Falcon ADF Provider Tutorial

1. Install Falcon

This tutorial utilizes Hortonworks Sandbox version 2.3.2. If you do not have a copy of the Sandbox installed, you can download it and get instructions for installing it from: http://hortonworks.com/products/hortonworks-sandbox/#install. For this tutorial and E2E demo, we provide Falcon binary with ADF Provider. This version will be in our future HDP release, so our customers don't need to worry about the installation.

If there is a previously installed Falcon and configured on your Sandbox, stop Falcon by issuing the following commands.

In the Sandbox, change to falcon user:

```
su - falcon
```

Change to the Falcon directory on Sandbox:

```
cd /usr/hdp/2.3*/falcon
```

Stop Falcon:

```
bin/falcon-stop
```

Step 1. Download installation files from OneDrive: *http://1drv.ms/1N7F2aL* and save to Sandbox */tmp/adf*. To copy local files to Sandbox, Mac users can use command *scp*; Windows users can use *WinSCP*: *https://www.siteground.com/tutorials/ssh/ssh_winscp.htm*

Step 2. SSH to Sandbox, e.g. *ssh -p 2222 root@127.0.0.1.* Default password is *hadoop*. You will be asked to change password the first time you ssh to sandbox.

Step 3. Copy scripts to HDP folder and run script *config_adf.sh* to install Falcon with ADF Provider:
cd /usr/hdp/2.3/*
cp /tmp/adf/.sh ./*
./config_adf.sh

Step 4. Set up port 15000 for Falcon homepage (e.g. http://127.0.0.1:15000/):
Add **.falcon.enableTLS=false* to falcon-0.7-SNAPSHOT/conf/startup.properties.

Step 5. Set up conf/client.properties to be on port 15000
falcon.url=http://localhost:15000/

Note: If you already have Falcon installed, please make sure you stop Falcon and delete the Falcon folder *falcon-0.7-SNAPSHOT* beforehand: *rm -rf falcon-0.7-SNAPSHOT*

2. Set up Azure Service Bus credentials

Falcon reads Azure Service Bus credentials from conf/startup.properties when it starts. So, the credential needs to be added before starting Falcon, and Falcon needs to be restarted if there is any change in the credential. Support for Azure Service Bus configurations through Ambari will be provided in a future release of HDP.

Use the namespace you defined when you setup the Azure Service Bus in the Wizard.

Example:

########## ADF Configurations start ##########

 # A String object that represents the namespace
**.microsoft.windowsazure.services.servicebus.namespace=<YourServiceBusName>*

 # Request and status queues on the namespace
**.microsoft.windowsazure.services.servicebus.requestqueuename=adfrequest*
**.microsoft.windowsazure.services.servicebus.statusqueuename=adfstatus*

 # A String object that contains the SAS key name
**.microsoft.windowsazure.services.servicebus.sasKeyName=<YourSASKey>*
 # A String object that contains the SAS key

**.microsoft.windowsazure.services.servicebus.sasKey=<YourSASKey>*

 # A String object containing the base URI that is added to your Service Bus namespace to form the URI to connect
 # to the Service Bus service. To access the default public Azure service, pass ".servicebus.windows.net"
**.microsoft.windowsazure.services.servicebus.serviceBusRootUri=.servicebus.windows.net*

 # Service bus polling frequency (in seconds)
**.microsoft.windowsazure.services.servicebus.polling.frequency=60*

3. Falcon Configuration on Oozie-site through Ambari

On HDP 2.3.2, Falcon is not by default enabled with Oozie. Oozie must be configured to manually to work with Falcon in this release. A future release of HDP will include configuration of Falcon with Oozie.

Go to Oozie page on Ambari. Make sure the properties in the file *custom-oozie-site.txt* on *OneDrive* are added to custom oozie-site (see Figure 1).

