Session-3 - Lab 1 – Install R & RStudio

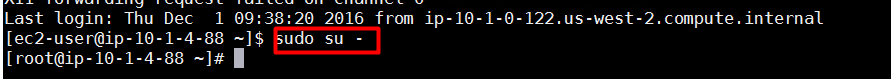
# Install R package dependencies

1. Login to Edge/CM node and install R package dependencies.

***Note: All commands must be executed ad “root” or “super user”***

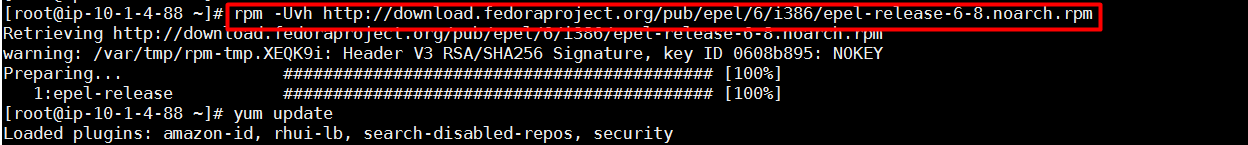
1. Switch user to root

|  |
| --- |
| sudo su - |



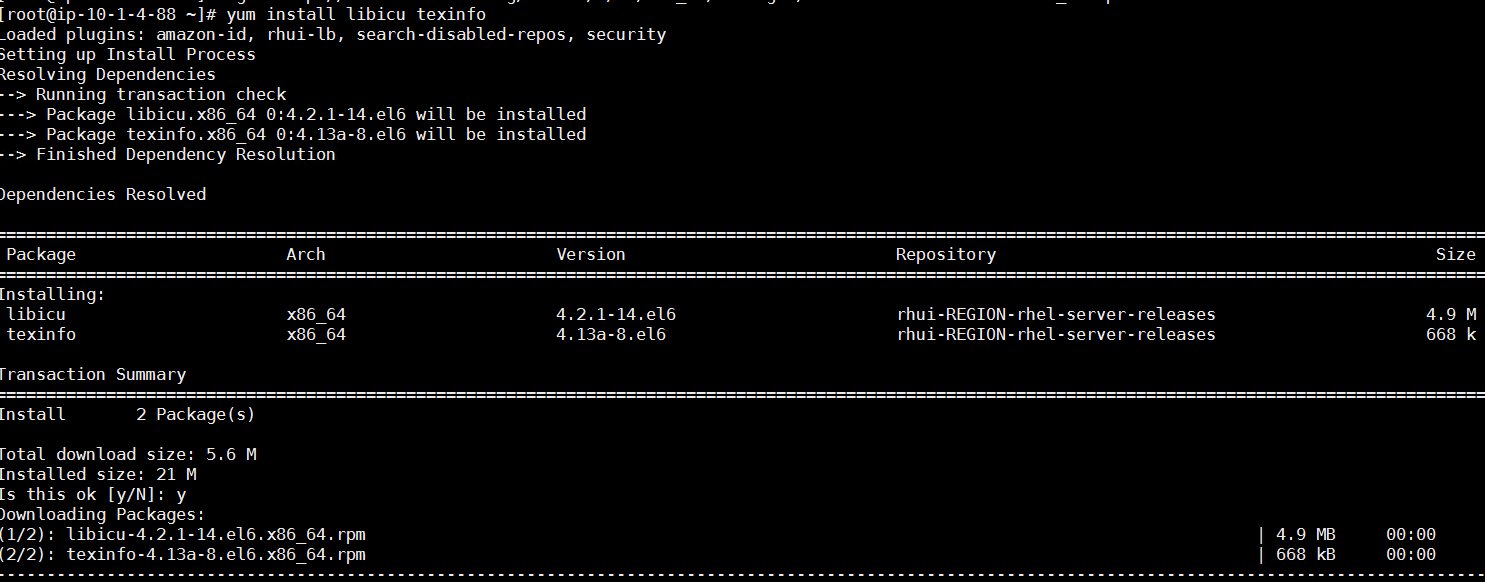
1. In order to get R running on RHEL 6, we need to add an additional repository that allows us to install the new packages. The EPEL (Extra Packages for Enterprise Linux) is a Fedora Special Interest Group that creates, maintains, and manages a high quality set of additional packages for Enterprise Linux, including, but not limited to, Red Hat Enterprise Linux (RHEL), CentOS and Scientific Linux (SL), Oracle Linux (OL).

|  |
| --- |
| rpm -Uvh http://download.fedoraproject.org/pub/epel/6/i386/epel-release-6-8.noarch.rpm |



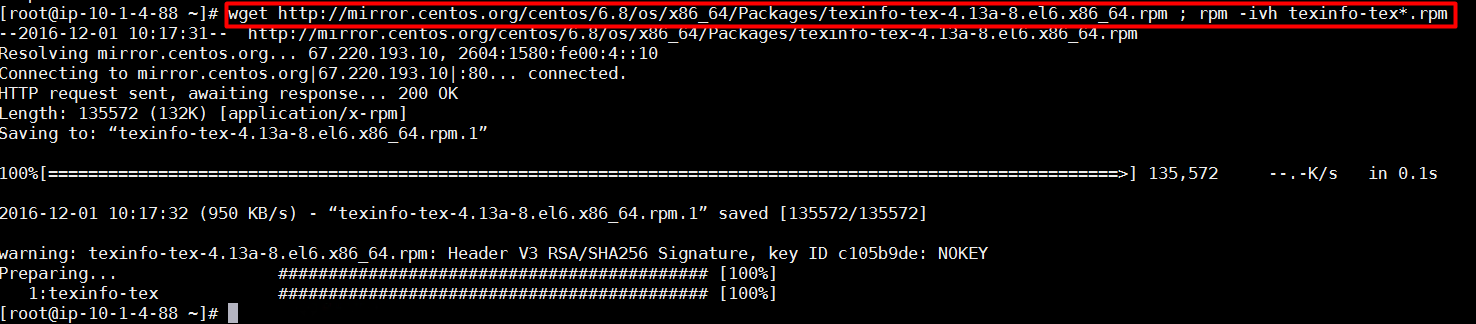
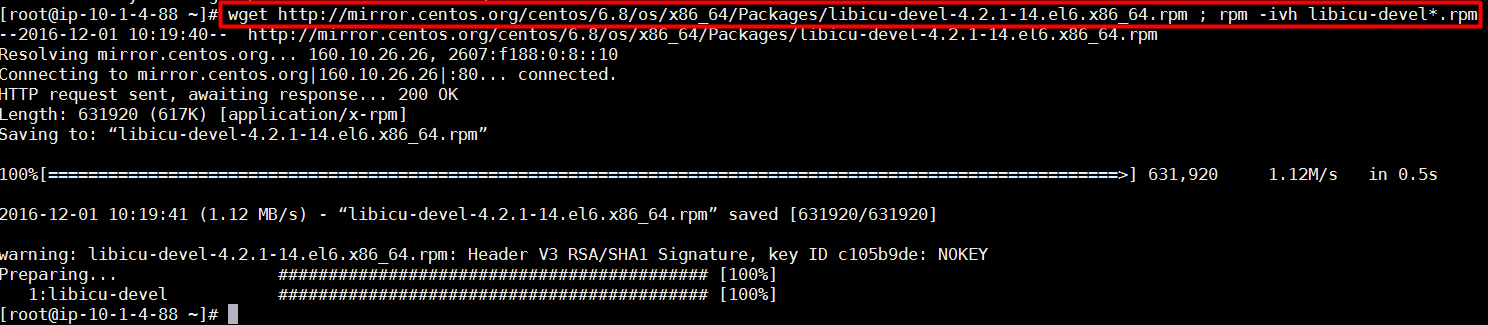
1. Install dependent packages before installing R packages.

