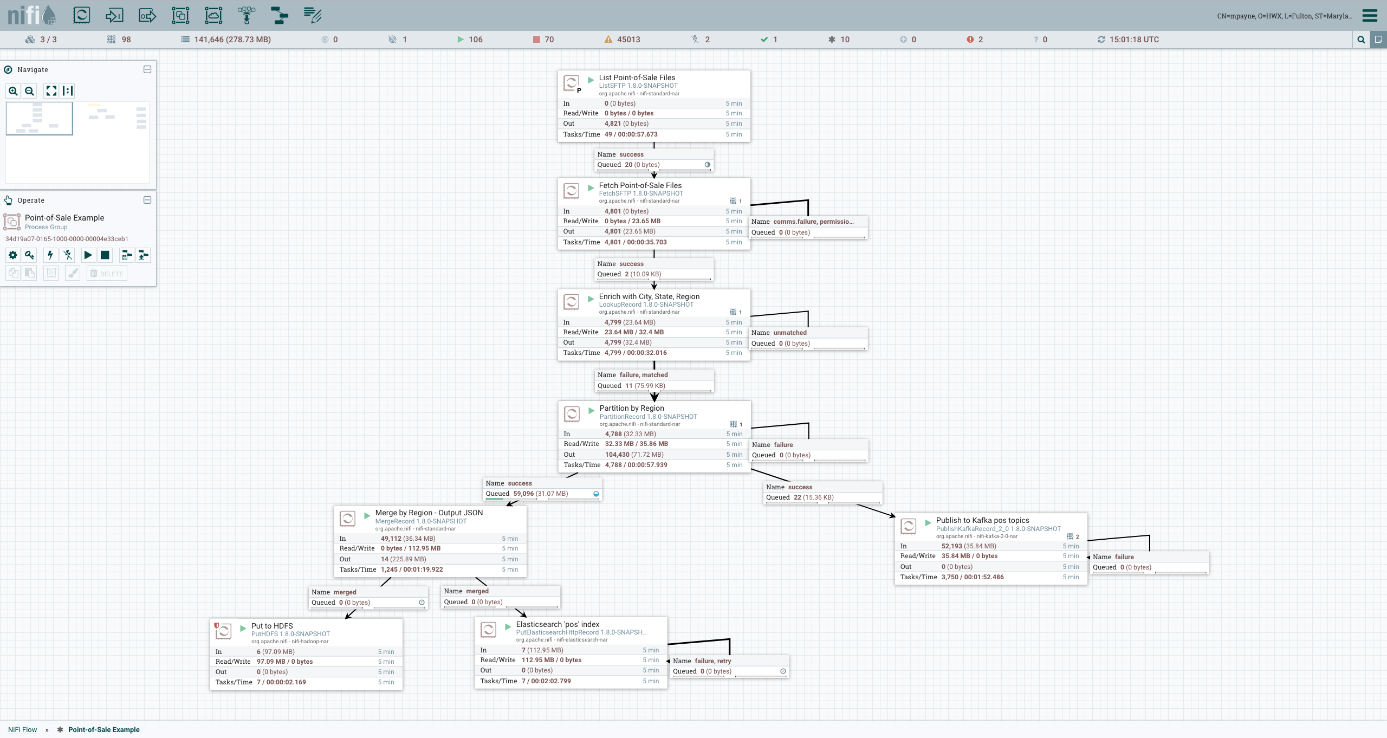
Load Balancing Use Case:

We need to get data from a remote SFTP server. The data is Point-of-Sale data in CSV format and is compressed using GZIP. One of the fields in the CSV data is the Store Identifier field, "storeId." We need to use this Store Identifier to perform some enrichment, adding in the City, State, and Region where the store is located. Once we have enriched the data, we want to send the data to three destinations: HDFS for archiving, Kafka for our real-time analytics to pull from, and Elasticsearch for querying and dashboarding. We want to push to Kafka in Avro format as quickly as possible. For HDFS, we want to merge together larger batches and store as JSON. For Elasticsearch, we can also benefit from having data bundled together because their Bulk API is more efficient than sending individual messages. We can build this flow fairly simply. Our entire flow now looks like this:

