Apache Storm 3

# Installing and Configuring Apache Storm

**Date of Publish:** 2019-03-08

https://docs.hortonworks.com

# Contents

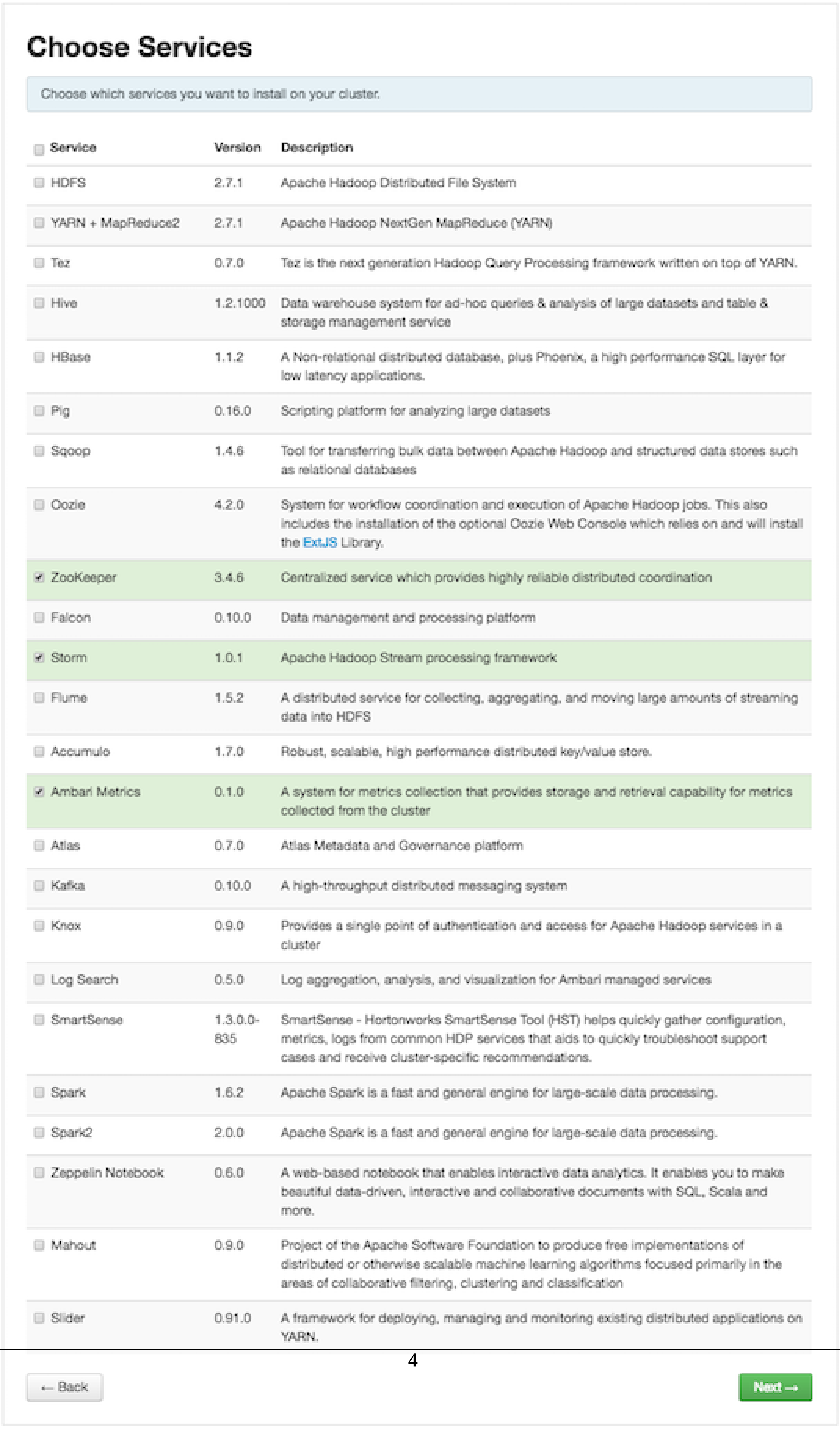
# Installing Apache Storm

### Before you begin

- HDP cluster stack version 2.5.0 or later.
- (Optional) Ambari version 2.4.0 or later.