|  |
| --- |
| yum -y install libicu texinfo tetex |



1. Install texinfo-tex

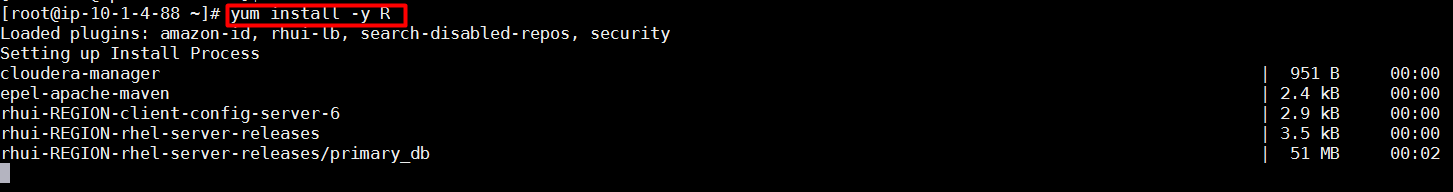
|  |
| --- |
| wget http://mirror.centos.org/centos/6.8/os/x86\_64/Packages/texinfo-tex-4.13a-8.el6.x86\_64.rpm ; rpm -ivh texinfo-tex\*.rpm |

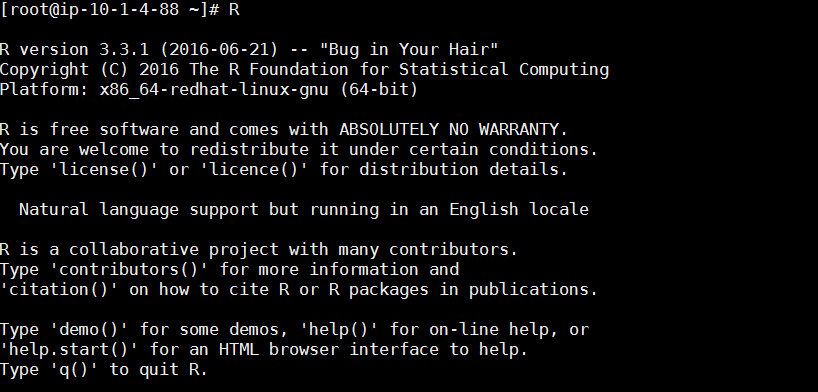
1. 
2. Install libicu-devel
3. 
4. **Repeat steps above on all cluster nodes.**

# Install R binaries and libraries

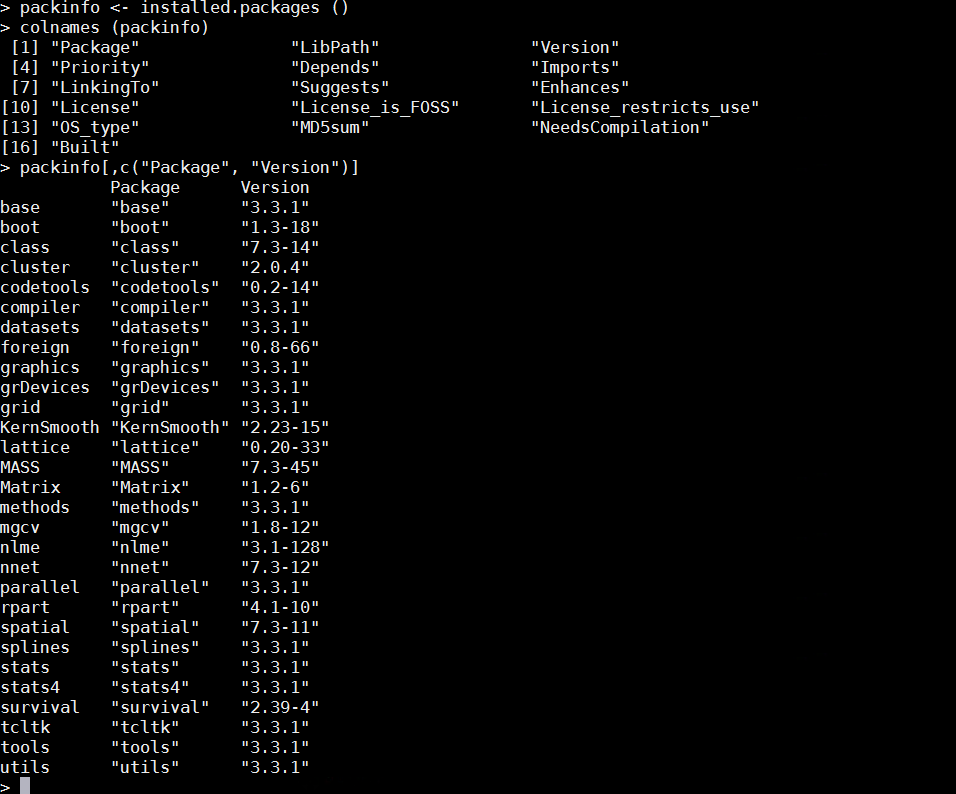
1. Login to Edge/CM node and binaries/libraries.
2. Install R packages using yum

|  |
| --- |
| yum install -y R |



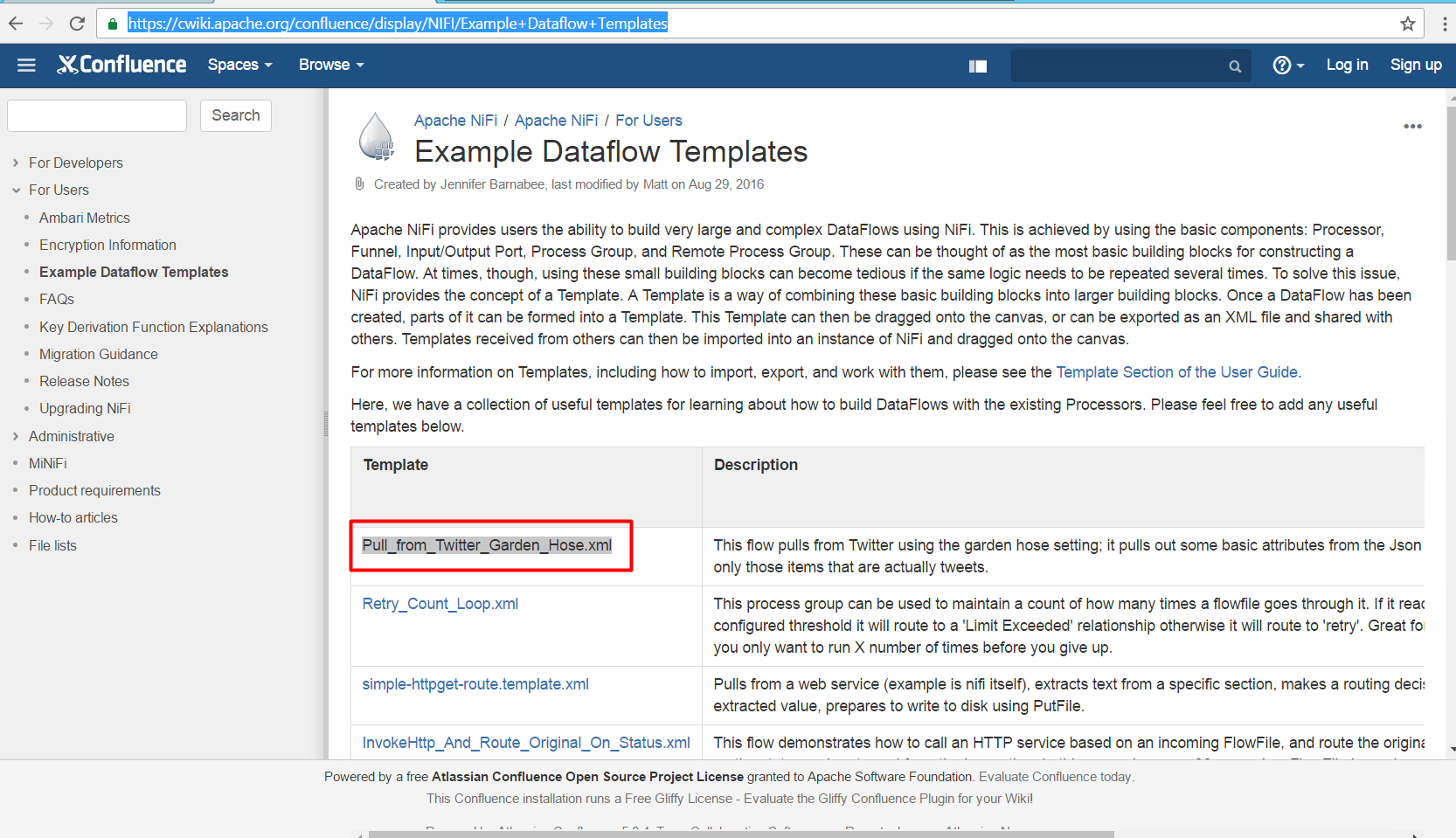
1. Execute “R” command
2. List installed R packages

|  |
| --- |
| packinfo <- installed.packages ()  colnames (packinfo)  packinfo[,c("Package", "Version")] |