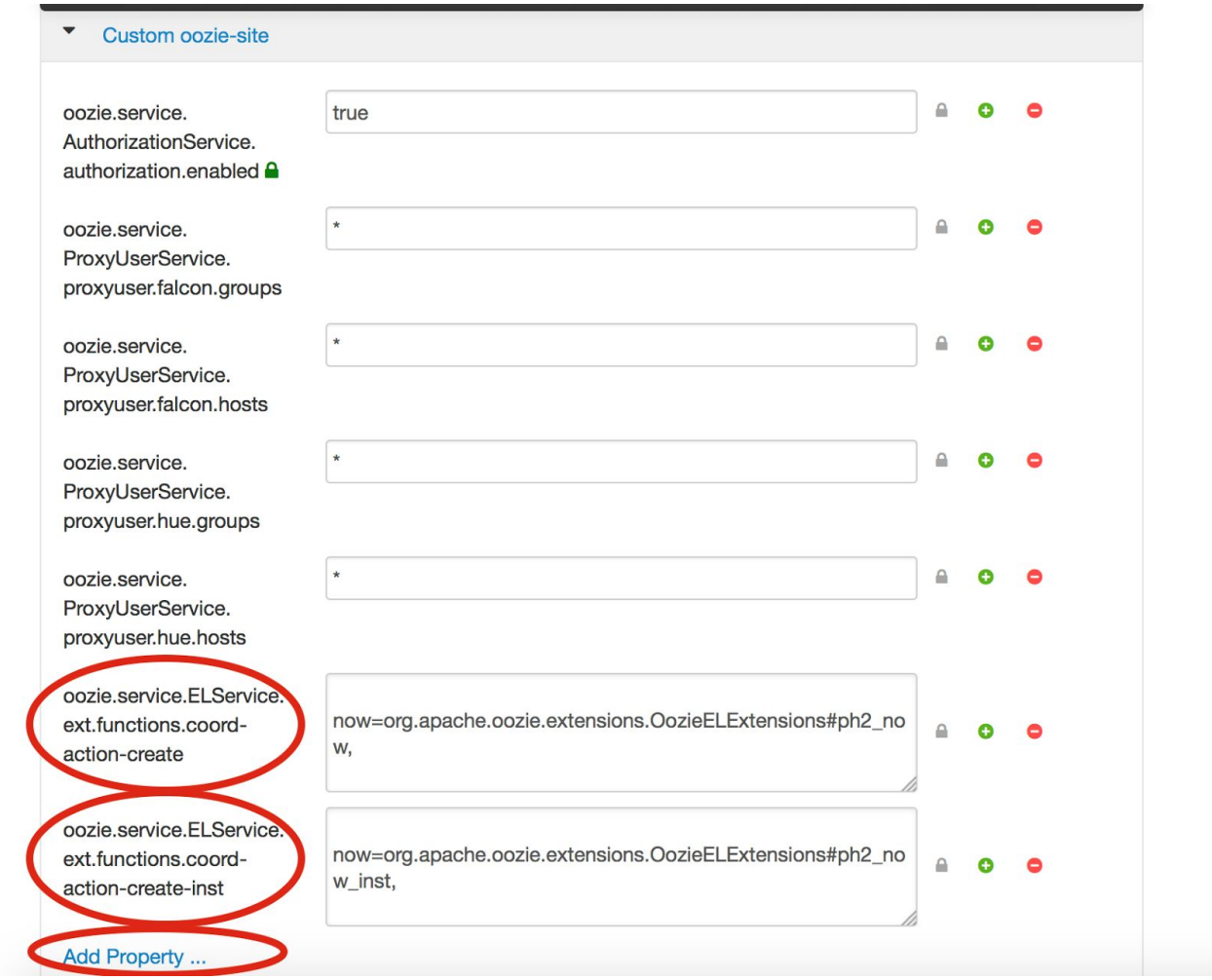
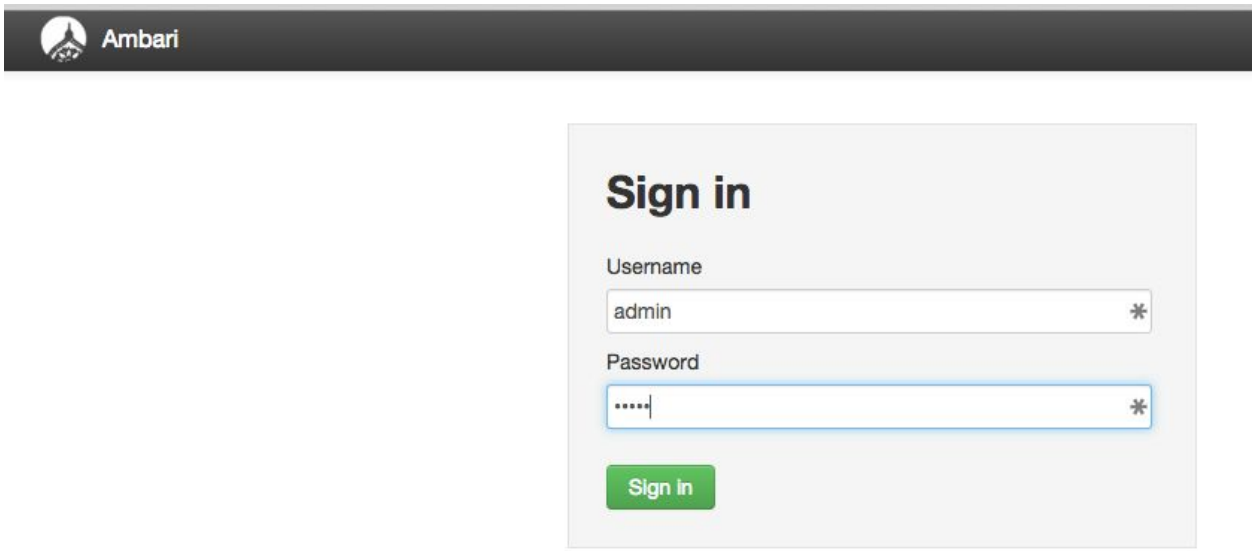