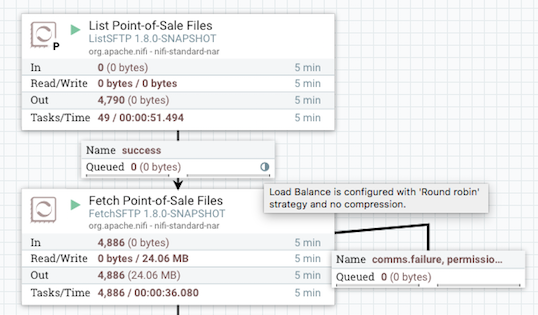
[](https://blogs.apache.org/nifi/mediaresource/8c89ac27-e75b-47c3-94ca-356ba7385b03)

The first connection, from ListSFTP to FetchSFTP uses a Load Balance Strategy of "Round Robin." We simply want to spread the data across the cluster so that we can have all nodes in the cluster working together to pull the data, decompress it, enrich it, and split it apart based on Region. The connection before MergeRecord is configured to use a Load Balance Strategy of "Partition by Attribute" and uses the "region" attribute that was added by the PartitionRecord Processor.

Now, notice the new icon on a couple of the connections:



This lets us know that the Connection is currently configured to Load Balance data across the cluster. If we hover over it, we get a little bit more information:



Now, when we start our flow, we can see the data quickly flowing. We can also see that this icon has now changed from a grey color to a blue color and the orientation is different:



This tells us that not only is Load Balancing configured for this connection but that data is currently being transferred between nodes in the cluster. This is very important information if, for example, we notice that the destination of our connection is not processing the data. If the indicator is blue, then we know that the reason the data is not being processed is probably because the data does not yet live on the correct node and is being transferred there now.

It's also important to note that if the Load Balance Strategy is changed while data is queued up in the connection, then the connection will immediately start re-balancing the data to ensure that the data goes to the correct node. For example, if "Do Not Load Balance" is configured, and then it is changed to "Round Robin," it will immediately start binning the data to go to the different nodes in the cluster, and you'll see the indicator in blue. Similarly, if the Load Balance Strategy is changed from "Round Robin" to "Partition by Attribute," the data will be partitioned and binned differently, to ensure that data ends up on the correct node. If the number of nodes in our cluster changes (i.e., a node is removed from the cluster or added to the cluster), then the data will also be re-partitioned.

Now, let's consider what happens in a failure scenario. Specifically, what happens if a node disconnects from the cluster, or if a Node A simply cannot communicate with Node B. In this case, what happens to the data that is queued up waiting to go to Node B? Will the data just queue up, or will it fail over to another node?

Well, that depends on the Load Balance Strategy configured. If "Round Robin" is used, and a failure occurs, the data will be rebalanced to another node. In this case, the data is rebalanced fairly slowly, though. Up to 1,000 FlowFiles, or 10 MB of data, will be rebalanced per second. This is done so that if the node reconnects to the cluster or is able to communicate again momentarily, that the data is not immediately redistributed to other nodes. But if that does not happen, the data is still redistributed throughout the cluster in a timely manner.

If the "Partition by Attribute" or "Single Node" strategy is used, then the data will just queue up, waiting to send, until the node is reconnected and able to communicate. This is because these strategies expect that a given piece of data go to a specific node, whereas "Round Robin" just expects that data gets spread across the cluster.

While this is a very powerful feature that will significantly improve the user experience offered, this feature also lays the groundwork for some other very exciting features. The newly added "Offload" feature makes use of this mechanism to allow a node in the cluster to be disconnected and then Offloaded. Doing this results in stopping any and all Processors on that node and distributing all data across the cluster to the still-connected nodes. This, in turn, enables easy decommissioning of a NiFi node and truly elastic clustering capabilities.