### Procedure

1. Click the Ambari "Services" tab.
2. In the Ambari "Actions" menu, select "Add Service." This starts the Add Service Wizard, displaying the Choose Services screen. Some of the services are enabled by default.
3. Scroll down through the alphabetic list of components on the Choose Services page, select "Storm", and click "Next" to continue:



## Choose Services

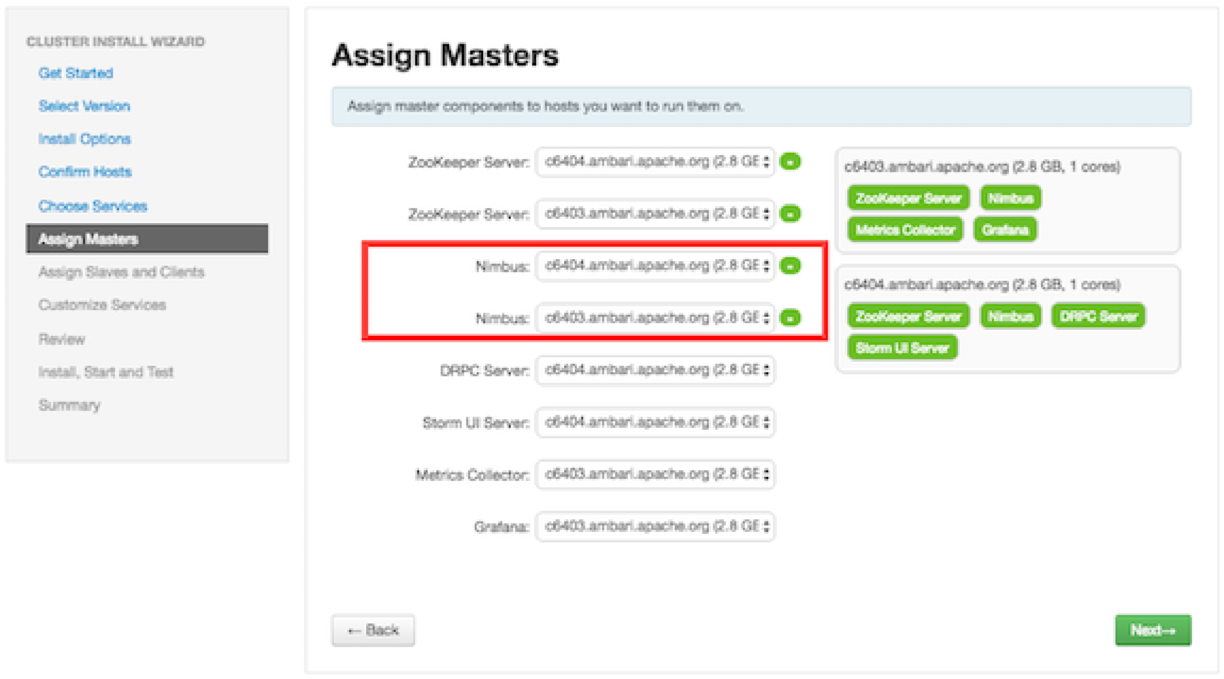
Choose which services you want to install on your cluster.