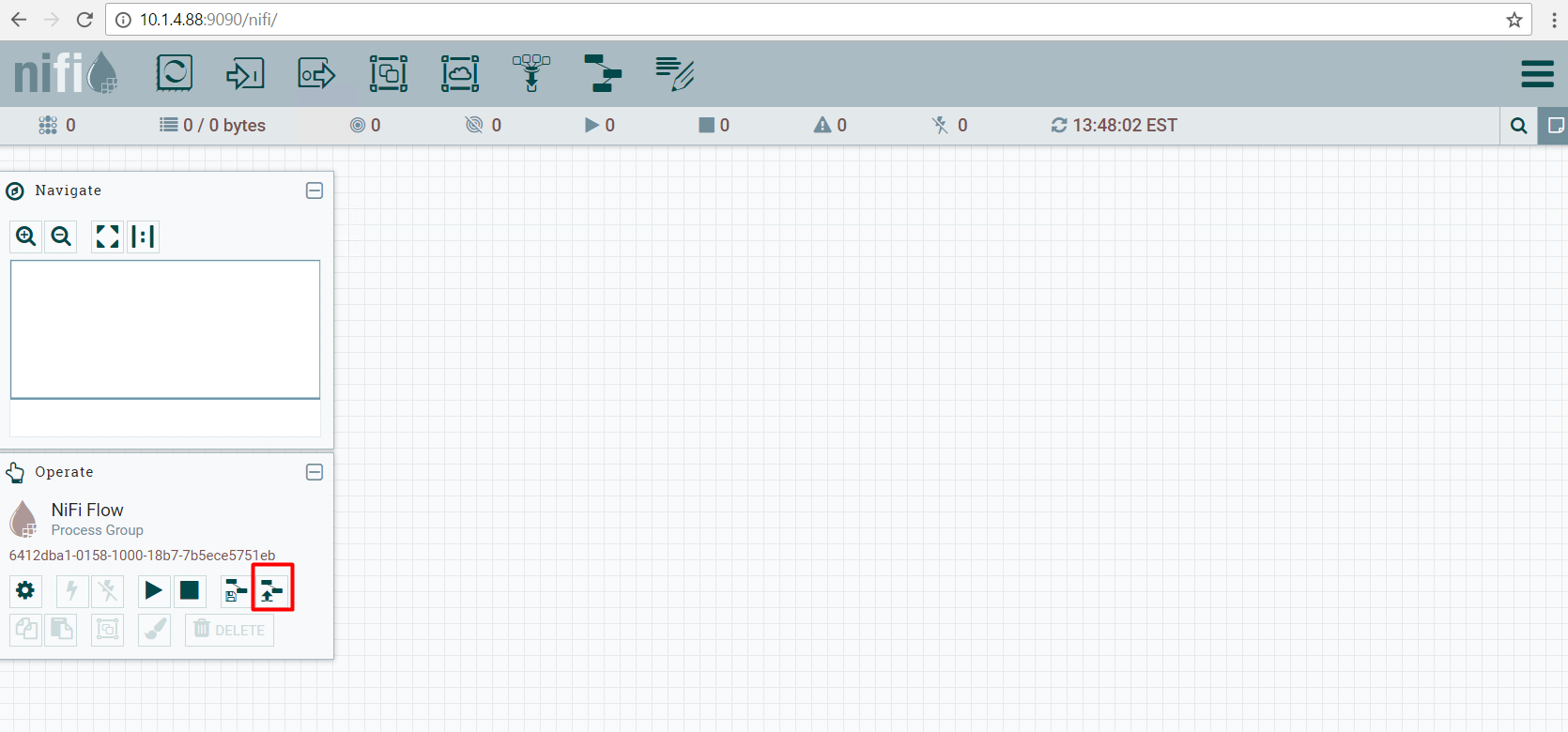


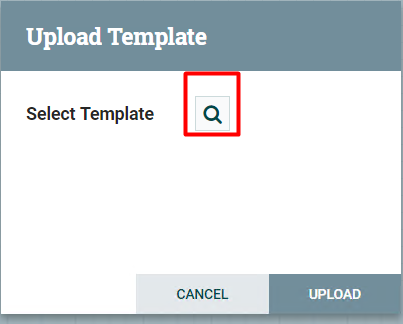
1. **Repeat steps above on all cluster nodes.**

