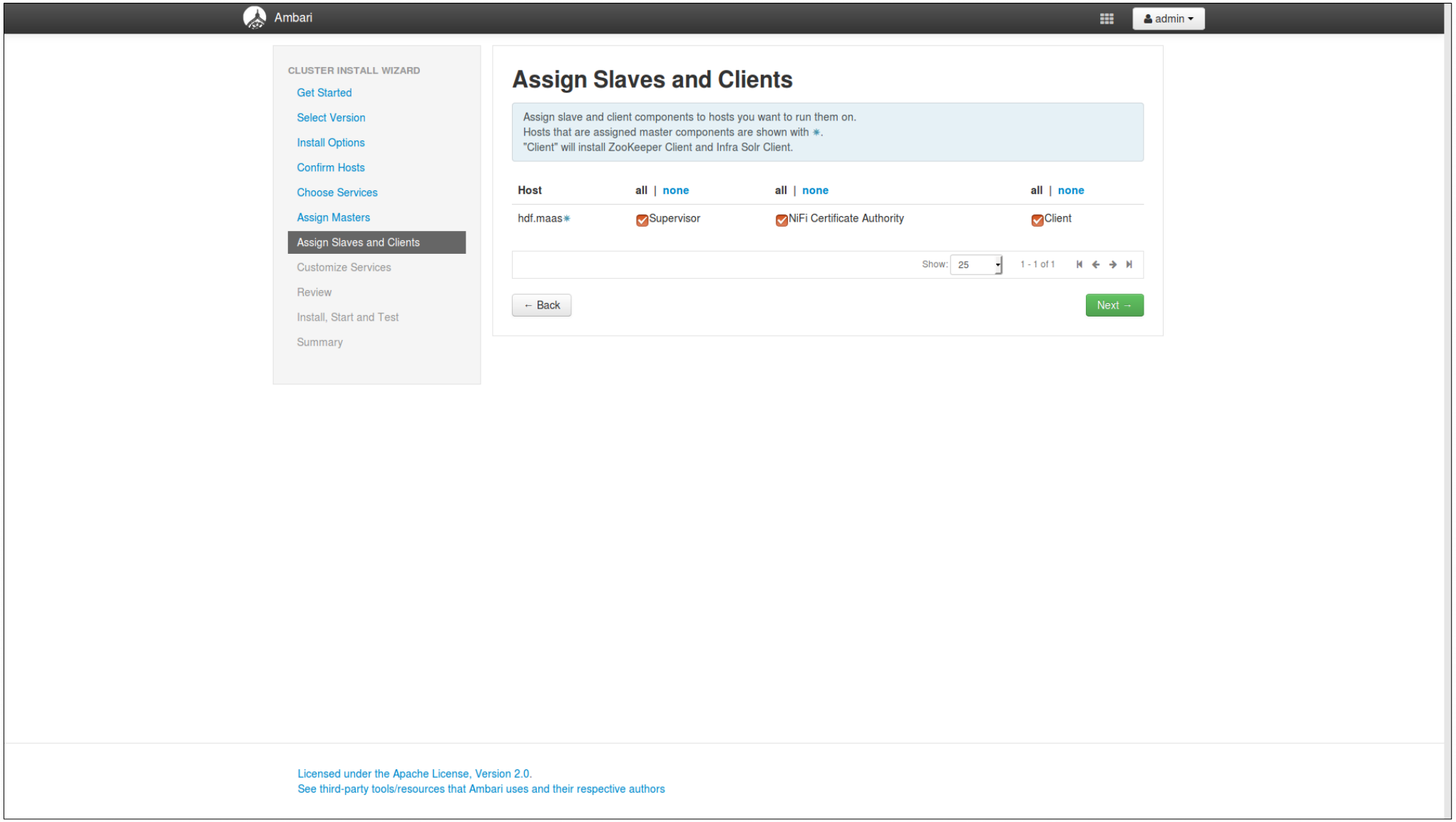
Install HDF



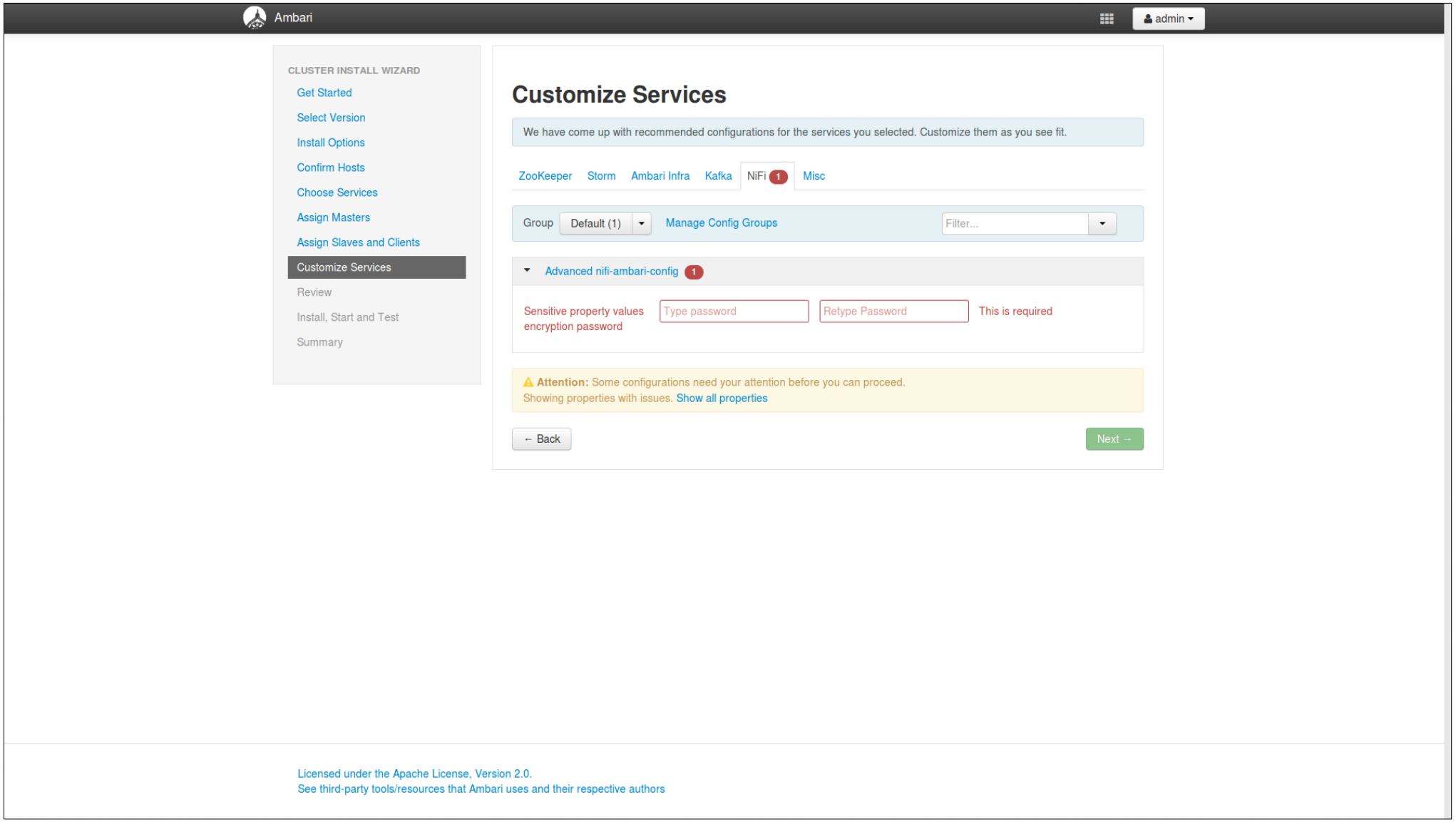
Install HDF



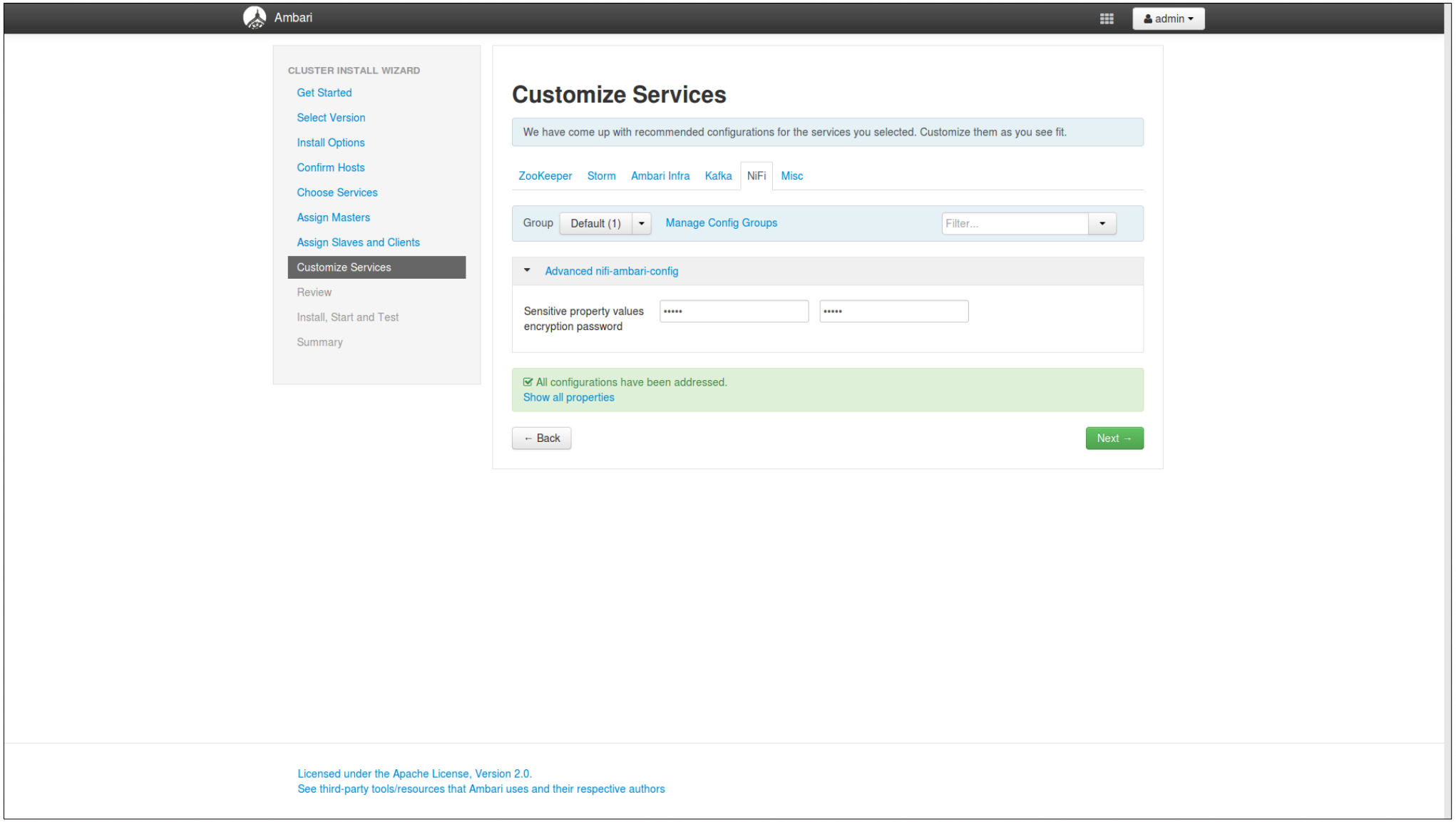
Install HDF



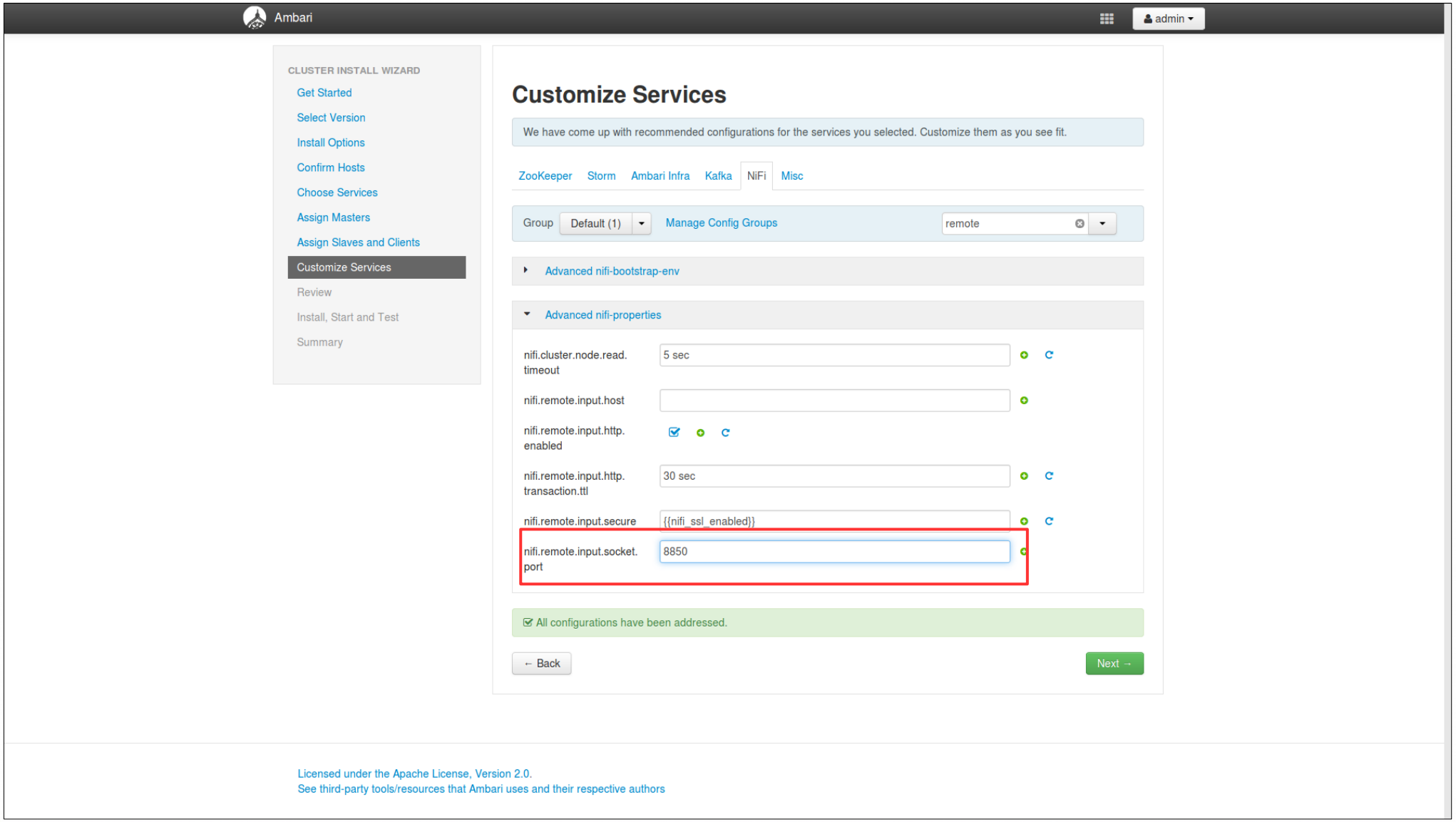
Install HDF



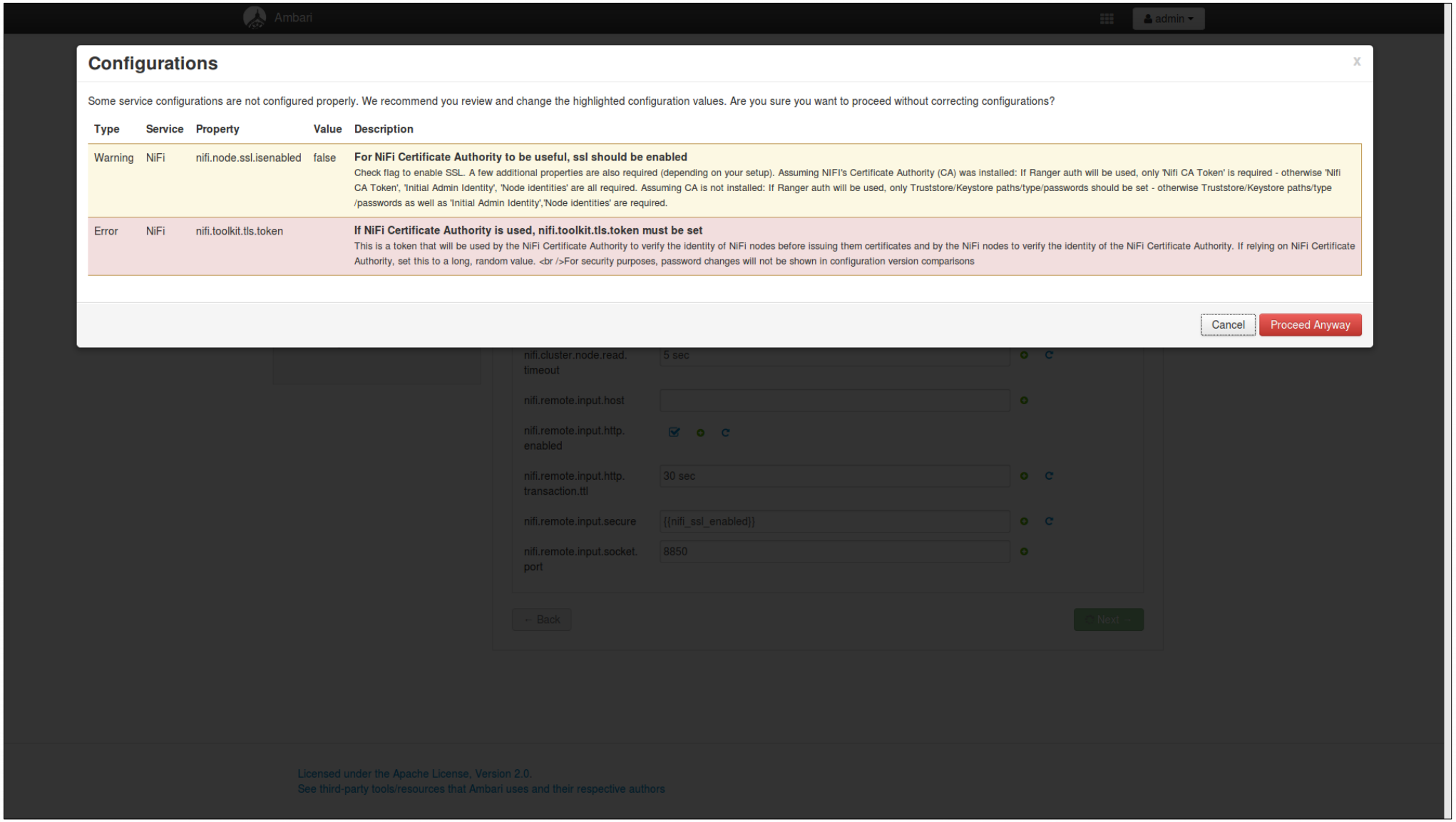
Install HDF



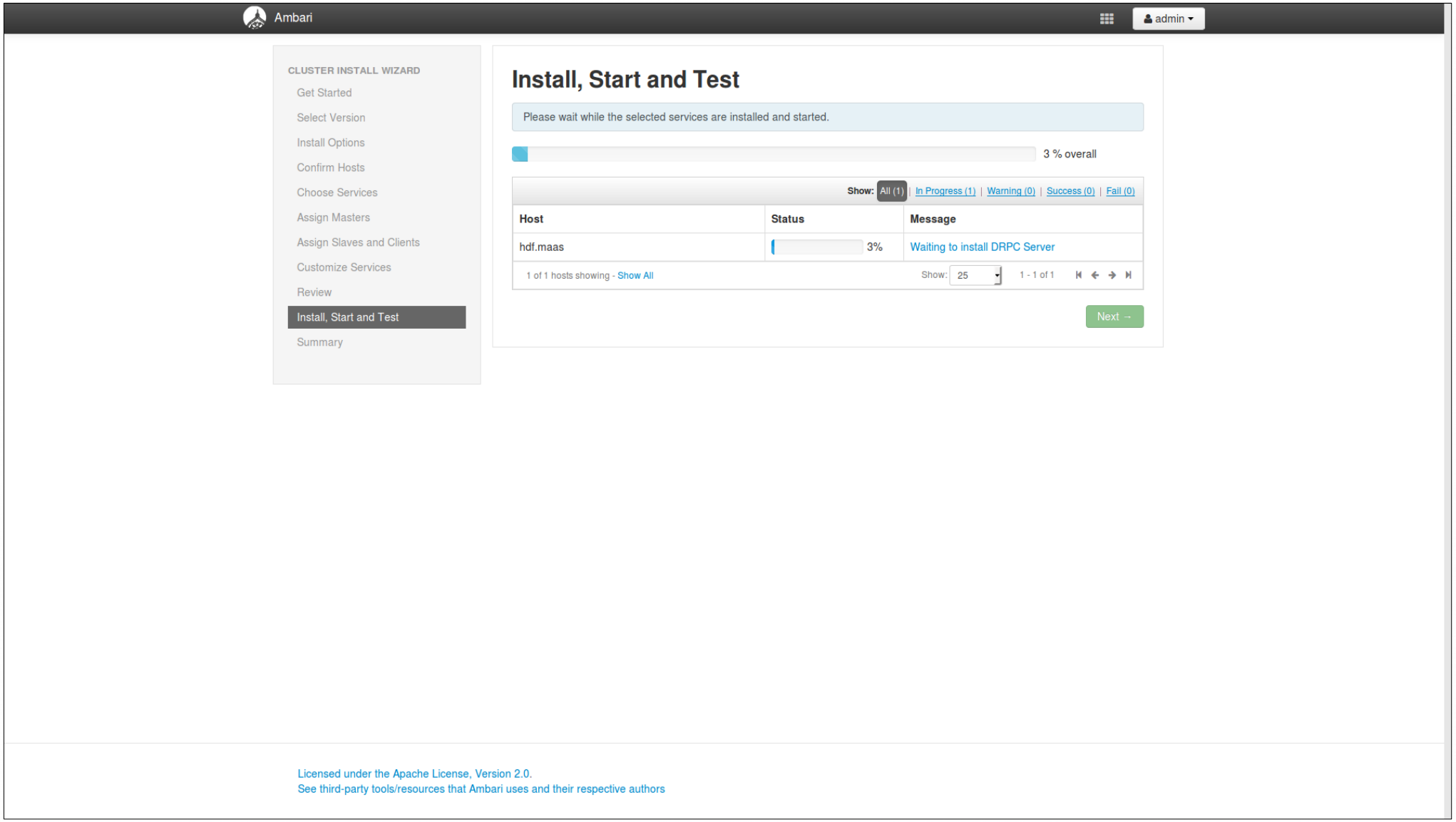
Install HDF



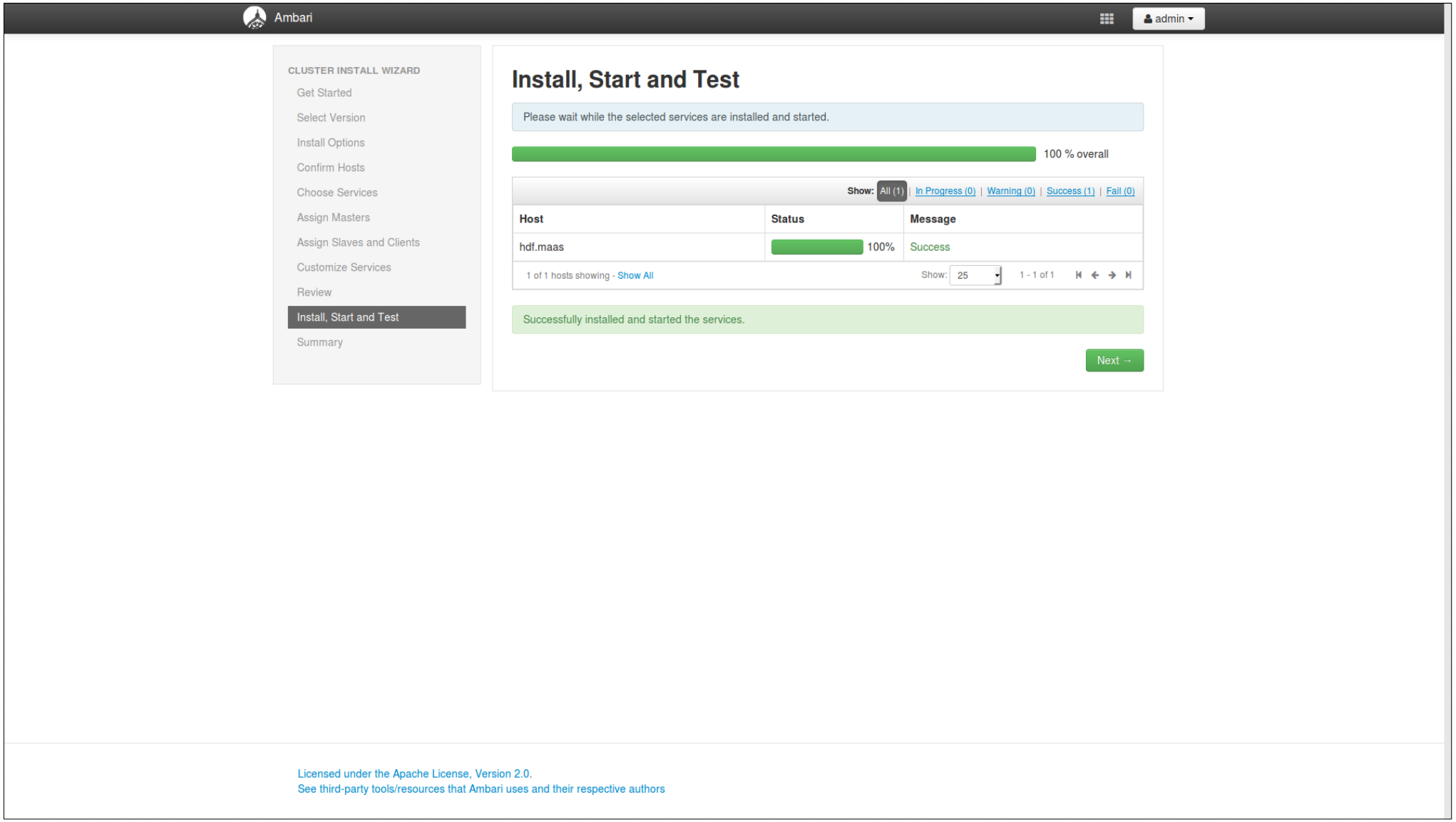
Install HDF



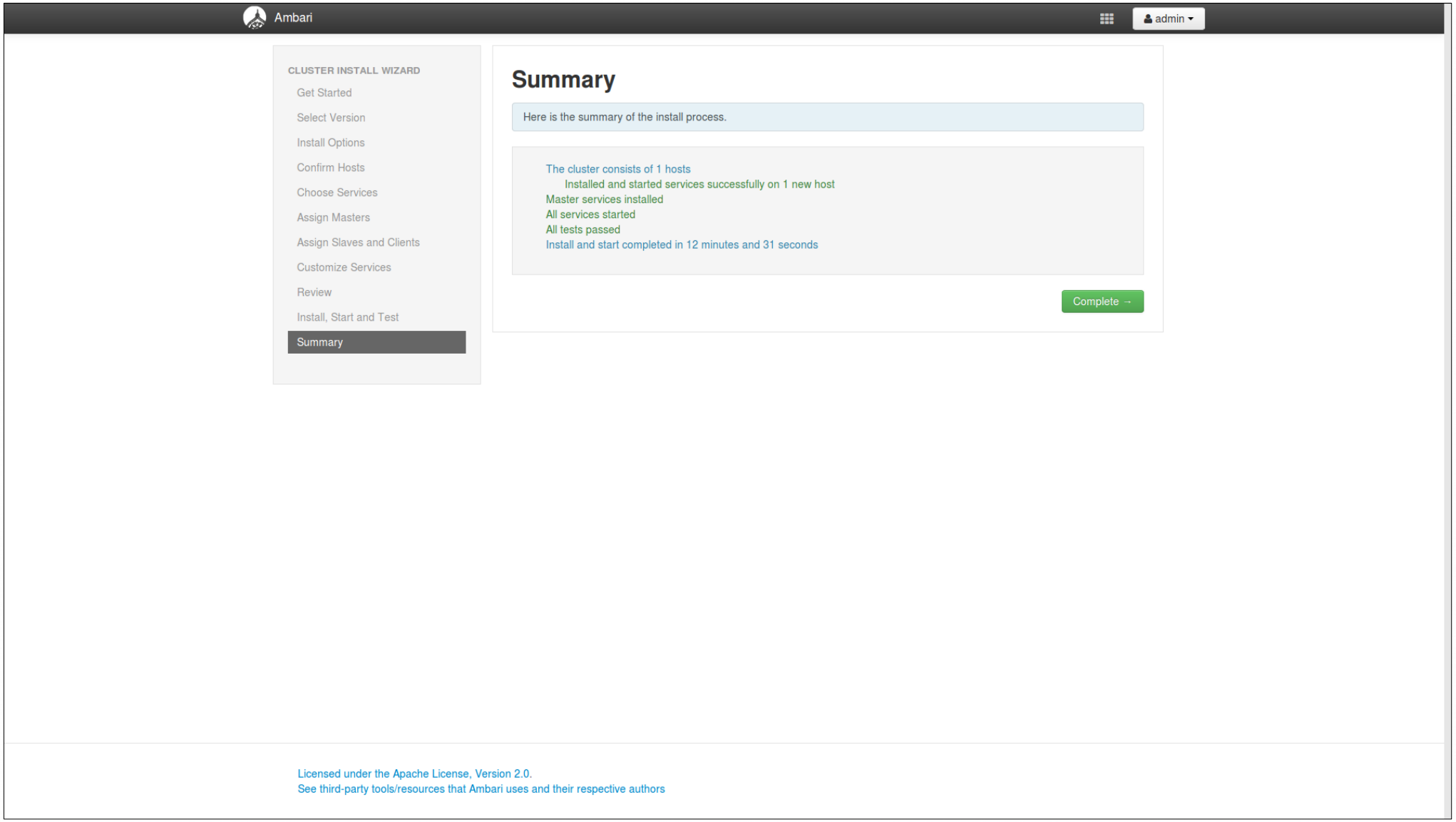
Install HDF



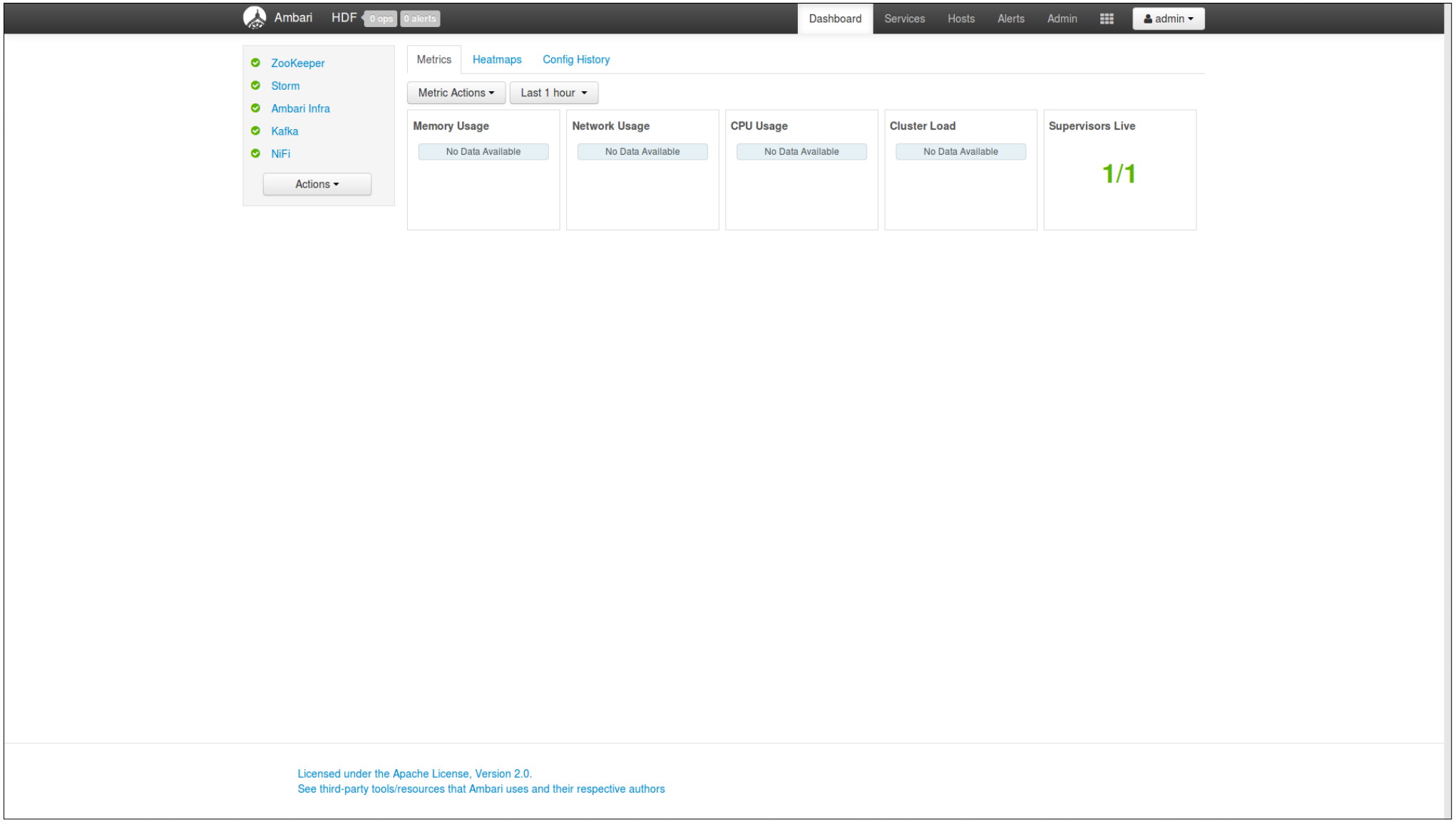
Install HDF



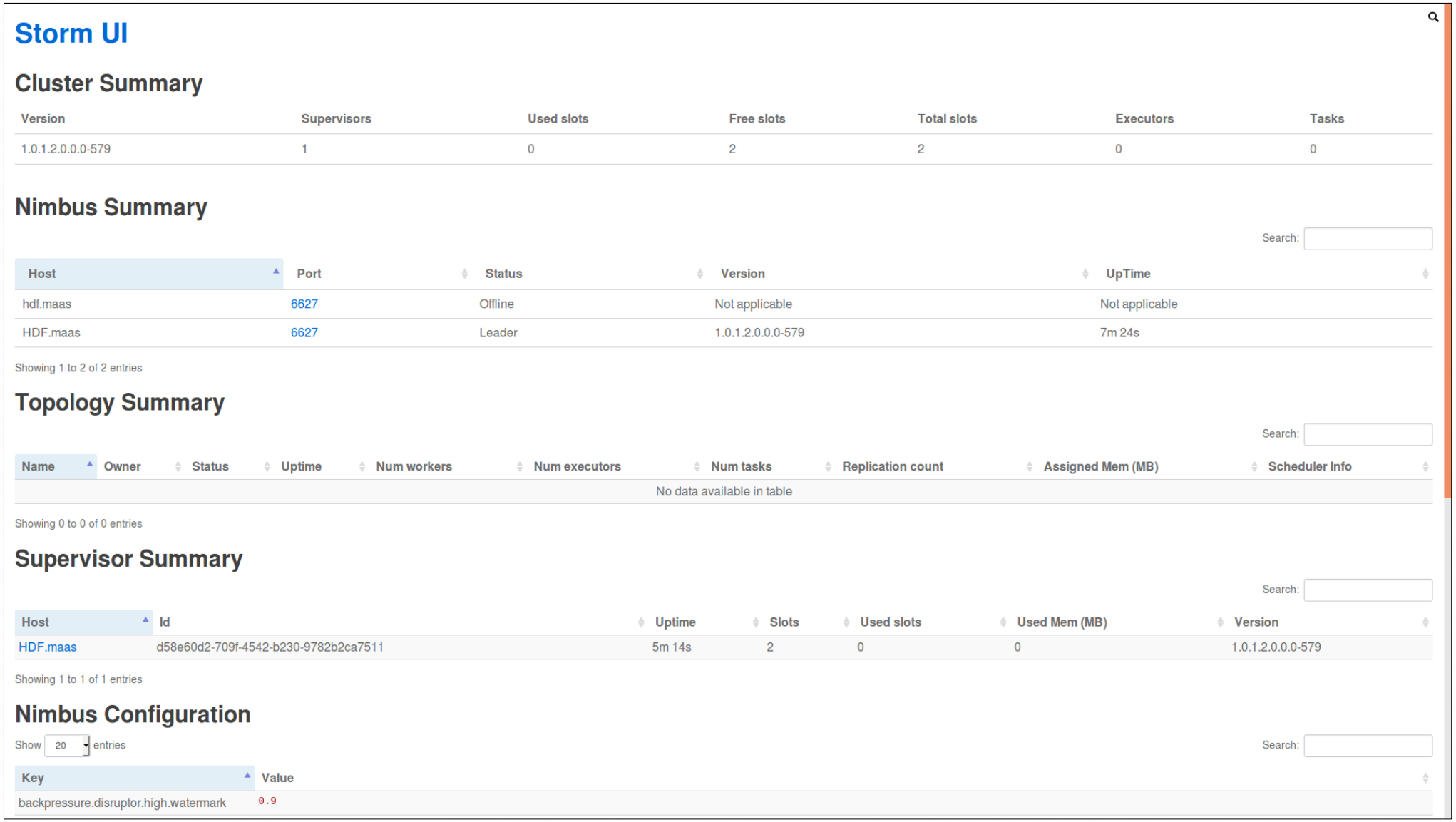
Install HDF



Result



Result



Storm UI

Cluster Summary

Version	Supervisors	Used slots	Free slots	Total slots	Executors	Tasks
1.0.1.2.0.0.0-579	1	0	2	2	0	0

Nimbus Summary

Search:

Host	Port	Status	Version	UpTime
hdf.maas	6627	Offline	Not applicable	Not applicable
HDF.maas	6627	Leader	1.0.1.2.0.0.0-579	7m 24s

Showing 1 to 2 of 2 entries

Topology Summary