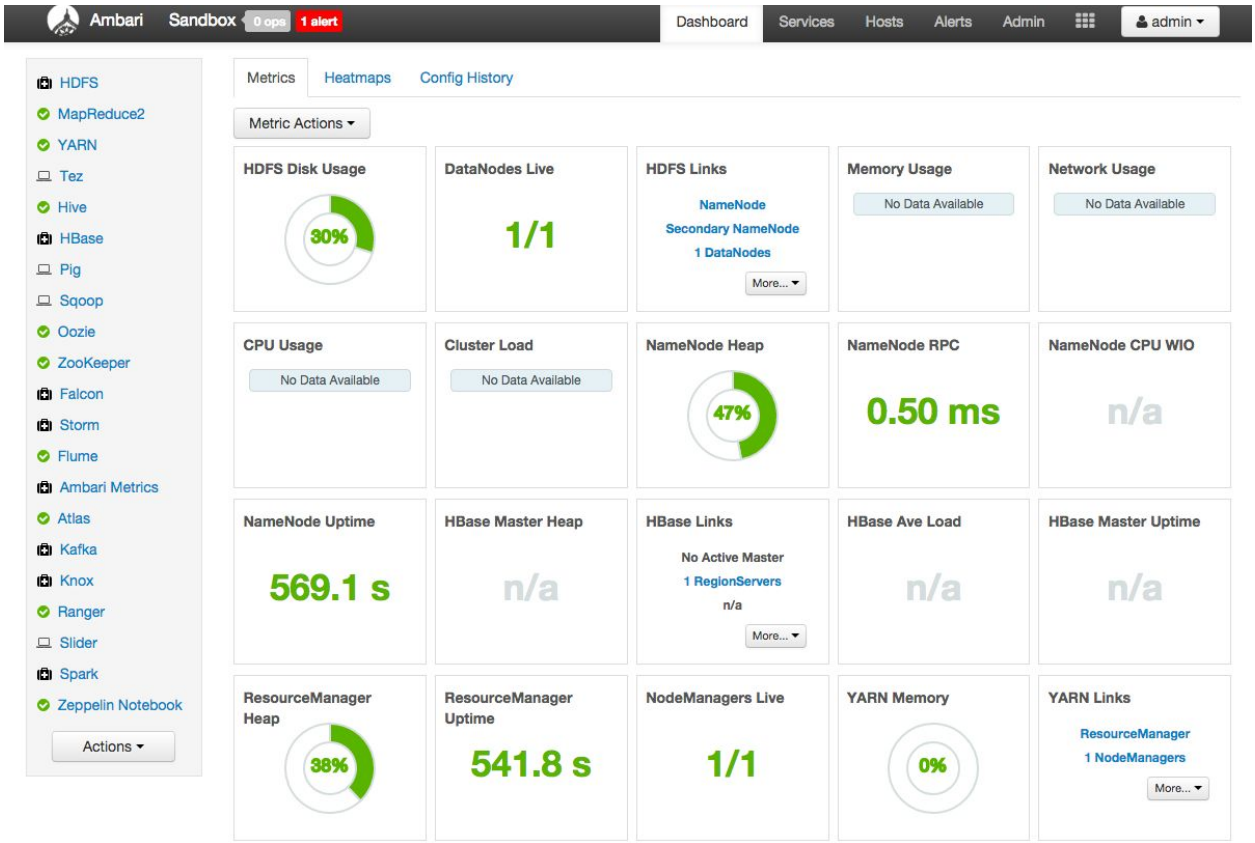
Figure 1. Add properties to custom oozie-site through Ambari

4. Check Status of HDP Components

Make sure HDP components needed in this tutorial (i.e. HDFS, Yarn, MapReduce2, Oozie, Hive, Pig) are running well. You can check their running status on Ambari. From your browser, log into Ambari with the user ID / password admin / admin and check the status of HDFS, Yarn, MapReduce2, Oozie, Hive, and Pig on the main metrics dashboard.

http://<YourSandboxURI>:8080


Ambari
Sign in
Username
admin
Password
Sign in


Ambari
Sandbox
0 ops
1 alert
Dashboard
Services
Hosts
Alerts
Admin
admin
HDFS
MapReduce2
YARN
Tez
Hive
HBase
Pig
Sqoop
Oozie
ZooKeeper
Falcon
Storm
Flume
Ambari Metrics
Atlas
Kafka
Knox
Ranger
Slider
Spark
Zeppelin Notebook
Actions
Metrics
Heatmaps
Config History
Metric Actions
HDFS Disk Usage
30%
DataNodes Live
1/1
HDFS Links
NameNode
Secondary NameNode
1 DataNodes
More...
Memory Usage
No Data Available
Network Usage
No Data Available
CPU Usage
No Data Available
Cluster Load
No Data Available
NameNode Heap
47%
NameNode RPC
0.50 ms
NameNode CPU WIO
n/a
NameNode Uptime
569.1 s
HBase Master Heap
n/a
HBase Links
No Active Master
1 RegionServers
n/a
More...
HBase Ave Load
n/a
HBase Master Uptime
n/a
ResourceManager Heap
38%
ResourceManager Uptime
541.8 s
NodeManagers Live
1/1
YARN Memory
0%
YARN Links
ResourceManager
1 NodeManagers
More...

5. Start Falcon

Start the Falcon server either by logging into the Sandbox and executing the following commands:

change to falcon user:

```
su - falcon
```

`Change to` Falcon directory on Sandbox:

```
cd /usr/hdp/2.3*/falcon-0.7-SNAPSHOT
```

To start Falcon:

```
bin/falcon-start
```

If Falcon fails to start, the Falcon logs can be found at *logs/falcon.application.log*

Cluster/Feed/Process: Submit/Search/Schedule

You can submit entity specification file through Falcon homepage (see Figure 2). You can also search entities by keywords and subsequence of the name (see Figure 3) and schedule entities to run (see Figure 4). You can refer to *http://falcon.apache.org/EntitySpecification.html* for more details of entity specification.