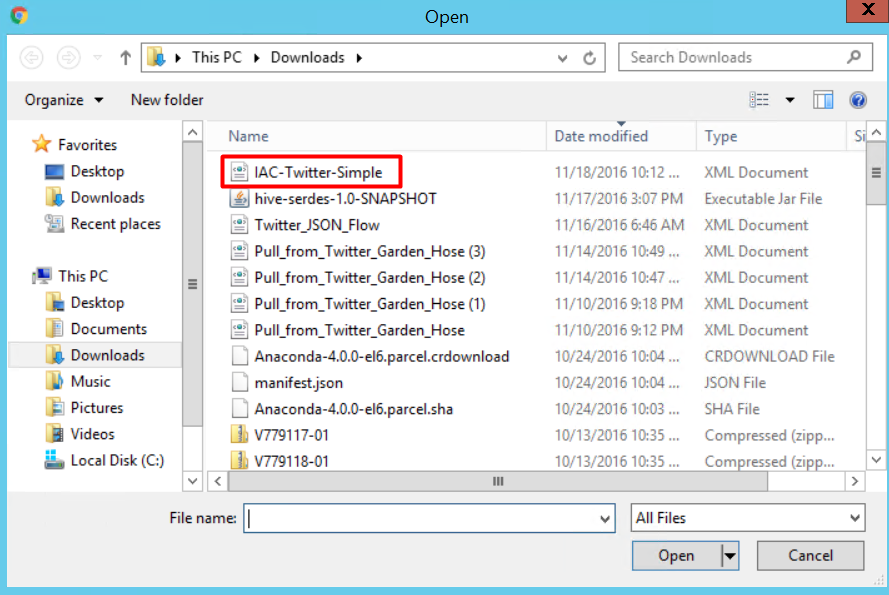
# Install R package

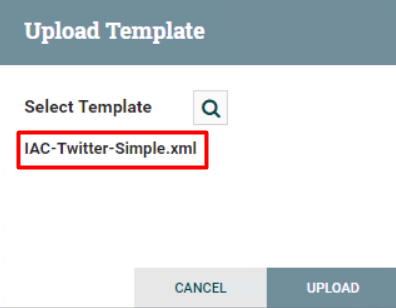


1. Import template into NiFi

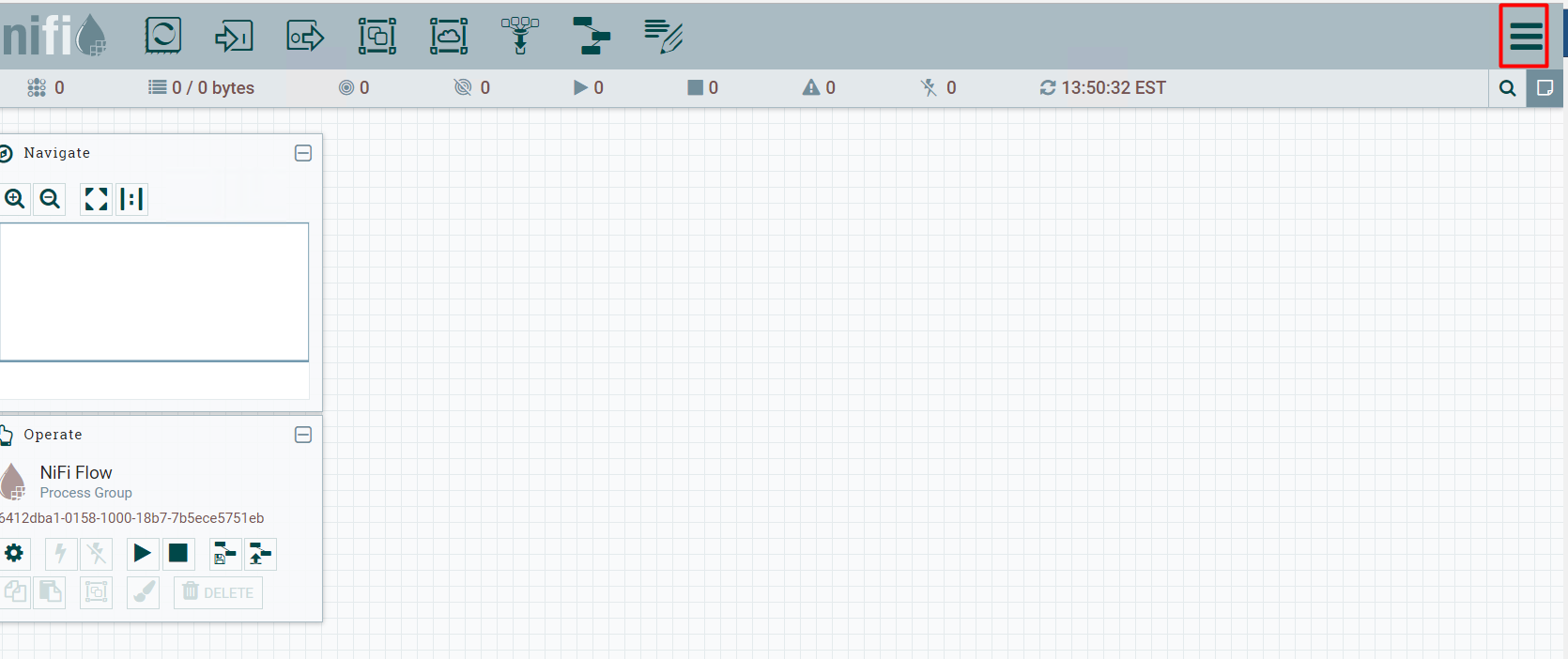


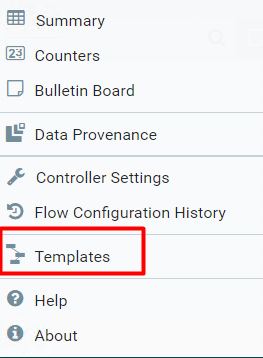




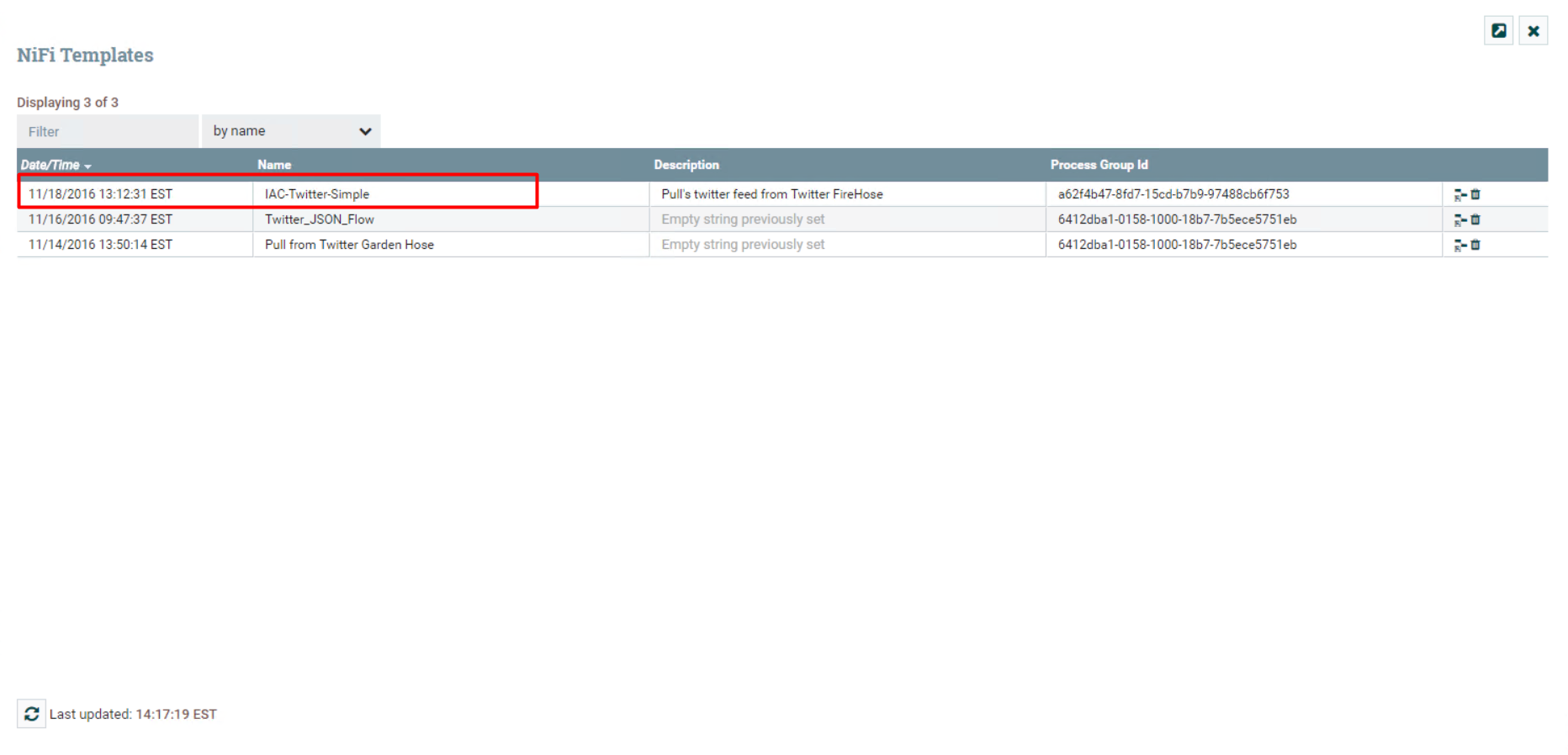


1. Verify template import was successful

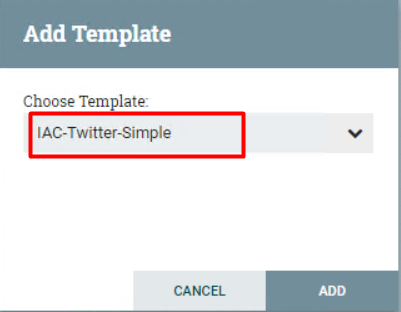


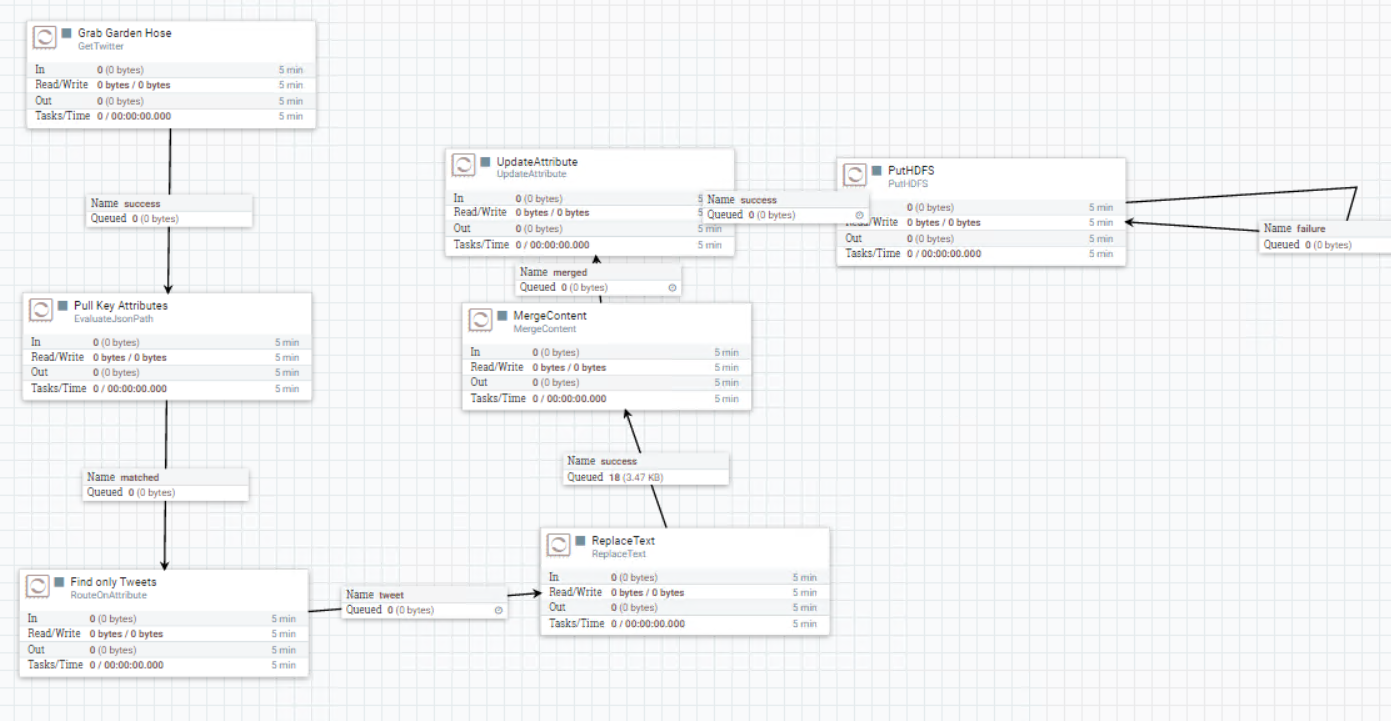


List below will show you all templates that have imported:

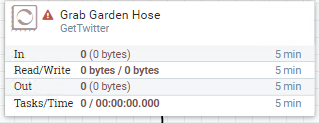


# Nifi Twitter configuration

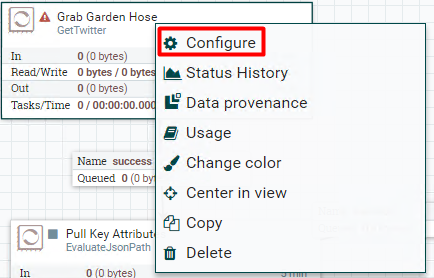
1. Initiate Twitter template by dragging icon  to worksheet



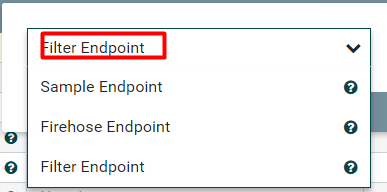
1. Configure Garden Hose with your twitter application
   1. Right click on “Grab Garden Hose”

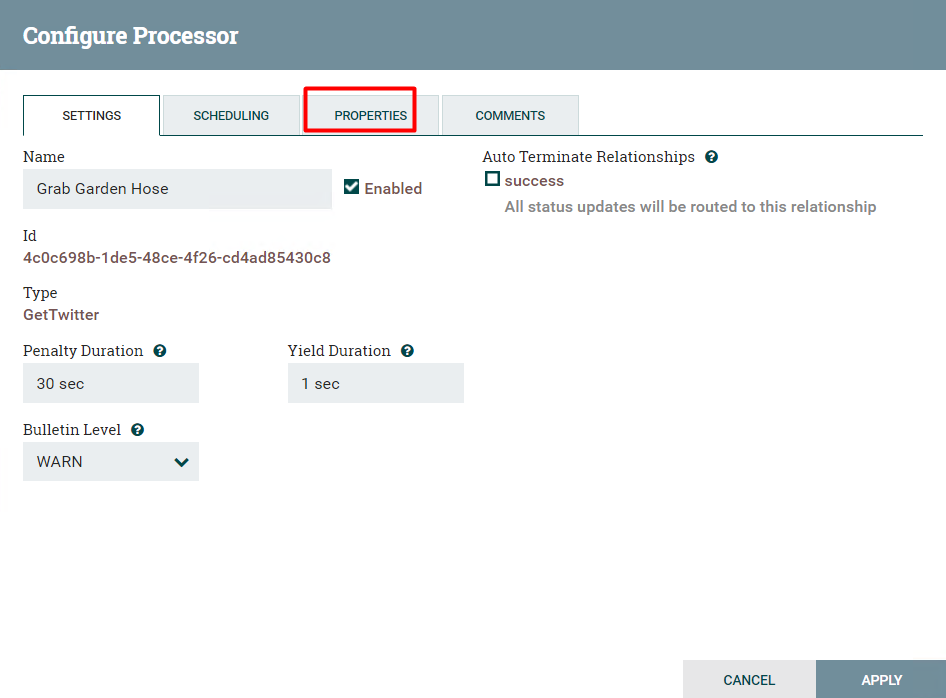