Search:

Name	Owner	Status	Uptime	Num workers	Num executors	Num tasks	Replication count	Assigned Mem (MB)	Scheduler Info
No data available in table									

Showing 0 to 0 of 0 entries

Supervisor Summary

Search:

Host	Id	Uptime	Slots	Used slots	Used Mem (MB)	Version
HDF.maas	d58e60d2-709f-4542-b230-9782b2ca7511	5m 14s	2	0	0	1.0.1.2.0.0.0-579

Showing 1 to 1 of 1 entries

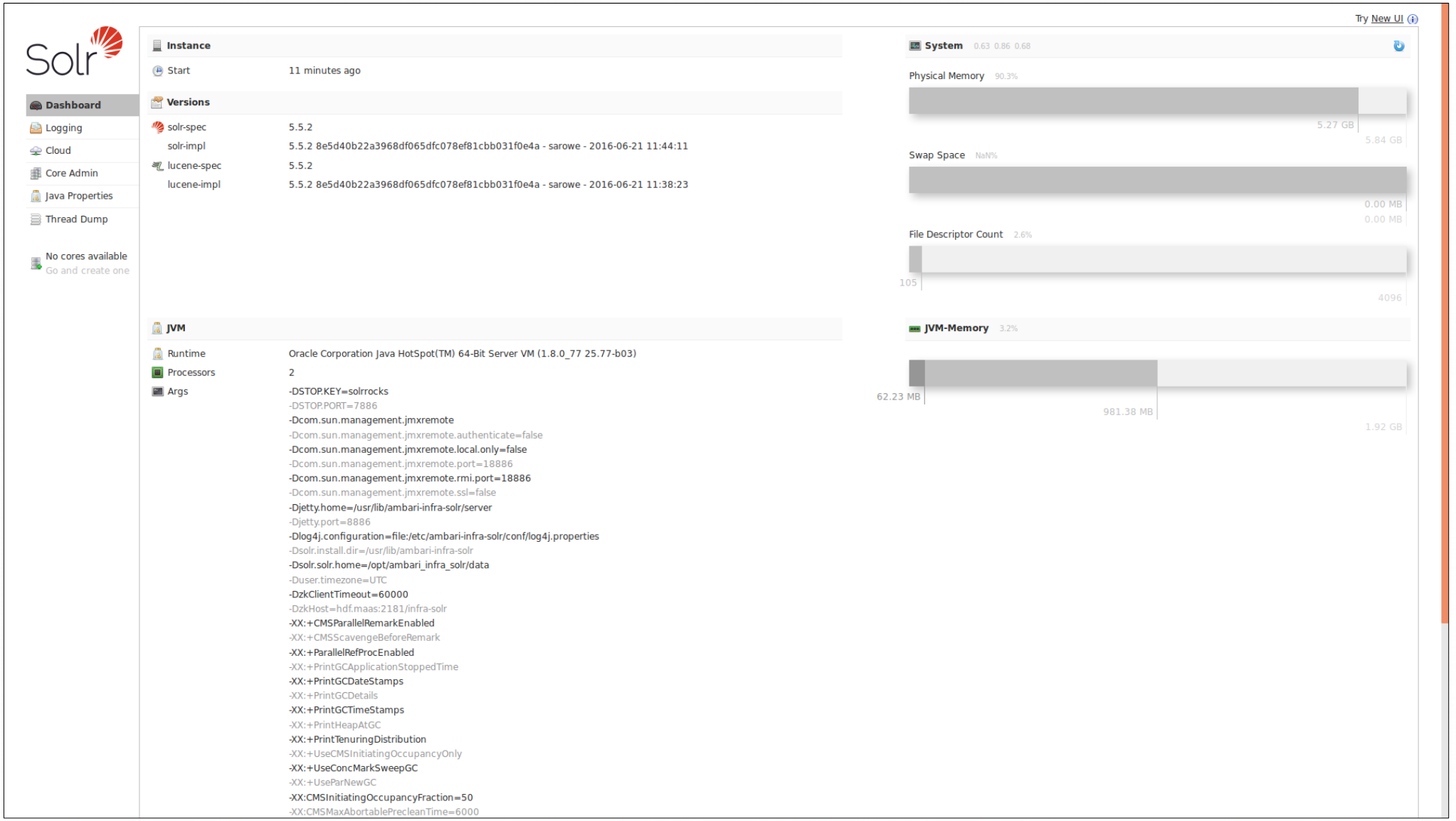
Nimbus Configuration

Show 20 entries

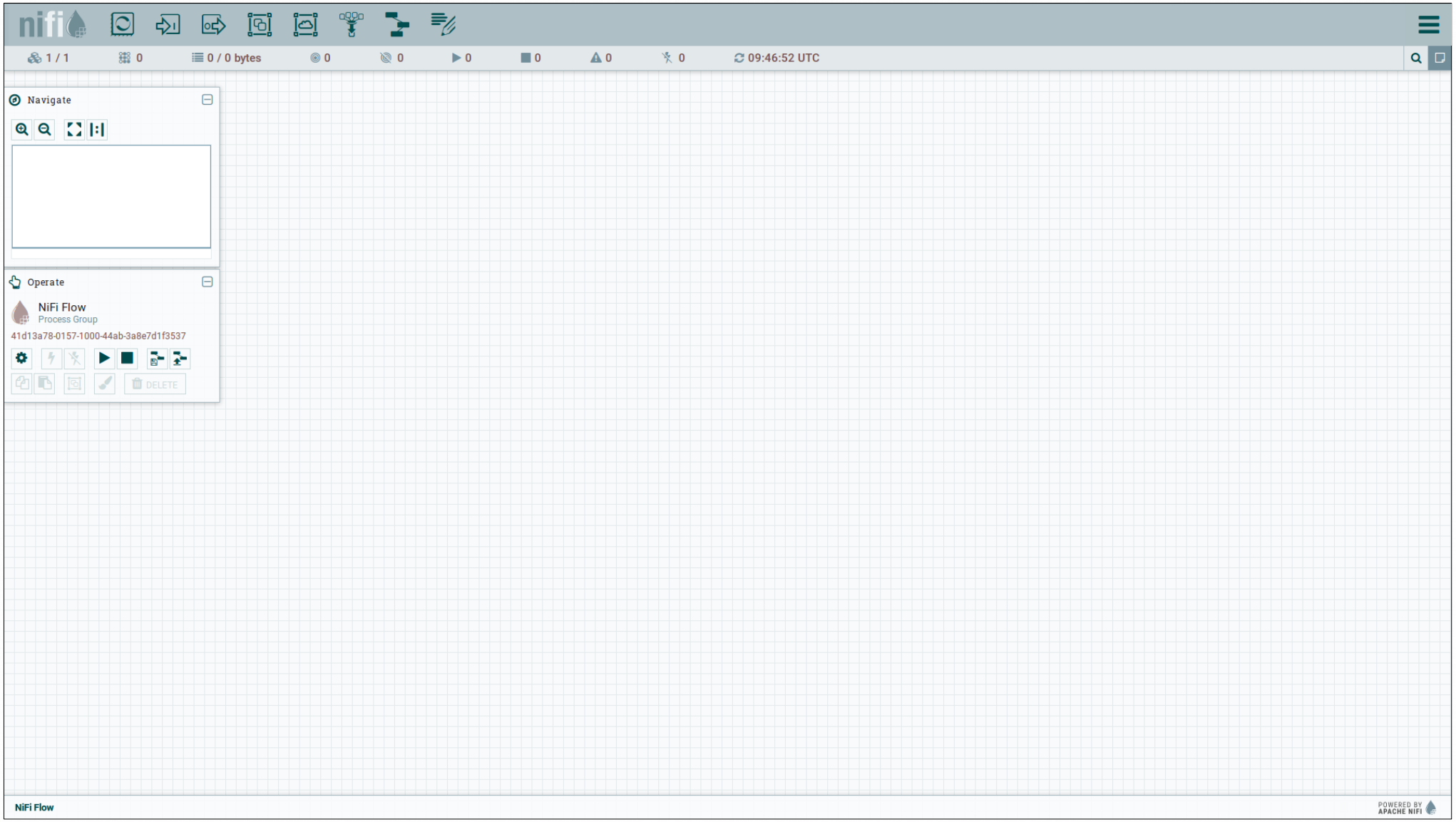
Search:

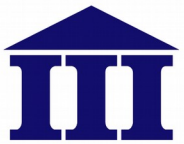
Key	Value
backpressure.disruptor.high.watermark	0.9

Result

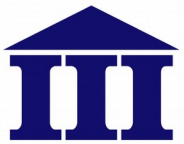


Result





NiFi with Kafka



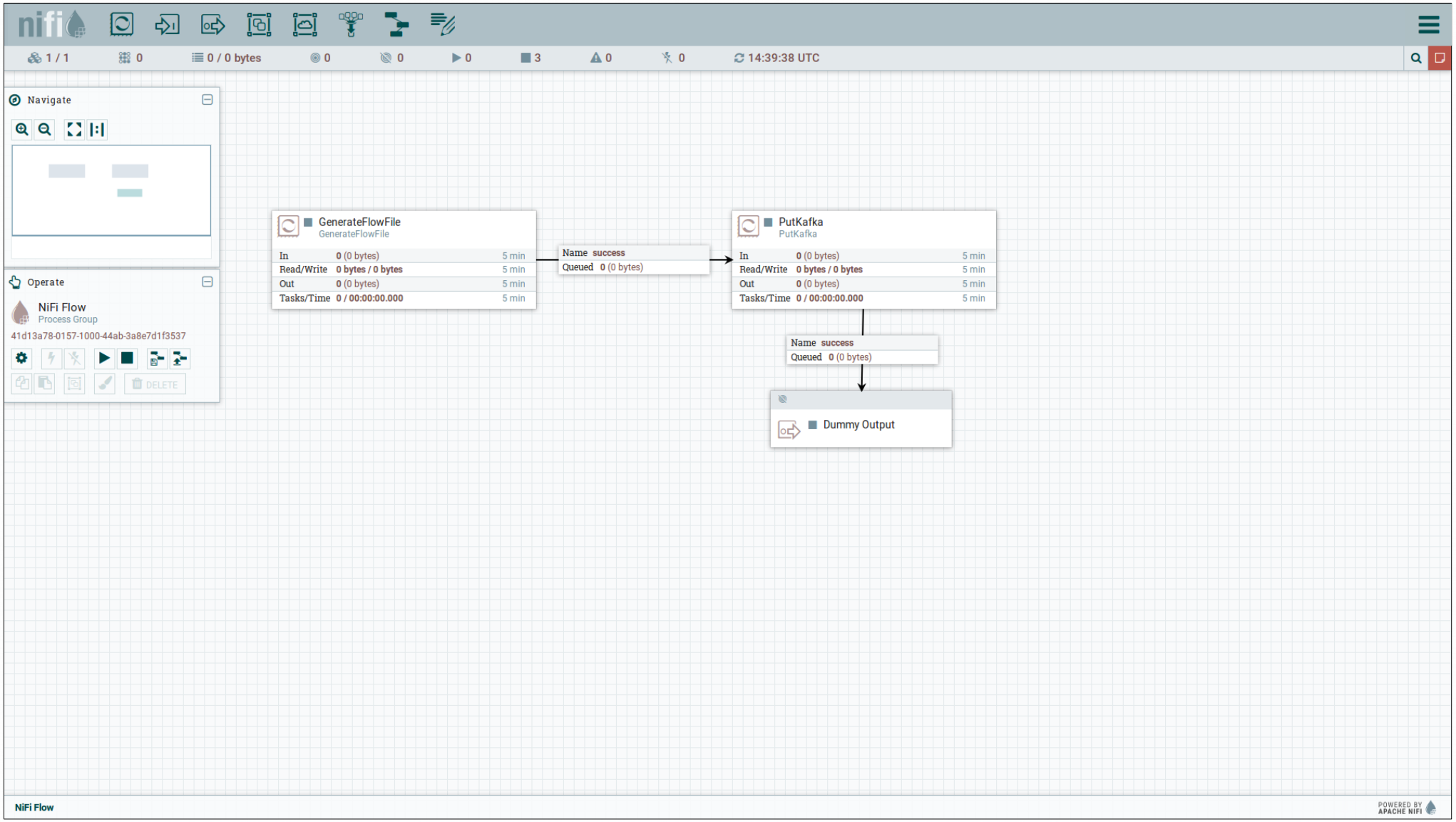
Create Kafka topic

cd /usr/hdf/2.0.0.0-579/kafka/bin/

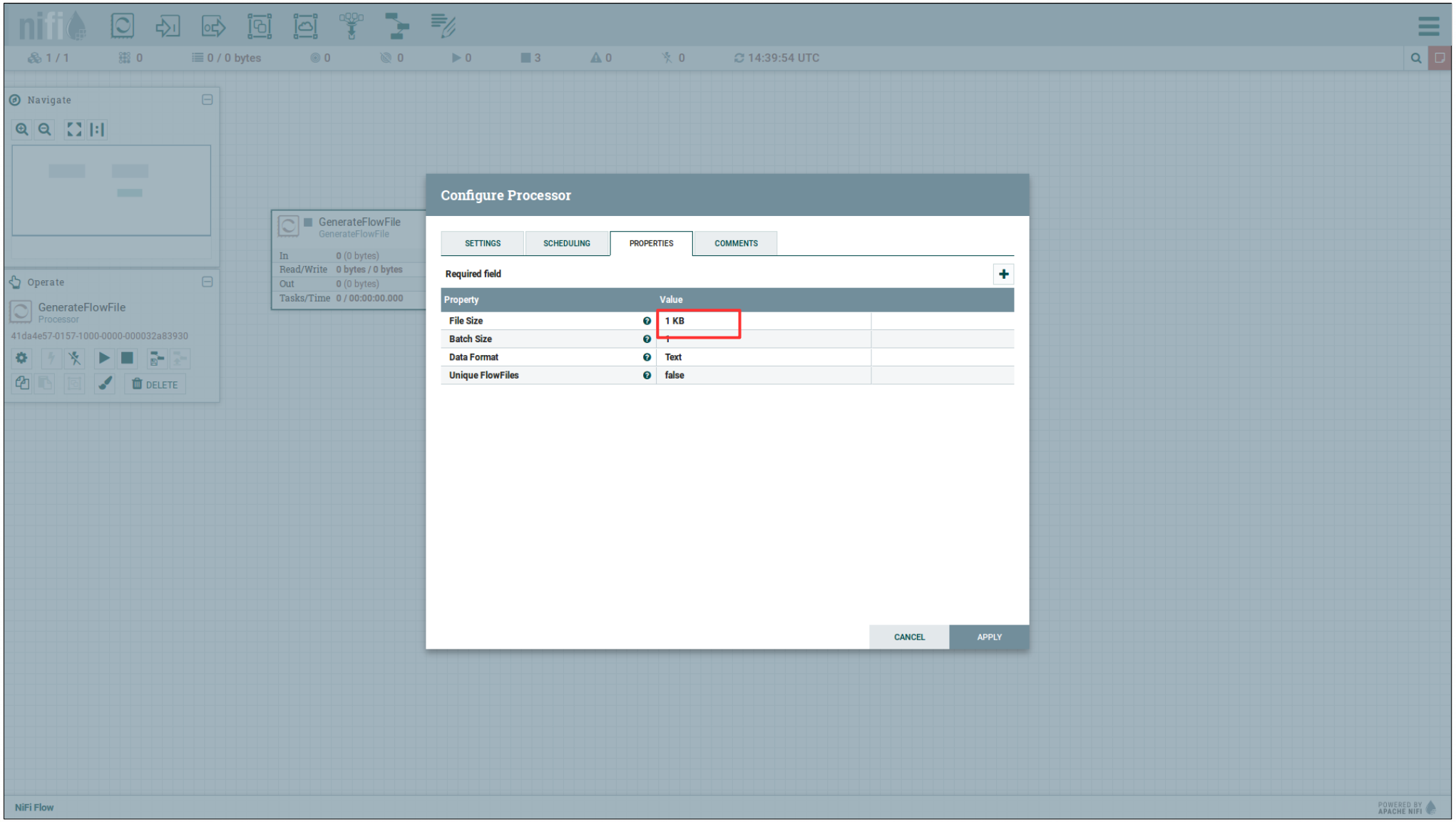
./kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 2 --topic test

./kafka-topics.sh --list --zookeeper localhost:2181

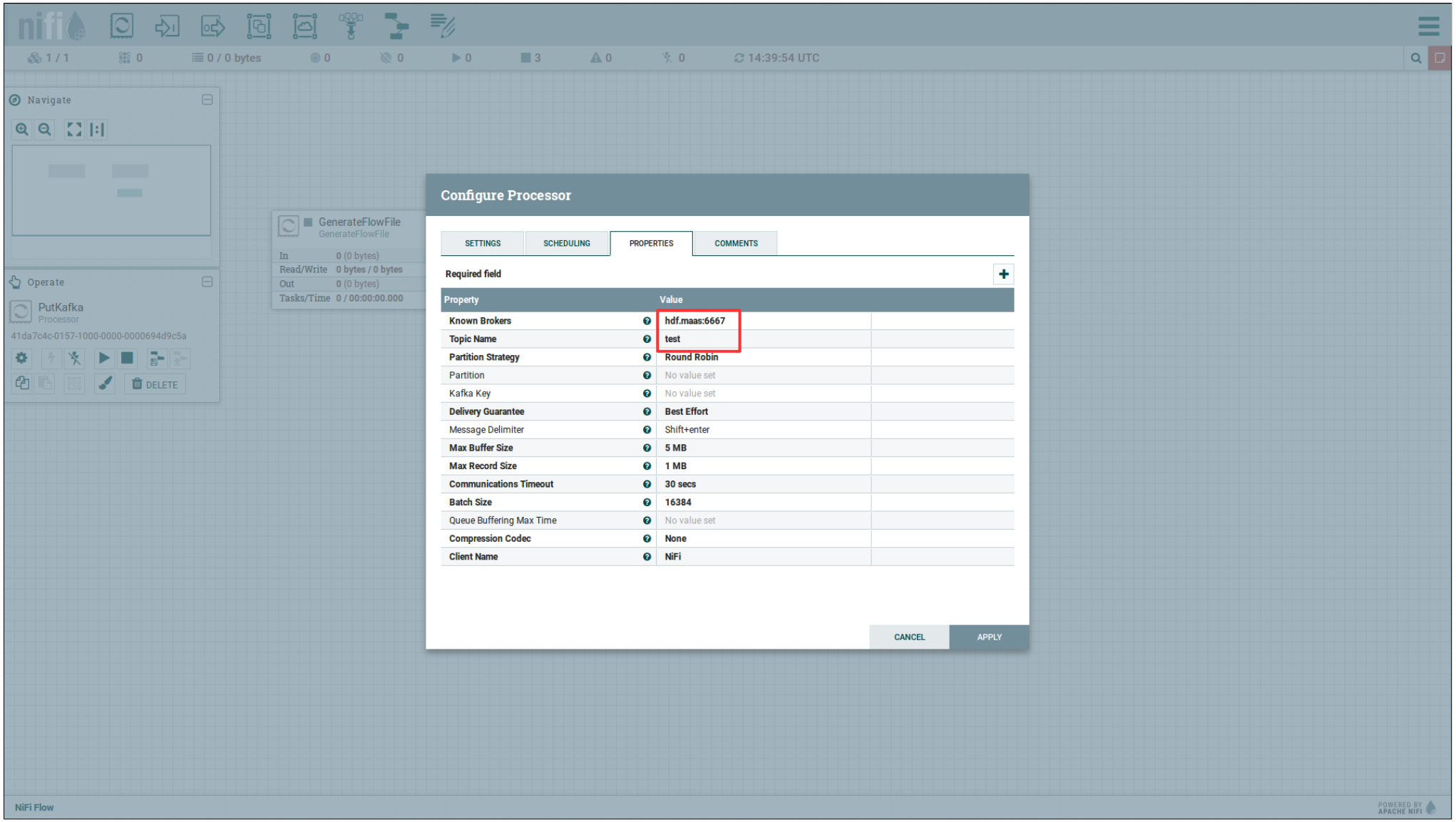
NiFi

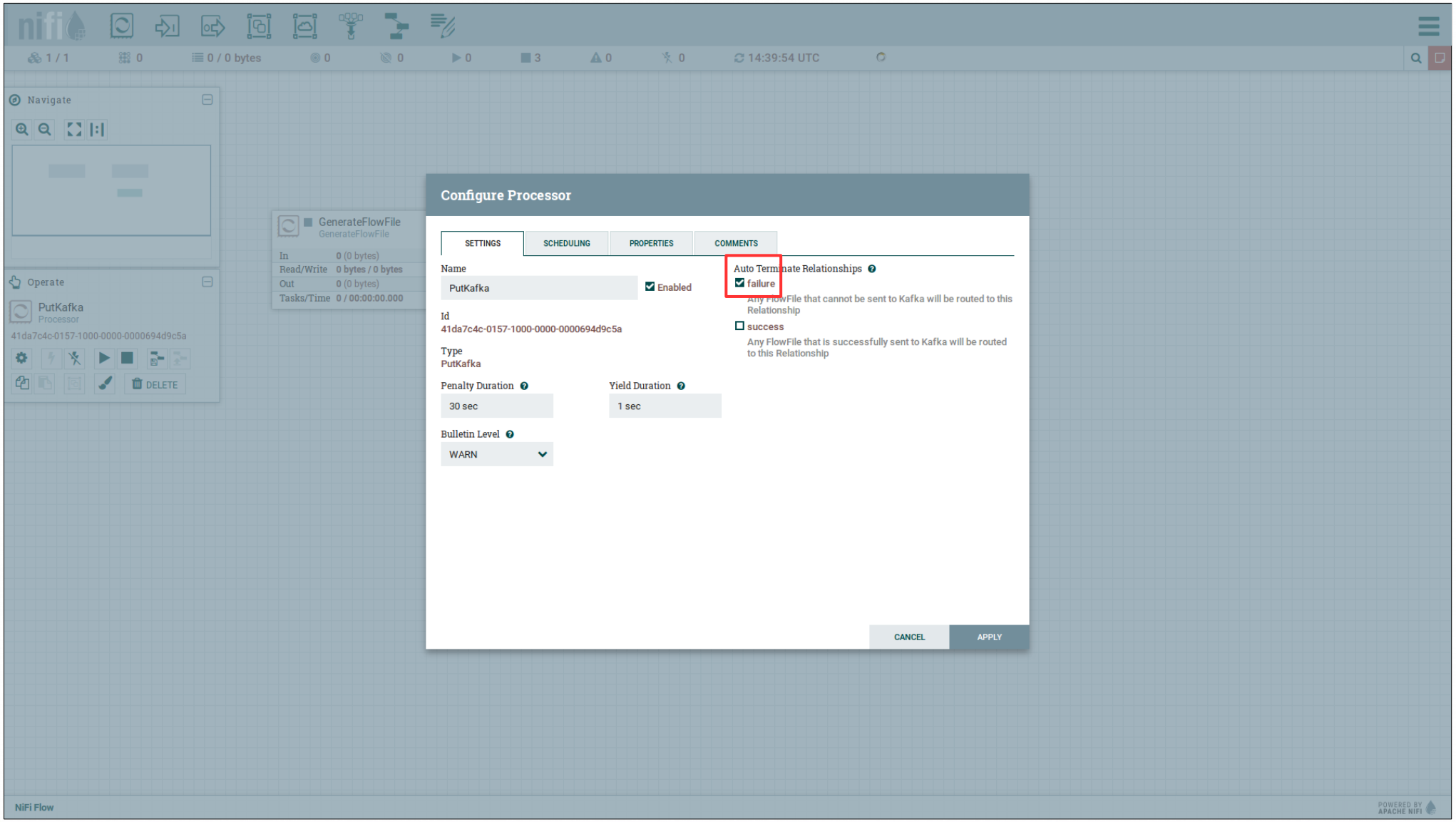


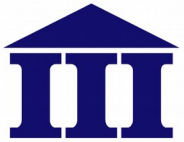
NiFi



NiFi

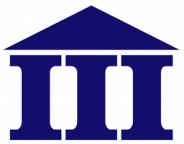




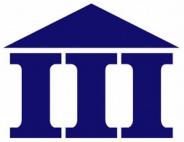


Check result

```
./kafka-console-consumer.sh --zookeeper
localhost:2181 --topic test --from-beginning
```



HDF to HDP



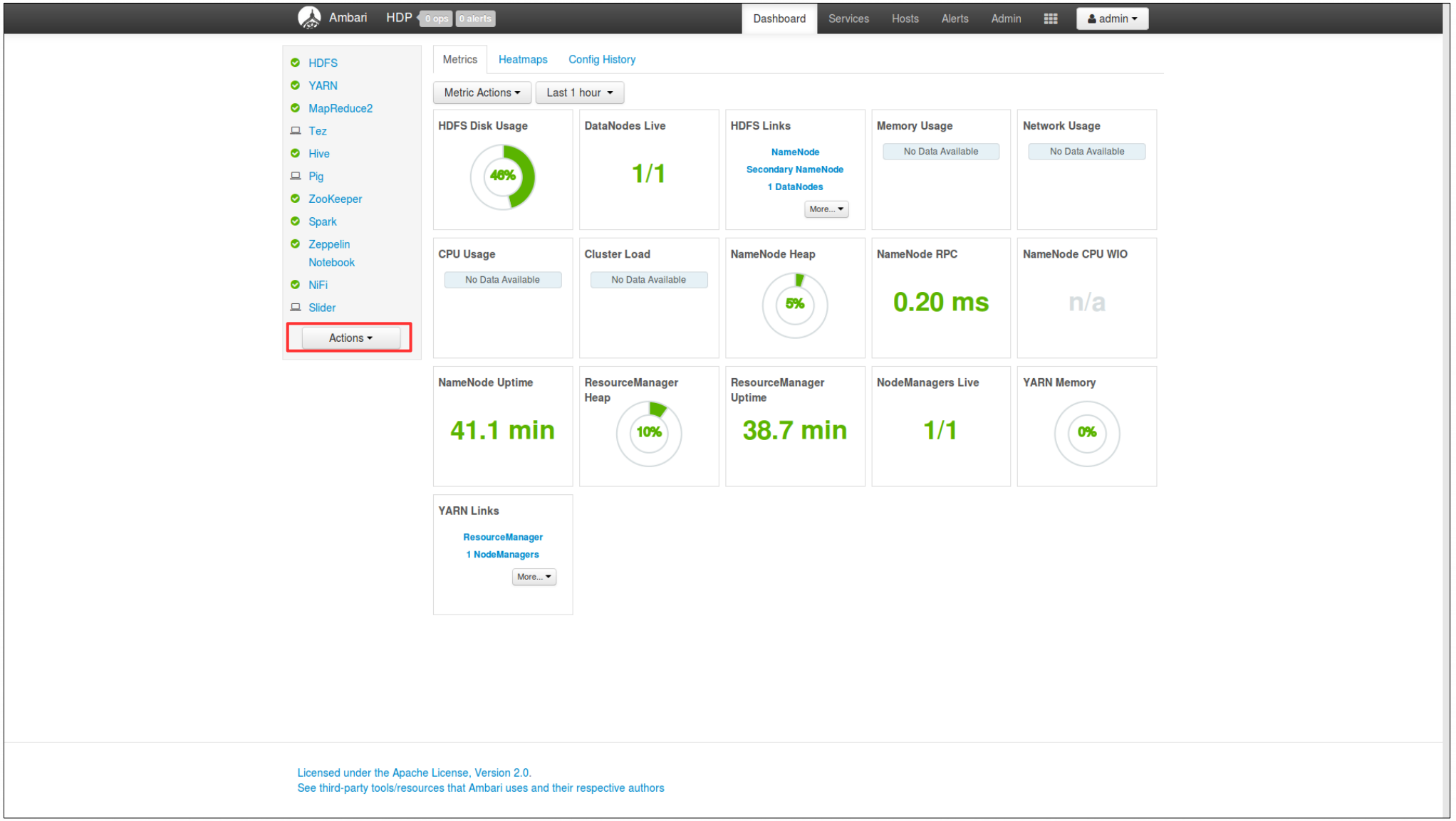
Install NiFi on HDP

sudo yum -y install git

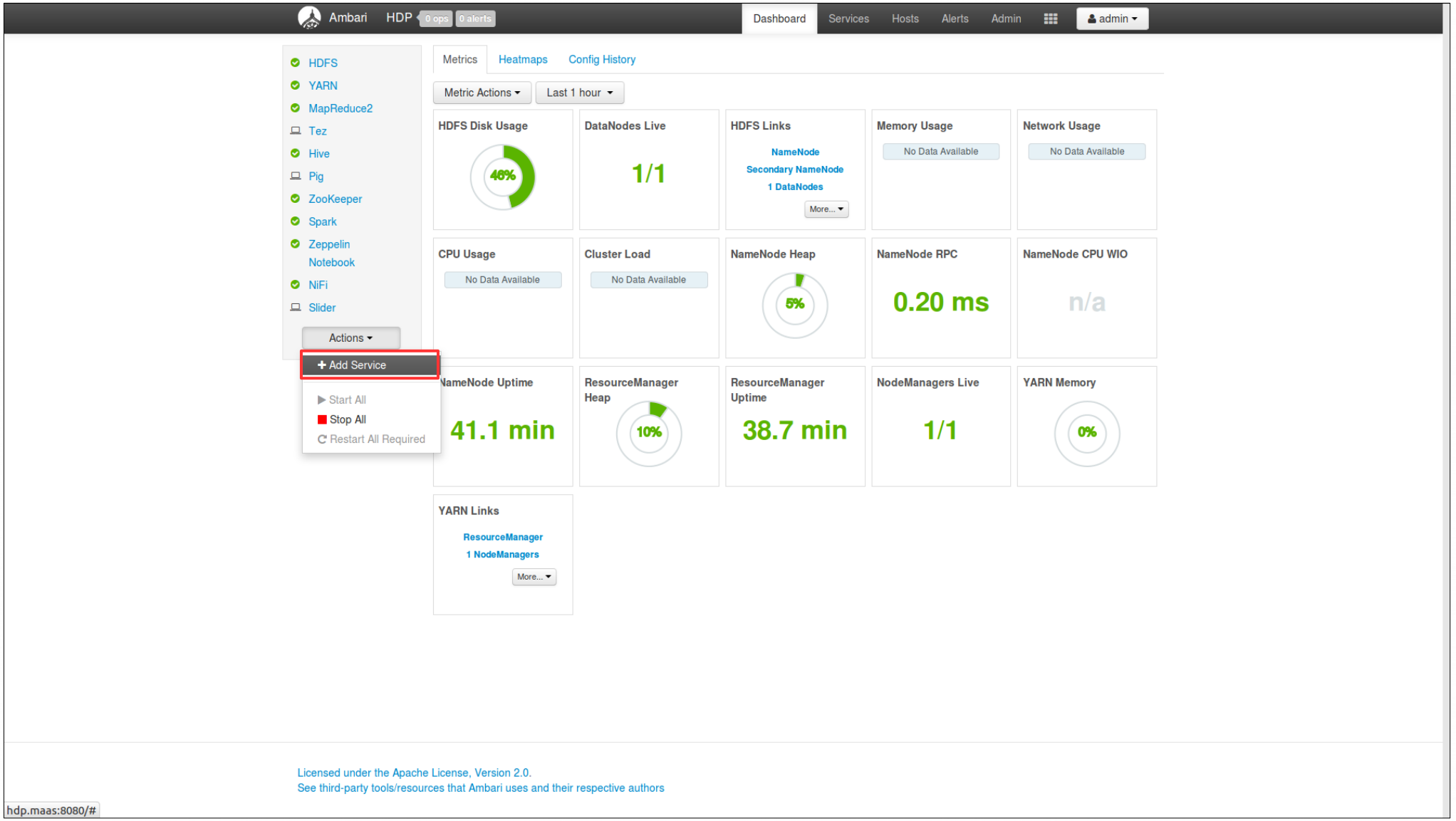
sudo git clone https://github.com/abajwa-hw/ambari-nifi-service.git /var/lib/ambari-server/resources/stacks/HDP/2.4/services/NIFI

sudo ambari-server restart

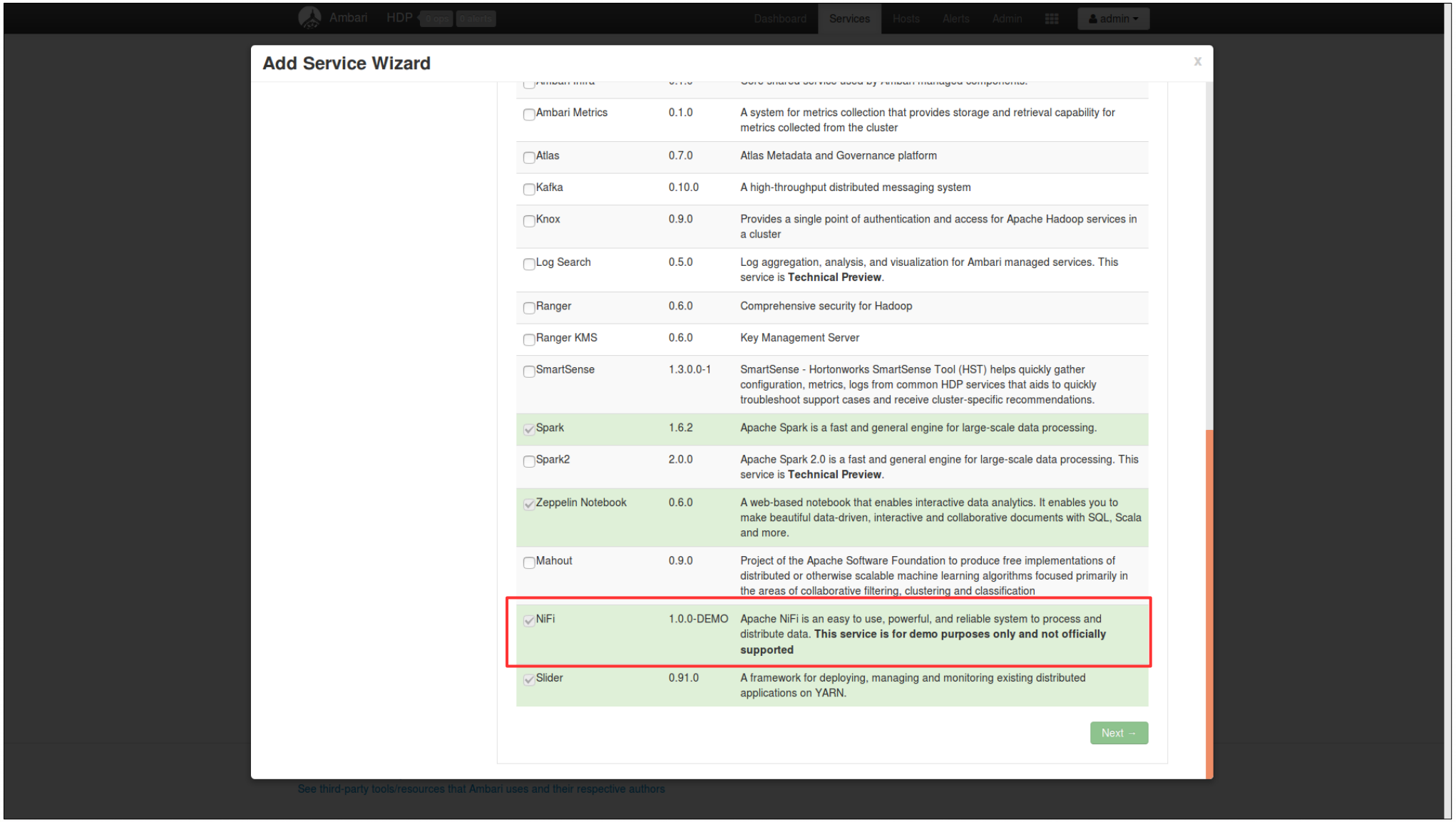
Install NiFi on HDP

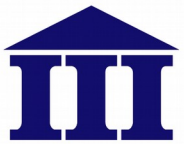


Install NiFi on HDP



Install NiFi on HDP

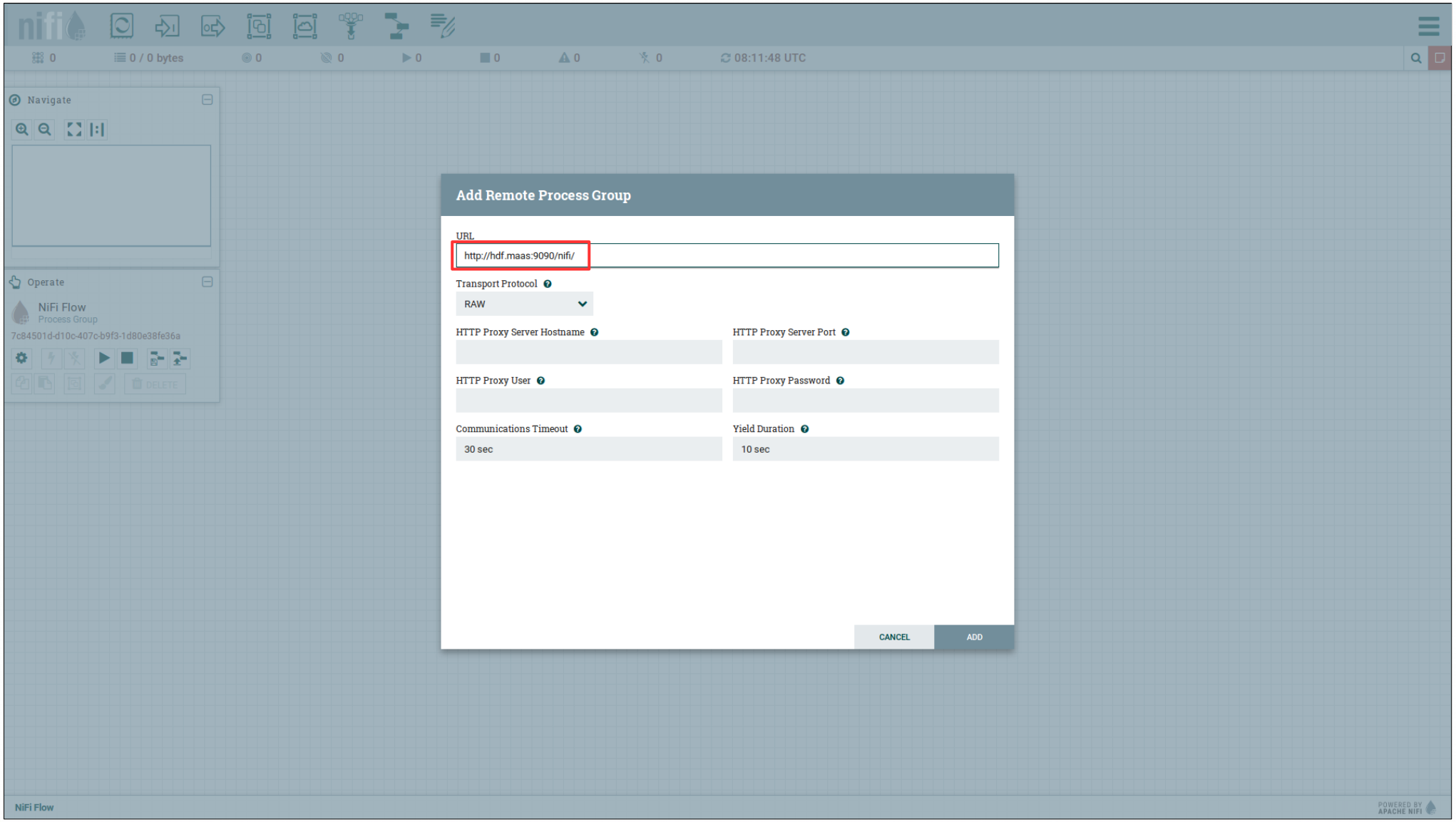




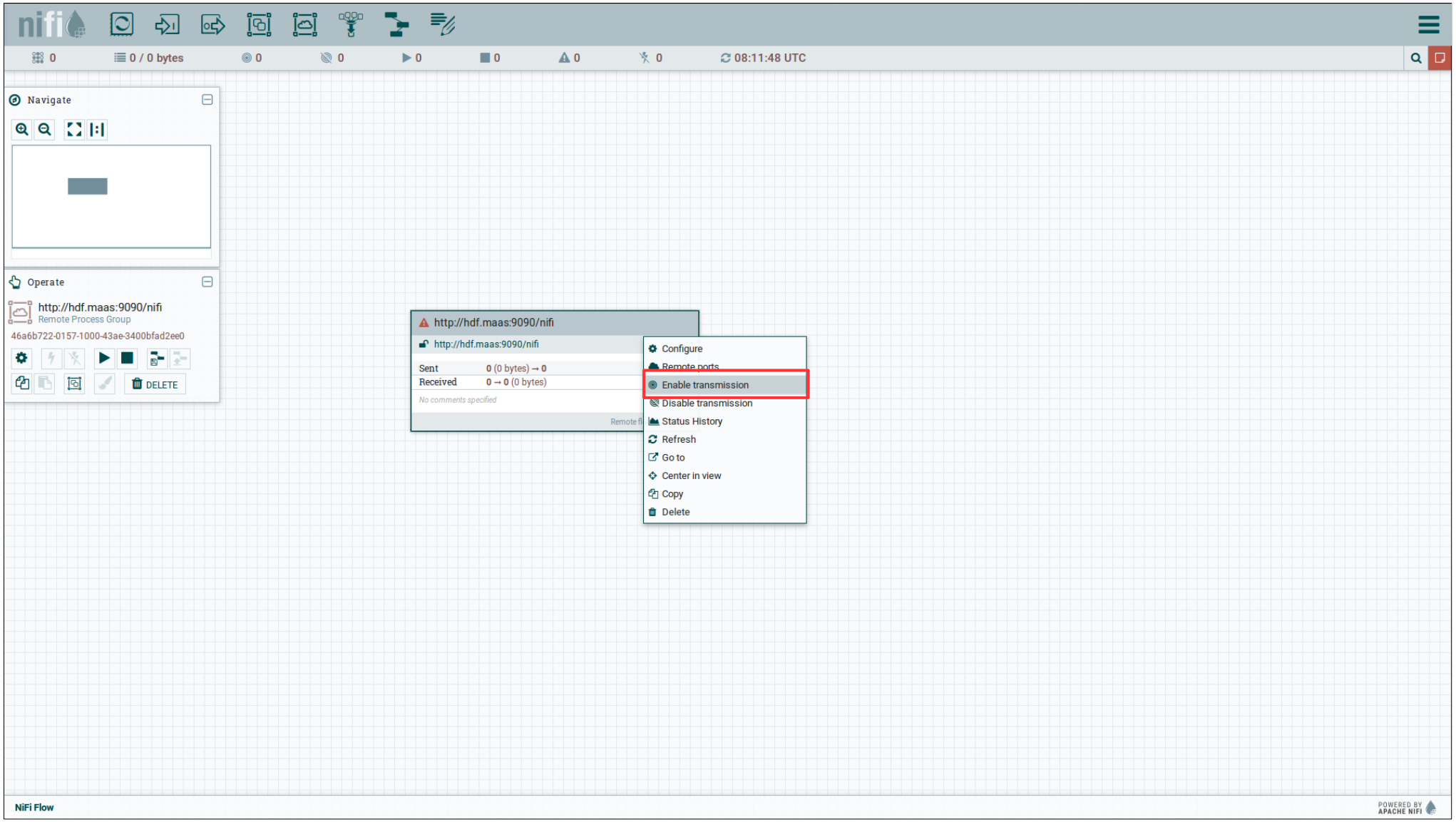
Create NiFi user for HDFS

```
sudo su hdfs -c 'hadoop fs -mkdir /user/nifi'

sudo su hdfs -c 'hadoop fs -chown nifi:hdfs /user/nifi'
```

