Image Name: Nifi/HDF2.1 with embedded Zookeepers

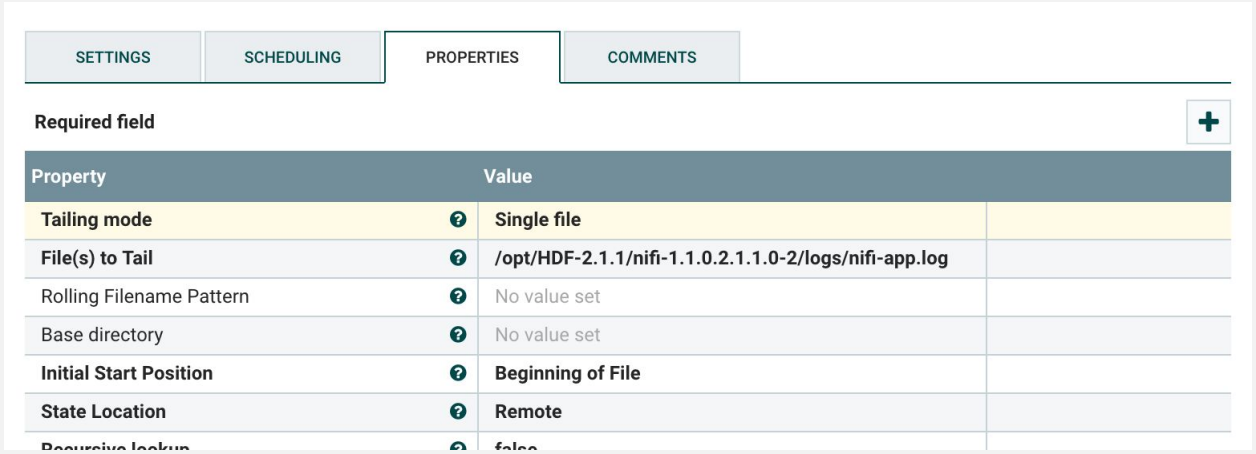
Location	https://s3.amazonaws.com/bluedata-catalog/solutions/bins/bdcatalog-centos-bluedata-hdf21base-3.0.bin
DistroId	bluedata/hdf21base
Version	3.0
Category (cluster Type)	ETLTools
Software Included	java version "1.8.0_40" nifi-1.1.0.2.1.2.0-10-bin.tar.gz zookeeper-3.4.6.tar.gz
Notebook access	Nifi UI (open to everyone, once deployed)
Systemv Service names and commands	sudo service HDF-master status (stop, start) sudo service HDF-slave status(stop, start) sudo service Zookeeper-service status (start, stop)
OS	Centos6. Works on both Centos and RHEL base machines

Sample workflow for testing:

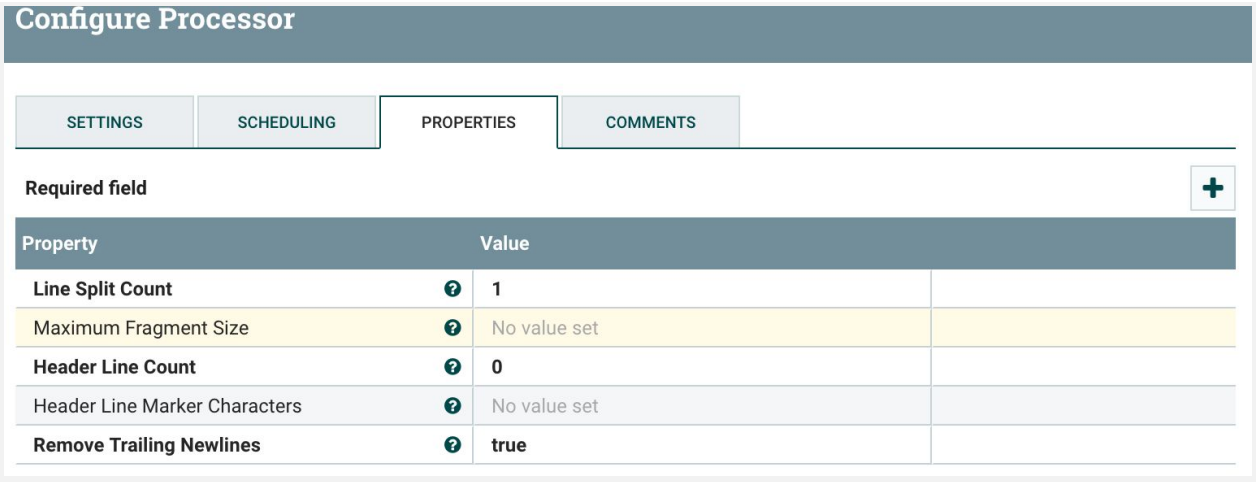
Building a Flow in NiFi to fetch and parse nifi-app.log

1) Let us build a small flow on NiFi canvas to read app log generated by NiFi itself to feed to Spark:

2) Drop a "TailFile" Processor to canvas to read lines added to "/opt/HDF-2.1.1/nifi-1.1.0.2.1.1.0-2/logs/nifi-app.log". Auto Terminate relationship Failure.