* 1. Click Configure & Properties

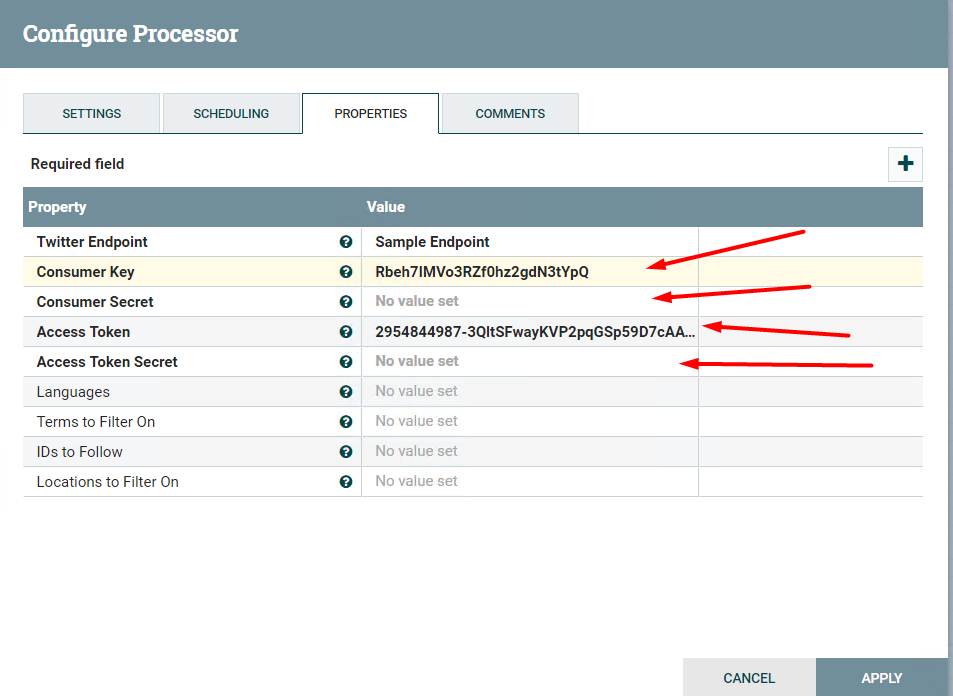


* 1. ensure the 'Twitter Endpoint' is set to 'Filter Endpoint'





* 1. At this point, you will need access to your twitter application API end-points we created in Session 1 – Lab2

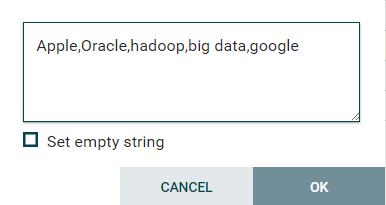


* 1. Update following values from your twitter application:

|  |  |
| --- | --- |
| **Nifi property** | Twitter property |
| **Consumer Key** | Consumer Key (API Key) |
| **Consumer Secret** | Consumer Secret (API Secret) |
| **Access Token** | Access Token |
| **Access Token Secret** | Access Token Secret |

* 1. Under the property “Terms to Filter On” add following or whatever you like to search on

|  |
| --- |
| Apple,Oracle,hadoop,big data,google |



* 1. Click “Apply” for properties to take effect.