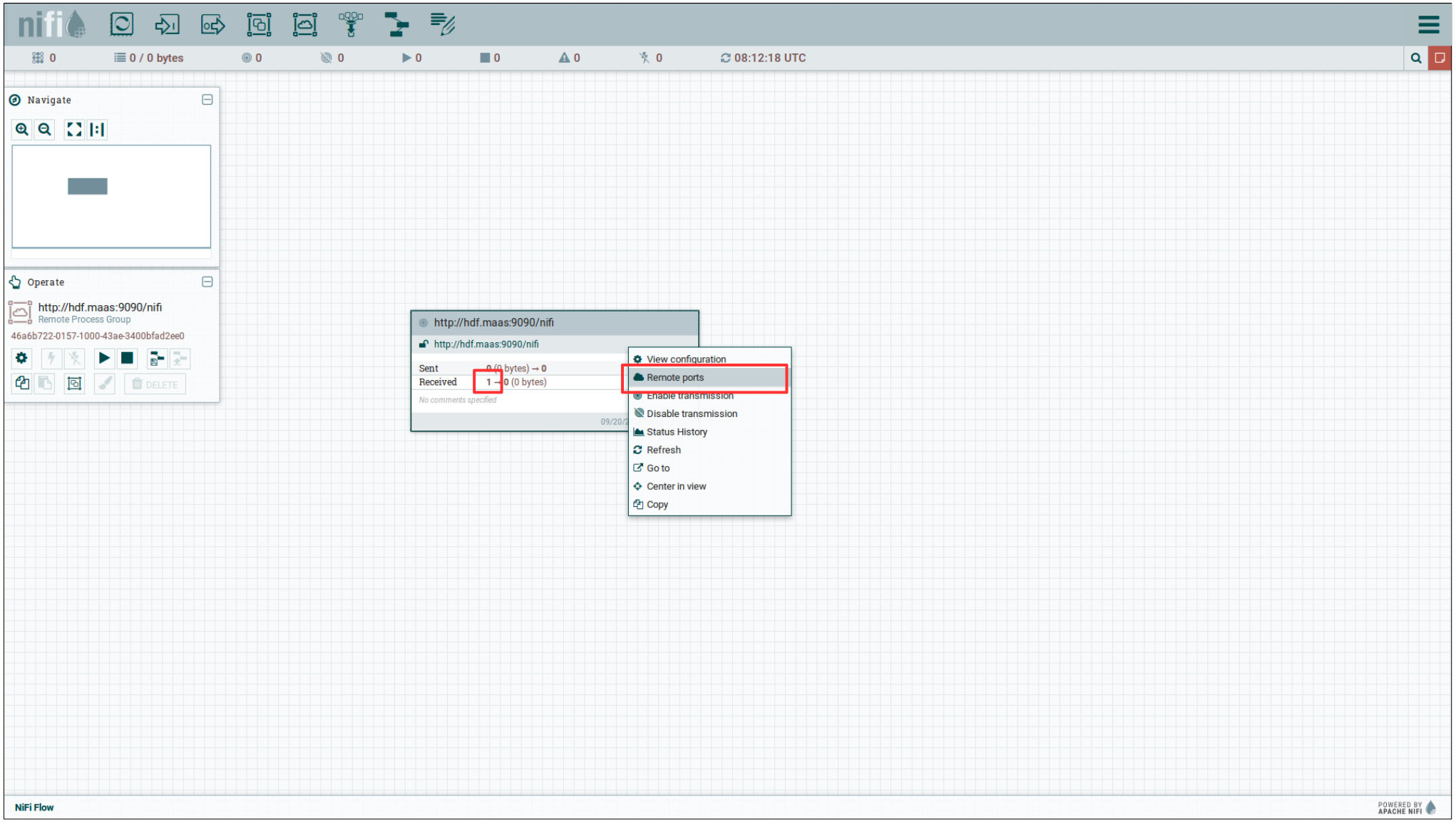
NiFi to HDFS



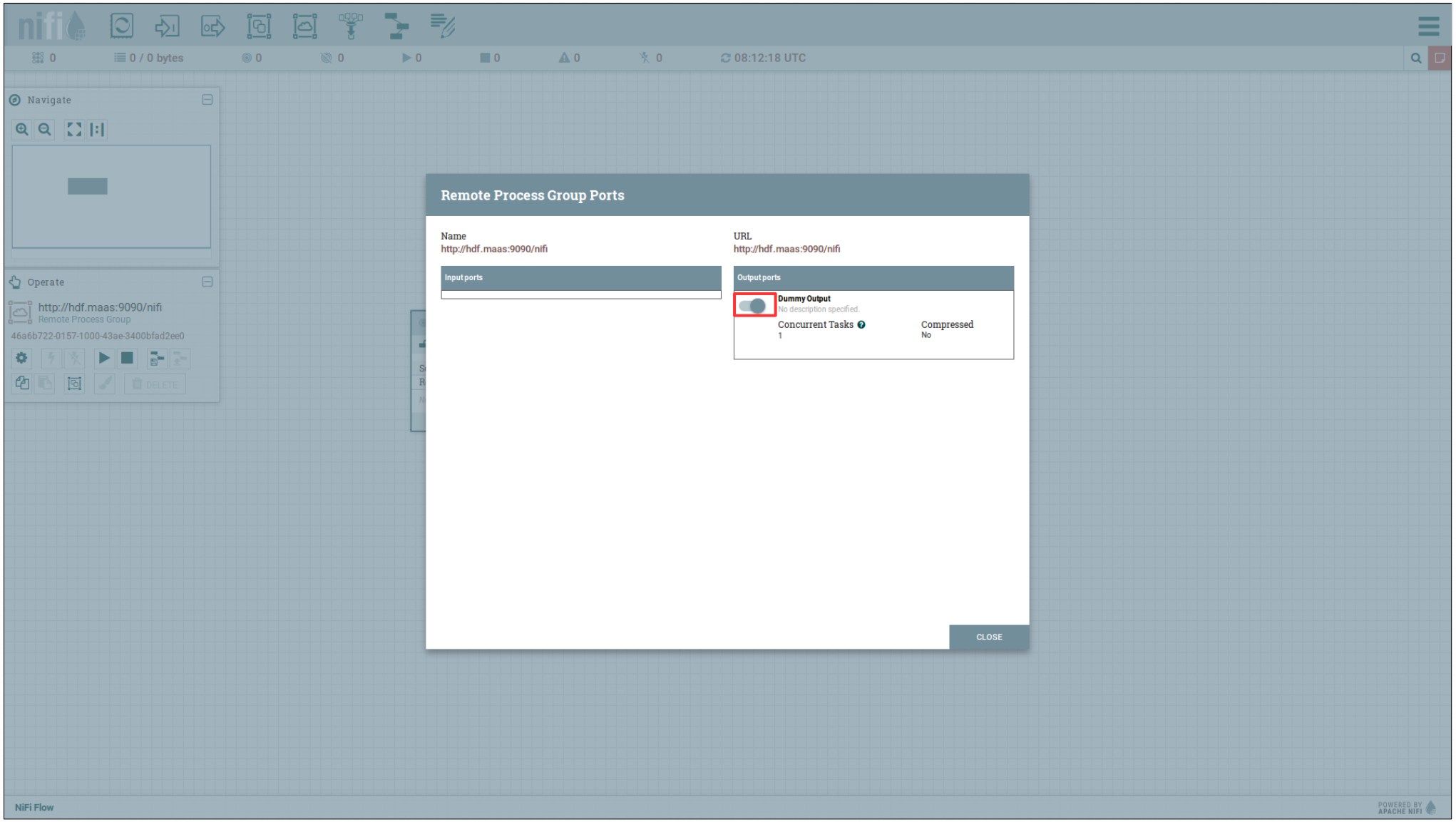
NiFi to HDFS



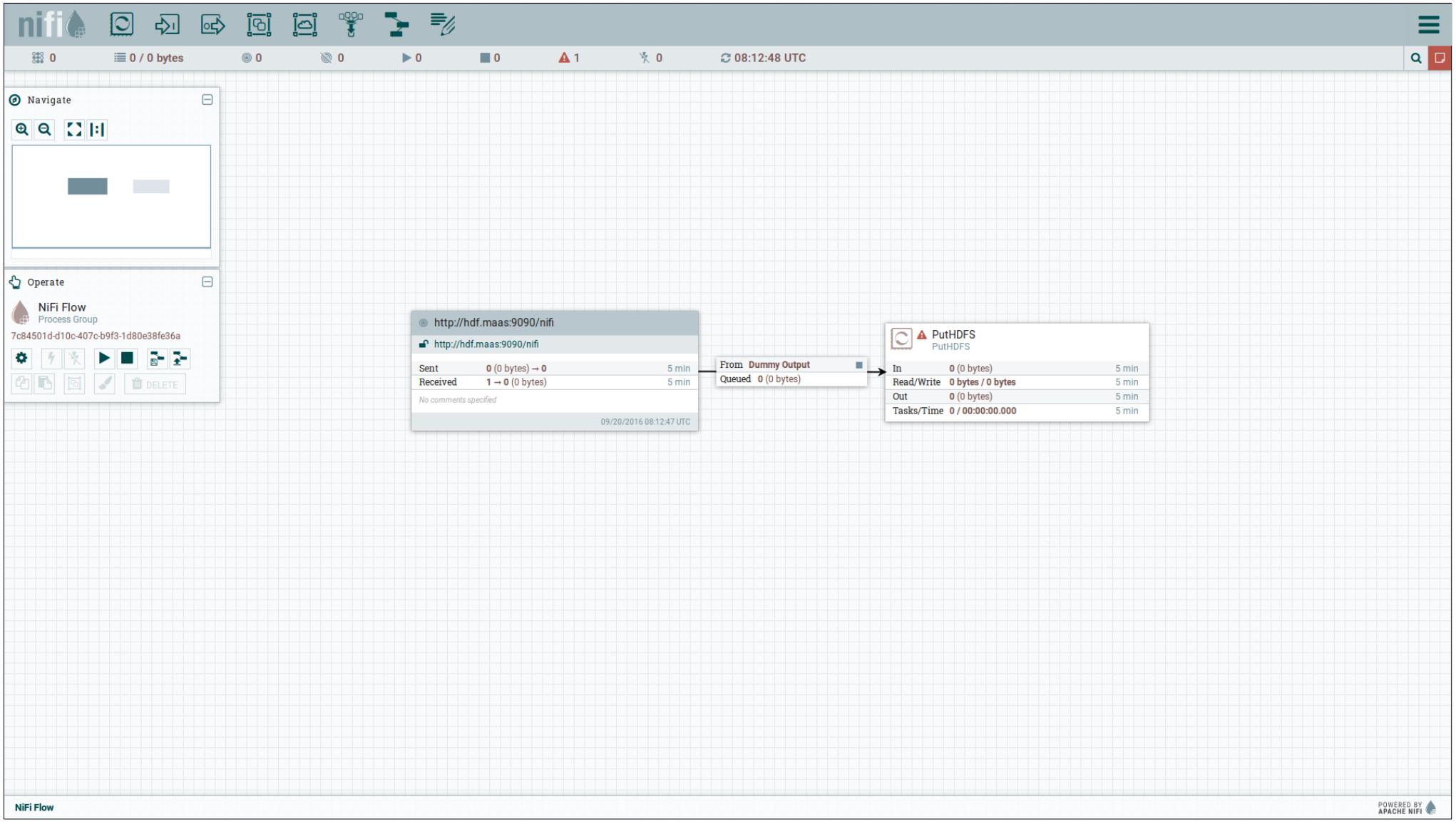
NiFi to HDFS



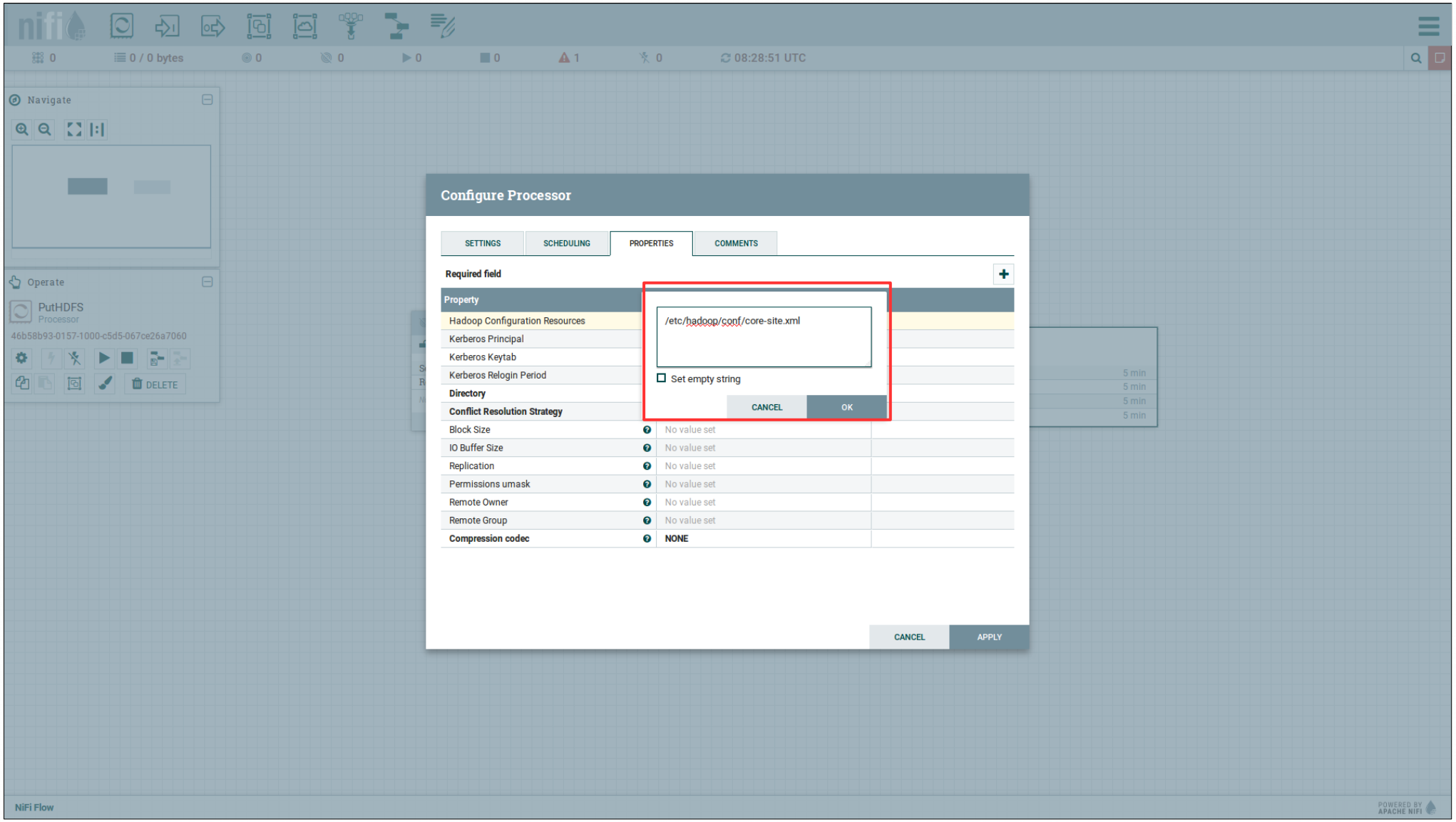
NiFi to HDFS



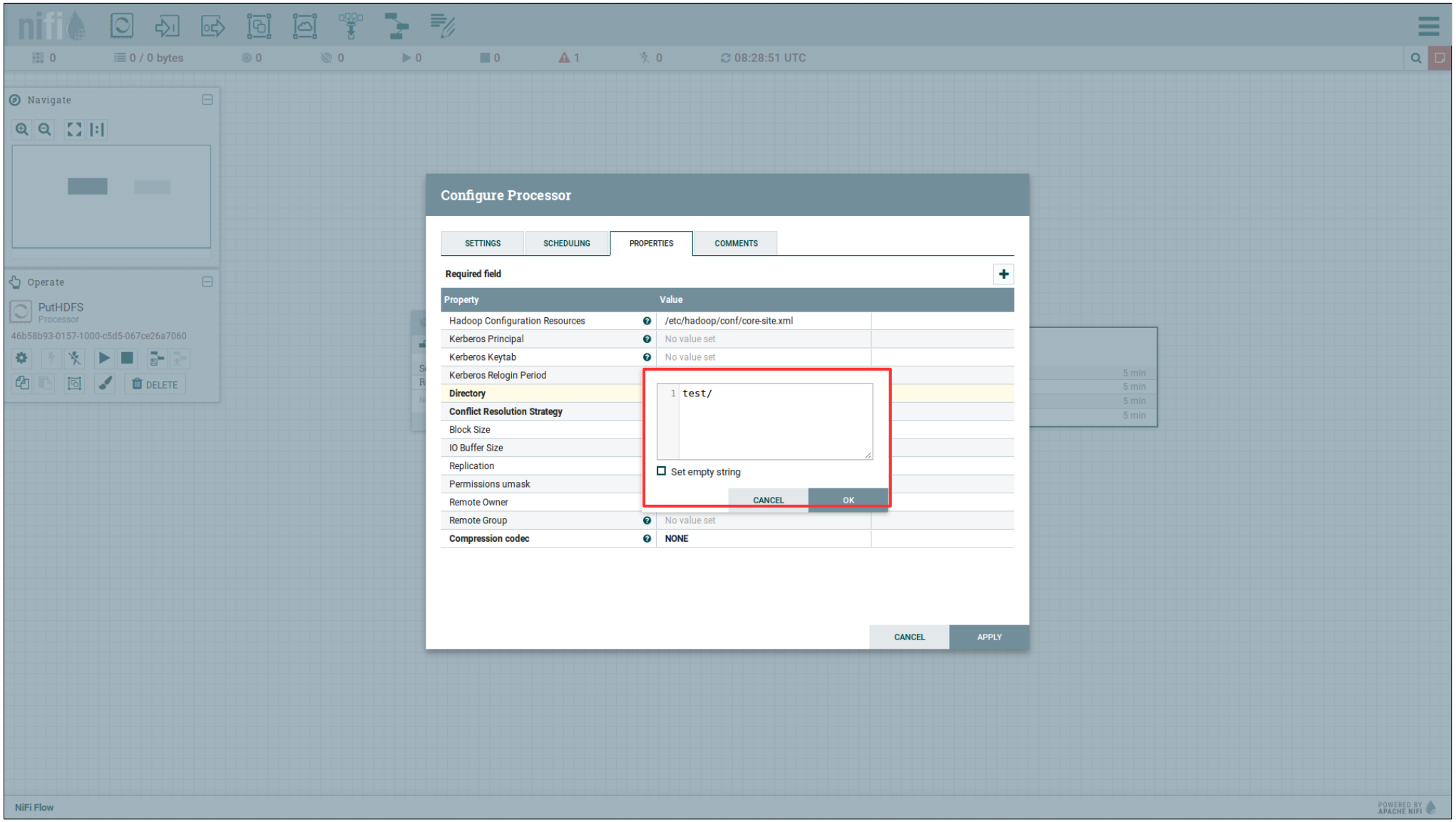
NiFi to HDFS



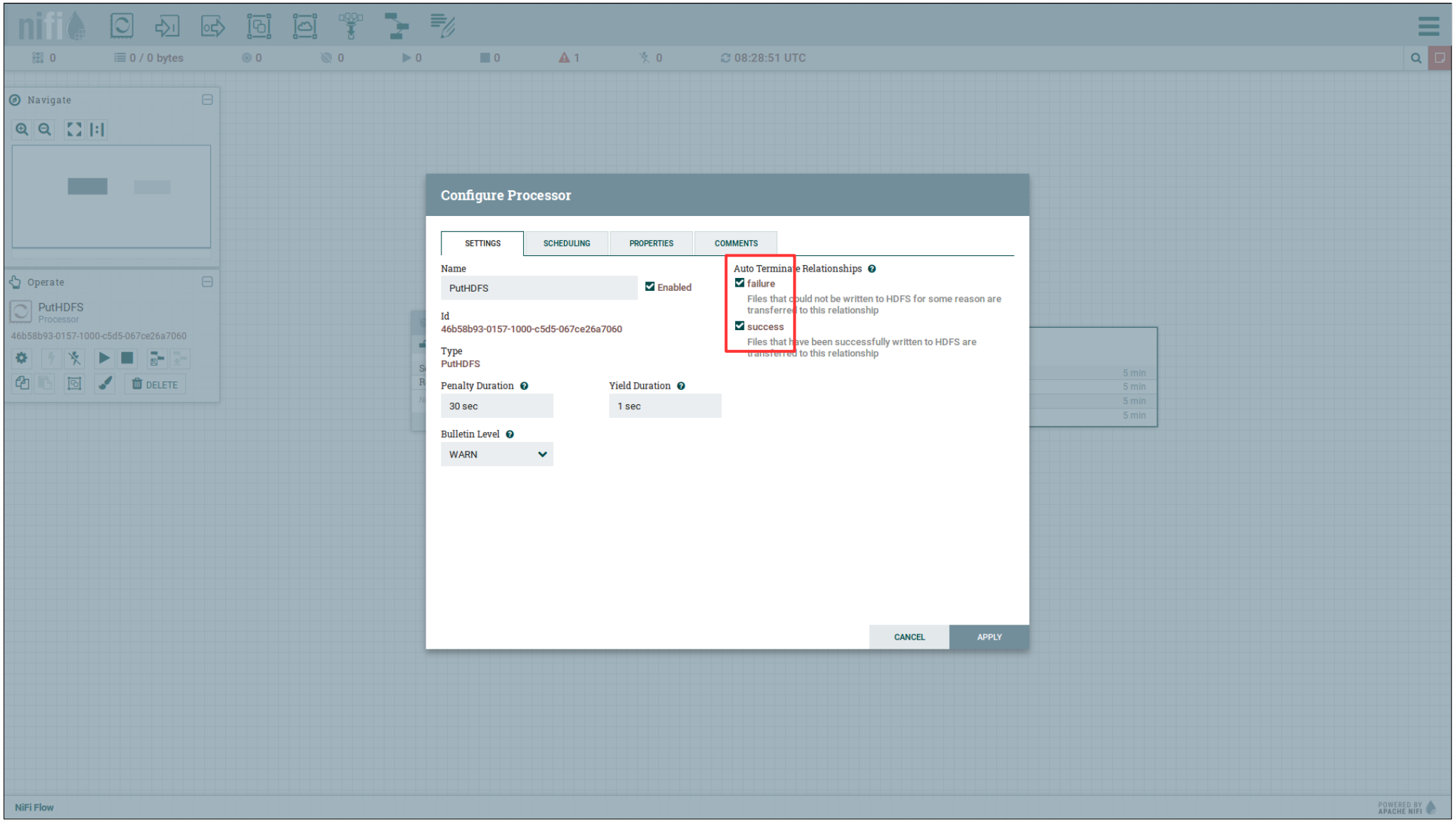
NiFi to HDFS



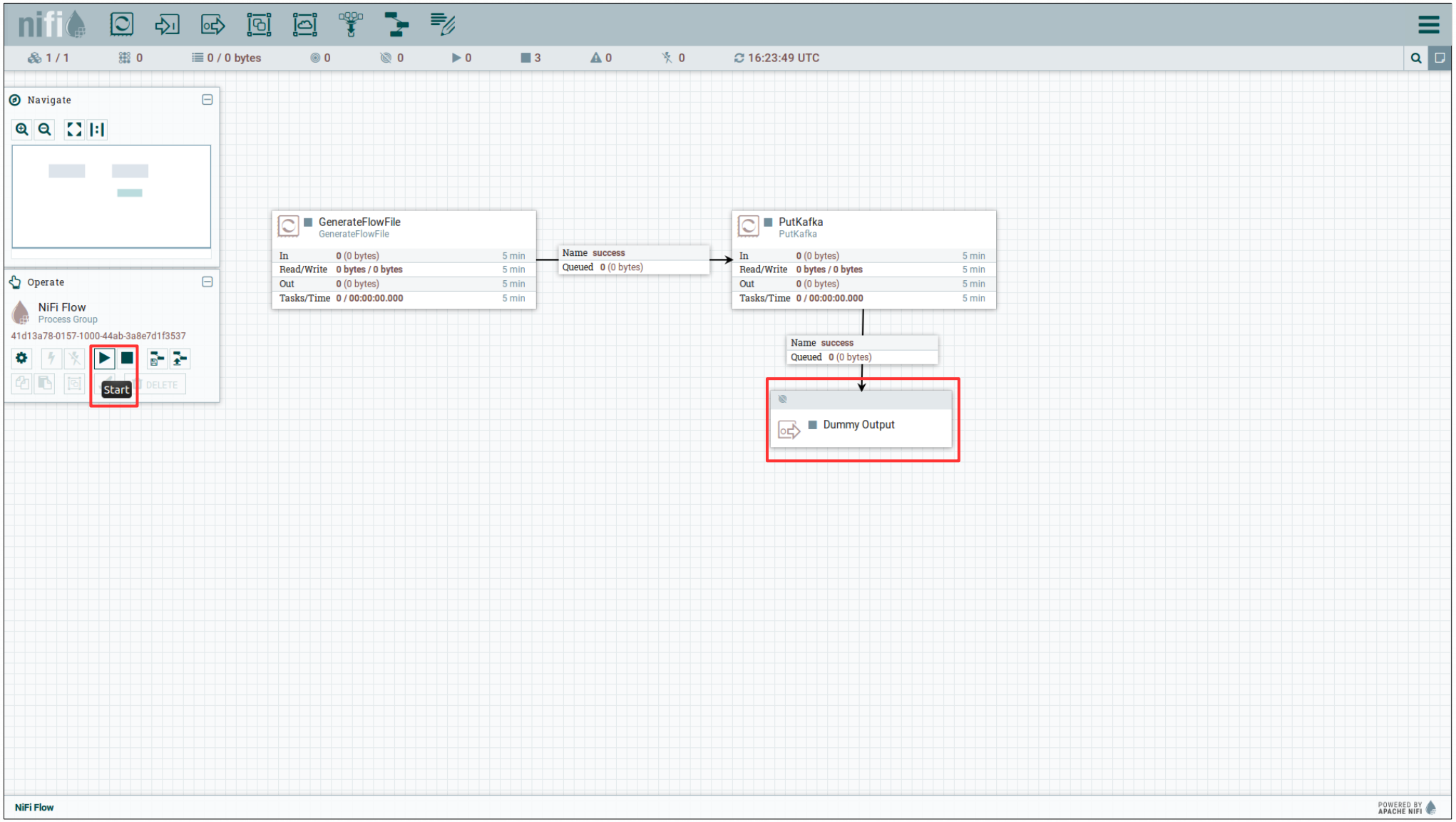
NiFi to HDFS



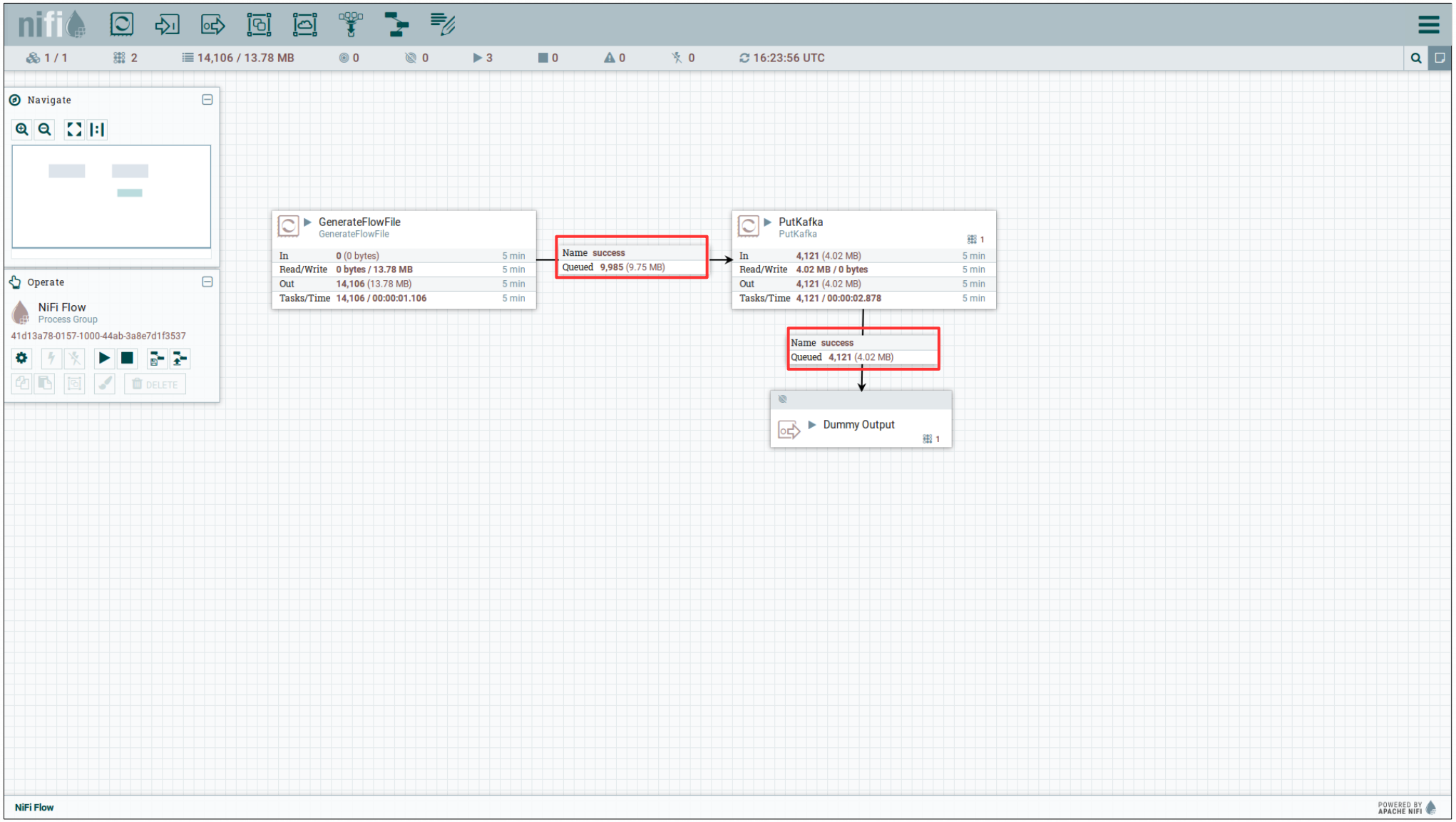
NiFi to HDFS



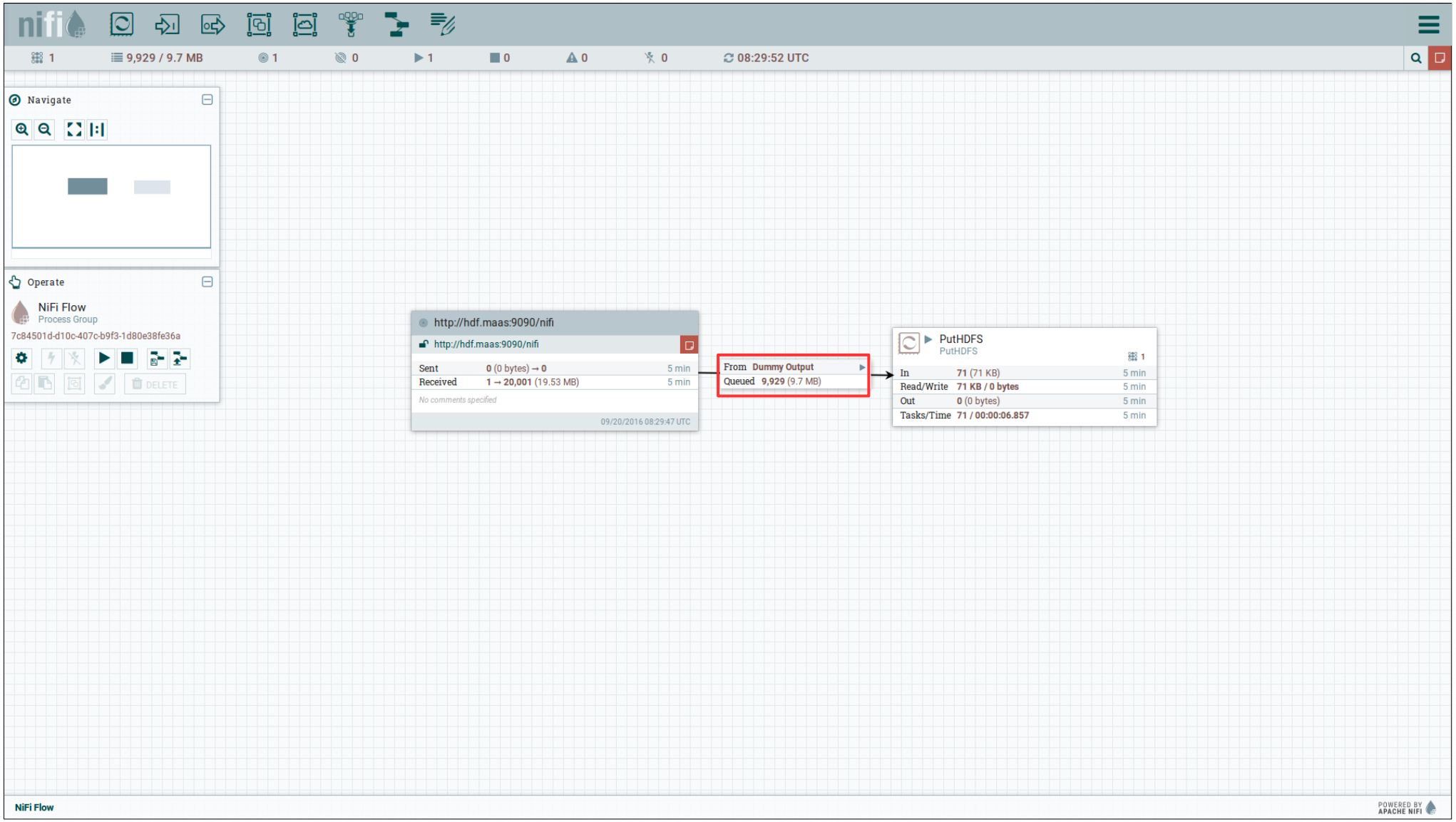
NiFi to HDFS



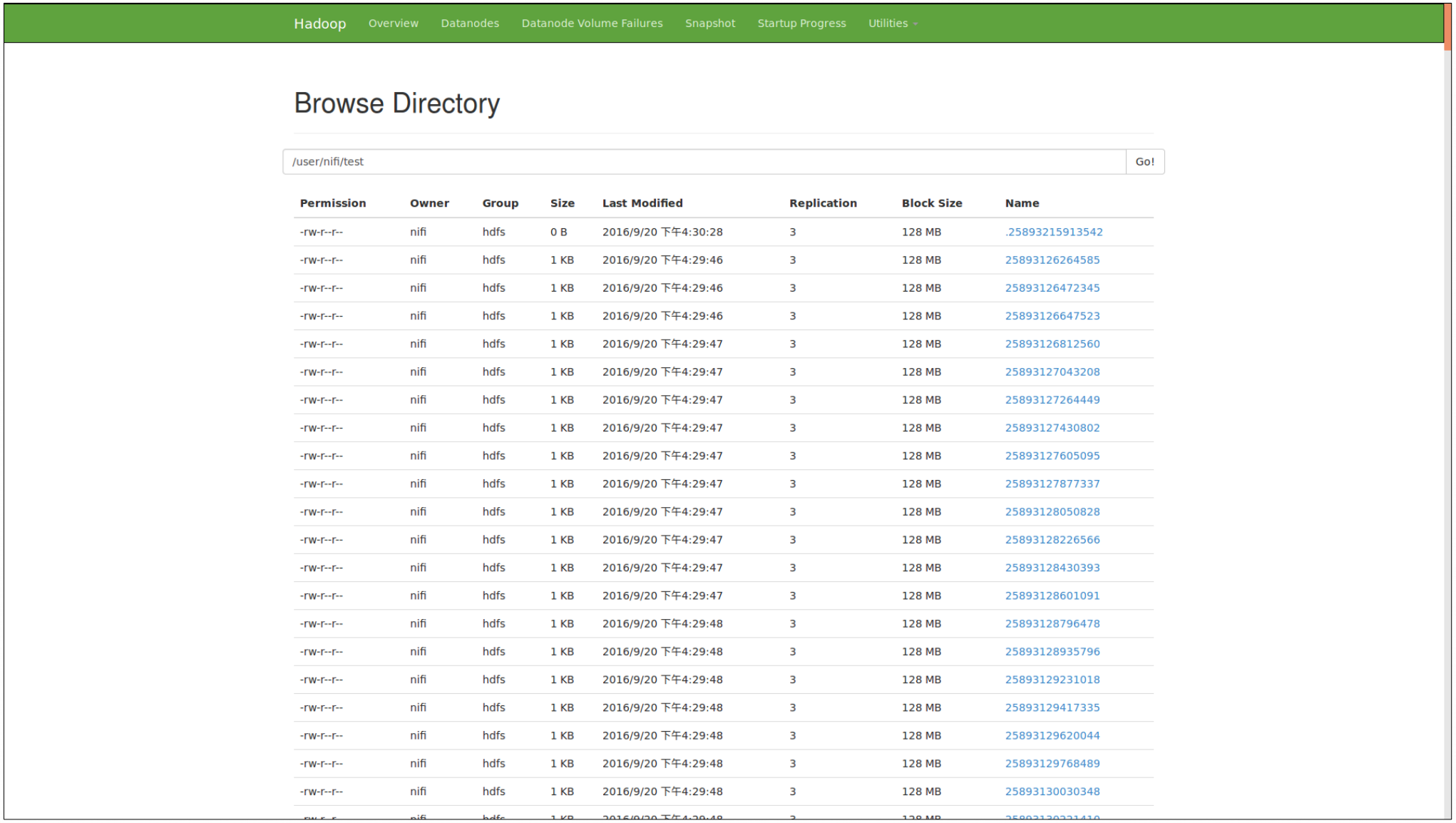
NiFi to HDFS



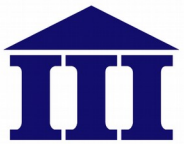
NiFi to HDFS



NiFi to HDFS

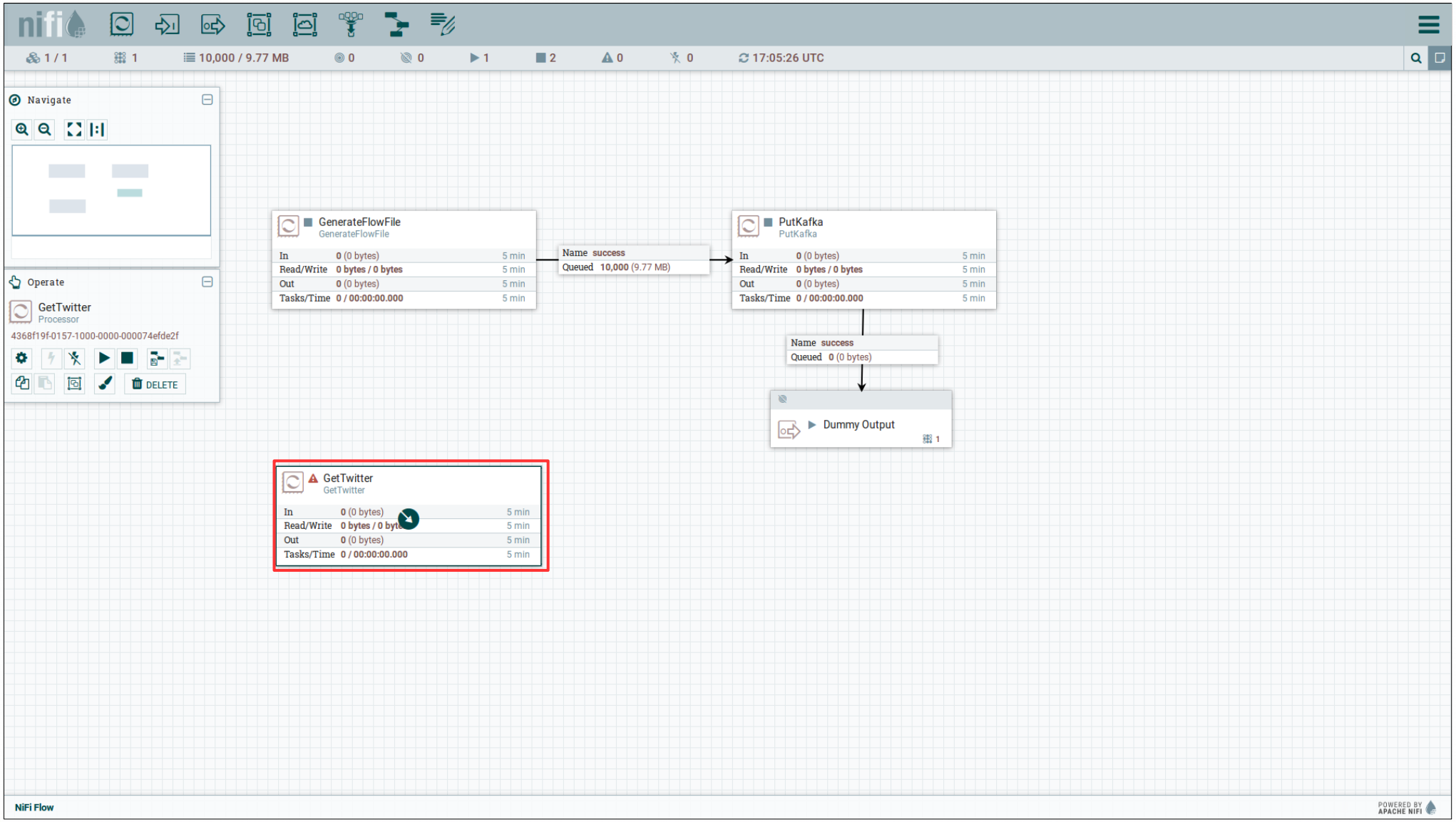


Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	nifi	hdfs	0 B	2016/9/20 下午4:30:28	3	128 MB	.25893215913542
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:46	3	128 MB	25893126264585
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:46	3	128 MB	25893126472345
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:46	3	128 MB	25893126647523
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:47	3	128 MB	25893126812560
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:47	3	128 MB	25893127043208
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:47	3	128 MB	25893127264449
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:47	3	128 MB	25893127430802
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:47	3	128 MB	25893127605095
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:47	3	128 MB	25893127877337
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:47	3	128 MB	25893128050828
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:47	3	128 MB	25893128226566
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:47	3	128 MB	25893128430393
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:47	3	128 MB	25893128601091
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:48	3	128 MB	25893128796478
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:48	3	128 MB	25893128935796
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:48	3	128 MB	25893129231018
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:48	3	128 MB	25893129417335
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:48	3	128 MB	25893129620044
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:48	3	128 MB	25893129768489
-rw-r--r--	nifi	hdfs	1 KB	2016/9/20 下午4:29:48	3	128 MB	25893130030348

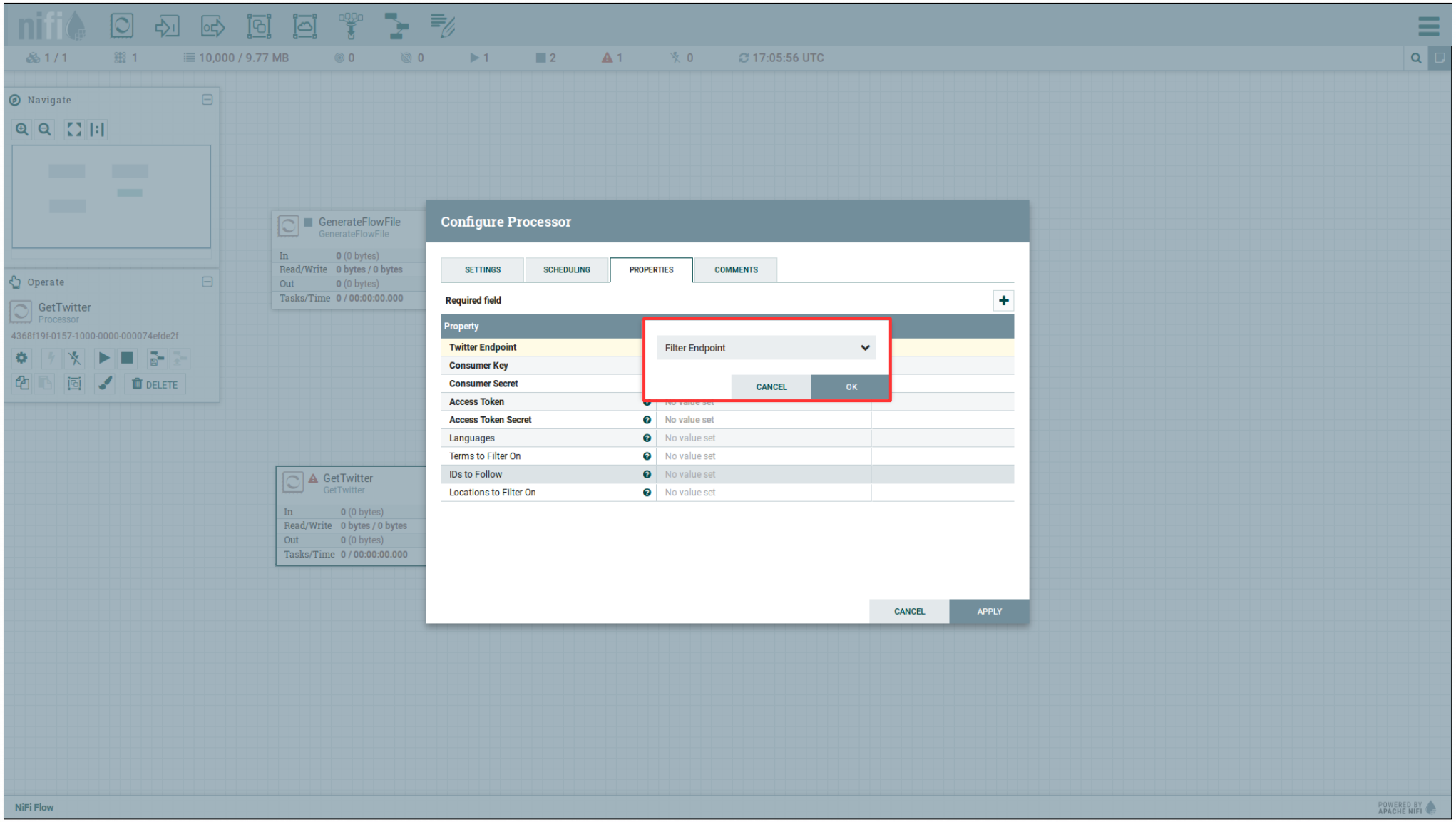


Let’s Tweet

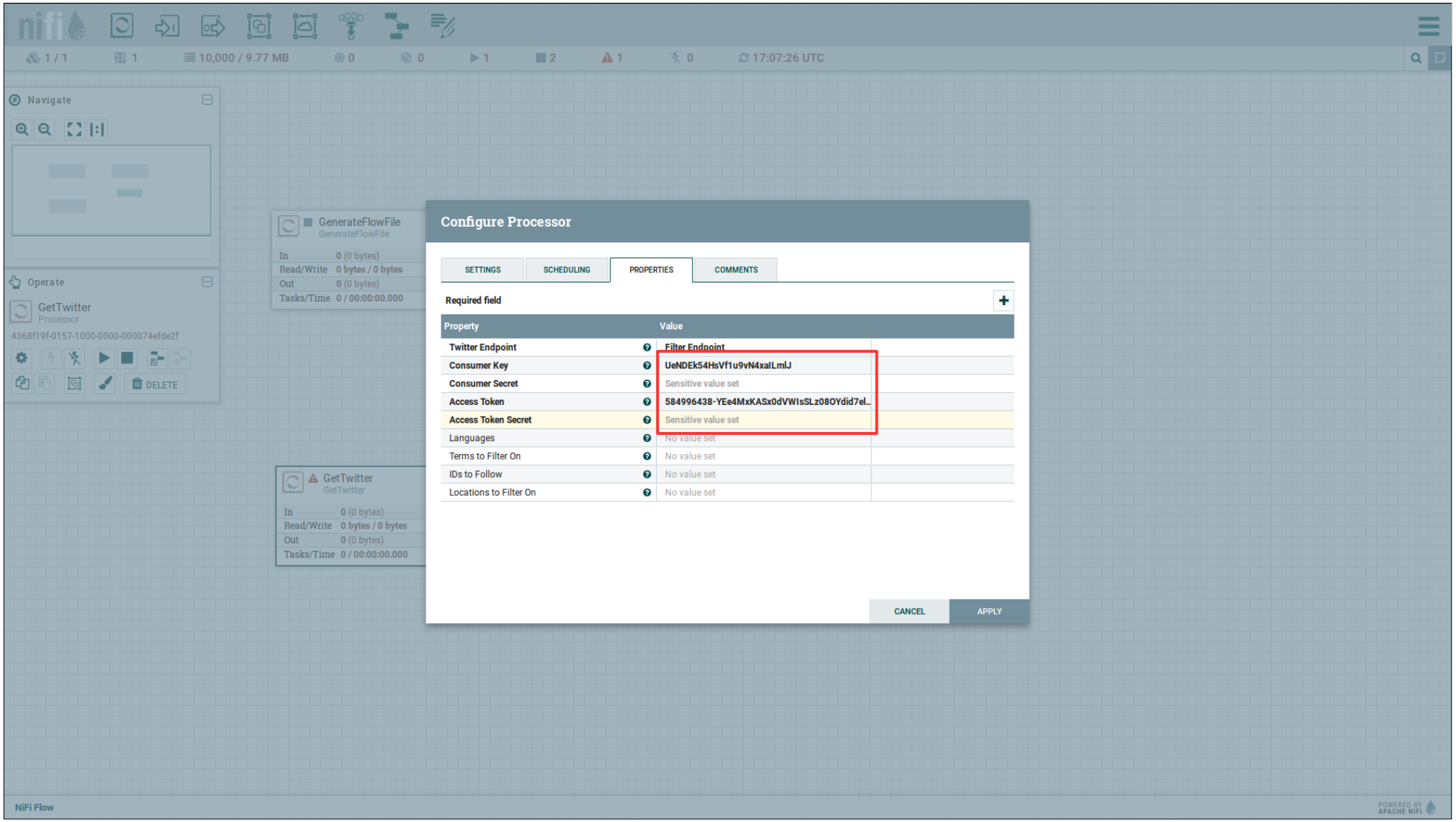
Twitter to HDF



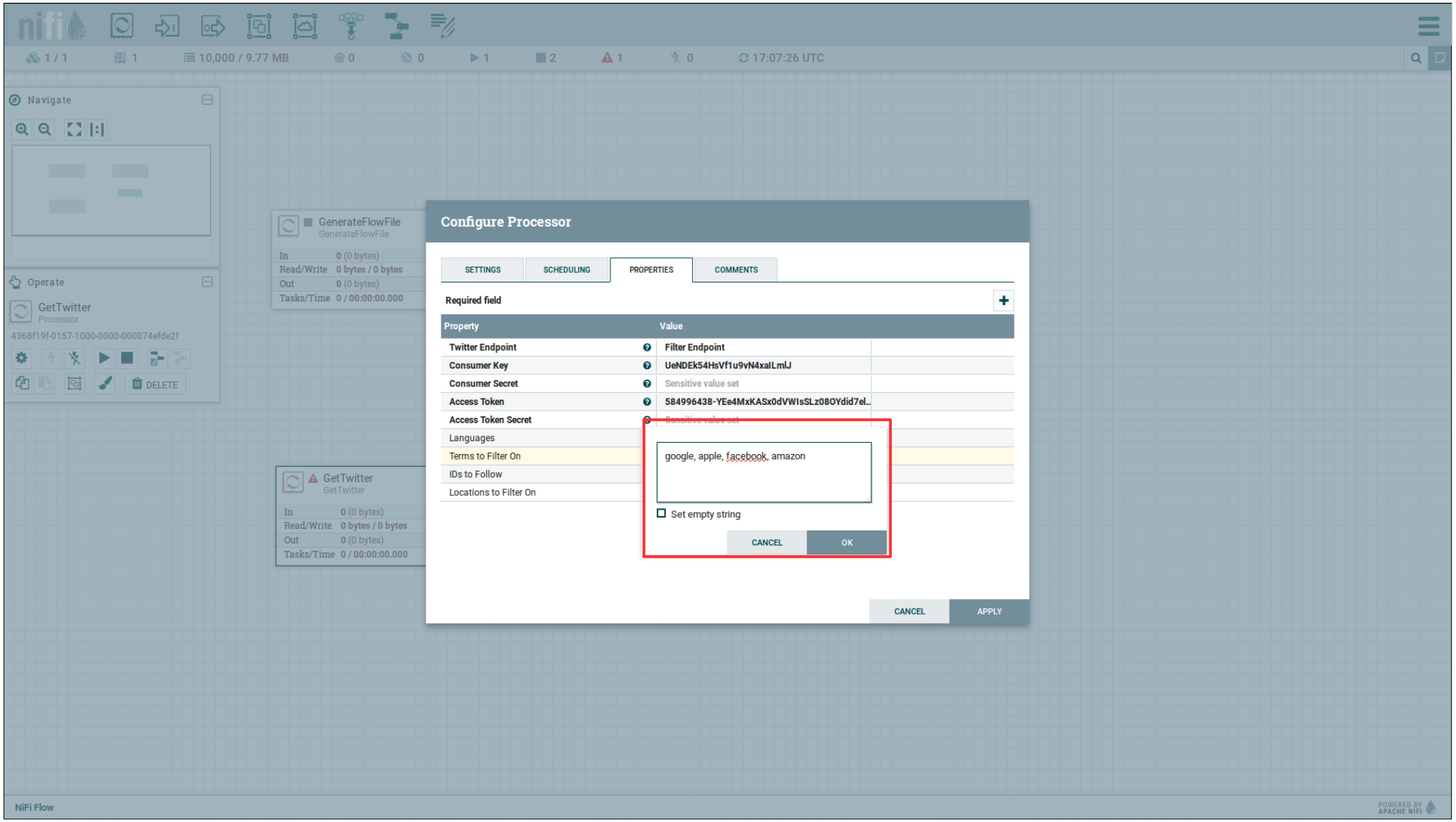
Twitter to HDF



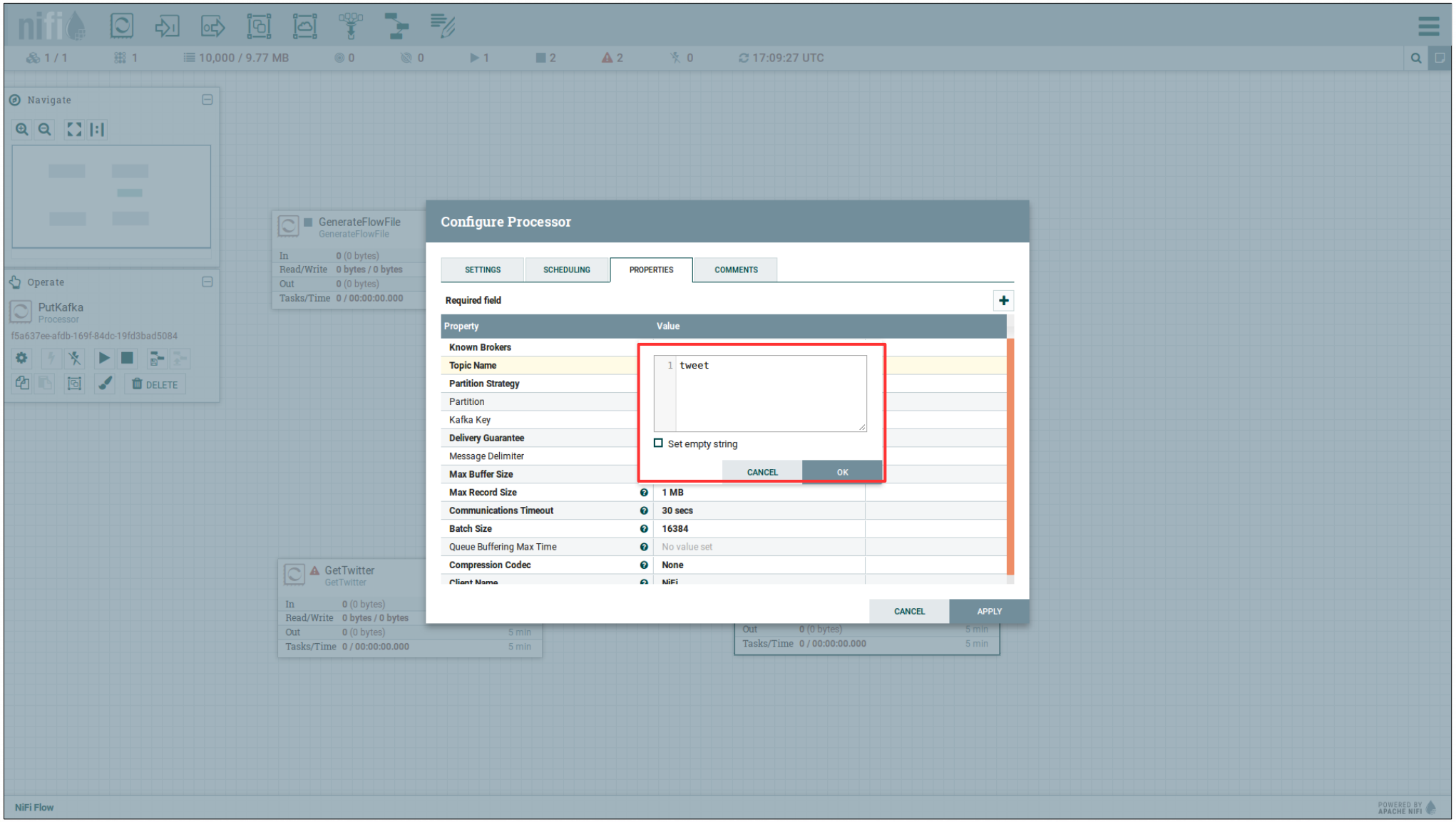
Twitter to HDF

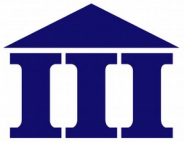


Twitter to HDF



Twitter to HDF



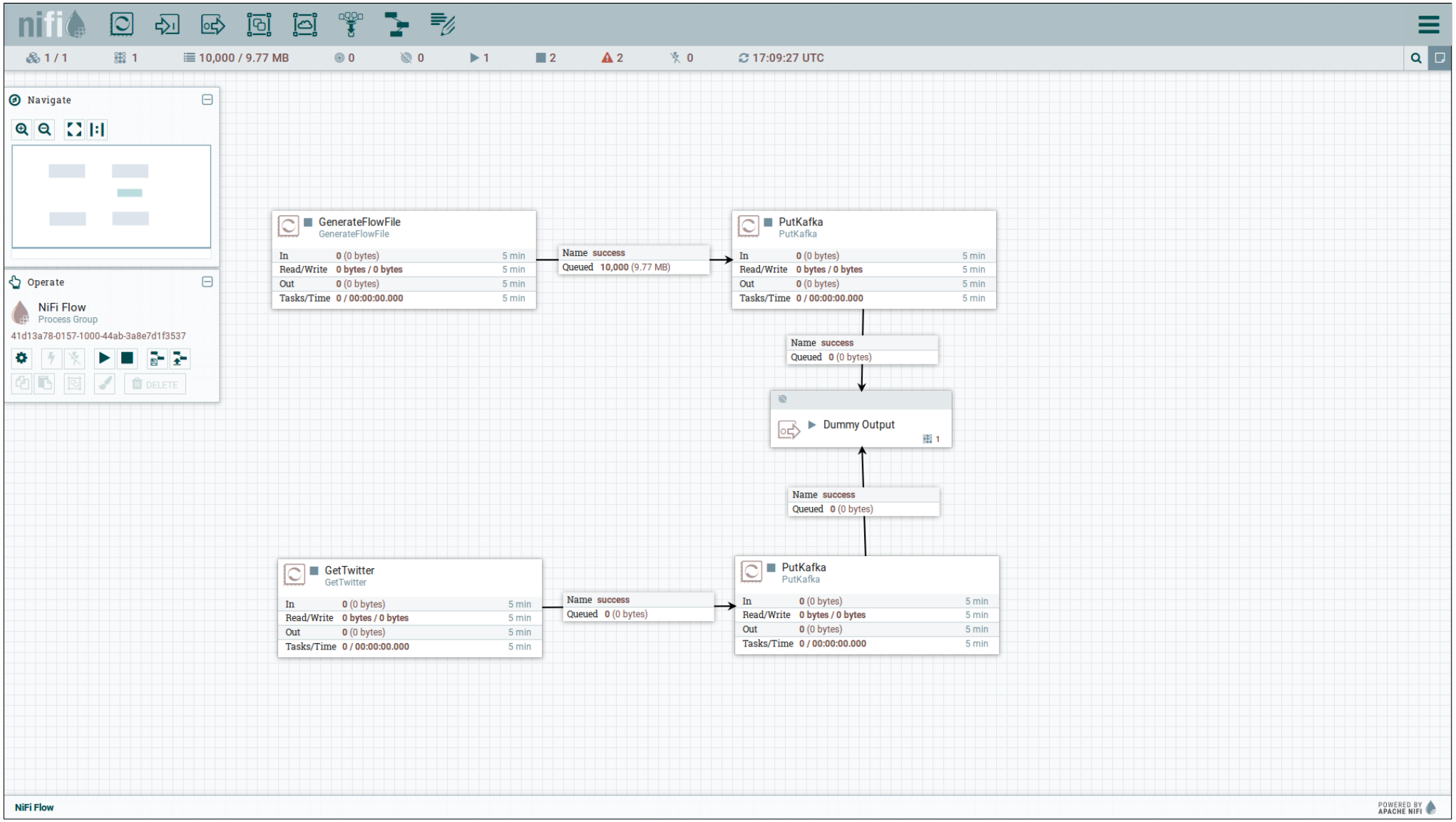


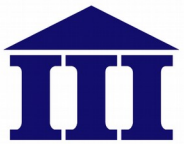
Twitter to HDF

cd /usr/hdf/2.0.0.0-579/kafka/bin/

./kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 2 --topic tweet

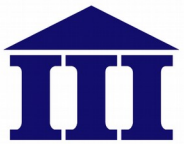
Twitter to HDF





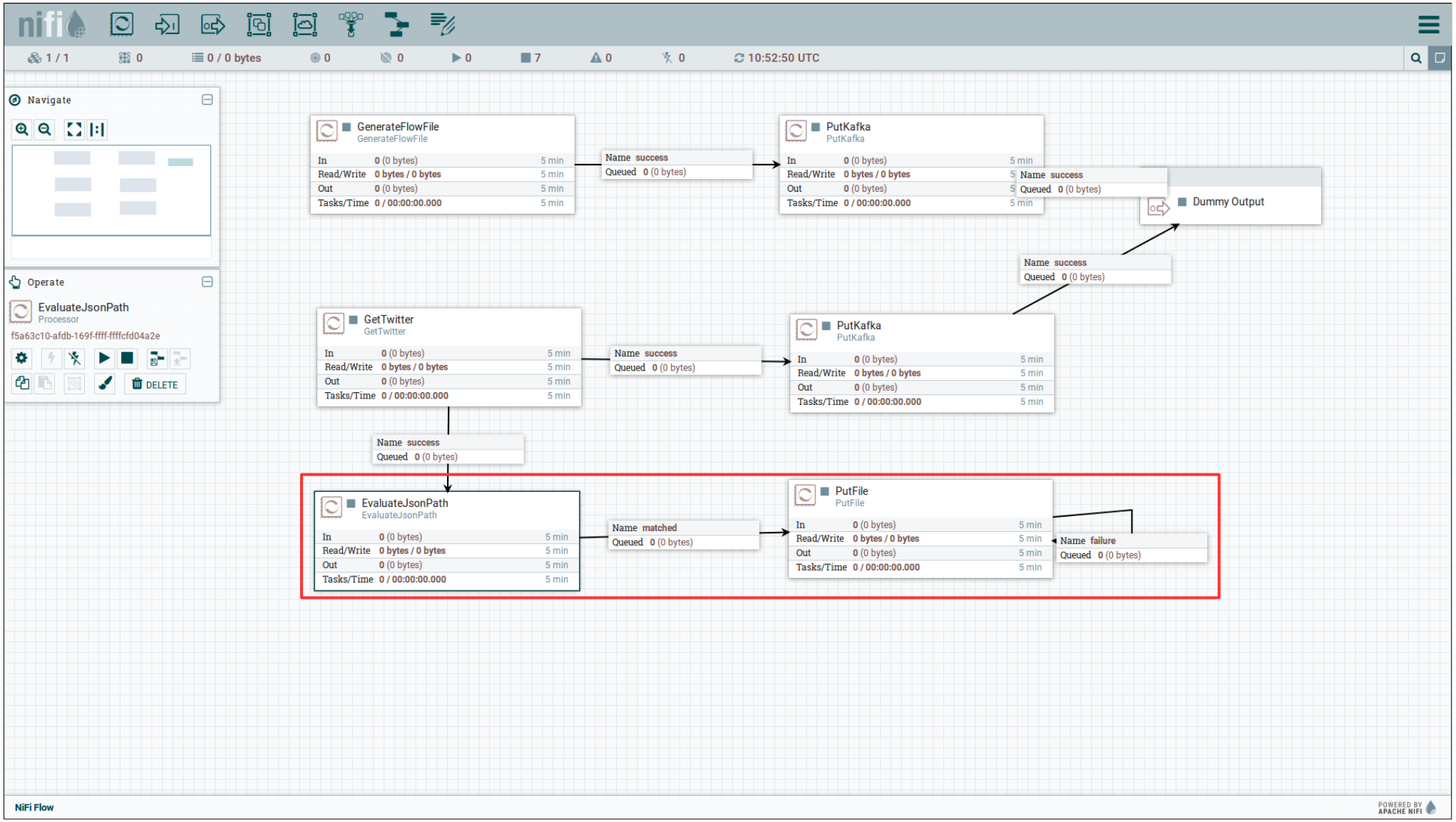
Check result on Kafka

```
./kafka-console-consumer.sh --zookeeper
localhost:2181 --topic tweet --from-beginning
```

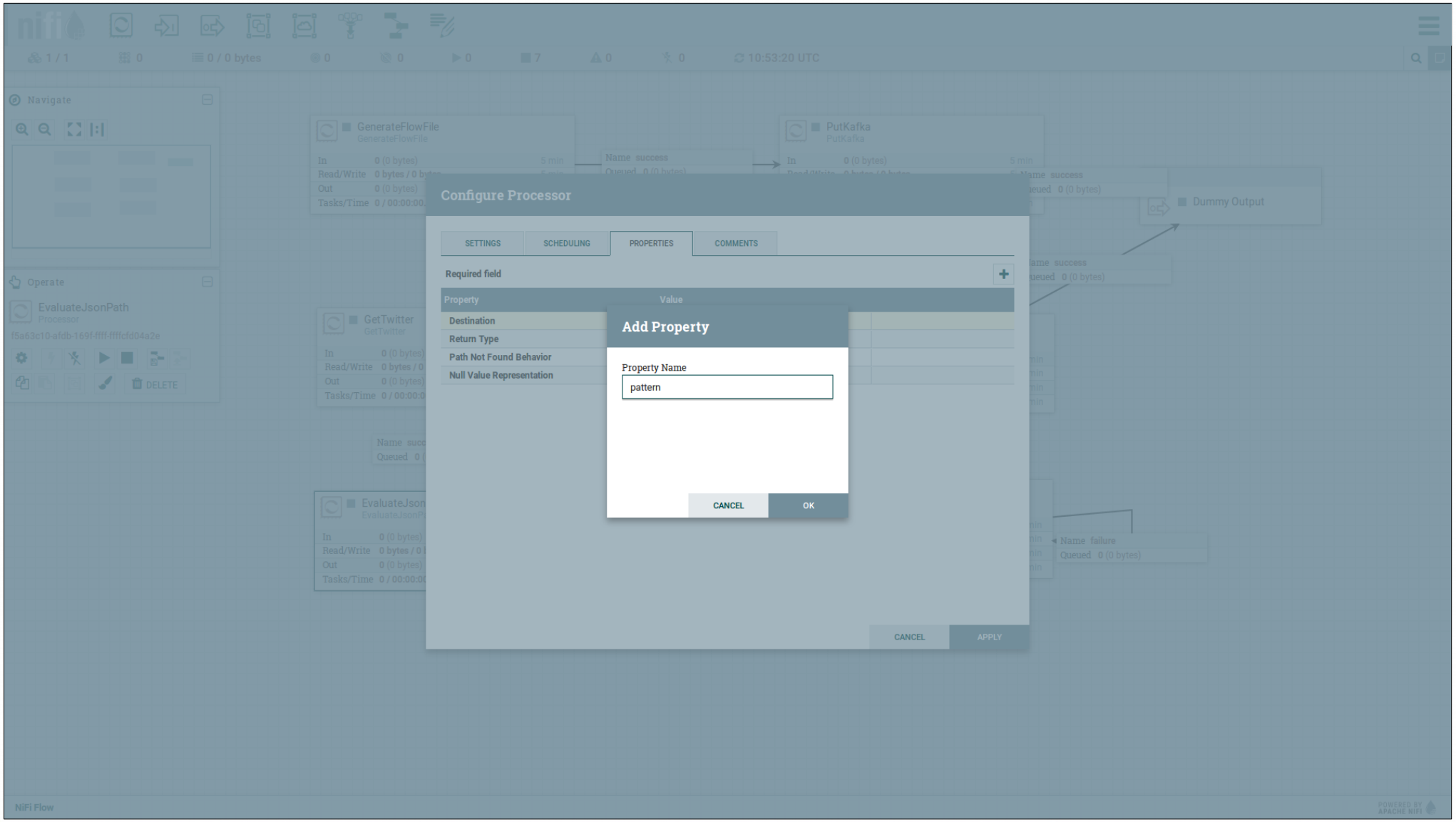


Pre-Process

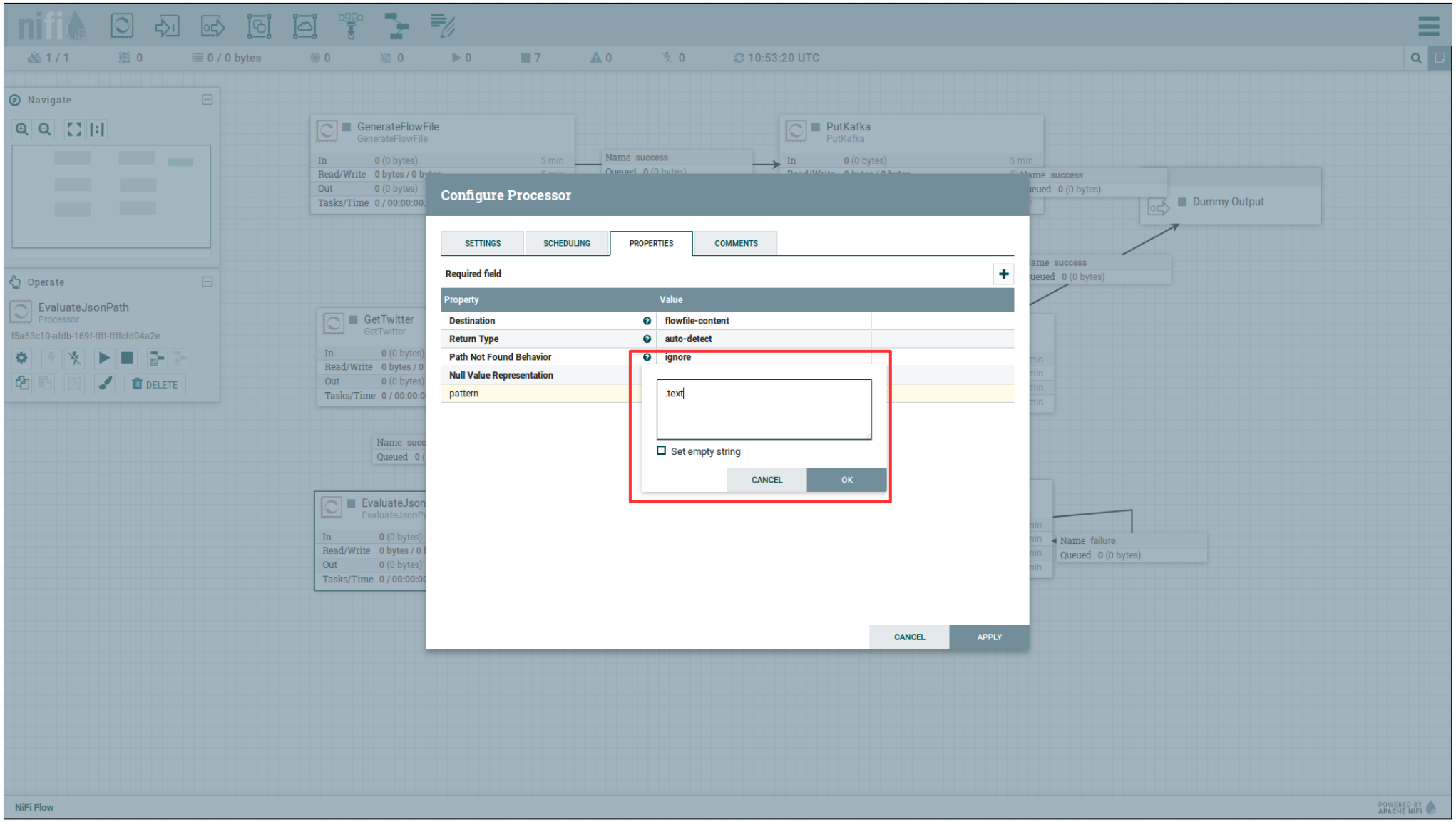
Content extraction



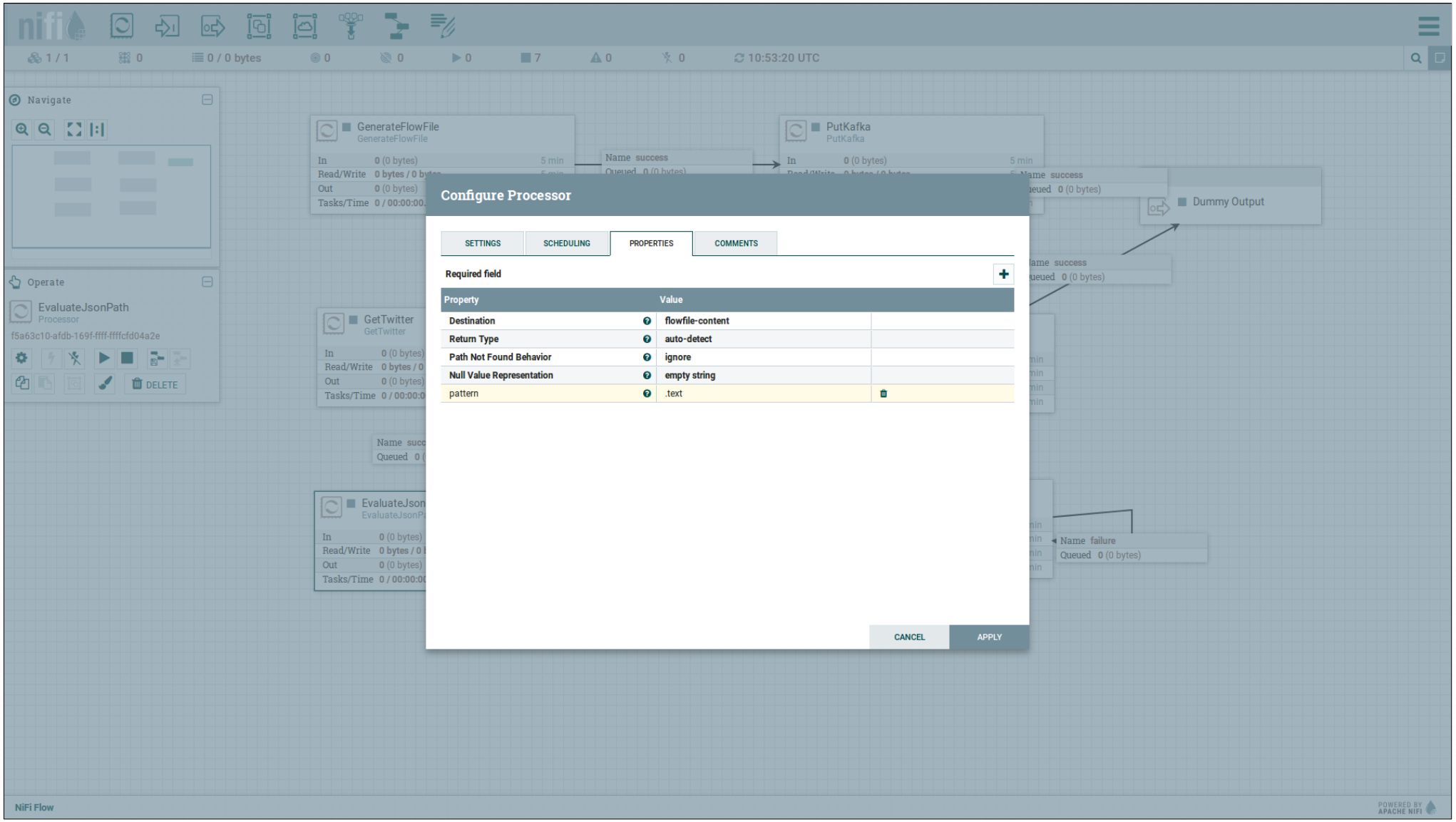
Content extraction



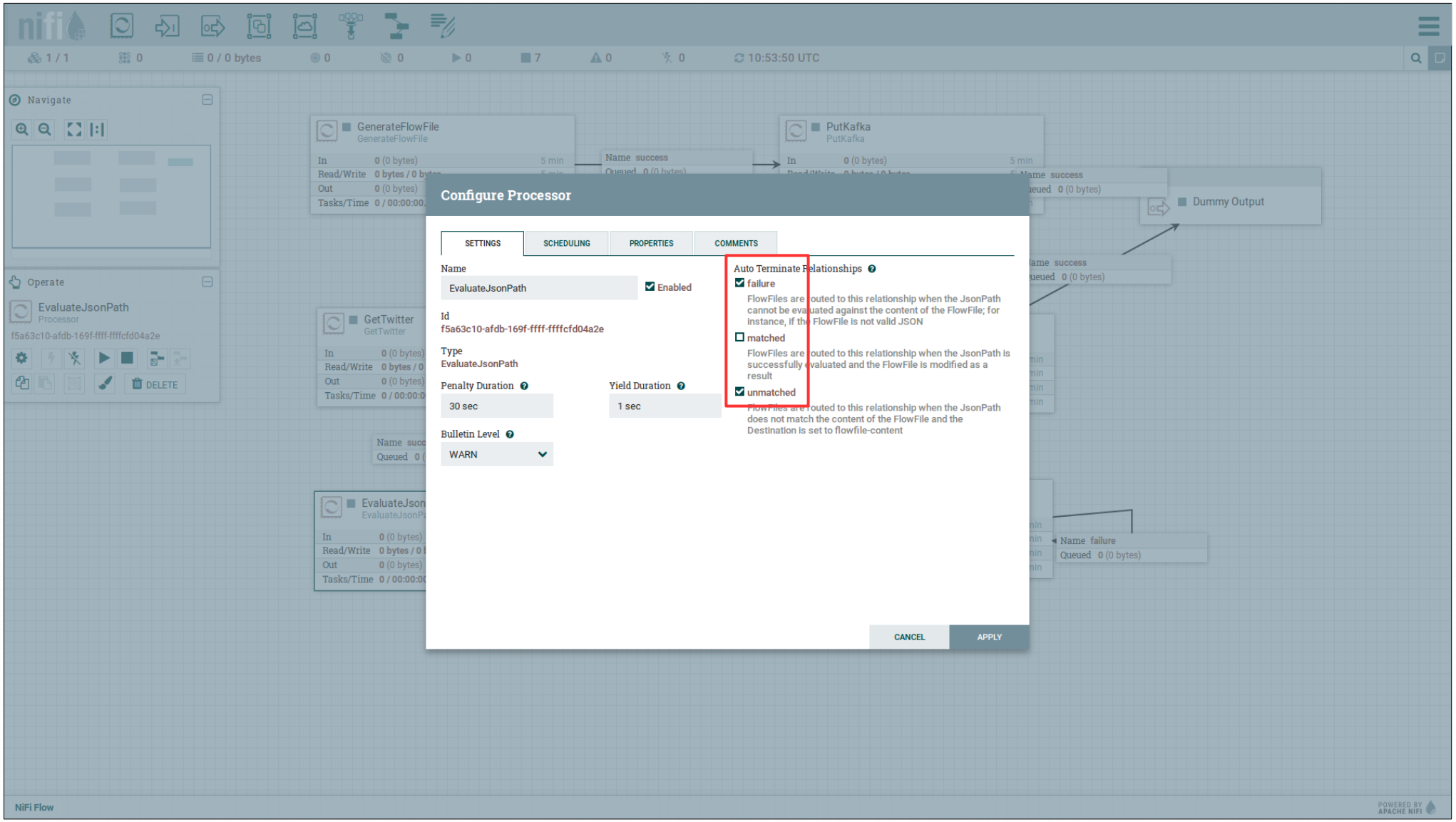
Content extraction



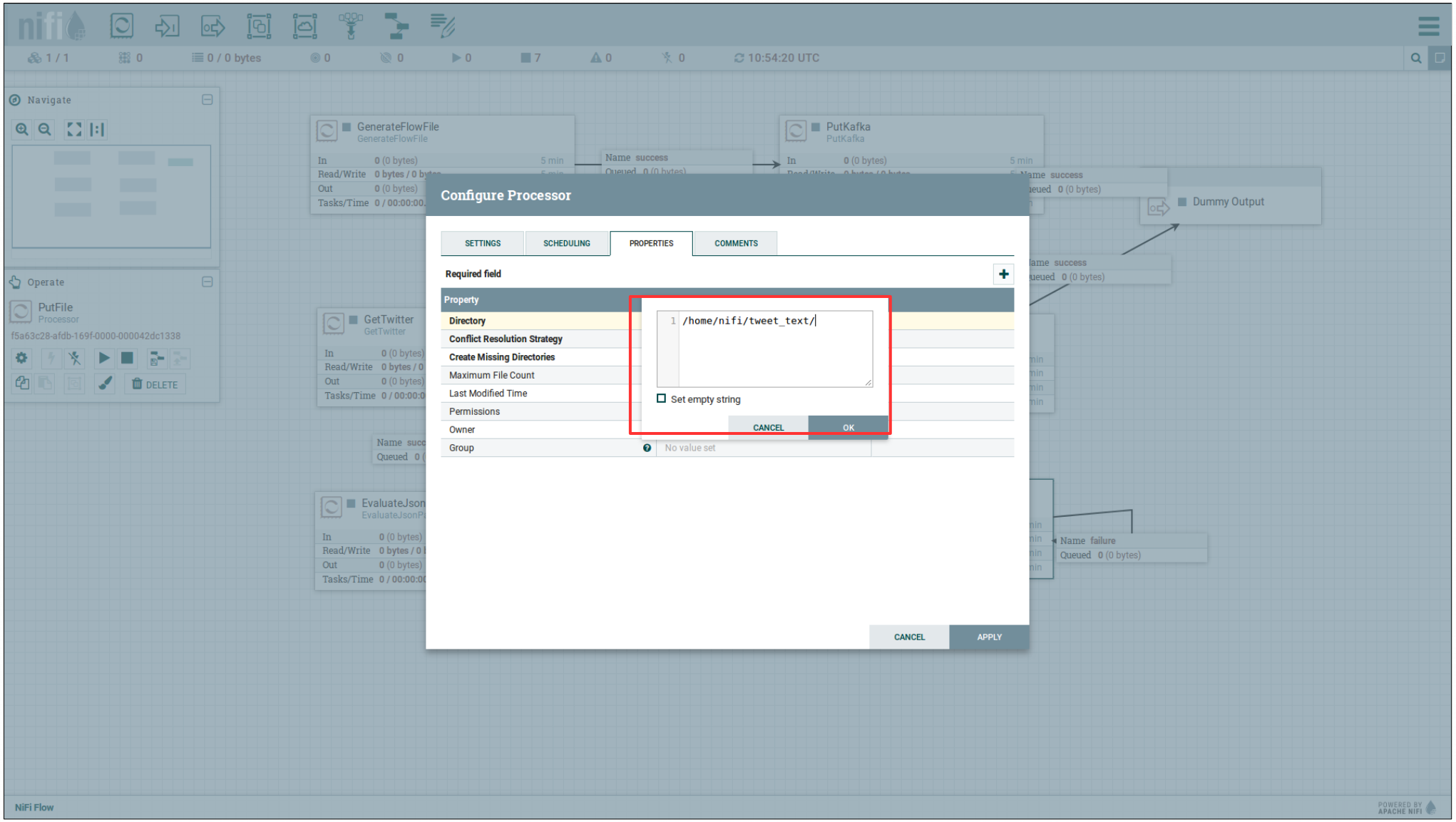
Content extraction



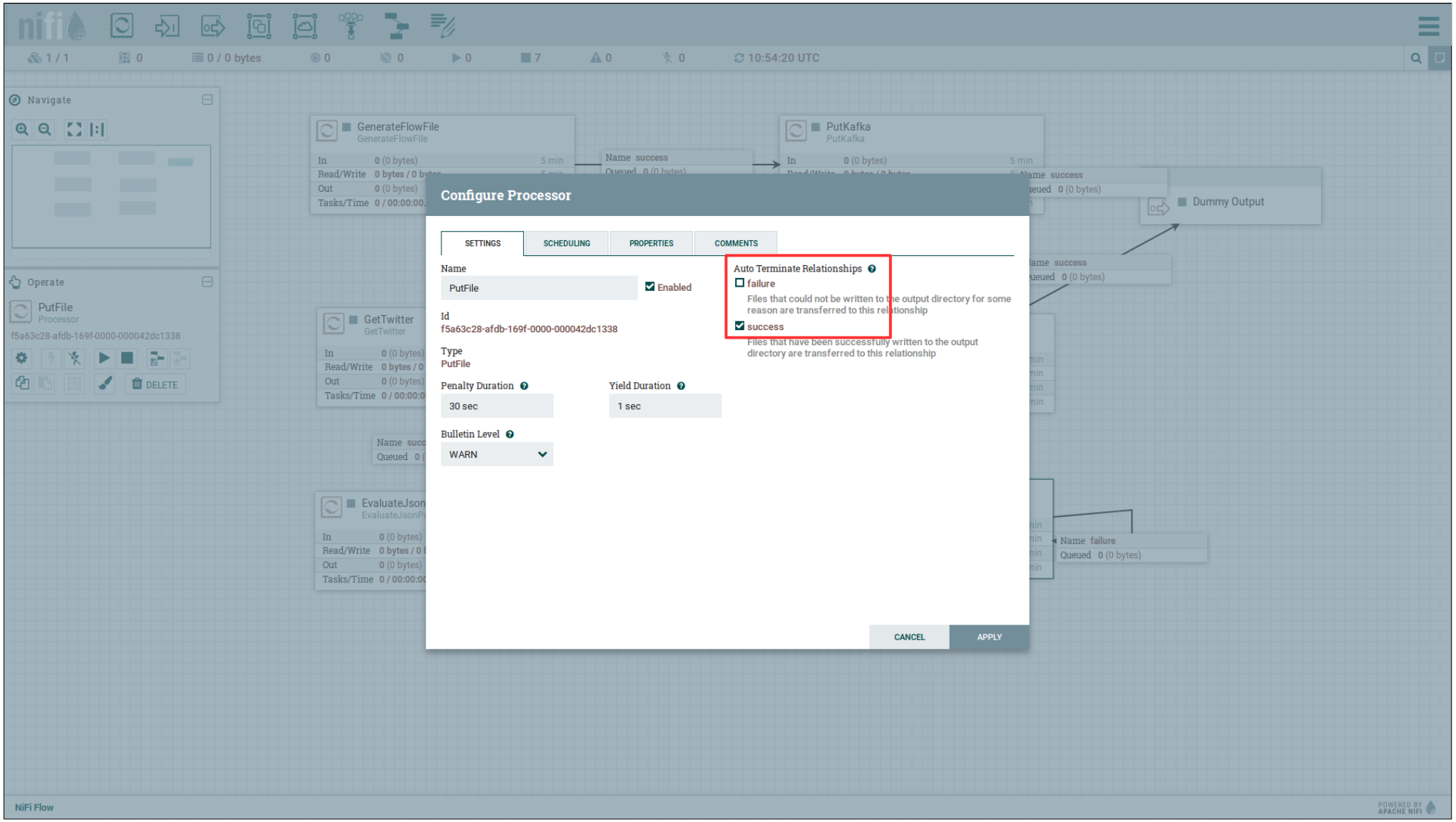
Content extraction



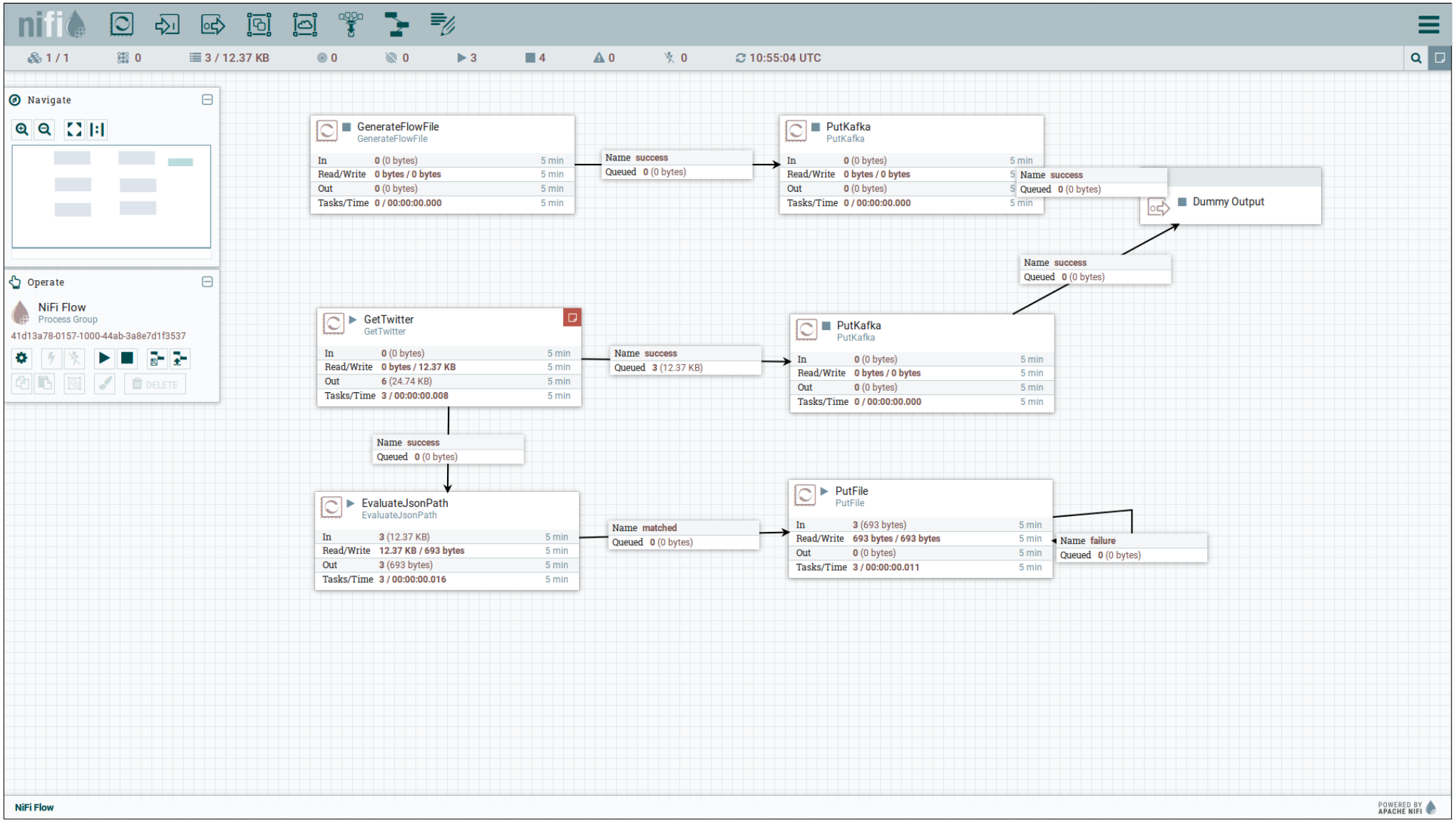
Content extraction



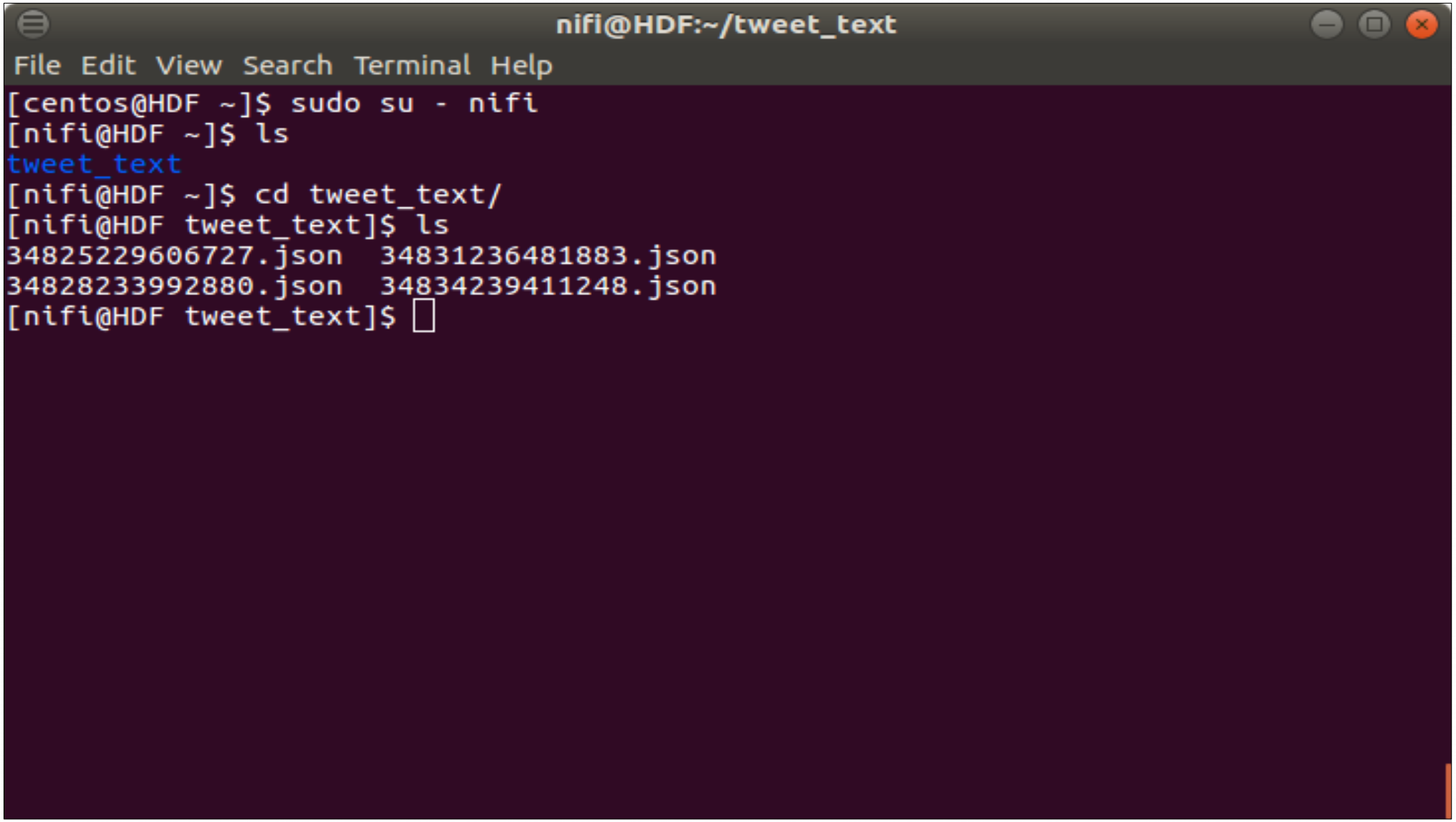
Content extraction

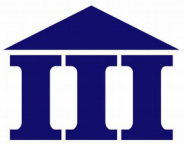


Start Content extraction

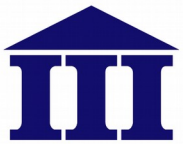


Check result in terminal





ExecuteProcess

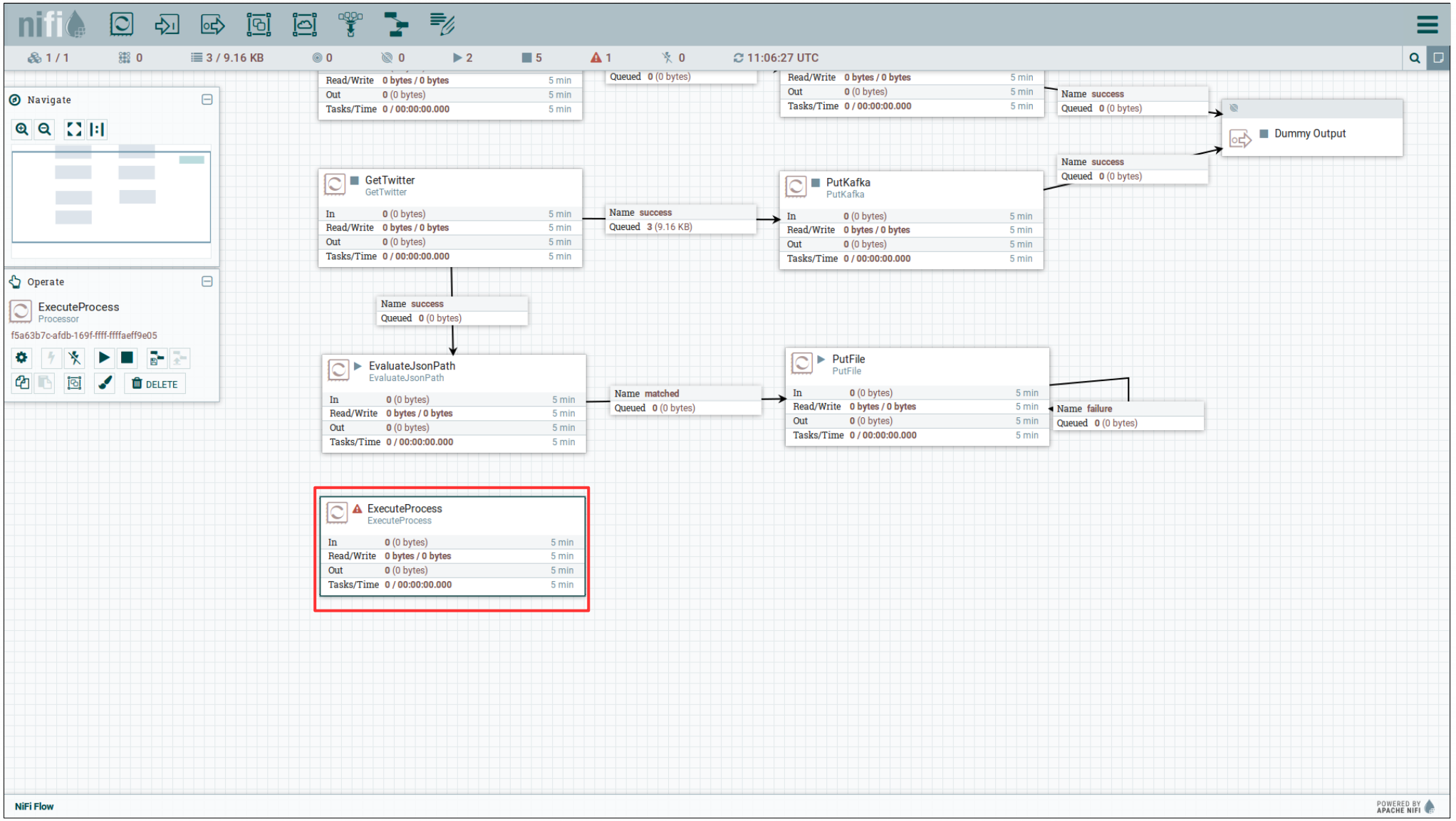


Create process

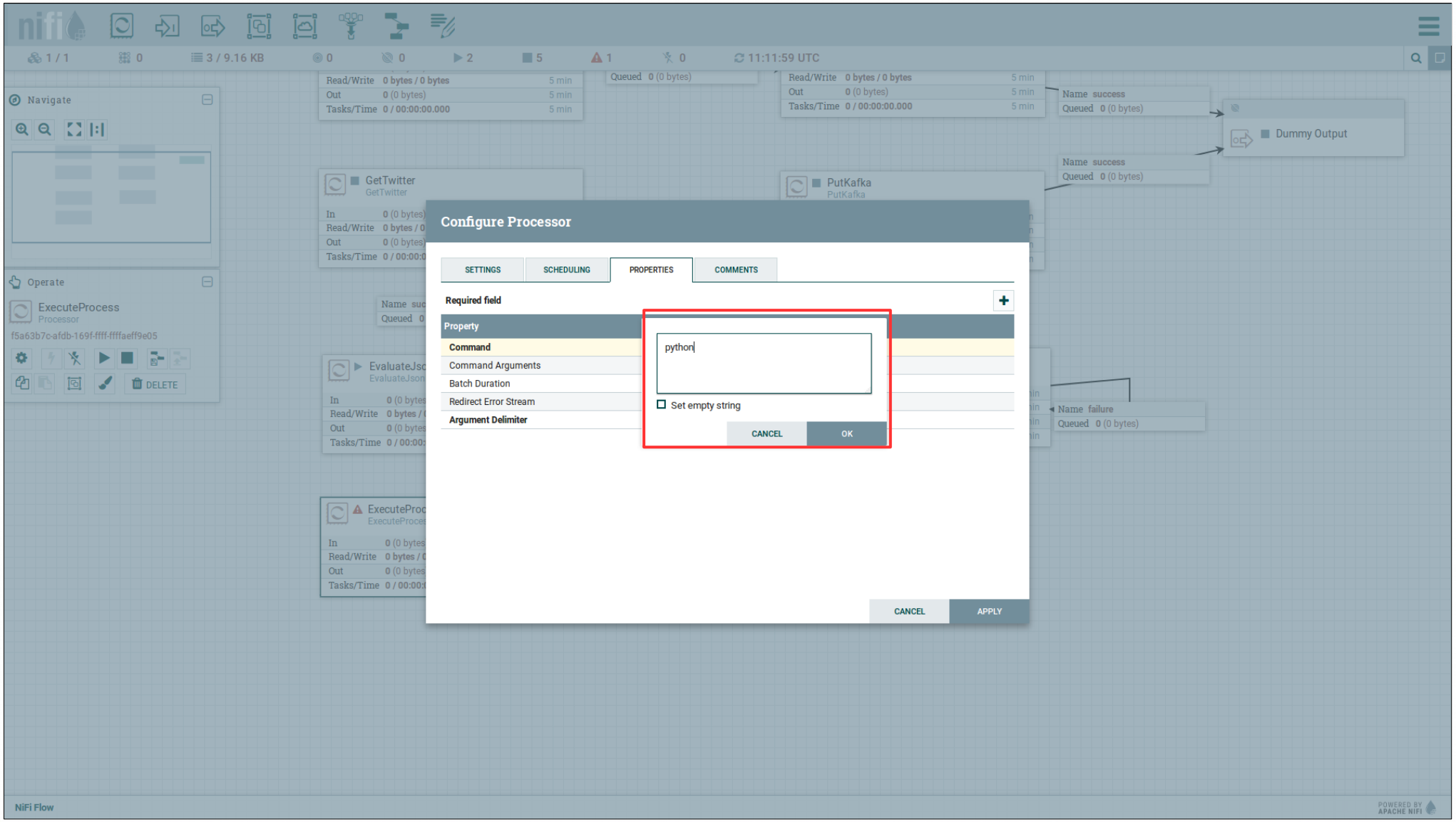
vi random_number.py

```
import random

print '%.0f' % random.uniform(1, 10) + ' ' +
str(random.uniform(100, 1000))
```

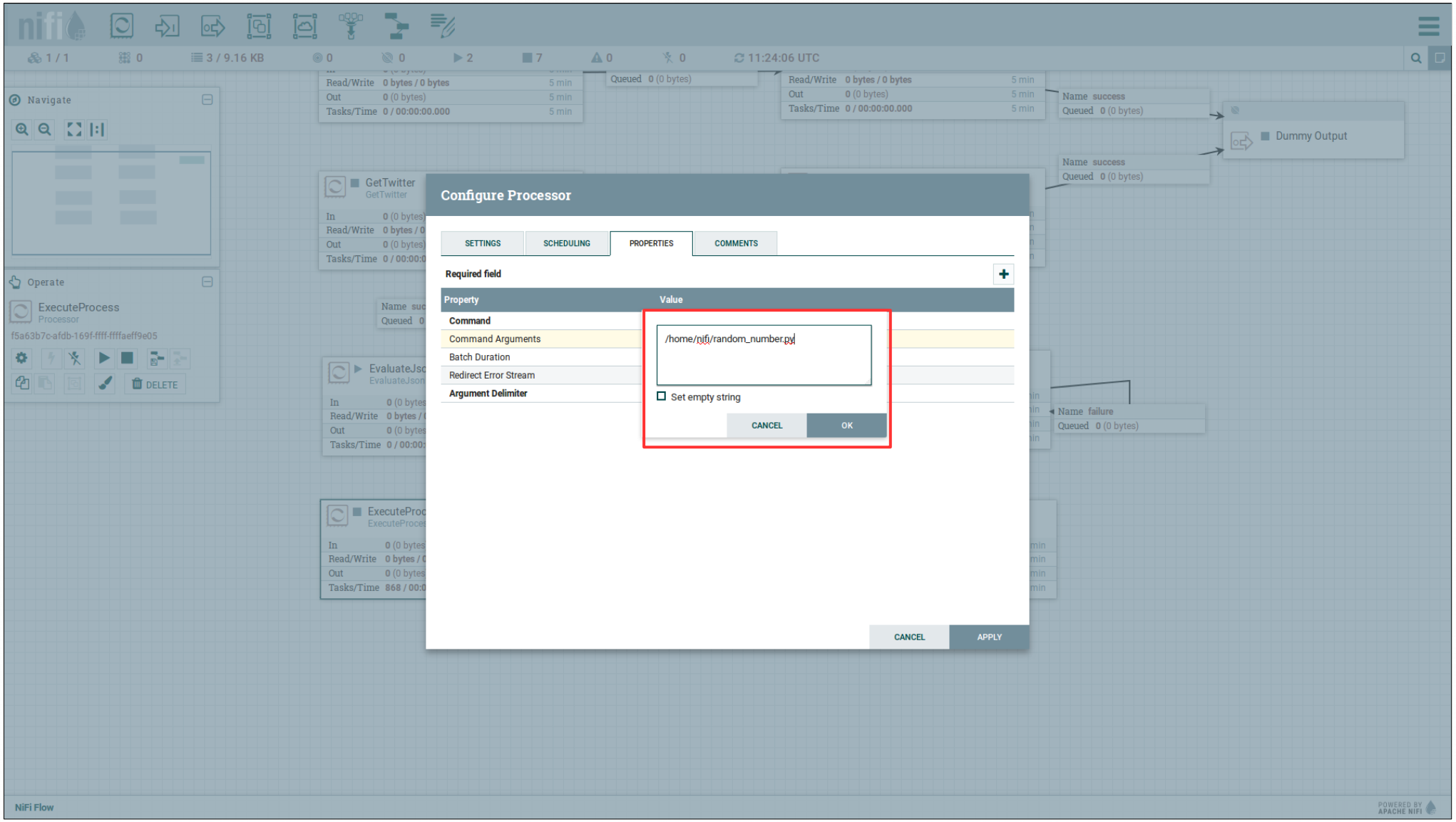
Create process



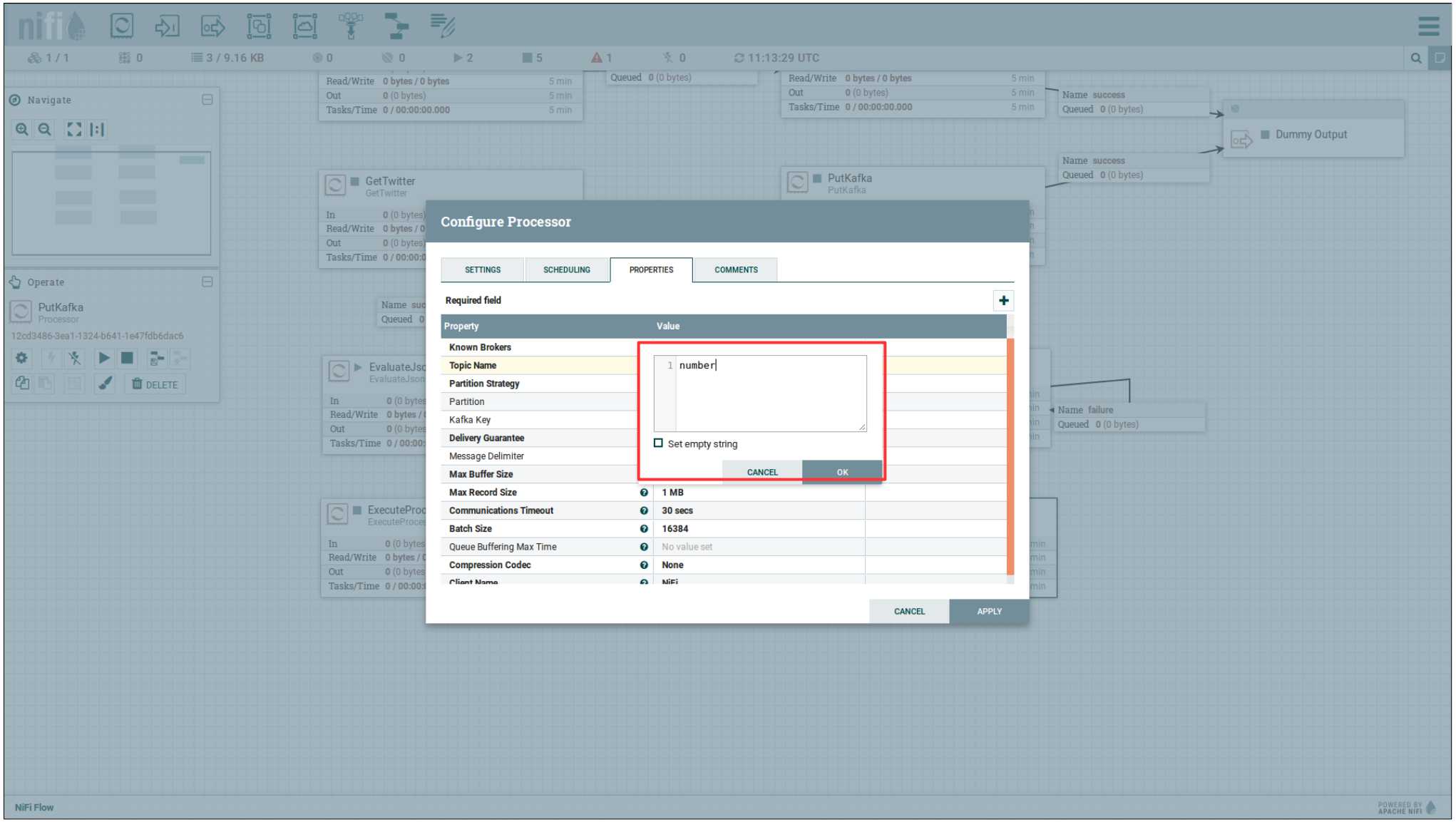
Create process

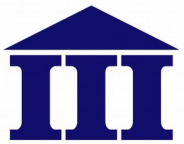


Create process



Create process



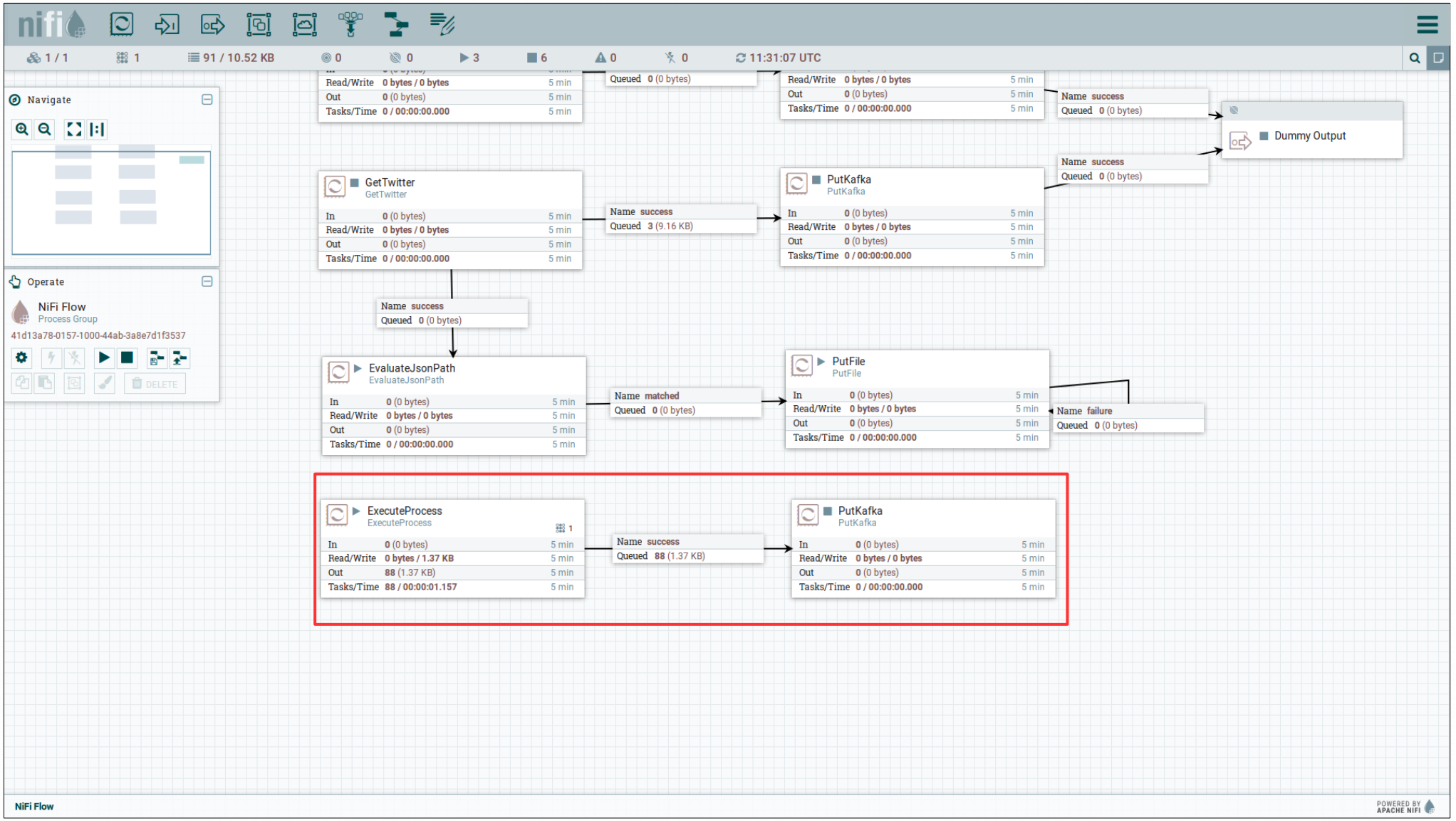


Create Kafka topic

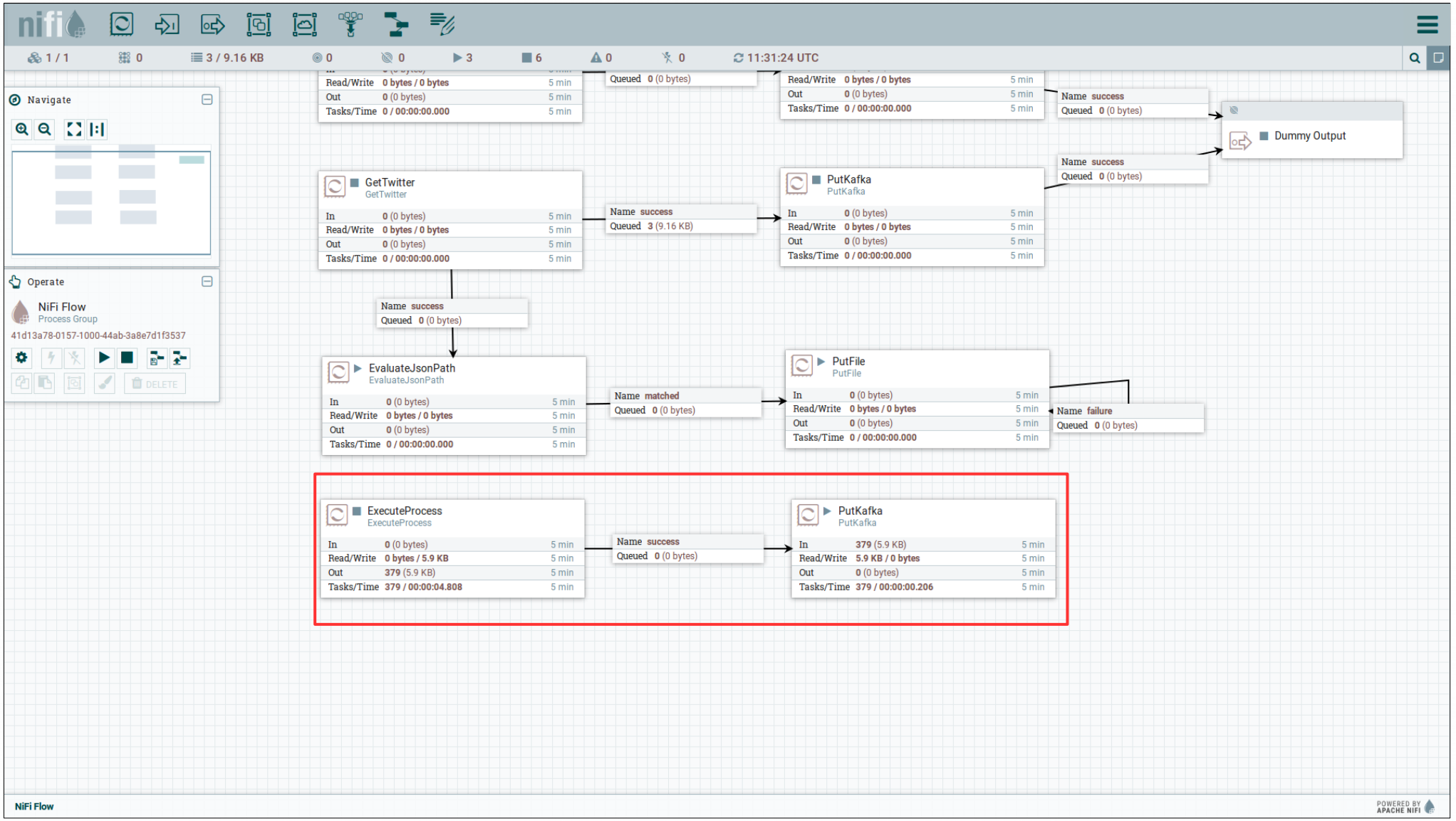
cd /usr/hdf/2.0.0.0-579/kafka/bin/

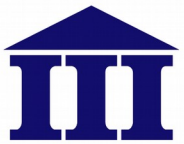
./kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 2 --topic number

Start ExecuteProcess



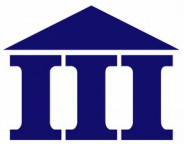
Start ExecuteProcess





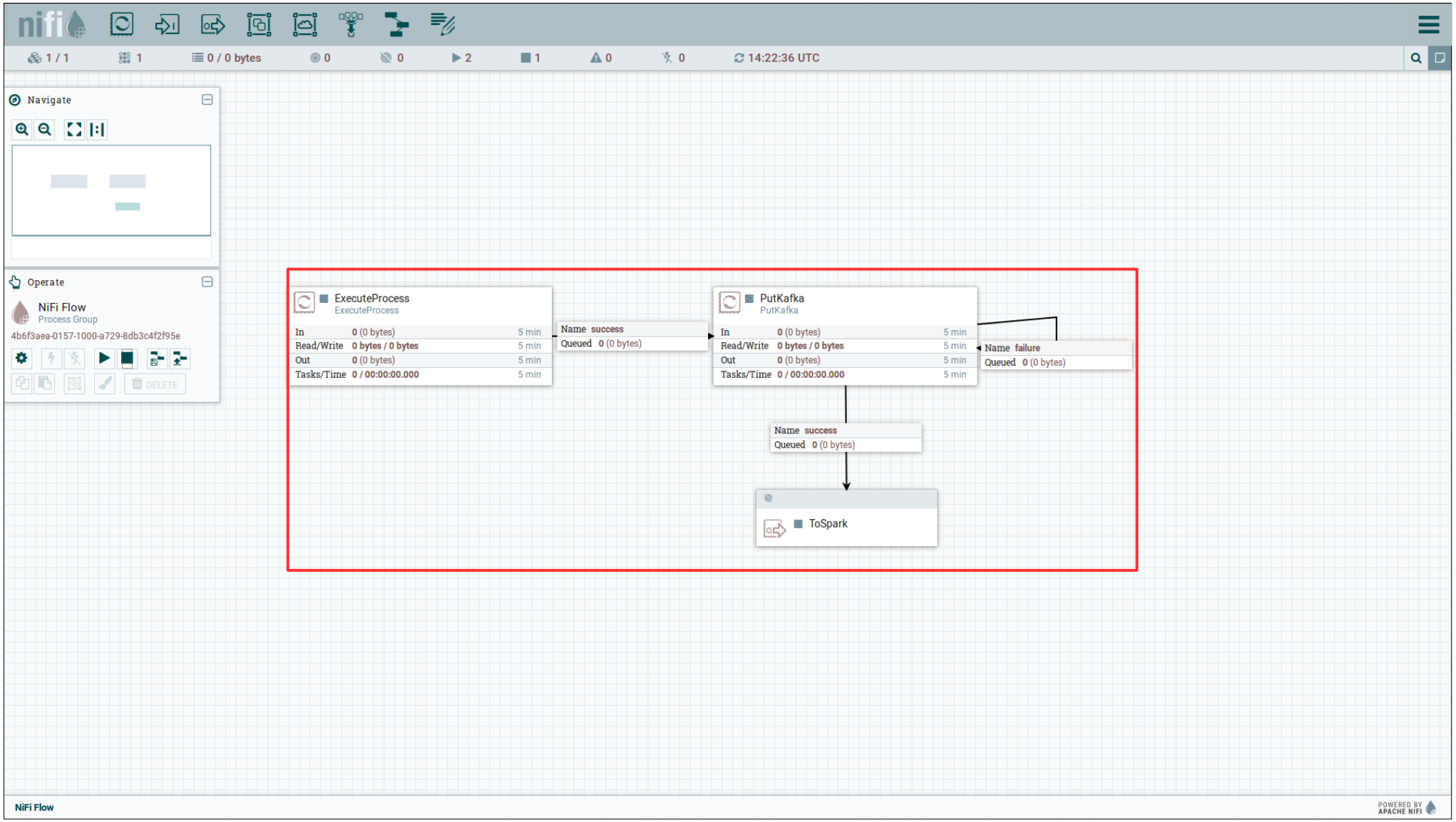
Check result on Kafka

```
./kafka-console-consumer.sh --zookeeper
localhost:2181 --topic number --from-beginning
```

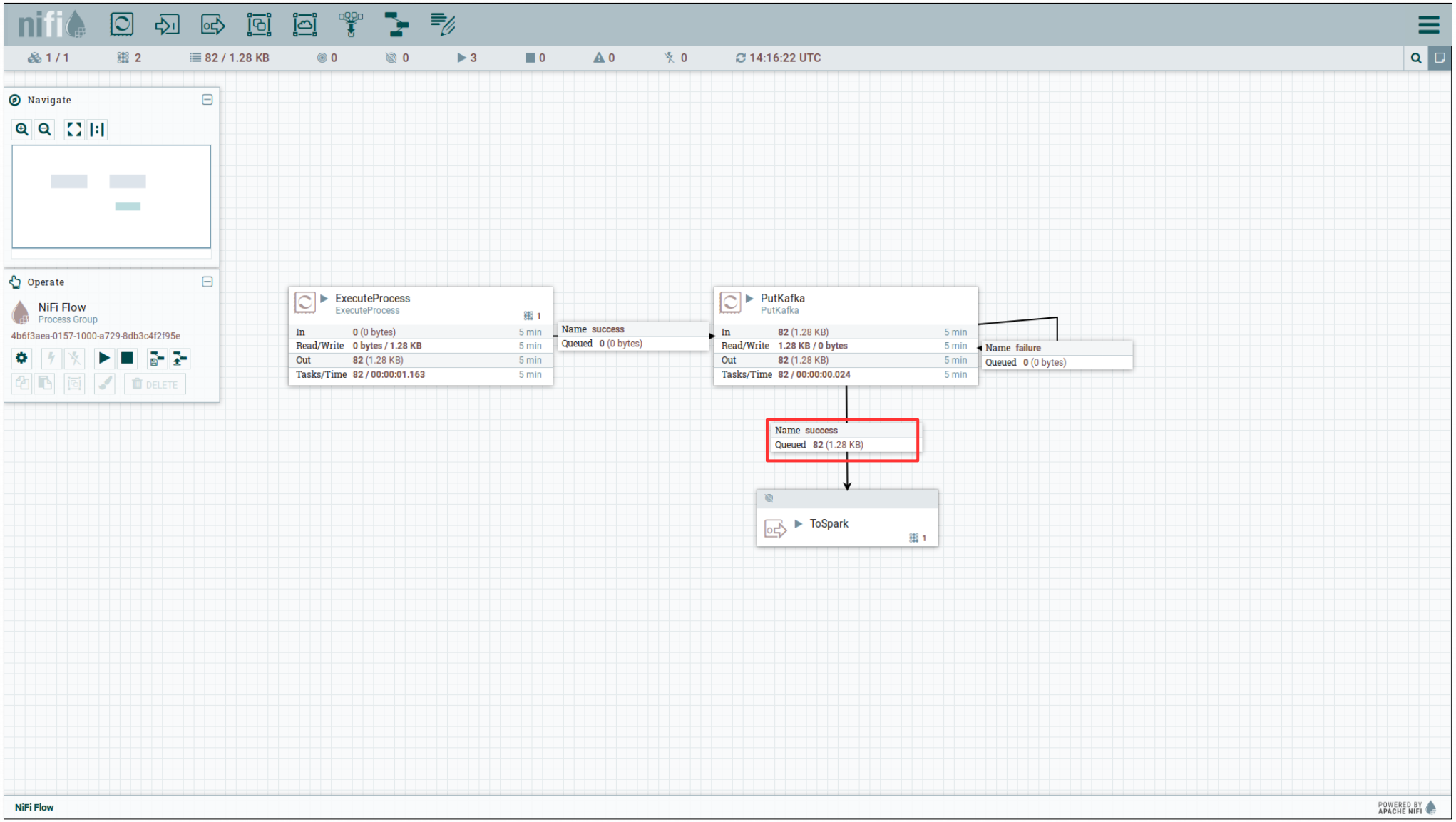


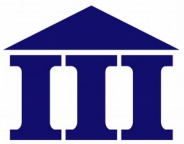
To Spark Streaming

NiFi



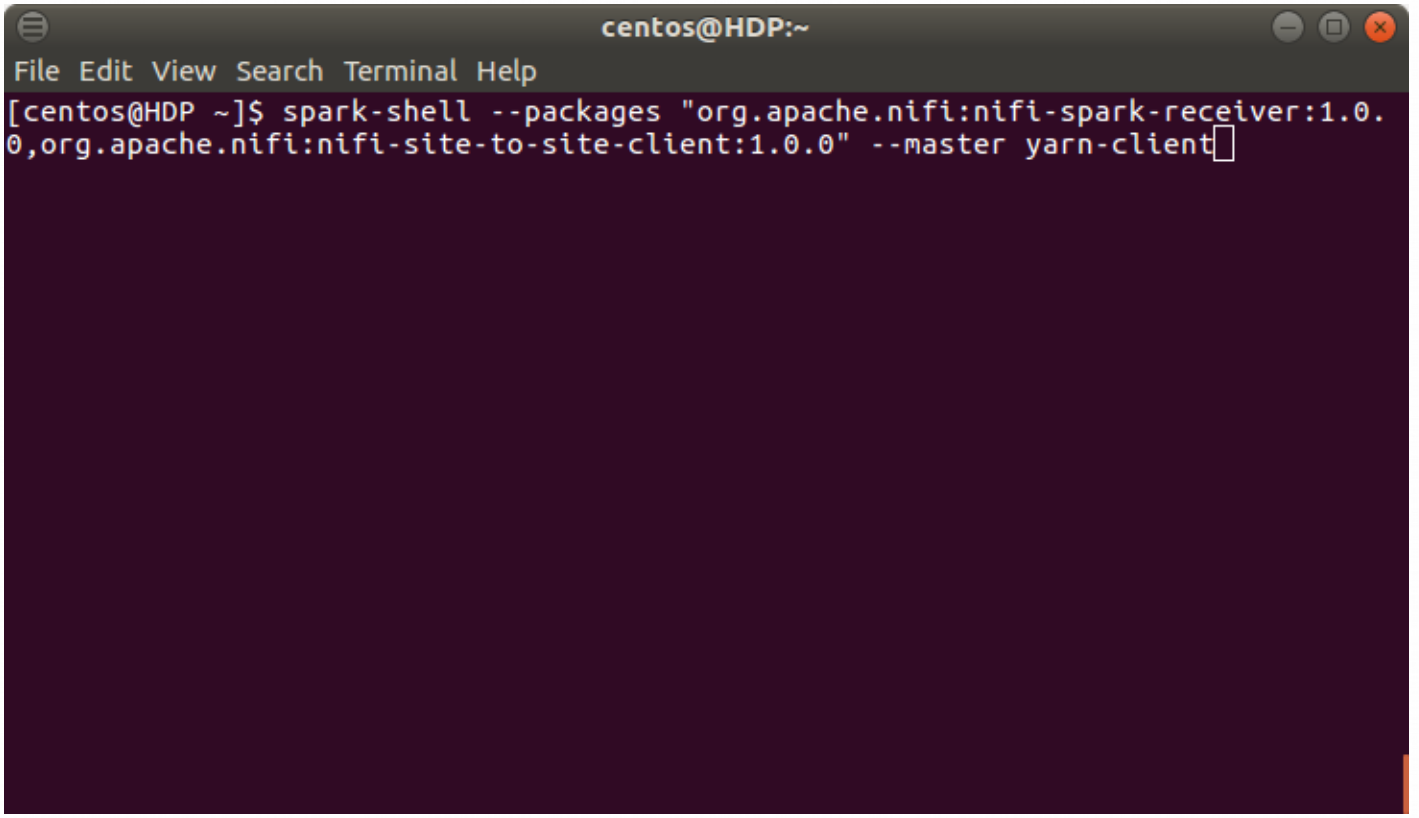
NiFi

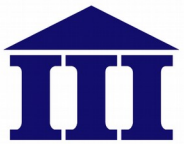




Spark-Shell

spark-shell --packages "org.apache.nifi:nifi-spark-receiver:1.0.0,org.apache.nifi:nifi-site-to-site-client:1.0.0" --master yarn-client

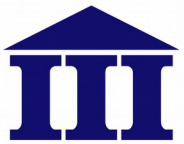




Spark-Shell

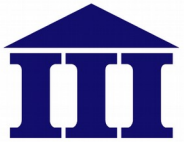
:paste

```
centos@HDP:~
File  Edit  View  Search  Terminal  Help
16/09/21 14:30:33 INFO ClientWrapper: Inspected Hadoop version: 2.7.3.2.5.0.0-12
45
16/09/21 14:30:33 INFO ClientWrapper: Loaded org.apache.hadoop.hive.shims.Hadoop
23Shims for Hadoop version 2.7.3.2.5.0.0-1245
16/09/21 14:30:33 INFO metastore: Trying to connect to metastore with URI thrift
://hdp.maas:9083
16/09/21 14:30:33 INFO metastore: Connected to metastore.
16/09/21 14:30:33 INFO SessionState: Created local directory: /tmp/75f36064-c2ae
-4541-b864-558e1ec2c5f6_resources
16/09/21 14:30:33 INFO SessionState: Created HDFS directory: /tmp/hive/centos/75
f36064-c2ae-4541-b864-558e1ec2c5f6
16/09/21 14:30:33 INFO SessionState: Created local directory: /tmp/centos/75f360
64-c2ae-4541-b864-558e1ec2c5f6
16/09/21 14:30:33 INFO SessionState: Created HDFS directory: /tmp/hive/centos/75
f36064-c2ae-4541-b864-558e1ec2c5f6/_tmp_space.db
16/09/21 14:30:33 INFO SparkILoop: Created sql context (with Hive support)..
SQL context available as sqlContext.

scala> :paste
// Entering paste mode (ctrl-D to finish)
```



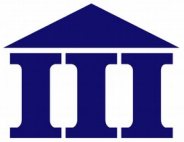
Spark-Shell

```
import java.nio.charset.StandardCharsets
import scala.collection.mutable.ListBuffer
import org.apache.spark.SparkConf
import org.apache.spark.SparkContext
import org.apache.spark.rdd.RDD
import org.apache.spark.sql.SQLContext
import org.apache.spark.storage.StorageLevel
```



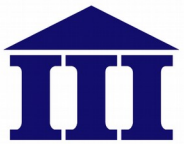
Spark-Shell

```
import org.apache.spark.streaming.{Seconds,
StreamingContext, Time}

import org.apache.nifi.remote.client.SiteToSiteClient

import
org.apache.nifi.remote.client.SiteToSiteClientConfig

import org.apache.nifi.spark.NiFiReceiver

import org.apache.nifi.spark.NiFiDataPacket
```



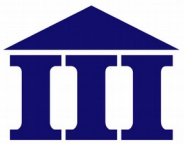
Spark-Shell

```
case class Record_sum(time: Time, id: String, sum: Double)

case class Record_count(time: Time, id: String, count: Int)

case class Record(time: Time, id: String, avg: Double)
```



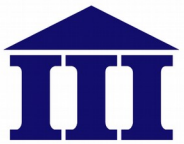
Spark-Shell

```
val clientConfig = new SiteToSiteClient.Builder()
                .url("http://nifi.maas:9090/nifi")
                .portName("ToSpark")
                .buildConfig();
```



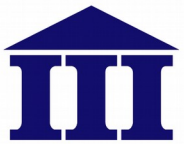
Spark-Shell

```
val sparkConf = new SparkConf().setAppName("NiFi_Spark Streaming example")

val ssc = new StreamingContext(sc, Seconds(5))

val stream = ssc.receiverStream(new NiFiReceiver(clientConfig, StorageLevel.MEMORY_ONLY))
```



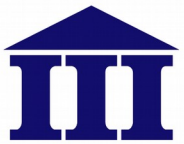
Spark-Shell

```
val lines = stream.map( (packet:
NiFiDataPacket) => new
String(packet.getContent(),
StandardCharsets.UTF_8) )

val sum = lines.map(line => (line(0).toString,
line(1).toDouble))
                        .reduceByKey((x, y) => x + y )
```



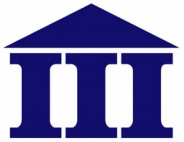
Spark-Shell

```
val count = lines.map(line => (line(0).toString, 1))
                 .reduceByKey((x, y) => x + y )
```



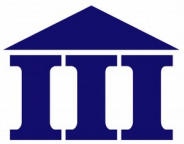
Spark-Shell

```
val avg = lines.map(line => (line(0).toString,
(line(1).toDouble, 1)))
                .reduceByKey((x, y) => (x._1 + y._1,
x._2 + y._2))
                .mapValues{ case (sum, count) =>
(1.0 * sum) / count }
```



Spark-Shell

```
sum.print()
count.print()
avg.print()
ssc.start()
```

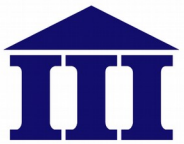


NiFi

centos@HDP:~

File Edit View Search Terminal Help

```
val sum = lines.map(line => (line(0).toString, line(1).toDouble))
                        .reduceByKey((x, y) => x + y )

val count = lines.map(line => (line(0).toString, 1))
                        .reduceByKey((x, y) => x + y )

val avg = lines.map(line => (line(0).toString, (line(1).toDouble, 1)))
                .reduceByKey((x, y) => (x._1 + y._1, x._2 + y._2))
                .mapValues{ case (sum, count) => (1.0 * sum) / count }

sum.print()

count.print()

avg.print()

ssc.start()

// Exiting paste mode, now interpreting.
```

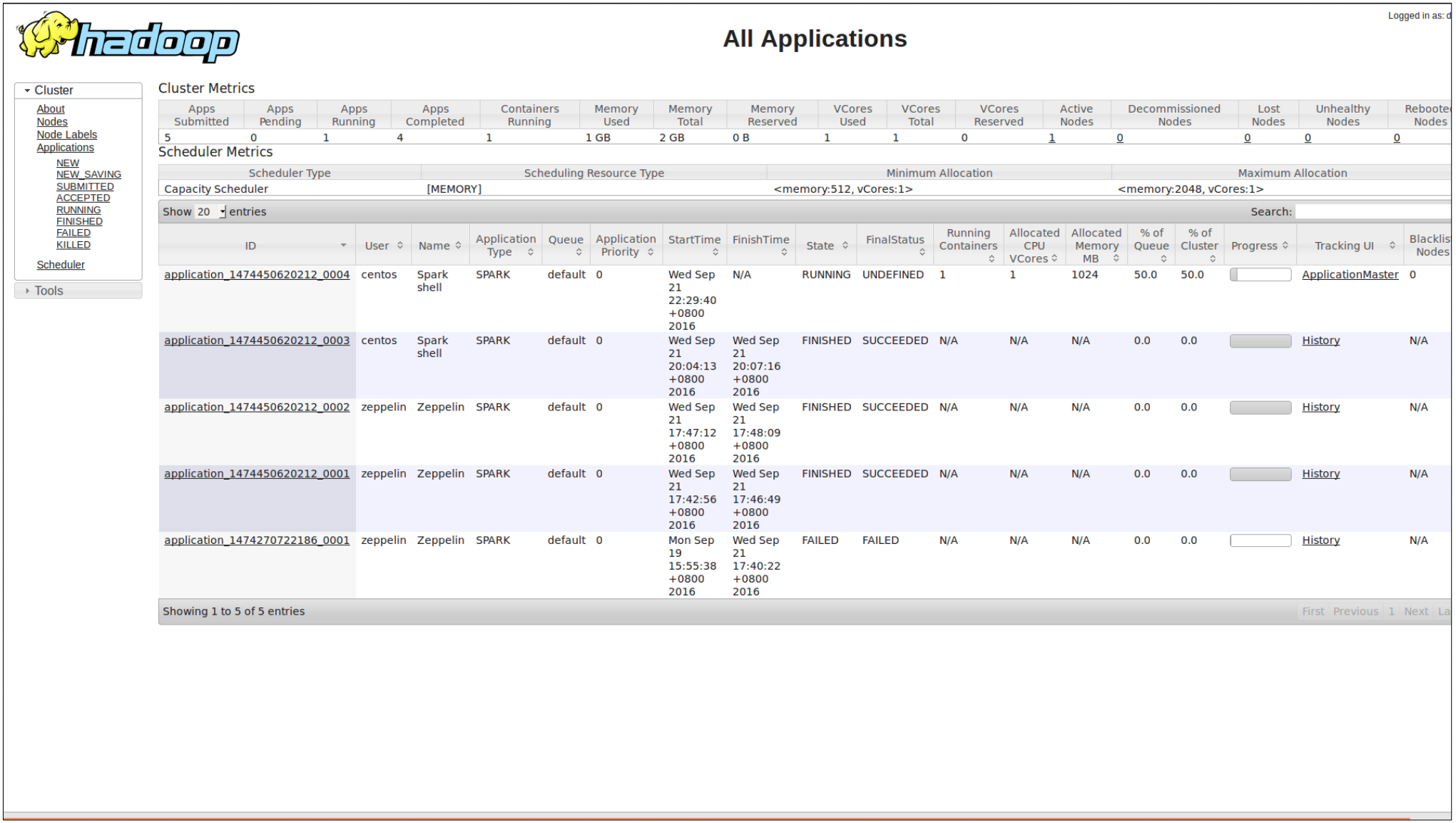


NiFi

centos@HDP:~

File Edit View Search Terminal Help

```
16/09/21 14:26:35 INFO DAGScheduler: Job 6 finished: print at <console>:65, took
 0.012255 s
-------------------------------------------
Time: 1474467995000 ms
-------------------------------------------
(4,32.0)
(8,32.0)
(6,32.0)
(2,32.0)
(7,32.0)
(5,32.0)
(9,32.0)
(3,32.0)
(1,40.12698412698413)

16/09/21 14:26:35 INFO JobScheduler: Finished job streaming job 1474467995000 ms
.2 from job set of time 1474467995000 ms
16/09/21 14:26:35 INFO JobScheduler: Total delay: 0.422 s for time 1474467995000
 ms (execution: 0.356 s)
16/09/21 14:26:35 INFO ReceivedBlockTracker: Deleting batches ArrayBuffer()
16/09/21 14:26:35 INFO InputInfoTracker: remove old batch metadata:
```



Logged in as: d

All Applications

- Cluster
  - About
  - Nodes
  - Node Labels
  - Applications
    - NEW
    - NEW_SAVING
    - SUBMITTED
    - ACCEPTED
    - RUNNING
    - FINISHED
    - FAILED
    - KILLED
  - Scheduler
- Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
5	0	1	4	1	1 GB	2 GB	0 B	1	1	0	1	0	0	0	0

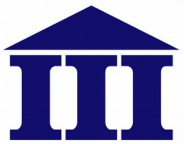
Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:512, vCores:1>	<memory:2048, vCores:1>

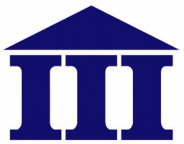
Show 20 entries — Search:

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	% of Queue	% of Cluster	Progress	Tracking UI	Blacklisted Nodes
application_1474450620212_0004	centos	Spark shell	SPARK	default	0	Wed Sep 21 22:29:40 +0800 2016	N/A	RUNNING	UNDEFINED	1	1	1024	50.0	50.0		ApplicationMaster	0
application_1474450620212_0003	centos	Spark shell	SPARK	default	0	Wed Sep 21 20:04:13 +0800 2016	Wed Sep 21 20:07:16 +0800 2016	FINISHED	SUCCEEDED	N/A	N/A	N/A	0.0	0.0		History	N/A
application_1474450620212_0002	zeppelin	Zeppelin	SPARK	default	0	Wed Sep 21 17:47:12 +0800 2016	Wed Sep 21 17:48:09 +0800 2016	FINISHED	SUCCEEDED	N/A	N/A	N/A	0.0	0.0		History	N/A
application_1474450620212_0001	zeppelin	Zeppelin	SPARK	default	0	Wed Sep 21 17:42:56 +0800 2016	Wed Sep 21 17:46:49 +0800 2016	FINISHED	SUCCEEDED	N/A	N/A	N/A	0.0	0.0		History	N/A
application_1474270722186_0001	zeppelin	Zeppelin	SPARK	default	0	Mon Sep 19 15:55:38 +0800 2016	Wed Sep 21 17:40:22 +0800 2016	FAILED	FAILED	N/A	N/A	N/A	0.0	0.0		History	N/A

Showing 1 to 5 of 5 entries — First Previous 1 Next La

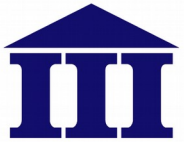


Homework



Homework

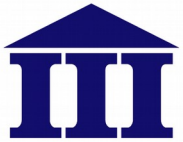
1. Create three topics.
2. Match Twitter’s text, user.name, created_at. (use unmatched)
3. Output text to **Spark Streaming** and write a wordcount.
4. Bonus: Output the result to hive.



Homework

What you have to give me:

1. ScreenShots.
2. Source Code.
3. Documents. (PDF)
4. GitHub or Bitbucket.



Reference

- http://hortonworks.com/
- https://community.hortonworks.com/articles/12708/nifi-feeding-data-to-spark-streaming.html
- https://blogs.apache.org/nifi/entry/stream_processing_nifi_and_spark