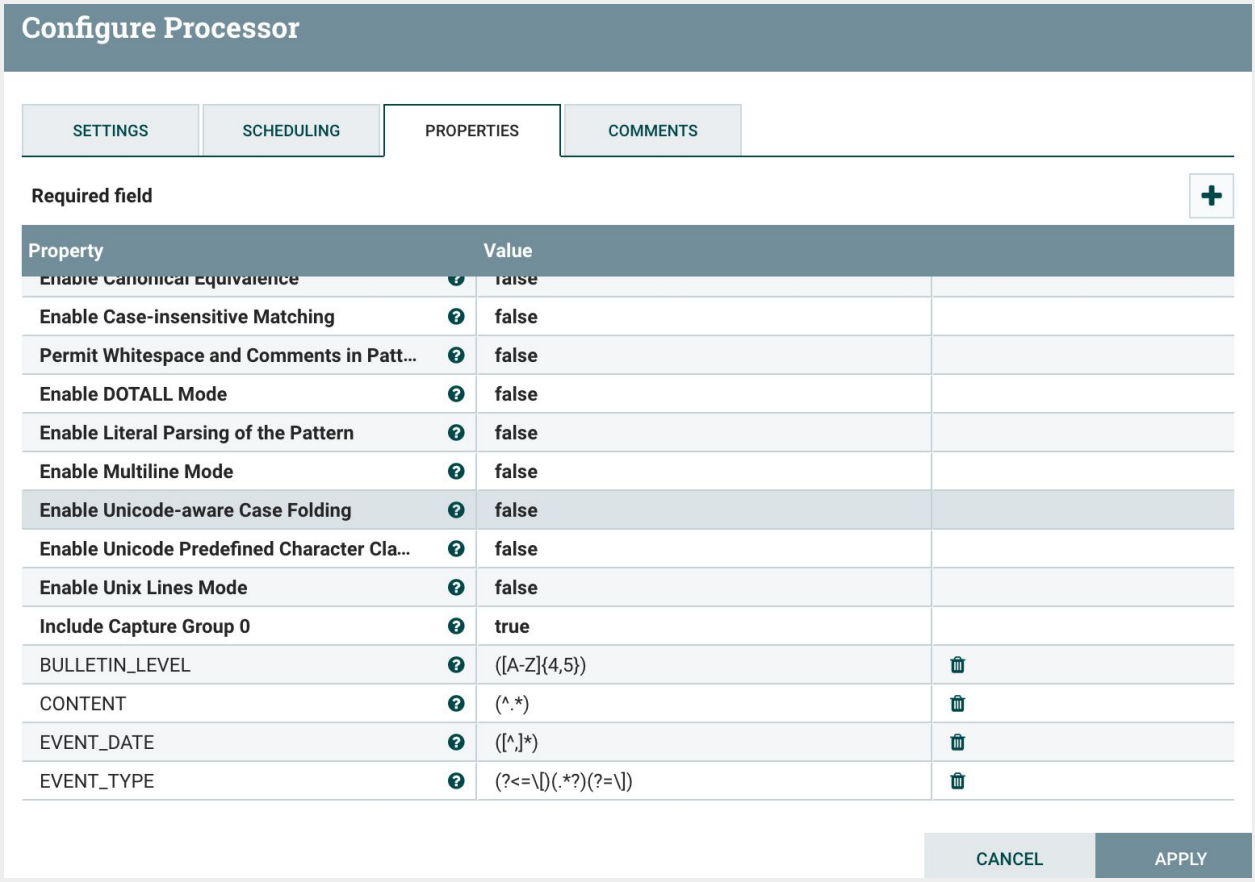


3) Drop a "SplitText" Processor to canvas to split the log file into separate lines. Auto terminate Original and Failure Relationship for now. Connect TailFile processor to SplitText Processor for Success Relationship.