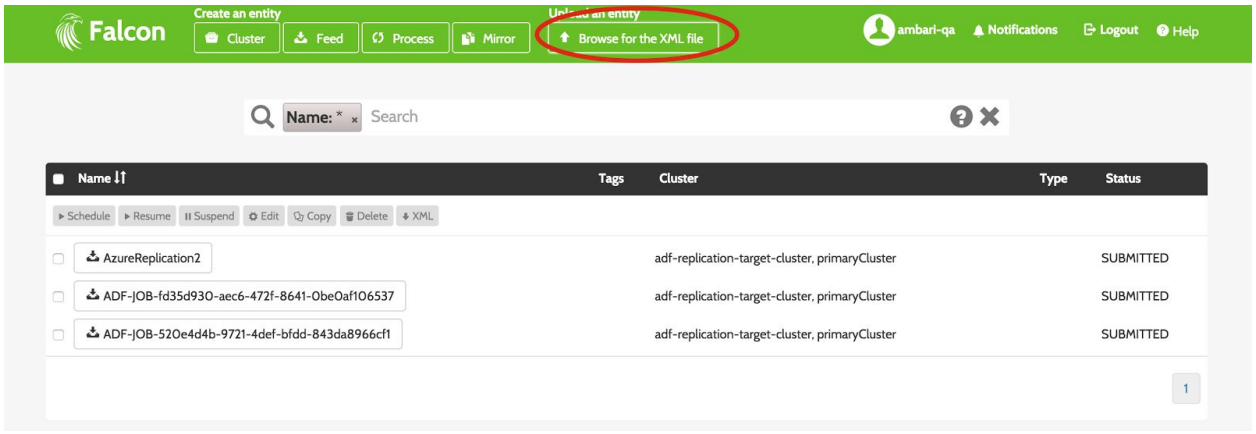


Figure 2. Submit entity specification *file*

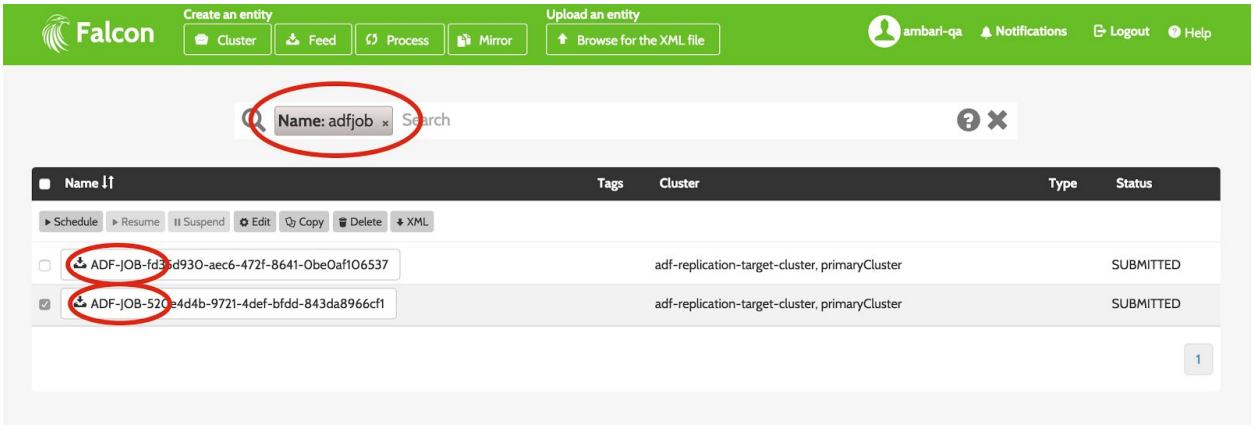


Figure 3. Search entities by subsequence of the name

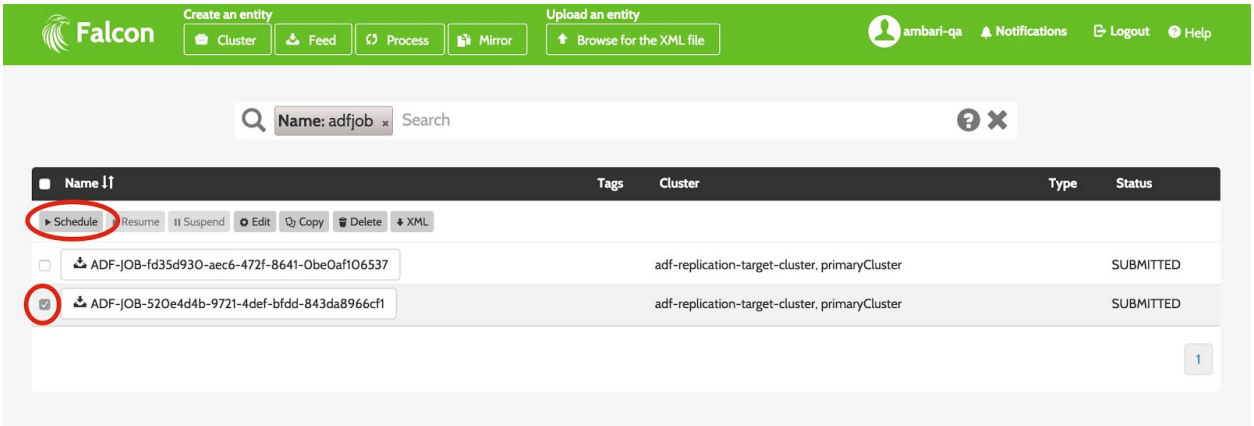


Figure 4. Schedule an entity

E2E Test Example Walkthrough

In this E2E test example, you can run HDP sandbox 2.3.2 on your local machine through VirtualBox: *http://hortonworks.com/hdp/downloads/*. Use the sample data factory for data replication from HDP cluster to Azure blob.

Azure Service Bus:
See Section "Set up Azure Service Bus credentials"

Azure Blob:
Account name: <AccountName>
Account key: <AccountKey>

Step 1. Install Falcon
Step 2. Set up Azure Service Bus credentials
Step 3. Falcon Configuration on Oozie-site through Ambari

Step 4. Make sure HDP components (i.e. HDFS, Yarn, MapReduce2, Oozie, Hive, Pig) are running well. You can check their running status on Ambari. Then, Start Falcon.
Step 5. Submit clusters. In our case, we need to create two clusters: *primaryCluster* and *adf-replication-target-cluster*. You can find cluster specification files in XML format on OneDrive.
Step 6. Now, you are ready for PIG E2E demo! Copy the input file *pig-input-2014-11* to the HDFS location *specified in your Azure Data Factory.* Log in to your data factory *account*. Run Azure pipeline *TestPigRunningOnHDPInline*! You should see new job entity created soon after Azure pipeline turns to "*in progress*". All the ADF jobs are named with prefix *ADF-JOB-*, so you can search ADF entities with name subsequence filter *adfjob* (case insensitive). See Figure 3 for illustration. You can monitor the job status (see Figure 5) by clicking the specific entity on Falcon homepage. Once the job is completed, you should see SUCCEEDED status on Azure portal soon (see Figure 6)!

Note: In Pig and Hive scripts, please use *indata* and *outdata* for input and output parameters, respectively. See examples under the folder *pig-example* and *hive-example*.

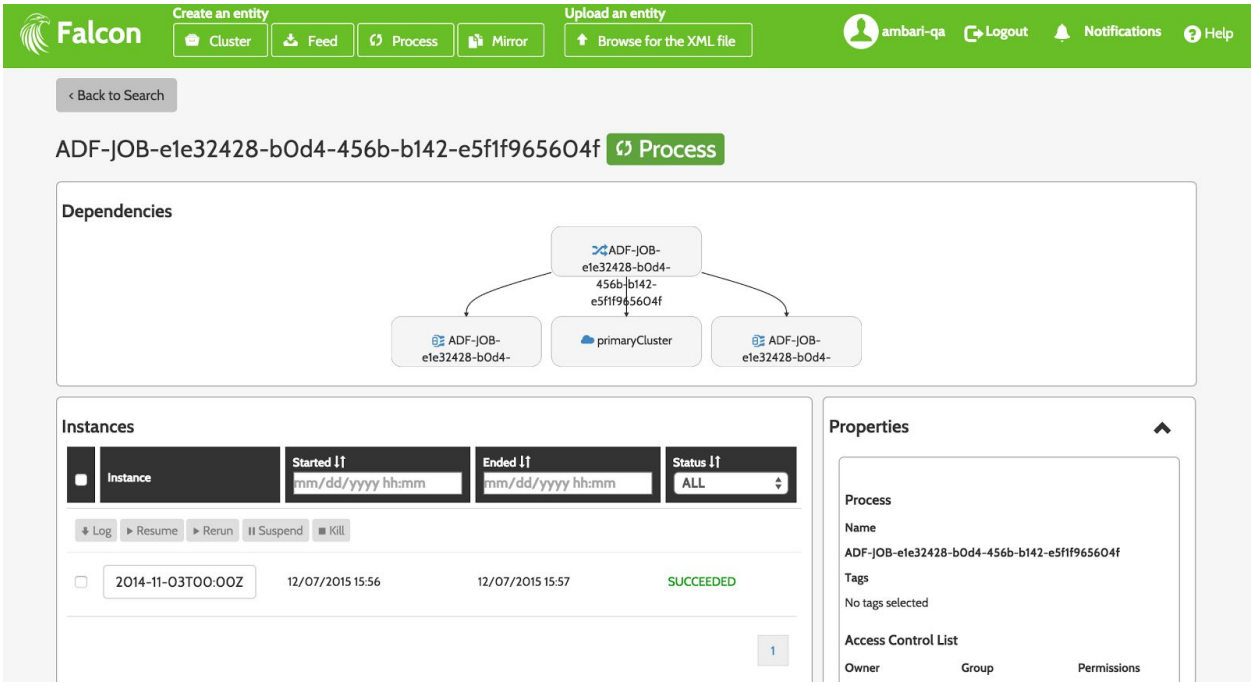


Figure 5. Monitor job status on Falcon entity page

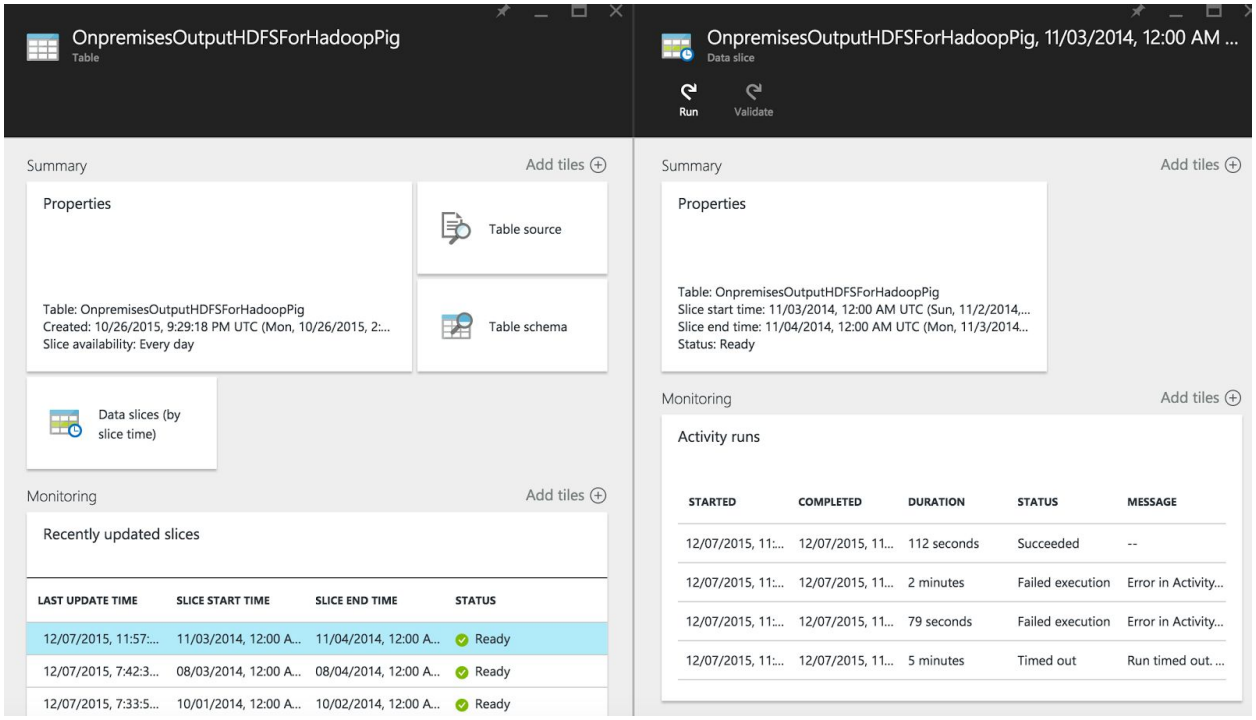


Figure 6. Up-to-date job status on Azure portal

Extra Configuration to Run Hive Jobs

On HDP 2.3.2, you need one extra configuration to run Hive jobs. On Ambari Hive Configs page, search for “hive.exec.post.hooks” and remove “org.apache.atlas.hive.hook.HiveHook” (See Figure 10). After the change, please restart Hive and then Oozie.

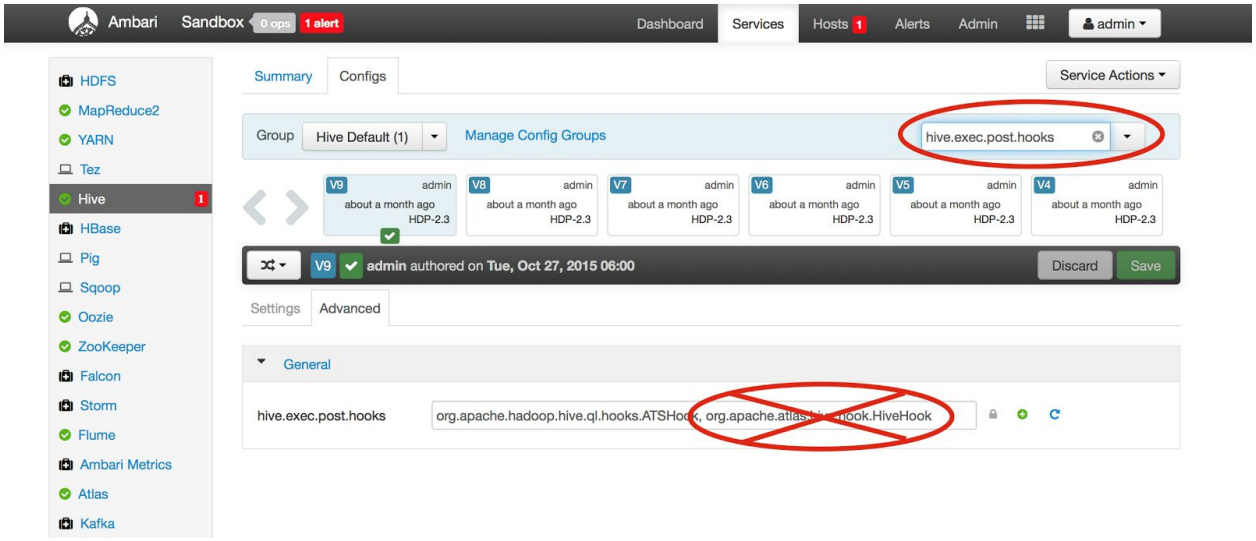


Figure 10. Remove HiveHook from Hive configuration

Extra Configuration to Replicate Data to Azure Blobs

To move data to and from Azure blobs with Falcon replication job, we need to add Azure blob credential to HDFS core-site through Ambari.

Step 1. Log in to Ambari: http://[cluster ip]:8080. The default user / password is admin / admin.

Step 2. Add Azure blob credential property to custom core-site. Go to advanced configs for HDFS (see Figure 7), expand custom core-site and click “add property” (see Figure 8) and add Azure credential as key/value property (see Figure 9). Use the following format for the key: *fs.azure.account.key.AZURE_BLOB_ACCOUNT_NAME.blob.core.windows.net*, e.g. *fs.azure.account.key.hwkadfstorage.blob.core.windows.net*. Use the Azure blob account key for the value.

Step 3. Restart the relevant components in the following order: HDFS, YARN, MapReduce2, Oozie, Falcon. The best way is to click on each component and use ‘restart all’ action on Ambari (see Figure 10). To restart the in-house version of Falcon, see Section “Start/Stop Falcon”.

Step 4. Test if we can access Azure blobs through HDFS:
e.g. *hadoop fs -ls wasb://replication-test@hwkadfstorage.blob.core.windows.net/*

Step 5. Now you can run sample Azure pipeline *TestReplicate2Azure*! Note that this pipeline copies the output data from PIG demo on HDFS */apps/falcon/adf-demo/pig-output-2014-11* to Azure blob *replication-test/2014/11*. So make sure you run pig pipeline to generate the input data for this replication pipeline or you manually copy your input data to the HDFS location.

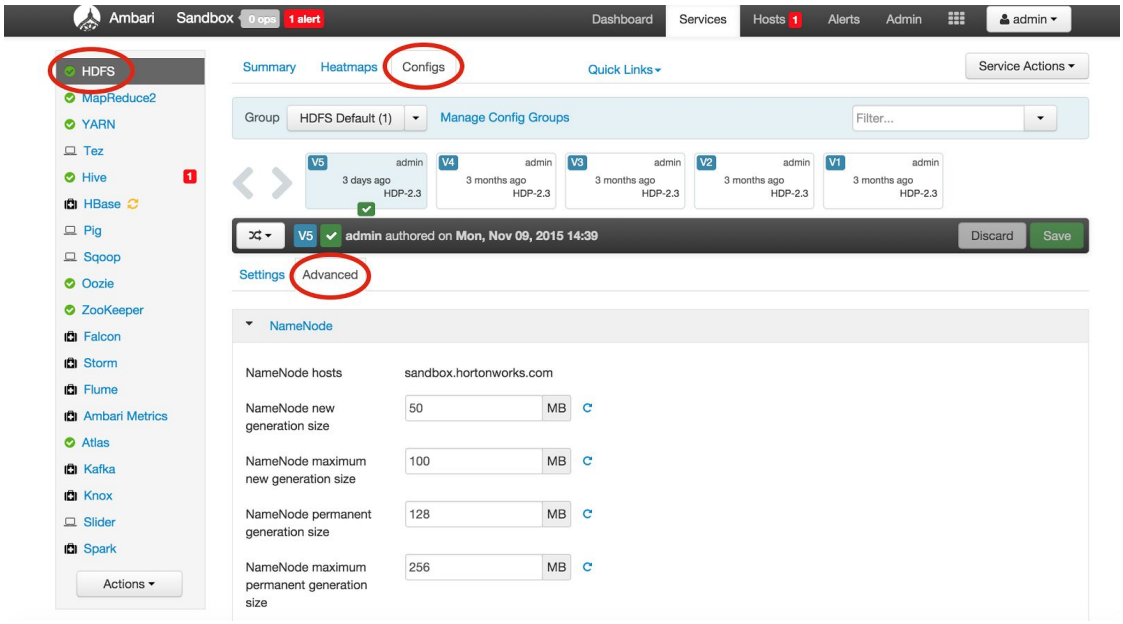


Figure 7. Access HDFS configuration through Ambari

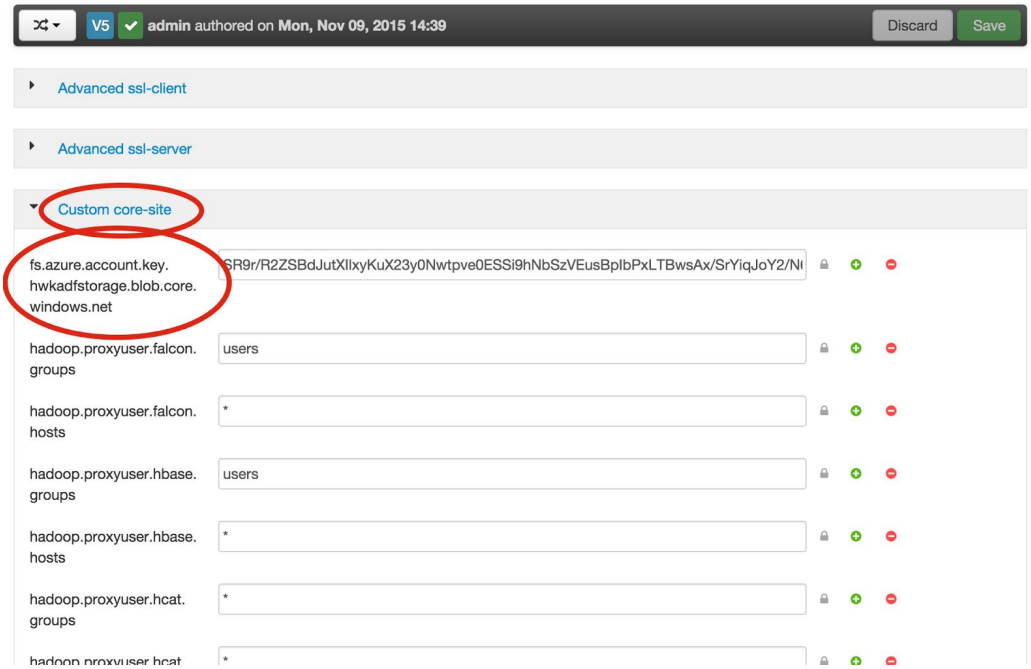


Figure 8. Expand custom core-site and add Azure blob credentials as a property

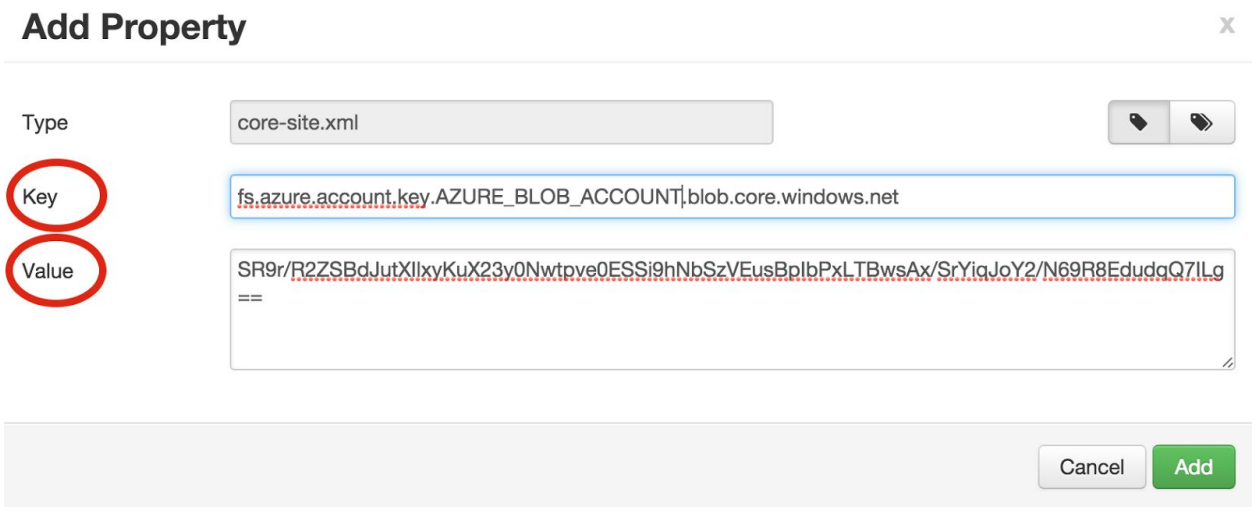


Figure 9. Add Azure blob credential property

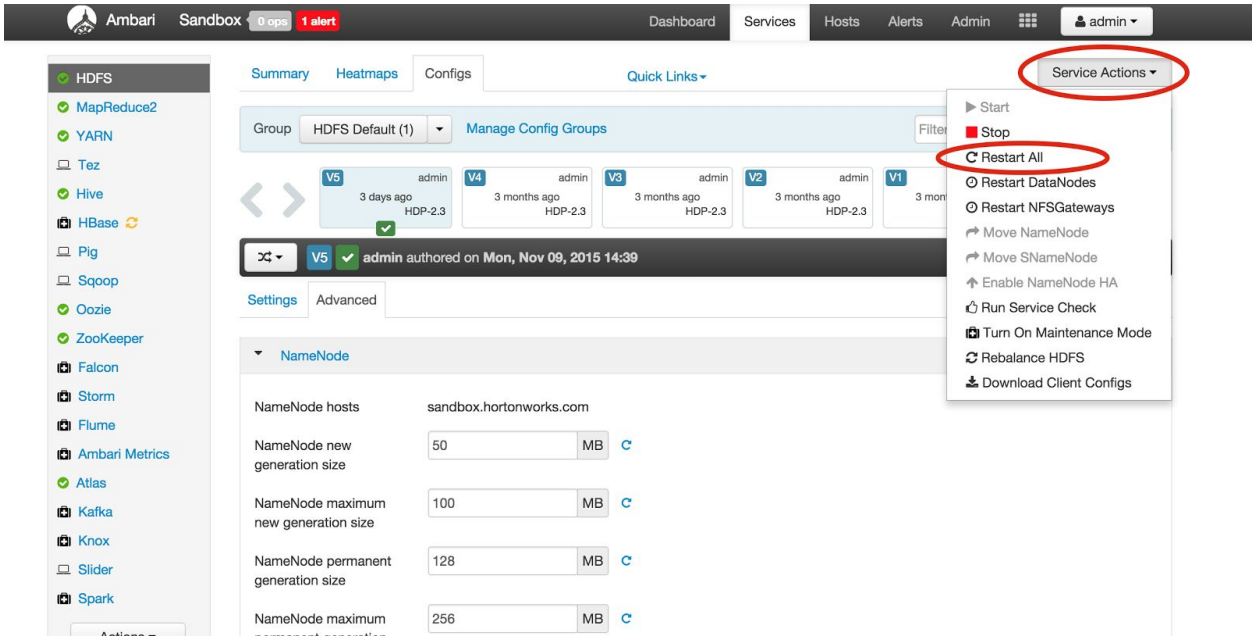


Figure 10. Restart HDP component