# HDFS process and HDFS Sink

1. Before we add new Tweets processer, let’s create the staging directory in HDFS where we will land the RAW dataset in JSON format.
   1. Create staging directory in HDFS as “hfds” user

|  |
| --- |
| hdfs dfs -mkdir /tmp/tweets\_staging |

* 1. Verify creation of directory is successful and change permission to “777”

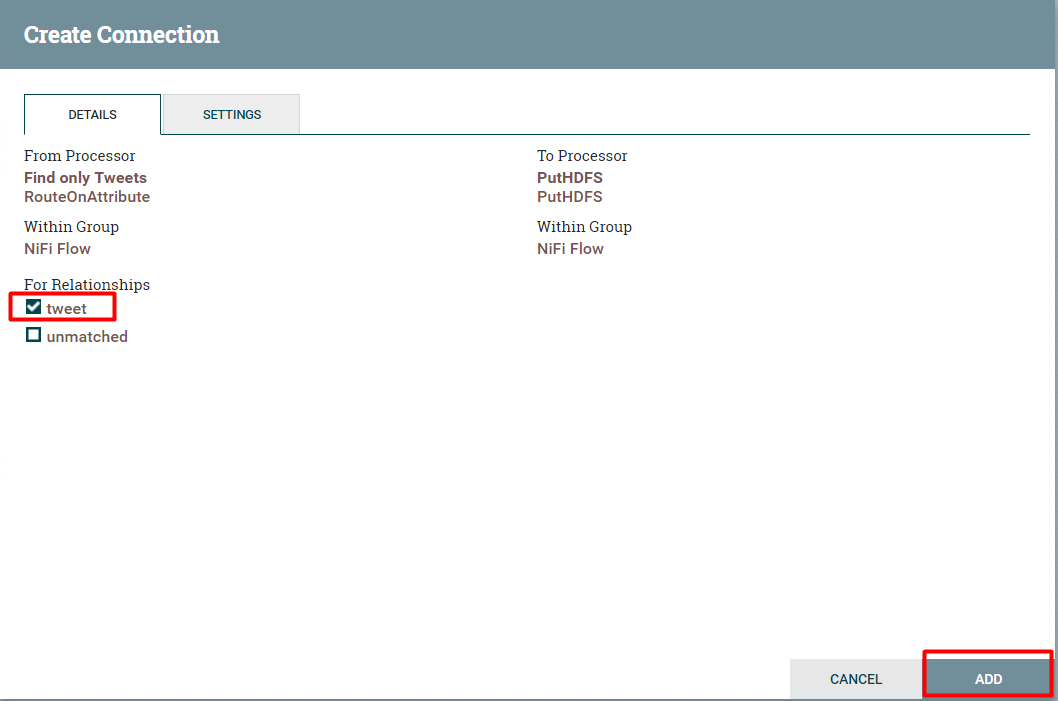
|  |
| --- |
| hdfs dfs -ls /tmp  hdfs dfs -chmod 777 /tmp/tweets\_staging |

**-------------------------------Optional, as processor created part of template --------------------------------**

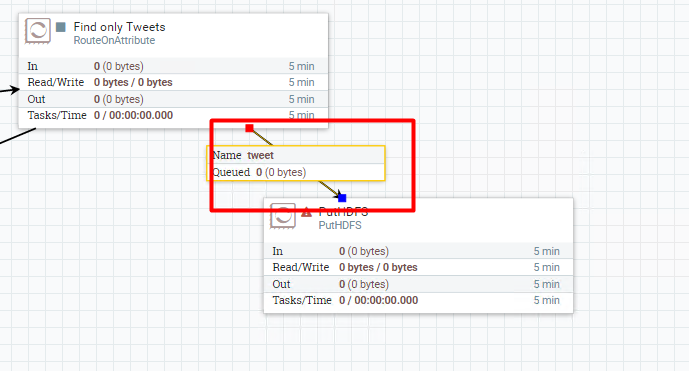
1. We will now create HDFS process which will capture data from twitter stream and write the sink to HDFS
   1. Create new process by dragging  to worksheet
   2. Add Processor window will pop-up, filter on HDFS in search box



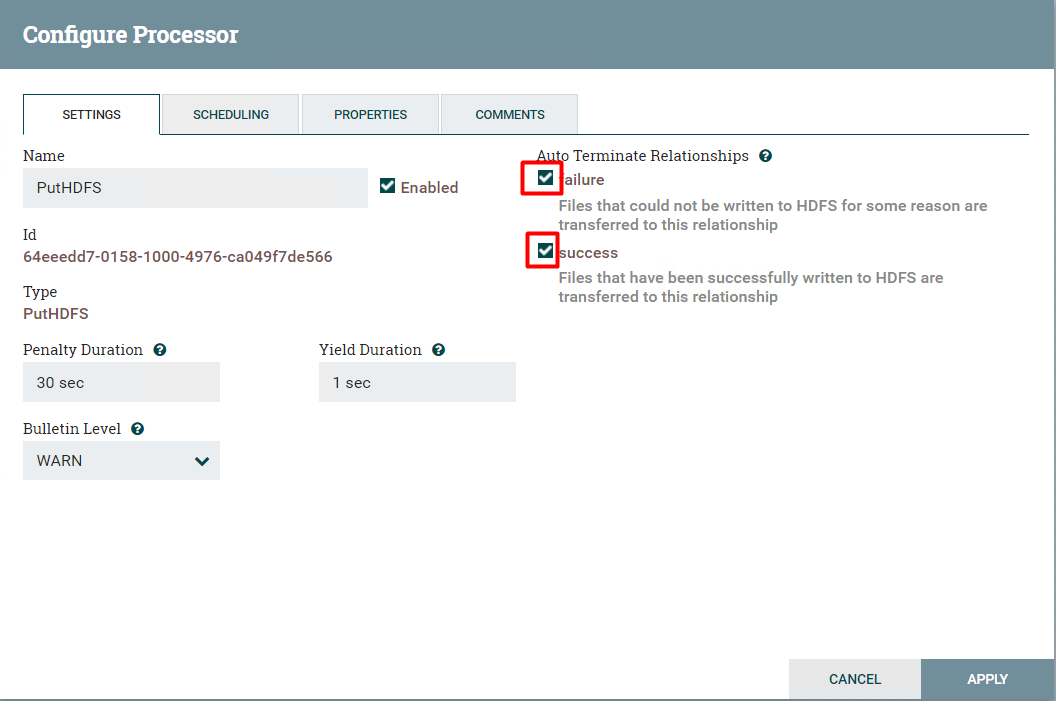
* 1. After adding processor, status of processor should be in  as there is no relationship with rest of workflow.
  2. We need to configure the processor and integrate it with workflow; to do this, click “Find only Tweets” to “PutHDFS” by dragging relationship arrow.



* 1. Check “tweet” box and click add, after this there should be pointer add between both processes.

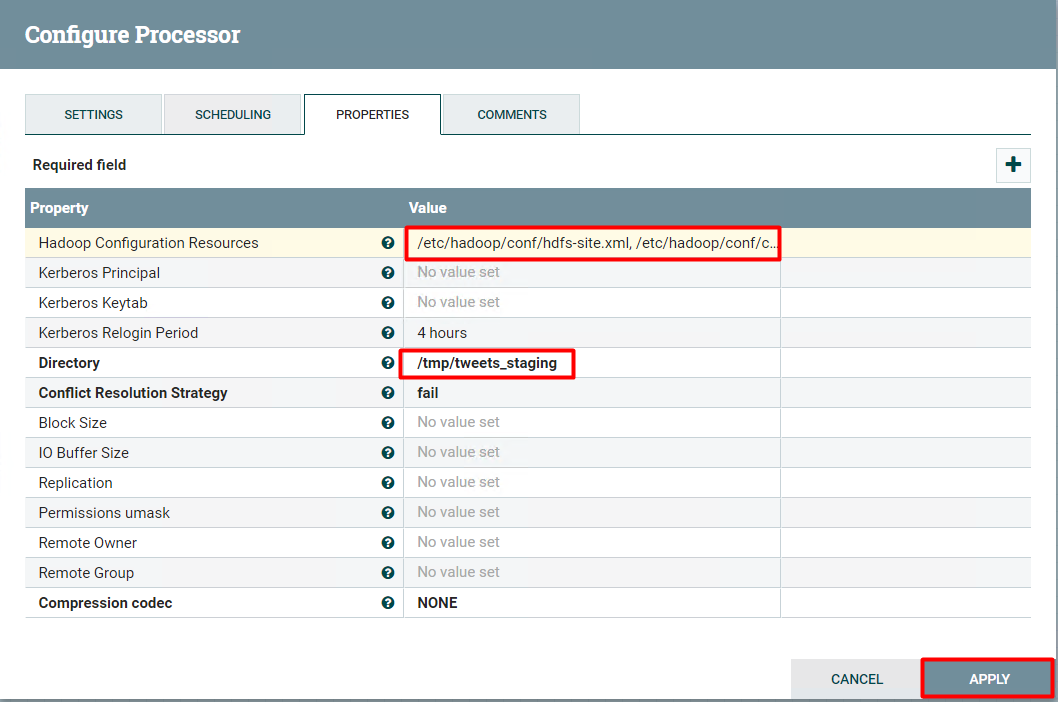


* 1. Right click on “PutHDFS” processor and click configure, you need to choose how the processor will terminate the flow, and since this is the last processor in the flow and it won’t pass any data beyond, tick all the auto-termination boxes.



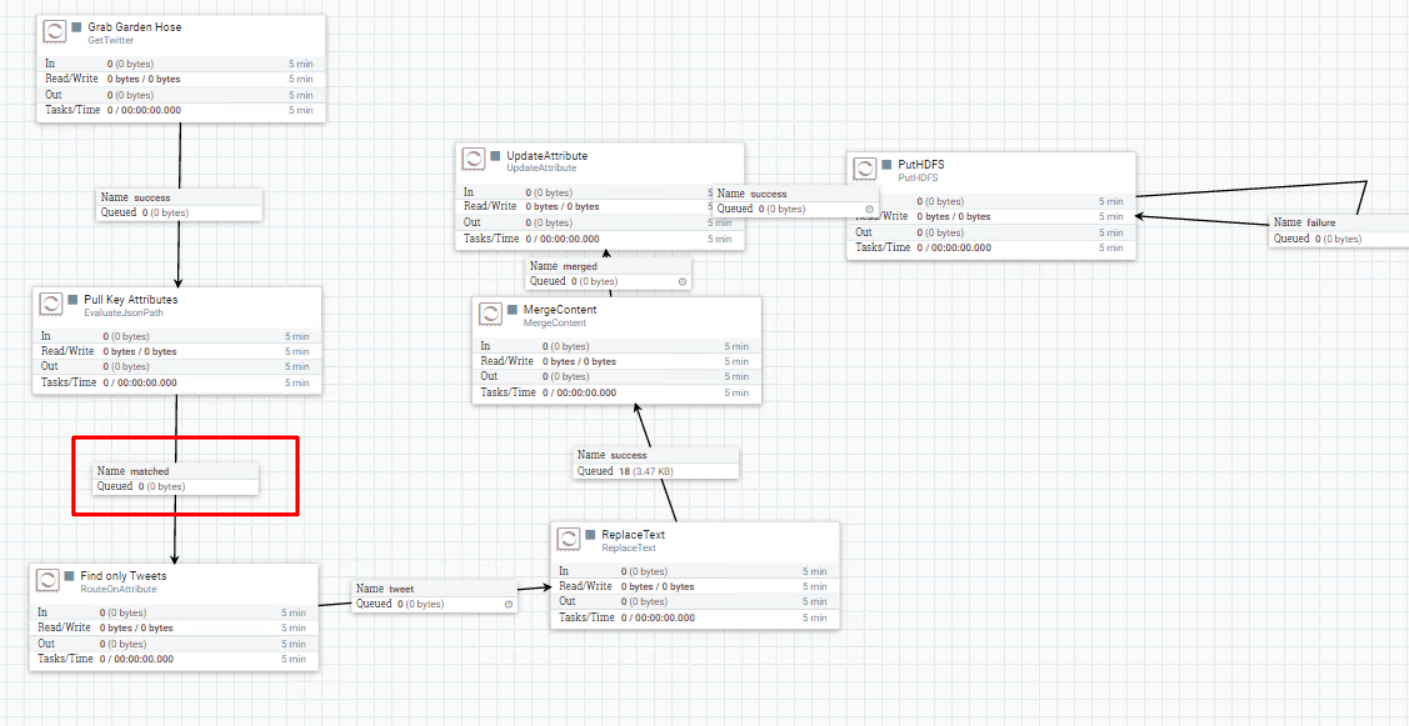
* 1. Click Properties and set the properties from table below:

|  |  |
| --- | --- |
| Property | Value |
| Hadoop Configuration Resources | /etc/hadoop/conf/hdfs-site.xml, /etc/hadoop/conf/core-site.xml |
| Directory | /tmp/tweets\_staging |
|  |  |
|  |  |

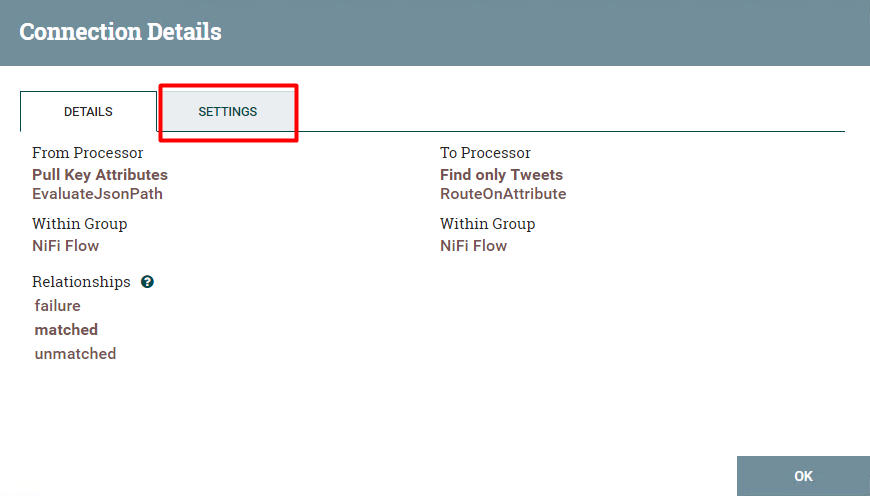


**-------------------------------Optional, as processor created part of template --------------------------------**

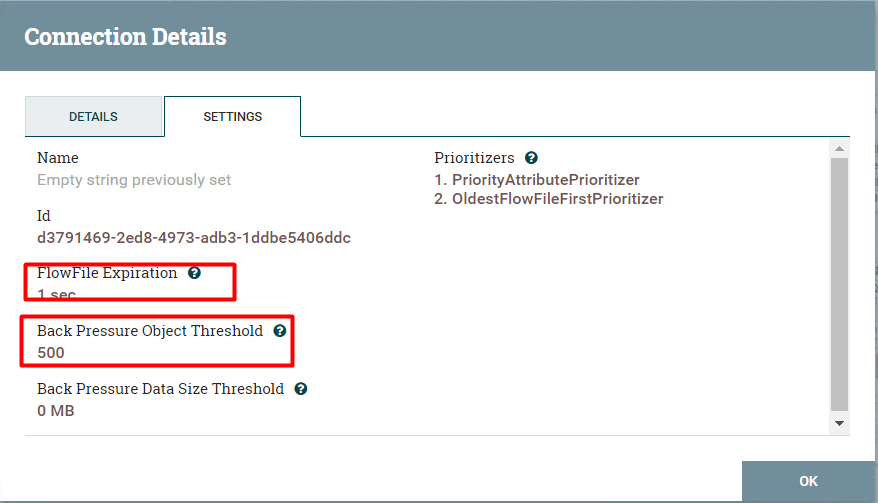
* 1. Set the queue’s FlowFile Expiration and Back Pressure for queue’s “matched” and “tweets”



Right click on “matched” queue, click view configuration and click settings

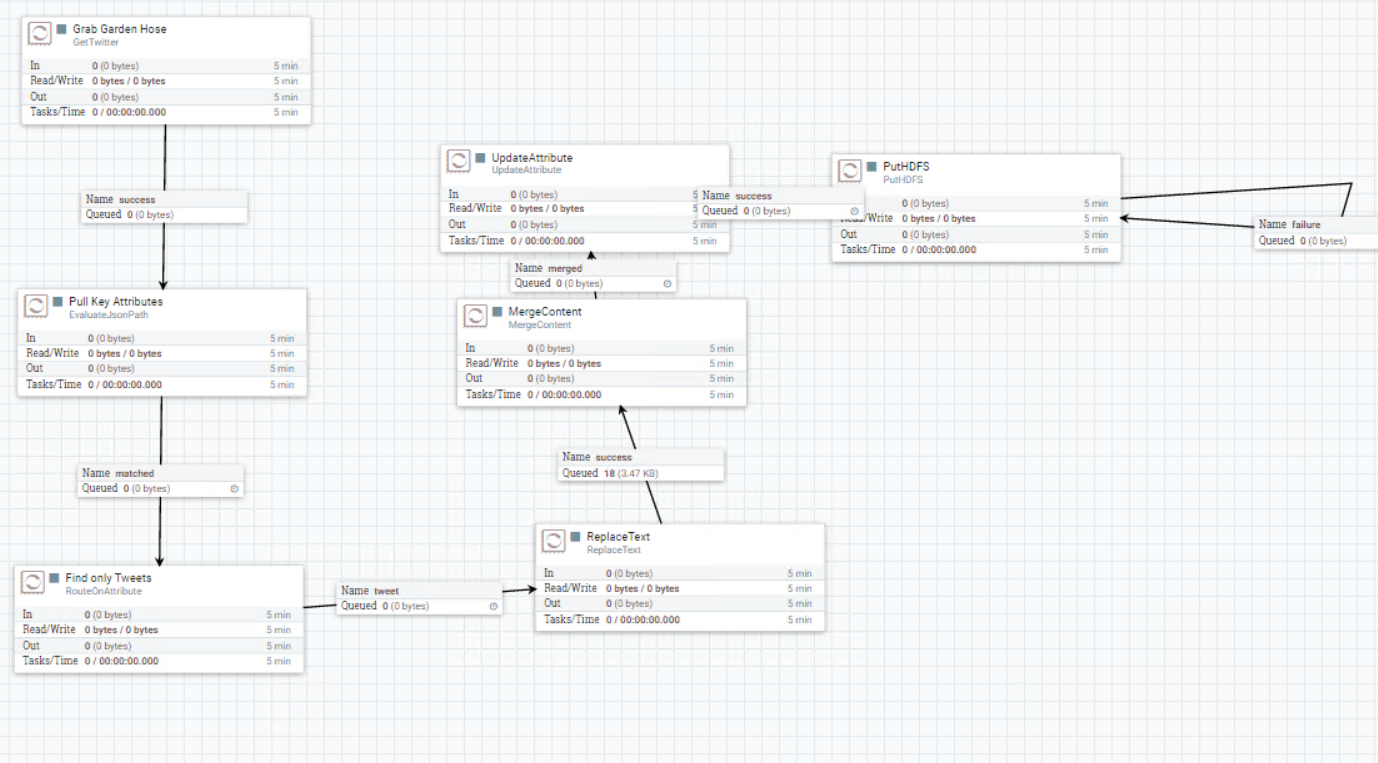


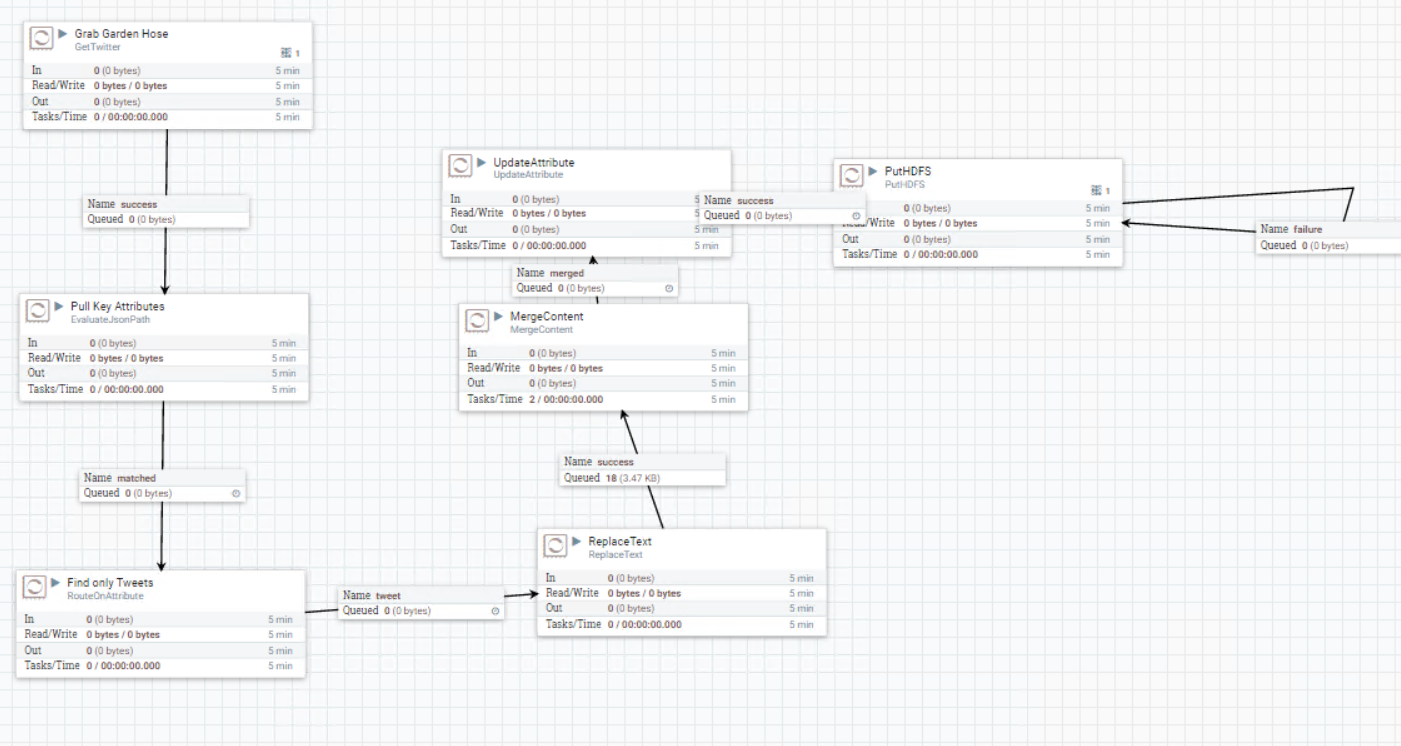
Set FlowFile Expiration to 1 sec and Back Pressure object threshold to 500



Repeat the steps for queue “tweet.”

* 1. If everything goes well, all your process status should turn into “stop” state “”



* 1. Now, let’s start our collection by right clicking each processor and clicking start button: 
  2. In few seconds we will start seeing tweets being collected from “Twitter Firehose” and you can check it by listing the contents of “/tmp/tweets\_staging”

|  |
| --- |
| hdfs dfs -ls /tmp/tweets\_staging |