| | Service | Version | Description |
|---|---|---|---|
| ☐ | HDFS | 2.7.1 | Apache Hadoop Distributed File System |
| ☐ | YARN + MapReduce2 | 2.7.1 | Apache Hadoop NextGen MapReduce (YARN) |
| ☐ | Tez | 0.7.0 | Tez is the next generation Hadoop Query Processing framework written on top of YARN. |
| ☐ | Hive | 1.2.1000 | Data warehouse system for ad-hoc queries & analysis of large datasets and table & storage management service |
| ☐ | HBase | 1.1.2 | A Non-relational distributed database, plus Phoenix, a high performance SQL layer for low latency applications. |
| ☐ | Pig | 0.16.0 | Scripting platform for analyzing large datasets |
| ☐ | Sqoop | 1.4.6 | Tool for transferring bulk data between Apache Hadoop and structured data stores such as relational databases |
| ☐ | Oozie | 4.2.0 | System for workflow coordination and execution of Apache Hadoop jobs. This also includes the installation of the optional Oozie Web Console which relies on and will install the ExtJS Library. |
| ✓ | ZooKeeper | 3.4.6 | Centralized service which provides highly reliable distributed coordination |
| ☐ | Falcon | 0.10.0 | Data management and processing platform |
| ✓ | Storm | 1.0.1 | Apache Hadoop Stream processing framework |
| ☐ | Flume | 1.5.2 | A distributed service for collecting, aggregating, and moving large amounts of streaming data into HDFS |
| ☐ | Accumulo | 1.7.0 | Robust, scalable, high performance distributed key/value store. |
| ✓ | Ambari Metrics | 0.1.0 | A system for metrics collection that provides storage and retrieval capability for metrics collected from the cluster |
| ☐ | Atlas | 0.7.0 | Atlas Metadata and Governance platform |
| ☐ | Kafka | 0.10.0 | A high-throughput distributed messaging system |
| ☐ | Knox | 0.9.0 | Provides a single point of authentication and access for Apache Hadoop services in a cluster |
| ☐ | Log Search | 0.5.0 | Log aggregation, analysis, and visualization for Ambari managed services |
| ☐ | SmartSense | 1.3.0.0-835 | SmartSense - Hortonworks SmartSense Tool (HST) helps quickly gather configuration, metrics, logs from common HDP services that aids to quickly troubleshoot support cases and receive cluster-specific recommendations. |
| ☐ | Spark | 1.6.2 | Apache Spark is a fast and general engine for large-scale data processing. |
| ☐ | Spark2 | 2.0.0 | Apache Spark is a fast and general engine for large-scale data processing. |
| ☐ | Zeppelin Notebook | 0.6.0 | A web-based notebook that enables interactive data analytics. It enables you to make beautiful data-driven, interactive and collaborative documents with SQL, Scala and more. |
| ☐ | Mahout | 0.9.0 | Project of the Apache Software Foundation to produce free implementations of distributed or otherwise scalable machine learning algorithms focused primarily in the areas of collaborative filtering, clustering and classification |
| ☐ | Slider | 0.91.0 | A framework for deploying, managing and monitoring existing distributed applications on YARN. |

4. On the Assign Masters page, review node assignments for Storm components.

   If you want to run Storm with high availability of nimbus nodes, select more than one nimbus node; the Nimbus daemon automatically starts in HA mode if you select more than one nimbus node.

   Modify additional node assignments if desired, and click "Next".



CLUSTER INSTALL WIZARD
- Get Started
- Select Version
- Install Options
- Confirm Hosts
- Choose Services
- Assign Masters
- Assign Slaves and Clients
- Customize Services
- Review
- Install, Start and Test
- Summary

Assign Masters

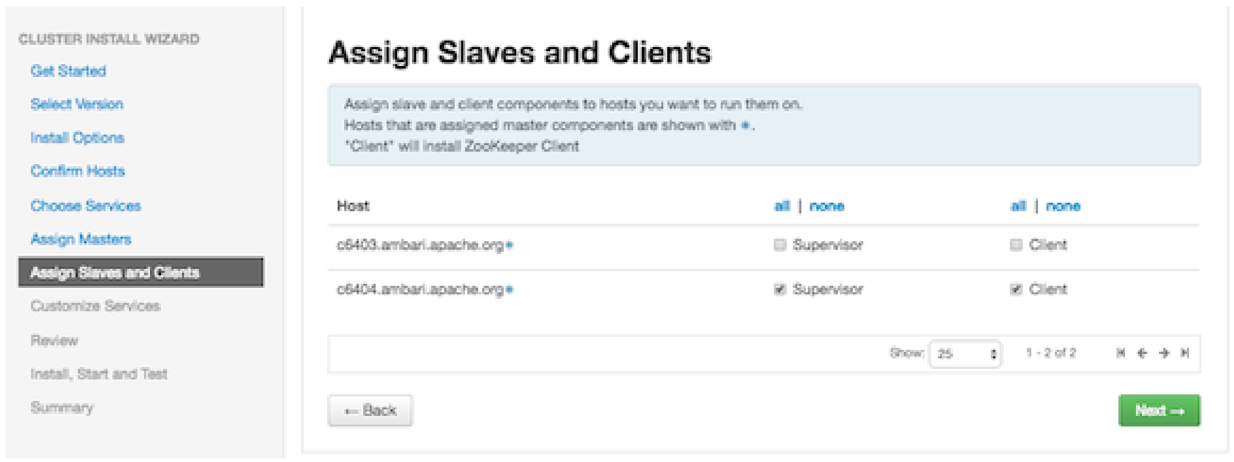
Assign master components to hosts you want to run them on.

| | |
|---|---|
| ZooKeeper Server: | c6404.ambari.apache.org (2.8 GE |
| ZooKeeper Server: | c6403.ambari.apache.org (2.8 GE |
| Nimbus: | c6404.ambari.apache.org (2.8 GE |
| Nimbus: | c6403.ambari.apache.org (2.8 GE |
| DRPC Server: | c6404.ambari.apache.org (2.8 GE |
| Storm UI Server: | c6404.ambari.apache.org (2.8 GE |
| Metrics Collector: | c6403.ambari.apache.org (2.8 GE |
| Grafana: | c6403.ambari.apache.org (2.8 GE |

c6403.ambari.apache.org (2.8 GB, 1 cores)
ZooKeeper Server, Nimbus, Metrics Collector, Grafana

c6404.ambari.apache.org (2.8 GB, 1 cores)
ZooKeeper Server, Nimbus, DRPC Server, Storm UI Server

5. On the Assign Slaves and Clients page, choose the nodes that you want to run Storm supervisors and clients:



CLUSTER INSTALL WIZARD
- Get Started
- Select Version
- Install Options
- Confirm Hosts
- Choose Services
- Assign Masters
- Assign Slaves and Clients
- Customize Services
- Review
- Install, Start and Test
- Summary

Assign Slaves and Clients

Assign slave and client components to hosts you want to run them on.
Hosts that are assigned master components are shown with ✱.
"Client" will install ZooKeeper Client

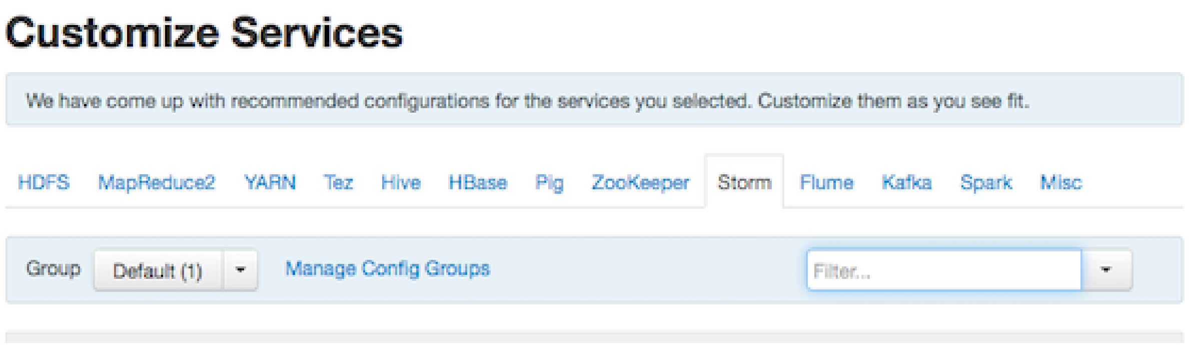
| Host | all \| none | all \| none |
|---|---|---|
| c6403.ambari.apache.org✱ | ☐ Supervisor | ☐ Client |
| c6404.ambari.apache.org✱ | ☑ Supervisor | ☑ Client |

Show: 25 1 - 2 of 2

   Storm supervisors are nodes from which the actual worker processes launch to execute spout and bolt tasks.

   Storm clients are nodes from which you can run Storm commands (jar, list, and so on).

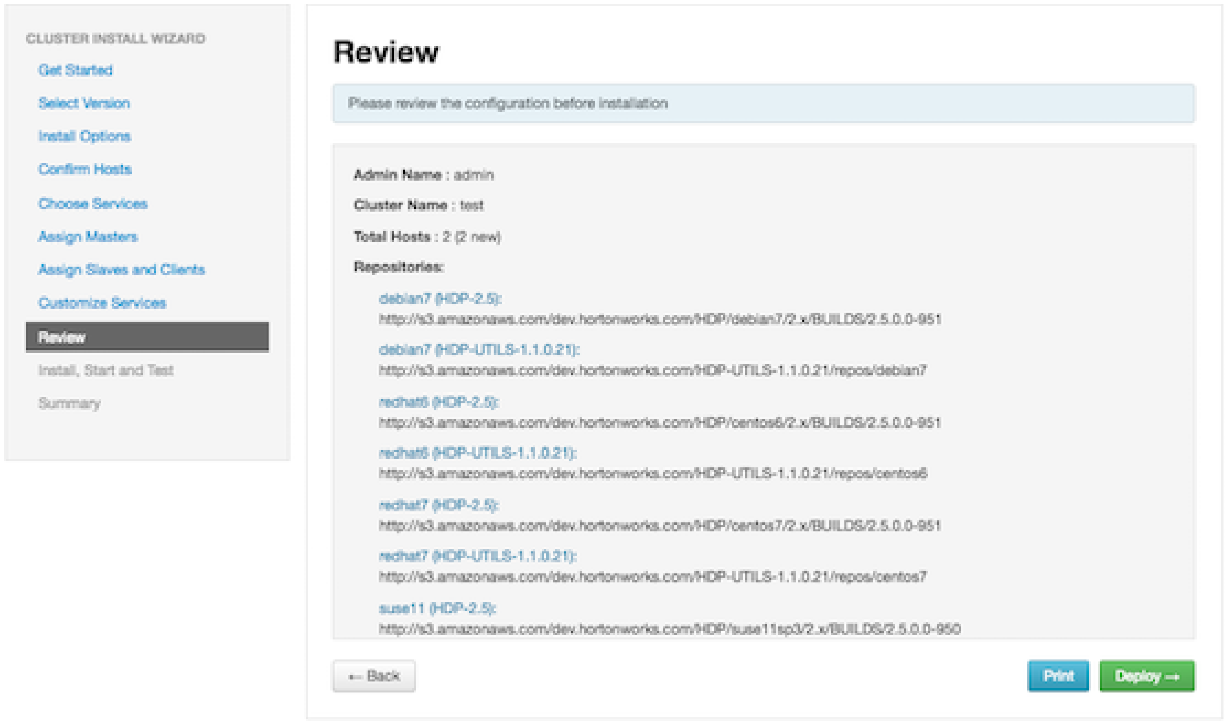
6. Click **Next** to continue.
7. Ambari displays the Customize Services page, which lists a series of services:


Customize Services

We have come up with recommended configurations for the services you selected. Customize them as you see fit.

For your initial configuration you should use the default values set by Ambari. If Ambari prompts you with the message "Some configurations need your attention before you can proceed," review the list of properties and provide the required information.

8. Click **Next** to continue.
9. When the wizard displays the Review page, ensure that all HDP components correspond to HDP 2.5.0 or later:


Review

Please review the configuration before installation

**Admin Name** : admin
**Cluster Name** : test
**Total Hosts** : 2 (2 new)
**Repositories:**

debian7 (HDP-2.5):
http://s3.amazonaws.com/dev.hortonworks.com/HDP/debian7/2.x/BUILDS/2.5.0.0-951

debian7 (HDP-UTILS-1.1.0.21):
http://s3.amazonaws.com/dev.hortonworks.com/HDP-UTILS-1.1.0.21/repos/debian7

redhat6 (HDP-2.5):
http://s3.amazonaws.com/dev.hortonworks.com/HDP/centos6/2.x/BUILDS/2.5.0.0-951

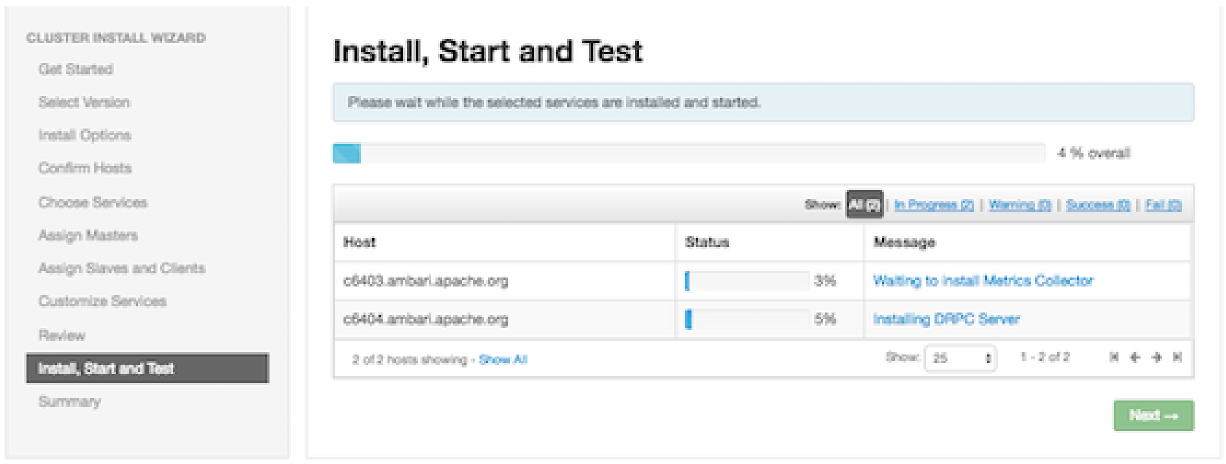
redhat6 (HDP-UTILS-1.1.0.21):
http://s3.amazonaws.com/dev.hortonworks.com/HDP-UTILS-1.1.0.21/repos/centos6

redhat7 (HDP-2.5):
http://s3.amazonaws.com/dev.hortonworks.com/HDP/centos7/2.x/BUILDS/2.5.0.0-951

redhat7 (HDP-UTILS-1.1.0.21):
http://s3.amazonaws.com/dev.hortonworks.com/HDP-UTILS-1.1.0.21/repos/centos7

suse11 (HDP-2.5):
http://s3.amazonaws.com/dev.hortonworks.com/HDP/suse11sp3/2.x/BUILDS/2.5.0.0-950

10. Click **Deploy** to begin installation.
11. Ambari displays the Install, Start and Test page. Review the status bar and messages for progress updates:



Install, Start and Test

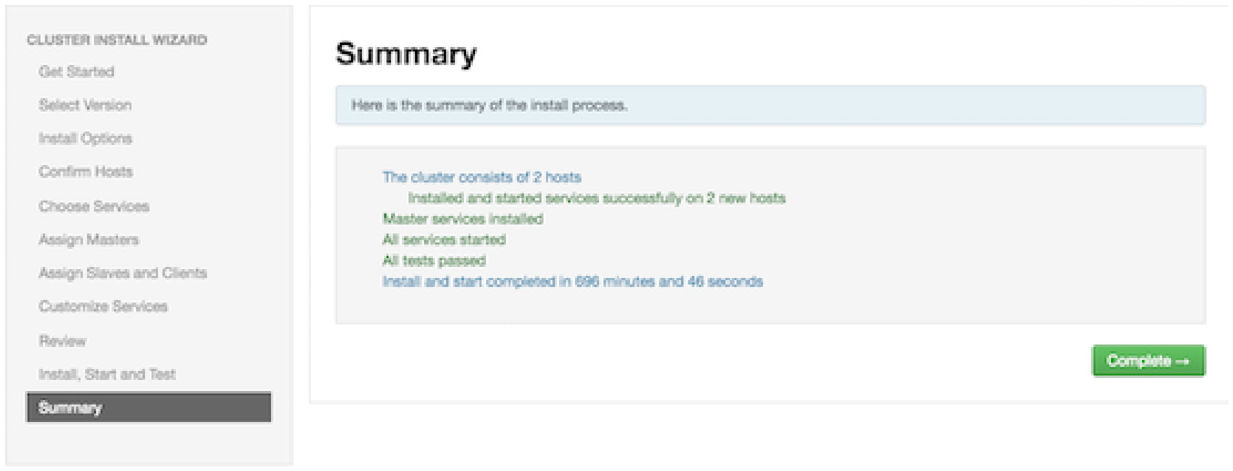
Please wait while the selected services are installed and started.

4 % overall

| Host | Status | Message |
|---|---|---|
| c6403.ambari.apache.org | 3% | Waiting to install Metrics Collector |
| c6404.ambari.apache.org | 5% | Installing DRPC Server |

2 of 2 hosts showing - Show All

12. When the wizard presents a summary of results, click "Complete" to finish installing Storm:



Summary

Here is the summary of the install process.

The cluster consists of 2 hosts
Installed and started services successfully on 2 new hosts
Master services installed
All services started
All tests passed
Install and start completed in 696 minutes and 46 seconds

## What to do next

To validate the Storm installation, complete the following steps:

1. Point your browser to the Storm UI URL for Ambari: http://<storm-ui-server>:8744 . You should see the Storm UI web page.
2. Submit the following command:

   storm jar /usr/hdp/current/storm-client/contrib/storm-starter/storm-starter-topologies-*.jar org.apache.storm.starter.WordCountTopology wordcount
3. The WordCount sample topology should run successfully.

# Configuring Apache Storm for a Production Environment

This chapter covers topics related to Storm configuration:

- Configuring Storm to operate under supervision
- Properties to review when you place topologies into production use
- Enabling audit to HDFS for a secure cluster

Instructions are for Ambari-managed clusters.

## Configuring Storm for Supervision

If you are deploying a production cluster with Storm, you should configure the Storm components to operate under supervision.

### Procedure

1. Stop all Storm components.
   a. Using Ambari Web, browse to Services > Storm > Service Actions.
   b. Choose Stop, and wait until the Storm service completes.
2. Stop Ambari Server:

   ambari-server stop
3. Change the Supervisor and Nimbus command scripts in the Stack definition. On Ambari Server host, run:

```
sed -ir "s/scripts\/supervisor.py/scripts\/supervisor_prod.py/g" /var/lib/
ambari-server/resources/common-services/STORM/0.9.1.2.1/metainfo.xml

sed -ir "s/scripts\/nimbus.py/scripts\/nimbus_prod.py/g" /var/lib/ambari-
server/resources/common-services/STORM/0.9.1.2.1/metainfo.xml
```

4. Install supervisord on all Nimbus and Supervisor hosts.
   a. Install EPEL repository:

      yum install epel-release -y
   b. Install supervisor package for supervisord:

      yum install supervisor -y
   c. Enable supervisord on autostart:

      chkconfig supervisord on
   d. Change supervisord configuration file permissions:

      chmod 600 /etc/supervisord.conf
5. Configure supervisord to supervise Nimbus Server and Supervisors by appending the following to /etc/supervisord.conf on all Supervisor host and Nimbus hosts:

```
[program:storm-nimbus]
command=env PATH=$PATH:/bin:/usr/bin/:/usr/jdk64/jdk1.7.0_67/bin/
 JAVA_HOME=/usr/jdk64/jdk1.7.0_67 /usr/hdp/current/storm-nimbus/bin/storm
 nimbus
user=storm
autostart=true
autorestart=true
startsecs=10
startretries=999
log_stdout=true
log_stderr=true
logfile=/var/log/storm/nimbus.out
logfile_maxbytes=20MB
logfile_backups=10

[program:storm-supervisor]
command=env PATH=$PATH:/bin:/usr/bin/:/usr/jdk64/jdk1.7.0_67/bin/
 JAVA_HOME=/usr/jdk64/jdk1.7.0_67 /usr/hdp/current/storm-supervisor/bin/
storm supervisor
user=storm
autostart=true
autorestart=true
```

```
startsecs=10
startretries=999
log_stdout=true
log_stderr=true
logfile=/var/log/storm/supervisor.out
logfile_maxbytes=20MB
logfile_backups=10
```

**Note:**

Change /usr/jdk64/jdk1.7.0_67 to the location of the JDK being used by Ambari in your environment.

6. Start supervisord on all Supervisor and Nimbus hosts:

   service supervisord start

7. Start Ambari Server:

   ambari-server start

8. Start all other Storm components:

   a. Using Ambari Web, browse to Services > Storm > Service Actions.
   b. Choose Start.

## Configuring Storm Resource Usage

The following settings can be useful for tuning Storm topologies in production environments.

Instructions are for a cluster managed by Ambari. For clusters that are not managed by Ambari, update the property in its configuration file; for example, update the value of topology.message.timeout.secs in the storm.yaml configuration file. (Do not update files manually if your cluster is managed by Ambari.)

Memory Allocation

| | |
|---|---|
| **Worker process max heap size: worker.childopts -XmX option** | Maximum JVM heap size for the worker JVM. The default Ambari value is 768 MB. On a production system, this value should be based on workload and machine capacity. If you observe out-of-memory errors in the log, increase this value and fine tune it based on throughput; 1024 MB should be a reasonable value to start with.<br><br>To set maximum heap size for the worker JVM, navigate to the "Advanced storm-site" category and append the -Xmx option to worker.childopts setting. The following option sets maximum heap size to 1 GB: -Xmx1024m |
| **Logviewer process max heap size: logviewer.childopts -Xmx option** | Maximum JVM heap size for the logviewer process. The default is 128 MB. On production machines you should consider increasing the logviewer.childopts -Xmx option to 768 MB or more (1024 MB should be a sufficient for an upper-end value). |

Message Throughput

| | |
|---|---|
| **topology.max.spout.pending** | Maximum number of messages that can be pending in a spout at any time. The default is null (no limit).<br><br>The setting applies to all core Storm and Trident topologies in a cluster: |

- For core Storm, this value specifies the maximum number of tuples that can be pending: tuples that have been emitted from a spout but have not been acked or failed yet.
- For Trident, which process batches in core, this property specifies the maximum number of batches that can be pending.

If you expect bolts to be slow in processing tuples (or batches) and you do not want internal buffers to fill up and temporarily stop emitting tuples to downstream bolts, you should set topology.max.spout.pending to a starting value of 1000 (for core Storm) or a value of 1 (for Trident), and increase the value depending on your throughput requirements.

You can override this value for a specific topology when you submit the topology. The following example restricts the number of pending tuples to 100 for a topology:

$ storm jar -c topology.max.spout.pending=100 jar args...

If you plan to use windowing functionality, set this value to null, or increase it to cover the estimated maximum number of active tuples in a single window. For example, if you define a sliding window with a duration of 10 minutes and a sliding interval of 1 minute, set topology.max.spout.pending to the maximum number of tuples that you expect to receive within an 11-minute interval.

This setting has no effect on spouts that do not anchor tuples while emitting.

**topology.message.timeout.secs**

Maximum amount of time given to the topology to fully process a tuple tree from the core-storm API, or a batch from the Trident API, emitted by a spout. If the message is not acked within this time frame, Storm fails the operation on the spout. The default is 30 seconds.

If you plan to use windowing functionality, set this value based on your windowing definitions. For example, if you define a 10 minute sliding window with a 1 minute sliding interval, you should set this value to at least 11 minutes.

You can also set this value at the topology level when you submit a topology; for example:

$ storm jar -c topology.message.timeout.secs=660 jar args...

Nimbus Node Resources

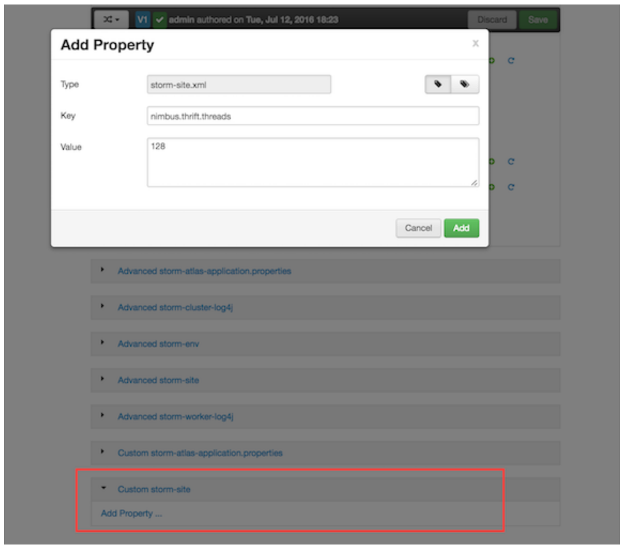
**nimbus.thrift.max_buffer_size**

Maximum buffer size that the Nimbus Thrift server allocates for servicing requests. The default is 1 MB. If you plan to submit topology files larger than 100 MB, consider increasing this value.

**nimbus.thrift.threads**

Number of threads to be used by the Nimbus Thrift server. The default is 64 threads. If you have more than ten hosts in your Storm cluster, consider increasing this value to a minimum of 196 threads, to handle the workload associated with multiple workers making multiple requests on each host.

You can set this value by adding the property and its value in the Custom storm-site category, as shown in the following graphic:



admin authored on Tue, Jul 12, 2016 18:23

Add Property

Type: storm-site.xml

Key: nimbus.thrift.threads

Value: 128

V1
Discard
Save
Cancel
Add
Advanced storm-atlas-application.properties
Advanced storm-cluster-log4j
Advanced storm-env
Advanced storm-site
Advanced storm-worker-log4j
Custom storm-atlas-application.properties
Custom storm-site
Add Property ...

Number of Workers on a Supervisor Node

**supervisor.slots.ports**

List of ports that can run workers on a supervisor node. The length of this list defines the number of workers that can be run on a supervisor node; there is one communication port per worker.

Use this configuration to tune how many workers to run on each machine. Adjust the value based on how many resources each worker will consume, based on the topologies you will submit (as opposed to machine capacity).

Number of Event Logger Tasks

**topology.eventlogger.executors**

Number of event logger tasks created for topology event logging. The default is 0; no event logger tasks are created.

If you enable topology event logging, you must set this value to a number greater than zero, or to null:

- topology.eventlogger.executors: <n> creates n event logger tasks for the topology. A value of 1 should be sufficient to handle most event logging use cases.
- topology.eventlogger.executors: null creates one event logger task per worker. This is only needed if

| | |
|---|---|
| | you plan to use a high sampling percentage, such as logging all tuples from all spouts and bolts. |
| Storm Metadata Directory | |
| **storm.local.dir** | Local directory where Storm daemons store topology metadata. You need not change the default value, but if you do change it, set it to a durable directory (not a directory such as /tmp). |