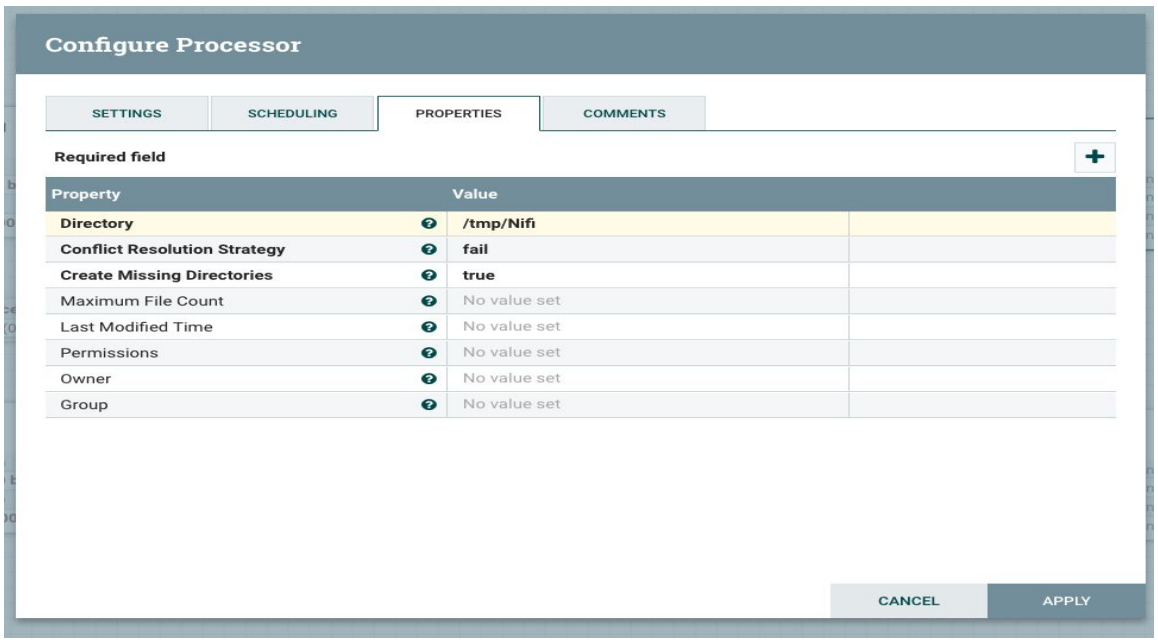


4) Drop a "ExtractText" Processor to canvas to extract portions of the log content to attributes as below. Connect SplitText processor to ExtractText Processor for splits relationship.

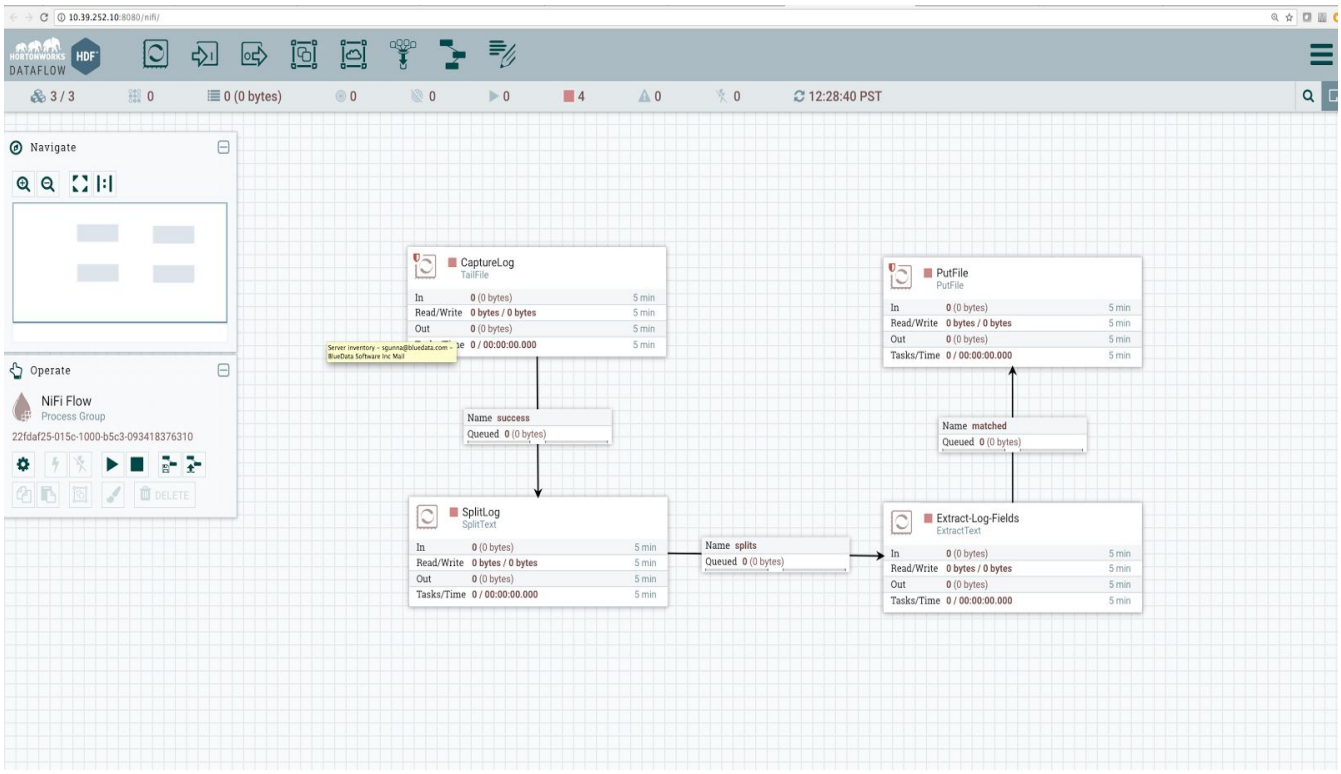
- BULLETIN_LEVEL:([A-Z]{4,5})
- CONTENT:(^.*)
- EVENT_DATE:([^,]*)
- EVENT_TYPE:(?<=\[)(.*?)(?=\])



5) Now create a PutFile Processor to store end result in local.



6) The final workflow looks like the image below.



The output for this workflow is generated under /tmp/Nifi location: