Hortonworks SmartSense

User Guide

(May 17, 2018)

docs.cloudera.com

Hortonworks SmartSense: User Guide

The Hortonworks Data Platform, powered by Apache Hadoop, is a massively scalable and 100% open source platform for storing, processing and analyzing large volumes of data. It is designed to deal with data from many sources and formats in a very quick, easy and cost-effective manner. The Hortonworks Data Platform consists of the essential set of Apache Hadoop projects including MapReduce, Hadoop Distributed File System (HDFS), HCatalog, Pig, Hive, HBase, ZooKeeper and Ambari. Hortonworks is the major contributor of code and patches to many of these projects. These projects have been integrated and tested as part of the Hortonworks Data Platform release process and installation and configuration tools have also been included.

Unlike other providers of platforms built using Apache Hadoop, Hortonworks contributes 100% of our code back to the Apache Software Foundation. The Hortonworks Data Platform is Apache-licensed and completely open source. We sell only expert technical support, training and partner-enablement services. All of our technology is, and will remain free and open source. Please visit the Hortonworks Data Platform page for more information on Hortonworks technology. For more information on Hortonworks services, please visit either the Support or Training page. Feel free to Contact Us directly to discuss your specific needs.

Table of Contents

List of Tables

1. Document Navigation

Hortonworks SmartSense gives all support subscription customers access to a unique service that analyzes HDP cluster diagnostic data, identifies potential issues, and recommends specific solutions and actions. These analytics proactively identify unseen issues and notify customers of potential problems before they occur.

The Hortonworks SmartSense Tool (HST) provides cluster diagnostic data collection capabilities, enabling customers to quickly gather configuration, metrics, and logs that they can use to analyze and troubleshoot SmartSense support cases.

Hortonworks SmartSense User Guide provides you with the latest information about using SmartSense. For SmartSense installation and upgrade instructions, see the Hortonworks SmartSense Installation. After installing SmartSense, refer to the *Hortonworks SmartSense User Guide* for information about using SmartSense and performing additional configuration.

2. Using SmartSense with Ambari

SmartSense is automatically included in Ambari 2.2.x and later. The integration between Ambari and SmartSense is facilitated by the Ambari stack and views extension mechanisms. These extensions enable you to add SmartSense as a native Ambari service, and they automatically deploy an Ambari view, enabling you to quickly capture data using the Ambari web UI.

2.1. Roles Required for Using SmartSense

The following table describes bundle capture-related actions and roles required to perform them:

Action	Ambari Administrator	Other Users*
Access Ambari View	✓	✓
Initiate SmartSense capture	✓	
Initiate support capture	✓	
View "Bundles" page	✓	✓
View bundle	✓	✓
Upload a bundle	✓	✓
Download encrypted bundles	✓	✓
Download unencrypted bundles	✓	✓
Delete bundles	✓	✓
View capture schedule	✓	✓
Update capture schedule	✓	
Pause capture schedule	✓	
Activate capture schedule	✓	
Delete capture schedule	✓	
View recommendations	✓	✓

Action	Ambari Administrator	Other Users*
Apply recommendations	✓	
Revert recommendations	✓	

"Other Users" include Cluster Administrator, Cluster User, Service Operator, Service Administrator, and Cluster Operator as defined in Understanding Cluster Roles.

2.2. Capturing Bundles

After you install the SmartSense service and view, data collection can begin.

To trigger an ad hoc capture, access the **SmartSense View** by clicking the ▦ icon and selecting **SmartSense View**, and then follow steps depending on your use case:

- If you would like to prevent issues, improve security, and/or increase availability and performance of your cluster: Capturing for Proactive Analysis.
- If you are working with support to troubleshoot a support case: Capturing for Troubleshooting.

2.2.1. Capturing for Proactive Analysis

To capture bundles for analysis, follow these steps:

1. Under **Select the intent for data capture**, select **Proactive Analysis**.
2. Click the **Capture** button.

   SmartSense will analyze cluster configuration and metrics for all cluster nodes, and will produce recommendations to prevent issues, improve security, availability and performance of your cluster.

Related Links

Downloading and Uploading Bundles [5]

2.2.2. Capturing for Troubleshooting

To capture bundles for analysis, follow these steps:

1. Under **Select the intent for data capture**, select **Support Case Troubleshooting**.
2. Enter your **Case Number**.
3. Select the type of diagnosis:

   - **Cluster Service**:

     a) Select services for diagnosis. The list services that can be captured is available in Services Available for Capture.

b) Next, select hosts for diagnosis: **All Hosts** or choose **Only Specific Hosts** and select specific hosts.

- **YARN Application**:

  Enter **Application ID**. The YARN application details, application master logs and a subset of container logs will be captured.

- **Hive Query**:

  Enter one of the following: **Tez DAG ID**, **YARN App ID**, **MR Job ID**, or **Hive Query ID**. The SQL query, execution plan, application logs will be captured.

4. Click the **Capture** button.

   This triggers Ambari agents on each node to invoke the HST agent to capture specific data.

   After HST agents complete their captures and report data to the HST server, the completed bundle is available in the bundles list for download, or it is automatically uploaded to the SmartSense Gateway, if configured.

Related Links

Downloading and Uploading Bundles [5]

2.3. Automatically Capturing and Uploading Bundles via SmartSense Gateway

When enabled, the gateway automatically uploads completed bundles to Hortonworks when a capture is completed. This includes SmartSense analysis as well as support case troubleshooting bundles. You can also schedule SmartSense Analysis bundles for capture and automatic upload in the SmartSense Ambari view.

2.3.1. Creating a New Capture Schedule

If you have deleted the default capture schedule, you can create a new one:

1. Access **SmartSense View** by clicking ▦ and selecting **SmartSense View**.
2. Click the **Schedule** link in the top right corner to access the scheduler settings.
3. Select the scheduling period (weekly or monthly) and the day of the week and time of day that you want the capture to take place.
4. Click **Set Capture Schedule**.

> **Note**
>
> Scheduler changes take up to one hour to take effect.

2.3.2. Updating the Capture Schedule

The SmartSense view provides a way to easily create, update, pause, resume, and remove the schedules used for automated bundle capture and upload. When you deploy it, SmartSense creates a default capture schedule. To view this default capture schedule and update it, follow these steps:

1. Access **SmartSense View** by clicking ▦ and selecting **SmartSense View**.
2. Click the **Schedule** link in the top right corner to access the scheduler settings.
3. Remove, pause, or resume existing schedules.

   You can also update the capture schedule by selecting a new scheduling period (weekly or monthly) or changing the day of the week and time of day that you want the capture to take place.

Note

Scheduler changes take up to one hour to take effect.

2.4. Downloading and Uploading Bundles

Completed bundles can be manually downloaded and uploaded. To view and download bundles, follow these steps:

1. Access the **SmartSense View** by clicking ▦ and selecting **SmartSense View**.
2. Click the **Bundles** link in the top right corner.

   This page shows all bundles that have been captured and their status. If data is still being captured, the UI automatically updates itself with the capture progress until completed.
3. After the bundle is in a completed state, you can:
   - Download it manually by clicking **Download** and selecting either **Download Encrypted** or **Download Unencrypted**.
   - Upload it manually by clicking **Upload**.

You can also automate and schedule this process by using the SmartSense Gateway. When using the SmartSense Gateway, all bundles are uploaded to Hortonworks. When support case troubleshooting bundles are received, they trigger a case notification. This case notification uses the case number provided during the capture initiation process. For more information about available ways to upload support bundles, see Bundle Transport.

2.5. Reviewing SmartSense Recommendations

From SmartSense View in Ambari Web UI, you can access your SmartSense recommendations.

Prerequisites

In order for recommendations to be generated for your cluster, you first need to capture a bundle and then upload it through HTTPS gateway for analysis.

Alternatively, you can schedule automatic capture and upload using the SmartSense Gateway.

Steps

1. Access the **SmartSense View** by clicking ▦ and selecting **SmartSense View**.

2. Click the **Recommendations** link in the top right corner.

3. From the recommendations page, you can view open recommendations, and view previously deferred and ignored recommendations.

4. To make sure that the recommendations are up-to-date, from the ... menu select **Get Latest**.

5. You can search and filter the recommendations:

    - Click on a column name to sort the recommendations accordingly.

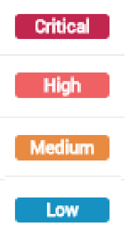
    - Use the search box in the top right corner to filter recommendations.

    - To review and apply, ignore, or defer a recommendation, click on its corresponding row.

2.5.1. Recommendations List

The following information is available for each recommendation:

Table 2.1.

Column	Description
Priority	One of:  Critical High Medium Low
Service	The HDP service to which the recommendation applies.
Recommendation	The summary of the recommendation.
Category	Broad category (such as "Operations", "Performance", or "Security") to which the recommendation belongs.
Avg. Rating	Average customer rating for the recommendation.

Column	Description
Classifiers	These markers indicate actions available for any recommendation: ⚙ means that the recommendation can be applied automatically through Ambari. 🔧 means that the configuration is not managed by Ambari and you must apply the recommendation manually. ☰ means that in order to apply the recommendation you must also apply dependent recommendations. ↺ means that the recommendation has previously been applied and then reverted.

2.5.2. Reviewing a Recommendation

To review a recommendation, click on its corresponding row.

The following actions are available for each recommendation:

- **Ignore** - Let us know that you do not want to apply this recommendation and help us understand why.
- **Defer** - Let us know that you will be applying this at a later date, but not right now.
- **Mark As Applied** (for recommendations that have to be applied manually), or **Proceed to Apply** (for recommendations that can be applied automatically)

If you ignore or defer a recommendation, you can still apply it later.

2.5.2.1. Recommendation Details

The following information is available for each recommendation:

Table 2.2.

Column	Description
Priority	One of: Critical High Medium Low
Status	One of: • Open

Column	Description
	• Applied • Ignored • Deferred • Reverted • Reopened
Recommendation Category	Broad category (such as "Operations", "Performance", or "Security") to which the recommendation belongs.
Affects	Describes to which specific software component the recommendation is related.
Rule Id	Unique ID that identifies the SmartSense rule related to the SmartSense recommendation.
Description	Background and context related to the recommendation.
Findings	Description of how your cluster deviates from the recommended configuration.
Recommendation	An outline of specific changes that need to be made to your cluster to apply the recommendation.
Configurations	Lists affected configuration properties, including: • **Config File** - The specific file that needs to be changed • **Property Name** - The specific property that needs to be changed • **Captured Value** - Configured value at the time of bundle capture • **Current Value** - Current configured value in Ambari • **Recommended Value** - Recommended value for this cluster
Affected hosts	Hosts on which the configuration change is required.

2.5.3. Applying a Recommendation

While some recommendations can be applied automatically, others have to be applied manually.

There are two ways to tell how a recommendation can be applied:

- When reviewing open recommendations, you can see in the **Classifiers** column, what options are available for which recommendation.

- When reviewing a specific recommendation, you can see one of the two options: **Mark As Applied** (for recommendations that must be applied manually) or **Proceed to Apply** (for recommendations that must be applied automatically).

Applying a Recommendation Automatically

1. From the **Recommendations** page, click on the table row corresponding to the recommendation that you want to review.

2. Click on **Proceed to Apply**.

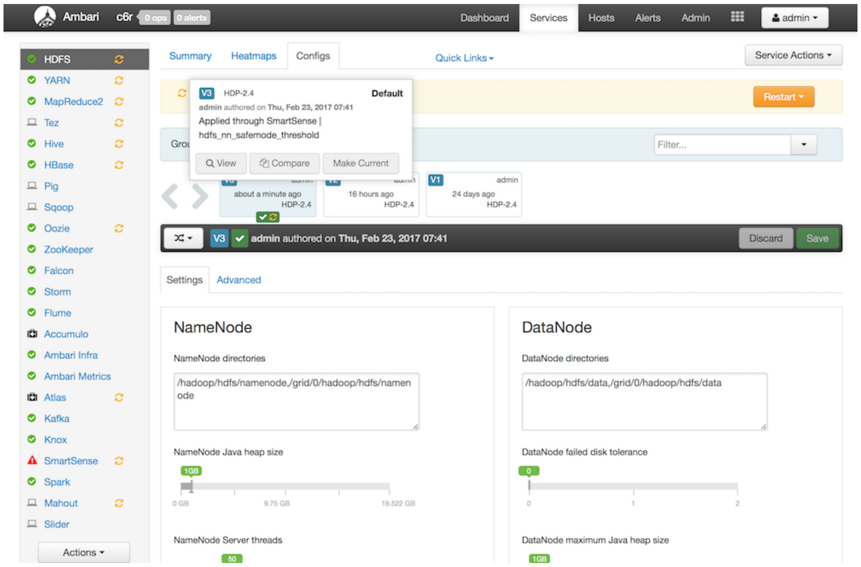
3. Review recommended changes.

4. Enter a comment in the **Change Notes** field. This comment will later allow you to track the Ambari configuration version created after applying a configuration.

5. Click on **Apply**.

6. You can optionally provide feedback for this recommendation and then click on **Submit Feedback**. Or you can opt out and click **I will provide later**. You can still provide feedback later, from the **History** page.

7. You can view the configuration change in Ambari configuration history.



V3 HDP-2.4 Default
admin authored on Thu, Feb 23, 2017 07:41
Applied through SmartSense | hdfs_nn_safemode_threshold

V3 admin authored on Thu, Feb 23, 2017 07:41

NameNode directories: /hadoop/hdfs/namenode,/grid/0/hadoop/hdfs/namenode

NameNode Java heap size: 1GB

NameNode Server threads: 50

DataNode directories: /hadoop/hdfs/data,/grid/0/hadoop/hdfs/data

DataNode failed disk tolerance: 0

DataNode maximum Java heap size: 1GB

Applying a Recommendation Manually

1. From the **Recommendations** page, click on the table row corresponding to the recommendation that you want to review.

2. Apply the recommendation manually.

3. Click on **Mark As Applied**.

4. Click on **I have** to confirm that you've applied the changes.

You can revert previously applied recommendations. This option is available on the **History** page.

2.5.4. Reviewing and Reverting Previously Applied Recommendations

To view the history of previously applied recommendations:

1. Click on the ••• menu and select **Show History**.

   The **History** tab allows you to review previously applied, ignored, deferred, and reverted recommendations, and, if needed, revert applied recommendations and review and reopen deferred and ignored recommendations.

2. To get more details about a specific recommendation, click on the ▶ .

3. You have an option to **Review and Revert**. If a recommendation has previously been reverted, this option is grayed out.

2.5.5. Exporting Recommendations as Excel Spreadsheet

You can export SmartSense recommendations to an Excel spreadsheet (XLSX file format).

To do that, click on the ⋯ menu and select **Export as Excel**. The spreadsheet will be downloaded to your default download location.

2.6. Accessing the Activity Explorer

The Activity Explorer includes an embedded instance of Apache Zeppelin, which hosts prebuilt notebooks that visualize cluster utilization data related to user, queue, job duration, and job resource consumption. To access the Activity Explorer:

> **Note**
>
> The quick link to the Activity Explorer is available only in Ambari 2.4 and later. If you are using **Ambari version earlier than 2.4**, you must access the Activity Explorer using the following URL: *http://<activity_explorer_host>:9060/*.

1. Navigate to the Ambari **Dashboard** and click the **SmartSense** service.

2. In the **Summary** tab, click **Quick Links** > **Activity Explorer**.

   This launches the Activity Explorer in a new browser tab.

3. Log in with your Activity Explorer admin credentials.

4. From the **Notebook** dropdown in the top toolbar, select the name of the notebook that you want to view.

   The following preconfigured notebooks are available:

   - Chargeback Dashboard [11]
   - HDFS Dashboard [12]
   - MapReduce & Tez Dashboard [13]
   - YARN Dashboard [13]

   Zeppelin organizes data in notebooks, where each notebook contains rows of paragraphs. Each paragraph visualizes the results of a single SQL statement using either a table, bar chart, pie chart, area chart, line chart, or scatter plot.

Once you opened a notebook, be aware of these three operations:

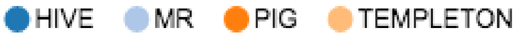
1. Since the notebooks represent a view of SmartSense utilization data at a specific point in time, they need to be refreshed. In order to refresh all of the data shown in all paragraph of a notebook, you need to:

   a. Hover over the row containing the notebook title, and a set of controls will appear.

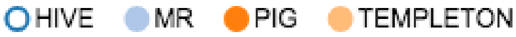
   b. Click on the ▷ button to "Run all paragraphs". The data for each paragraph in the notebook will be refreshed.

2. Top N paragraphs show the top 10 entries by default, but you can change this number by entering a new number in the **Top** input field and then typing enter.

3. Charts have interactive filters that let you select and deselect specific resources by clicking on the circle in the chart legend. For example, if there are four resources being displayed in a chart, and you only want to see four, you can click on a colored circle in the legend to filter it out:

   Once clicked, the inside of the circle will change to white, and the entry will not be displayed in the chart. For example, if you deselect "Hive", the legend will look like this:

2.6.1. Chargeback Dashboard

The Chargeback Dashboard helps operators understand which resources are being consumed and what costs are associated with these resources. This dashboard exposes five types of resources:

- **CPU Hours** (in hours) - The amount of CPU used by MapReduce and Tez jobs
- **Memory Hours** (in gigabytes) - The amount of memory consumed by MapReduce and Tez jobs, and length of consumption
- **Storage** (in gigabytes) - The amount of HDFS space being consumed
- **Data IO** (in gigabytes) - The amount of data read and written to HDFS
- **Network IO** (in gigabytes) - The amount of data sent and received over the cluster's network

Paragraph	Description
Chargeback Report	This paragraph lets you associate a financial cost with each of the five resources presented in the previous paragraph. Based on these per unit financial costs, you can see how much should be charged for each resource type. The report also sums up the charge per resource to a per user total, so it's easy to see how much should be charged back to that specific user for their total resource consumption. The goal is to show how much money each user should be charged for the cluster resources that they have consumed.

2.6.2. HDFS Dashboard

The HDFS Dashboard helps operators better understand how HDFS is being used and which users and jobs are consuming the most resources within the file system.

This dashboard includes the following paragraphs:

- File Size Distribution
- Top N Users with Small Files
- Top N Largest HDFS Users
- Average File Size
- HDFS File Size Distribution Trend
- HDFS Utilization Trend
- HDFS File Size Distribution Trend by User
- HDFS File Size Distribution Trend by User
- Jobs With High Number of HDFS Operations
- HDP 2.5: Jobs Creating Many HDFS Files
- Jobs With Large Amount of Data Written

Most of these paragraphs have titles that are self-explanatory. A few of them are described below to provide more context:

Paragraph	Description
File Size Distribution	For any large multi-tenant cluster, it’s important to identify and keep the proliferation of small files in check. The paragraph displays a pie chart showing the relative distribution of files by file size categorized by Tiny (0-10K), Mini (10K-1M), Medium (30M-128M), and Large (128M+) files. The goal is to show how dominant specific file size categories are within HDFS. If there are many small files, you can easily identify (in the next paragraph) who is contributing to those small files.
Top N Users with Small Files	Understanding how prevalent files of specific sizes are is helpful, but the next step is understanding who is responsible for creating those files. The goal of this paragraph is to show who is responsible for creating the majority of small files within HDFS.
Top N Largest HDFS Users	This paragraph helps you understand where all of the HDFS capacity is being consumed, and who is consuming it. The goal is to help you quickly understand which user or users are storing the most data in HDFS.
HDFS File Size Distribution Trend by User	Each “by User” paragraph allows you to see how an individual user’s file sizes are trending. This paragraph helps answer questions related to points in time where large or small files start becoming more or less prevalent for specific users, and can help measure the success of coaching users on Hadoop best practices.
HDP 2.5: Jobs Creating Many HDFS Files	When troubleshooting issues related to HDFS NameNode performance, it’s helpful to understand which jobs are creating the most files, and potentially putting the largest amount of load on the NameNode. In HDP 2.5, new counters have been added to track how many files are created by each YARN application. This is helpful in troubleshooting erroneous jobs that

Paragraph	Description
	are unintentionally creating hundreds of thousands, or even millions of files within HDFS.

2.6.3. MapReduce & Tez Dashboard

The MapReduce & Tez Dashboard was created to provide key information for workloads that use MapReduce or Tez for execution.

This dashboard includes the following paragraphs:

- Top N Longest Running Jobs
- Top N Resource Intensive Jobs
- Top N Resource Wasting Jobs
- Job Distribution By Type
- Top N Data IO Users
- CPU Usage By Queue
- Job Submission Trend By Day.Hour

Most of these paragraphs have titles that are self-explanatory. A few of them are described below to provide more context:

Paragraph	Description
Top N Resource Wasting Jobs	Resource wasting is calculated by calculating the difference between the memory asked for and the memory that was actually used. For example, if a job asks for 100 8GB containers but only uses 5GB per container, 3GB per container is considered wasted. This is calculated per job, and the top 10 are listed.
Job Submission Trend By Day.Hour	This paragraph shows the number of jobs submitted by day and hour with the notation being <day>.<hour>. For example: • Monday.1 - 1am on Monday • Monday.20 - 8pm on Monday The goal of this dashboard is to identify specific job submission hotspots during the week and day. You can use this information to identify the best time to schedule resource intensive jobs to execute.

2.6.4. YARN Dashboard

The YARN Dashboard provides key information for queue, application, container, and NodeManager host metrics.

This dashboard includes the following paragraphs:

- Application Runtime Duration by Queue
- Top N Applications by Number of Containers Requested

- Top N Applications by Number of Containers Failed
- Top N Hosts by Number of Containers Executed
- Top N Hosts by Number of Application Failures
- Top N Hosts by Localization Time
- Top N Hosts by Container Launch Delay

Most of these paragraphs have titles that are self-explanatory. One of them is described below to provide more context:

Paragraph	Description
Top N Applications by Number of Containers Failed	This paragraph shows the top jobs with the highest number of failed containers and the reason for each failure, so that you can quickly identify which containers failed and why.

3. Configuring SmartSense

This chapter guides you through common configuration tasks such as changing capture levels, configuring data anonymization rules, and changing server and agent configurations.

3.1. Configuring Data Anonymization Rules

As data is captured, specific types of data are automatically anonymized. By default, IP addresses and the domain component of host names are anonymized. To customize these anonymization rules, follow these steps:

1. Navigate to the Ambari **Dashboard** and click the **SmartSense** service.
2. Click the **Config** tab.
3. Navigate to the **Data Capture** section.
4. Add the new anonymization rule (or change the existing rule) by following the details provided in Anonymization Rule Types.

3.1.1. Anonymization Rule Types

Anonymization rules define regular expressions to anonymize sensitive data (like IP addresses, and so on). Each rule uses JSON format to define what to match and the value to replace.

You can define the following types of anonymization rules:

- Pattern-based - Anonymize data by pattern, using the ***extract*** field to match and extract content to anonymize.
- Property-based - Anonymize structured content. The supported formats are: XML, property, ini, and YAML files.
- XPath-based - Anonymize XML data using XPATH.
- JSONPath-based - Anonymize JSON data using JSONPATH.

In addition, there are **domain-based rules** that can be used to anonymize domain names. They are a special case of pattern rules where the anonymization pattern is build from local host FQDN. The domain-based rules cannot be customized.

For a detailed description of all the fields required to define annonymization rules, refer to Fields Used for Defining Anonymization Rules.

Note

Anonymization rule formats vary between different SmartSense versions. Make sure that you consult the documentation that matches your SmartSense version.

3.1.1.1. Fields Used for Defining Anonymization Rules

To define anonymization rules, use the following fields:

Table 3.1.

Field	Description
name	Provides a descriptive name for data anonymized by the rule. It has to be unique across all rules.
description	Provides a description for the rule.
rule_id	Defines the class of rules the current rule belongs to. The supported rule IDs are: *PATTERN*, *PROPERTY*, *XPATH*, JSONPATH. This parameter is case-insensitive.
patterns	Defines a list of data patterns to be anonymized. It is applicable only to *Pattern* rules, where rule_id=PATTERN. These patterns are matched in a case-insensitive manner, which means that the following pattern `keystore.pass=([^\\s]*)` matches with any of the following values: • keystore.pass=123 • KeyStore.Pass=123 • KEYSTORE.PASS=123

extract	Specifies a pattern to extract data matched through the list of patterns. The extract pattern is matched in a case-insensitive manner.
	For example, in order to anonymize the **oozie.https.keystore.pass** password, the following pattern and extract values are used:
	`"patterns": ["oozie.https.keystore.pass=([^\\s]*)"]`
	`"extract": "=([^\\s]*)",`
	This pattern is matched with values such as `oozie.https.keystore.pass=1234`.
	The extract pattern is used to extract and anonymize only the values after the = (which in this example is `1234`). The `[^\\s]*` denotes all non-whitespace characters, and the capturing group `( )` is used the exclude = from the anonymized value.
	If the extract pattern is not configured, the entire value matched with the pattern is anonymized (which in this example is `oozie.https.keystore.pass=1234`), regardless of capturing groups used in the patterns.

properties	Specifies a list of property name patterns to anonymize; these are case-insensitively matched. It is applicable only to *Property* rules.
parentNode	This field is applicable to property anonymization in XML files. It allows you to define the parent node of the property that you want to anonymize. By default, *parentNode* is set to `"parentNode": "property"`, because typically the XML block to anonymize has the parent node *property*, like in the following example:

Field	Description

	`<property>`
	`  <name>fs.s3a.proxy.password</name>`
	`  <value>Abc7j*4$aTh</value>`
	`  <description>Password for authenticating with`
	` proxy server.</description>`
	`</property>`
	
	For example, you can anonymize `main.ldapRealm.contextFactory.systemPassword` in the following XML block that has a parent node called *param* by setting `"parentNode": "param"` in the anonymization rule:
	
	`<param>`
	`  <name>main.ldapRealm.contextFactory.`
	`systemPassword</name>`
	`  <value>pass</value>`
	`</param>`
	
	The rule to anonymize the above content configures *param* as the root tag `"parentNode": "param"`:
	
	`  {`
	`    "name": "KNOX LDAP Password",`
	`    "rule_id": "Property",`
	`    "properties": ["main.ldapRealm.contextFactory.`
	`systemPassword"],`
	`    "include_files": ["topologies/*.xml"],`
	`    "action" : "REPLACE",`
	`    "parentNode": "param",`
	`    "replace_value": "Hidden"`
	`  }`

action	The supported actions are: *ANONYMIZE*, *DELETE*, *REPLACE*. The *action* value is **not** case sensitive, so *Anonymize* or *delete* are also accepted values. ANONYMIZE action encrypts the data using the key indicated by *shared* flag, DELETE deletes the data, and REPLACE replaces the data with a predefined value, which can be customized using *replace_value*.
replace_value	This field is used by the *REPLACE* action to specify a replacement for the data to anonymize. The default value is *Hidden*.
shared	Indicates which key to use for anonymization (shared or private). This value is used when the anonymization action is set to *ANONYMIZE*. It is a boolean type property (true/false). If set to true - the Hortonworks support team can unmask data if needed for diagnostic purposes; for example, host names and IP addresses for resolving issues on specific hosts or communication between hosts. Note that unmasked data is not stored in Hortonworks repositories; it is discarded as soon as the analysis finishes. The default value is true. Rules configured with `shared = false` cannot be unmasked by Hortonworks (and in some cases might become a roadblock for support case analysis.)
include_files	Specifies a list of *glob* file patterns for which the rule applies. If not configured, the rule is applicable to all files.
exclude_files	Specifies a list of *glob* file patterns which are excluded from anonymization. If not configured, no file is excluded from the rule application.

Field	Description
enabled	A flag (true/false) which specifies if the rule is enabled to be executed. By default, it is set to *true*.

3.1.1.2. Pattern-Based Anonymization Rules

Write pattern-based rules to anonymize data by pattern, using the *extract* pattern to extract content to anonymize.

Required and Optional Fields

- name
- description (optional)
- rule_id (should be set to PATTERN)
- patterns
- extract (optional)
- include_files (optional)
- exclude_files (optional)
- action (optional, default value is ANONYMIZE)
- replace_value (optional, applicable only when action=REPLACE)
- shared (optional, default value is *true*)
- enabled (optional, default value is *true*)

For more information on each field, refer to Fields Used for Defining Anonymization Rules.

Rule Definition Example (without *extract*)

```
    {
      "name": "EMAIL",
      "rule_id": "Pattern",
      "patterns": ["(?<![a-z0-9._%+-])[a-z0-9._%+-]+@[a-z0-9.-]+\\.[a-z]{2,6}
(?![a-z0-9._%+-])$?",
      "shared": false
    }
```

The content of the input file *version.txt* is:

```
Hadoop 2.7.3.2.5.0.0-1245
Subversion git@github.com:hortonworks/hadoop.git -r
 cb6e514b14fb60e9995e5ad9543315cd404b4e59
Compiled by jenkins on 2016-08-26T00:55Z
```

The content of the output file *version.txt*, with anonymized email address, is:

```
Hadoop 2.7.3.2.5.0.0-1245
Subversion ‡qpe@unqfay.mjp‡:hortonworks/hadoop.git -r
 cb6e514b14fb60e9995e5ad9543315cd404b4e59
Compiled by jenkins on 2016-08-26T00:55Z
```

Rule Definition Example (with *extract*)

```
    {
      "name": "KEYSTORE",
      "rule_id": "Pattern",
      "patterns": ["oozie.https.keystore.pass=([^\\s]*)",
 "OOZIE_HTTPS_KEYSTORE_PASS=([^\\s]*)"],
      "extract": "=([^\\s]*)",
      "include_files": ["java_process.txt", "pid.txt", "ambari-agent.log",
 "java_process.txt", "oozie-env.cmd"],
      "shared": false
    }
```

The content of the input file *oozie-env.cmd* is:

```
oozie.https.keystore.pass=abcde
set OOZIE_HTTPS_KEYSTORE_PASS=12345
```

To anonymize the content of the input file, the following anonymization patterns configured in the rule will be used:

```
"oozie.https.keystore.pass=([^\\s]*)", "OOZIE_HTTPS_KEYSTORE_PASS=([^\\s]*)"
```

`oozie.https.keystore.pass=([^\\s]*)` and `OOZIE_HTTPS_KEYSTORE_PASS=([^\\s]*)` match with `oozie.https.keystore.pass=abcde` and `OOZIE_HTTPS_KEYSTORE_PASS=12345` respectively.

Next, the extract pattern `"=([^\\s]*)` is used to identify *12345* and *abcde*, which are the values to be anonymized.

The content of the output file *oozie-env.cmd* is:

```
oozie.https.keystore.pass=‡vvdwa‡
set OOZIE_HTTPS_KEYSTORE_PASS=‡zdowg‡
```

The values of `oozie.https.keystore.pass` and `OOZIE_HTTPS_KEYSTORE_PASS` have been anonymized.

For more examples, refer to Examples of Pattern-Based Anonymization Rules.

3.1.1.2.1. Examples of Pattern-Based Anonymization Rules

This section includes examples of commonly used pattern-based anonymization rules.

Example 1: Mask by pattern across all log files, without *extract* pattern

To mask all email addresses in all log files, use the following rule definition:

```
{
  "name": "EMAIL",
  "rule_id": "Pattern",
  "patterns": ["(?<![a-z0-9._%+-])[a-z0-9._%+-]+@[a-z0-9.-]+\\.[a-z]{2,6}(?!
[a-z0-9._%+-])"],
  "include_files": ["*.log*"],
  "shared": false
}
```

Example 2: Mask by pattern across all log files, with extract pattern

To mask encryption keys, logged in the following format *Key=12..* with a value consisting of 64 hexadecimal characters, use the following rule definition:

```
{
  "name": "ENC_KEYS",
  "rule_id": "Pattern",
  "patterns": ["Key=[a-f\\d]{64}\\s"],
  "extract": "=([a-f\\d]{64})",
  "include_files": ["*.log*"],
  "shared": false
}
```

Input data, *test.log* is:

```
encryption key=
1234567890adc1234567aaabc1234567890adc1234567aaabc12345678901234 for keystore
derby.system.home=null
```

Output data, *test.log*, with the encryption keys anonymized, is:

```
encryption key=
‡8697685738fnx1736987qigyx7611731027yds0096404hlsph91727138403654‡ for
 keystore
derby.system.home=null
```

Example 3: Mask by pattern across all files, except a few files

To mask email addresses in all files, except *hdfs-site.xml* and *.property* files, use the following rule definition:

```
{
  "name": "EMAIL",
  "rule_id": "Pattern",
  "patterns": ["(?<![a-z0-9._%+-])[a-z0-9._%+-]+@[a-z0-9.-]+\\.[a-z]{2,6}(?!
[a-z0-9._%+-])"],
  "exclude_files" : ["*.properties", "hdfs-site.xml"],
  "shared": false
}
```

Input data, *version.txt*, is:

```
Hadoop 2.7.3.2.5.0.0-1245
Subversion git@github.com :hortonworks/hadoop.git -r
 cb6e514b14fb60e9995e5ad9543315cd404b4e59
Compiled by jenkins on 2016-08-26T00:55Z
```

Output file *version.txt*, with an anonymized email address, is:

```
Hadoop 2.7.3.2.5.0.0-1245
Subversion ‡qpe@unqfay.mjp‡ :hortonworks/hadoop.git -r
 cb6e514b14fb60e9995e5ad9543315cd404b4e59
Compiled by jenkins on 2016-08-26T00:55Z
```

3.1.1.3. Property-Based Anonymization Rules

Property-based rules anonymize structured content. The supported formats are: XML, property, ini, and YAML files.

Required and Optional Fields

- name
- description (optional)

- rule_id (should be set to PROPERTY)
- properties
- parentNode (optional, applicable only for XML, default value is "property")
- include_files
- exclude_files (optional)
- action (optional, default value is ANONYMIZE)
- replace_value (optional, applicable only when action=REPLACE)
- shared (optional, default value is *true*)
- enabled (optional, default value is *true*)

For more information on each field, refer to Fields Used for Defining Anonymization Rules.

Rule Definition Example

```
    {
      "name": "PASSWORDS",
      "rule_id": "Property",
      "properties": [".*password.*", ".*awsAccessKeyId.*"],
      "include_files": ["*.xml", "*.properties", "*.yaml", "*.ini"],
      "exclude_files" : ["capacity-scheduler.xml"],
      "action" : "REPLACE",
      "replace_value": "Hidden"
    }
```

The following examples show how the rule defined above anonymizes specific password-related properties in XML, property, ini, and YAML files.

- **XML file content:**

  ```
  <property>
    <name>fs.s3a.proxy.password</name>
    <value>Abc7j*4$aTh</value>
    <description>Password for authenticating with proxy server.</description>
  </property>
  ```

  The XML file content, with password value anonymized:

  ```
  <property>
    <name>fs.s3a.proxy.password</name>
    <value>Hidden</value>
    <description>Password for authenticating with proxy server.</description>
  </property>
  ```

- **Property file content:**

  ```
  javax.jdo.option.ConnectionPassword=pswd
  ```

  The property file content, with password value anonymized:

  ```
  javax.jdo.option.ConnectionPassword=Hidden
  ```

- **Ini file content:**

```
connection_password=pswd
```

The ini file content, with password value anonymized:

```
connection_password=Hidden
```

- **YAML file content:**

```
"metrics_collector:\n" +
                "  truststore.path : \"/etc/security/clientKeys/all.jks\"\n"
 +
                "  truststore.type : \"jks\"\n" +
                "  truststore.password : \"bigdata\"\n"
```

The YAML file content, with password value anonymized:

```
"metrics_collector:\n" +
                "  truststore.path : \"/etc/security/clientKeys/all.jks\"\n"
 +
                "  truststore.type : \"jks\"\n" +
                "  truststore.password : Hidden\n"
```

For more examples, refer to Examples of Property-Based Anonymization Rules.

3.1.1.3.1. Examples of Property-Based Anonymization Rules

This section includes examples of commonly used property-based anonymization rules.

Example 1: Mask one configuration parameter in multiple files

Rule definition example:

```
{
  "name": "JPA_PASSWORD",
  "rule_id": "Property",
  "properties": ["oozie.service.JPAService.jdbc.password"],
  "include_files": ["oozie-site.xml", "sqoop-site.xml"],
  "action" : "REPLACE",
  "replace_value": "Hidden"
}
```

This rule anonymizes the value of `oozie.service.JPAService.jdbc.password` in oozie-site.xml and sqoop-site.xml:

Input data, *sqoop-site.xml*:

```
    <property>
      <name>oozie.service.JPAService.jdbc.px</name>
      <value>at@!_*rue</value>
    </property>
```

Output data, *sqoop-site.xml*, with anonymized `oozie.service.JPAService.jdbc.px` parameter value:

```
    <property>
      <name>oozie.service.JPAService.jdbc.px</name>
      <value>Hidden</value>
    </property>
```

Example 2: Mask multiple configuration parameters in multiple files

Rule definition example:

```
{
  "name": "JDBC_JPA_PASSWORDS",
  "rule_id": "Property",
  "properties": ["oozie.service.JPAService.jdbc.password", "javax.jdo.option.
ConnectionPassword"],
  "include_files": ["oozie-site.xml", "sqoop-site.xml", "hive-site.xml"],
  "action" : "REPLACE",
  "replace_value": "Hidden"
}
```

Example 3: Mask a configuration that matches a pattern

Rule definition example:

```
{
  "name": "GLOBAL_JDBC_PASSWORDS",
  "rule_id": "Property",
  "properties": [".*password"],
  "include_files": ["*.xml"],
  "action" : "REPLACE",
  "replace_value": "Hidden"
}
```

Input data:

ssl-server.xml

```
    <property>
      <name>ssl.server.keystore.keypassword</name>
      <value>big123!*</value>
    </property>
```

ssl-client.xml

```
    <property>
      <name>ssl.client.keystore.password</name>
      <value>NBg7j*4$aTh</value>
    </property>
```

Output data:

Anonymized *ssl-server.xml*

```
    <property>
      <name>ssl.server.keystore.keypassword</name>
      <value>Hidden</value>
    </property>
```

Anonymized *ssl-client.xml*

```
    <property>
      <name>ssl.client.keystore.password</name>
      <value>Hidden</value>
    </property>
```

3.1.1.4. XPath-Based Anonymization Rules

XPath-based rules anonymize XML data using XPath.

Required and Optional Fields

- name
- description (optional)
- rule_id (should be set to XPATH)
- paths
- include_files
- exclude_files (optional)
- action (optional, default value is ANONYMIZE)
- replace_value (optional, applicable only when action=REPLACE)
- shared (optional, default value is *true*)
- enabled (optional, default value is *true*)

For more information on each field, refer to Fields Used for Defining Anonymization Rules.

Rule Definition Example

```
  {
    "name": "XPATH_RULE",
    "rule_id": "XPATH",
    "paths": ["/data/record[1]/value"],
    "include_files": ["*test_config.xml"],
    "shared": true
  }
```

Sample Input XML Data

```
<data>
    <record>
        <name>password</name>
        <value>valueToAnonymize</value>
    </record>
    <record>
        <name>name</name>
        <value>value</value>
    </record>
</data>
```

Sample Output XML Data (After Anonymization)

```
<data>
    <record>
        <name>password</name>
        <value>¶smfz923swc¶</value>
    </record>
    <record>
        <name>name</name>
        <value>value</value>
    </record>
</data>
```

For more examples, refer to Examples of XPath-Based Anonymization Rules.

You can use this reference documentation for XPath.

3.1.1.4.1. Examples of XPath-Based Anonymization Rules

This section includes examples of commonly used XPath-based anonymization rules.

Example 1: Rule with nested XML structure

Rule definition example:

```
{
  "name": "NESTED_XPATH_RULE",
  "rule_id": "XPATH",
  "paths": ["/configs/properties/passwd"],
  "include_files": ["*config.xml"],
  "shared": true
}
```

Input data:

```
<?xml version="1.0" encoding="UTF-8" ?>
<configs>
    <properties>
        <user>abc</user>
        <passwd>1234</passwd>
    </properties>
</configs>
```

Output data (after anonymization):

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<configs>
    <properties>
        <user>abc</user>
        <passwd>¶9165¶</passwd>
    </properties>
</configs>
```

Example 2: Rule with XML array structure

Rule definition example:

```
{
  "name": "ARRAY_XPATH_RULE",
  "rule_id": "XPATH",
  "paths": ["/configs/properties[2]/passwd"],
  "include_files": ["*config.xml"],
  "shared": true
}
```

Input data:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<configs>
    <properties>
        <database>mysql</database>
        <url>user@host:port</url>
    </properties>
    <properties>
        <user>abc</user>
        <passwd>1234</passwd>
    </properties>
</configs>
```

Output data (after anonymization):

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<configs>
    <properties>
        <database>mysql</database>
        <url>user@host:port</url>
    </properties>
    <properties>
        <user>abc</user>
        <passwd>¶9165¶</passwd>
    </properties>
</configs>
```

Example 3: Rule with XML map structure

Rule definition example:

```
{
  "name": "MAP_XPATH_RULE",
  "rule_id": "XPATH",
  "paths": ["/configs/properties/passwd"],
  "include_files": ["*config.xml"],
  "shared": true
}
```

Input data:

```
<?xml version="1.0" encoding="UTF-8" ?>
<configs>
    <db>mysql</db>
    <properties>
        <user_name>sa</user_name>
        <passwd>sa_pass</passwd>
    </properties>
    <pooli_size>32</pooli_size>
    <timeout>10</timeout>
</configs>
```

Output data (after anonymization):

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><configs>
    <db>mysql</db>
    <properties>
        <user_name>sa</user_name>
        <passwd>¶vm_wtto¶</passwd>
    </properties>
    <pooli_size>32</pooli_size>
    <timeout>10</timeout>
</configs>
```

Example 4: Rule to mask all array elements

Rule definition example:

```
{
  "name": "ALL_FROM_ARRAY_XPATH_RULE",
  "rule_id": "XPATH",
  "paths": ["/configs/properties[*]/passwd"],
  "include_files": ["*config.xml"],
  "shared": true
}
```

Input data:

```
<?xml version="1.0" encoding="UTF-8" ?>
<configs>
    <properties>
        <user>abc1</user>
        <passwd>pass1</passwd>
    </properties>
    <properties>
        <user>abc2</user>
        <passwd>pass2</passwd>
    </properties>
</configs>
```

Output data (after anonymization):

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<configs>
    <properties>
        <user>abc1</user>
        <passwd>¶smfz7¶</passwd>
    </properties>
    <properties>
        <user>abc2</user>
        <passwd>¶smfz8¶</passwd>
    </properties>
</configs>
```

Example 5: Rule to mask some array elements which have *passwd*

Rule definition example:

```
{
  "name": "SOME_FROM_ARRAY_XPATH_RULE",
  "rule_id": "XPATH",
  "paths": ["/configs/properties[passwd]/passwd"],
  "include_files": ["*config.xml"],
  "shared": true
}
```

Input data:

```
<?xml version="1.0" encoding="UTF-8" ?>
<configs>
    <properties>
        <user>abc1</user>
        <passwd1>pass1</passwd1>
    </properties>
    <properties>
        <user>abc2</user>
        <passwd2>pass2</passwd2>
    </properties>
    <properties>
        <user>abc3</user>
        <passwd>pass3</passwd>
    </properties>
</configs>
```

Output data (after anonymization):

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<configs>
    <properties>
        <user>abc1</user>
        <passwd1>pass1</passwd1>
    </properties>
    <properties>
        <user>abc2</user>
        <passwd2>pass2</passwd2>
    </properties>
    <properties>
        <user>abc3</user>
        <passwd>¶smfz9¶</passwd>
    </properties>
</configs>
```

3.1.1.5. JSONPath-Based Anonymization Rules

JSONPath-based rules anonymize JSON data using JSONPath.

Required and Optional Fields

- name
- description (optional)
- rule_id (should be set to JSONPATH)
- paths
- include_files
- exclude_files (optional)
- action (optional, default value is ANONYMIZE)
- replace_value (optional, applicable only when action=REPLACE)
- shared (optional, default value is *true)*
- enabled (optional, default value is *true*)

For more information on each field, refer to Fields Used for Defining Anonymization Rules.

Rule Definition Example

```
  {
    "name": "JSONPATH_RULE",
    "rule_id": "JSONPATH",
    "paths": ["$.users[0].password"],
    "include_files": ["*test_config.json"],
    "shared": true
  }
```

Sample Input JSON Data

```
{
  "users": [
    {
      "name": "Logsearch Admin",
      "username": "admin",
      "password": "testdata"
    },
    {
      "name": "Admin",
      "username": "admin",
      "password": "test data"
    }
  ]
}
```

Sample Output JSON Data (After Anonymization)

```
{
  "users": [
    {
      "name": "Logsearch Admin",
      "username": "admin",
      "password": "¶smfvvcz9¶"
    },
    {
      "name": "Admin",
      "username": "admin",
      "password": "test data"
    }
  ]
}
```

For more examples, refer to Examples of JSONPath-Based Anonymization Rules.

You can use this reference documentation for JSONPath.

3.1.1.5.1. Examples of JSONPath-Based Anonymization Rules

This section includes examples of commonly used JSONPath-based anonymization rules.

Example 1: Rule with nested JSON elements

Rule definition example:

```
{
   "name": "NESTED_JSONPATH_RULE_1",
   "rule_id": "JSONPATH",
   "paths": ["$.configs.properties.passwd"],
   "include_files": ["*config.json"],
   "shared": true
 }
```

Input data:

```
{
    "configs": {
        "properties":
            {
                "user": "abc",
                "passwd": "12!@"
            }
    }
}
```

Output data (after anonymizarion):

```
{
  "configs": {
    "properties": {
      "user": "abc",
      "passwd": "¶91!@¶"
    }
  }
}
```

Example 2: Rule with indexed JSON array objects

Rule definition example:

```
  {
    "name": "ARRAY_JSONPATH_RULE",
    "rule_id": "JSONPATH",
    "paths": ["$.configs.properties[1].passwd"],
    "include_files": ["config.json"],
    "shared": true
  }
```

Input data:

```
{
    "configs": {
        "properties": [
            {
                "database": "mysql",
                "url": "user@host:port"
            },
            {
                "user": "abc",
                "passwd": "12!@"
            }
 ]
    }
}
```

Output data (after anonymization):

```
{
  "configs": {
    "properties": [
      {
        "database": "mysql",
        "url": "user@host:port"
      },
      {
        "user": "abc",
        "passwd": "¶91!@¶"
      }
    ]
  }
}
```

Example 3: Rule with JSON map

Rule definition example:

```
    {
      "name": "MAP_JSONPATH_RULE",
      "rule_id": "JSONPATH",
      "paths": ["$.properties.passwd"],
      "include_files": ["*config.json"],
      "shared": true
    }
```

Input data:

```
 {
   "db":"mysql",
   "properties":
   {
     "user_name":"sa",
     "passwd":"sa_pass"
   },
   "pooli_size":32,
   "timeout":10
 }
```

Output data (after anonymization):

```
{
  "db": "mysql",
  "properties": {
    "user_name": "sa",
    "passwd": "¶vm_wtto¶"
  },
  "pooli_size": 32,
  "timeout": 10
}
```

Example 4: Rule to mask all JSON objects from list

Rule definition example:

```
 {
   "name": "ALL_FROM_ARRAY_JSONPATH_RULE",
   "rule_id": "JSONPATH",
   "paths": ["$.configs.properties[*].passwd"],
   "include_files": ["*config.json"],
   "shared": true
 }
```

Input data:

```
{
   "configs": {
       "properties": [
           {
               "user": "abc1",
               "passwd": "pass1"
           },
           {
               "user": "abc2",
               "passwd": "pass2"
           }
   ]
   }
}
```

Output data (after anonymization):

```
{
 "configs": {
   "properties": [
     {
       "user": "abc1",
       "passwd": "¶smfz7¶"
     },
     {
       "user": "abc2",
       "passwd": "¶smfz8¶"
     }
   ]
 }
}
```

3.2. Configuring Bundle Upload

SmartSense Gateway is automatically configured with HTTPS so you don't normally need to perform this configuration. However, if a specific custom configuration is required by your corporate network firewall policies, you can use these instructions to configure SmartSense Gateway to upload bundles by using HTTPS:

- Configuring the Gateway to Use HTTPS [32]

3.2.1. Configuring the Gateway to Use HTTPS

You can configure the gateway to use HTTPS to upload bundles to Hortonworks by using the connectivity and configuration details available in this article: https://support.hortonworks.com/s/article/SmartSense-Gateway-setup (To view this article, you need a valid Hortonworks support account).

To use an authenticated HTTP or HTTPS proxy to upload bundles to Hortonworks, follow these steps:

Note

If you would like to set up HTTPS proxy, you must contact Hortonworks Support.

1. On the SmartSense Gateway host, edit the `/etc/hst/conf/gateway/hst-gateway.ini` file and supply the appropriate values for your environment:

   ```
   ; All proxy configurations are applicable only for HTTPS provider type
   ;#set to true#to#set up#a#proxy#between#gateway#and#SmartSense#environment
   ;default:false
   provider.https.proxy.enabled=true
   ;#fully#qualified#proxy#hostname
   provider.https.proxy.hostname=your.proxy.host
   ;#proxy#port#that#will#be#used#by#gateway#for#outbound#access
   provider.https.proxy.port=3128
   ;#supported proxy#types#:#HTTP#/#HTTPS#[default:HTTP]
   provider.https.proxy.type=HTTP
   ; supported proxy authentication #types#:#NONE#/#BASIC#/#DIGEST#[default:NONE]
   provider.https.proxy.auth.type=BASIC
   ;#proxy#username#for#identified#auth.type
   provider.https.proxy.auth.username=proxyuser
   ;#proxy#password#for#identified#auth.type
   provider.https.proxy.auth.password=proxypassword
   ;#[optional]#any#additional#proxy#setup#parameters
   ; use#"|" to#separate#multiple#parameters
   ;#for example:#digest#requires#setting#parameters#such as
   ;#realm=default|nonce=12GHtqeZA!7Ke43
   provider.https.proxy.auth.parameters=
   ```

2. After you update the configuration file, restart the SmartSense Gateway:

   ```
   hst gateway restart
   ```

3.3. Configuration Guidelines

The following sections describe configuration properties related to SmartSense components and provide tuning guidelines.

- HST Server [34]
- HST Agent [41]
- SmartSense Gateway [44]
- Activity Analyzer [45]
- Activity Explorer [50]

The "Default Value" of a parameter is listed as "(no value)" if by default the parameter is set to an empty value. In order to set the parameter, find it in the Ambari configuration tab listed and set it to a desired value.

The "Default Value" of a parameter is listed as "(unspecified)" if the parameter is unset by default. In order to set the parameter, you must add it as a custom configuration.

3.3.1. HST Server

The following configuration properties are available for HST server:

Table 3.2. HST Server Configuration Properties

Property Name	Description	Where to Configure	Guidelines
customer.smartsense.id	Your SmartSense ID uniquely identifies your account. You can obtain it from Hortonworks Support. This is a mandatory field during SmartSense installation. **Type:** string **Default Value:** (unspecified)	**Ambari Config:** Basic **Config File:** /etc/hst/conf/hst-server.ini	You can obtain your existing SmartSense ID from the Hortonworks Support portal.
customer.account.name	The name of your organization as it is registered with Hortonworks Support. This is a mandatory field during SmartSense setup and it is one of the important identifiers for clusters belonging to the same customer.. **Type:** string **Default Value:** (unspecified)	**Ambari Config:** Basic **Config File:** /etc/hst/conf/hst-server.ini	You must enter the organization name exactly as it is registered in the Hortonworks Support portal.
customer.notification.email	Email address used to send bundle upload and recommendation availability notifications. This is a mandatory field during SmartSense setup. **Type:** string **Default Value:** (unspecified)	**Ambari Config:** Basic **Config File:** /etc/hst/conf/hst-server.ini	Check your junk mailbox in case you do not receive notifications.
customer.enable.flex.subscription	Enables Flex Subscription for the cluster. **Type:** boolean **Default Value:** false	**Ambari Config:** Basic **Config File:** /etc/hst/conf/hst-server.ini	Enable only if you have a valid Flex Subscription ID obtained from Hortonworks Support.
customer.flex.subscription.id	Your Flex Subscription ID obtained from Hortonworks Support. Flex Subscription offers flexible support subscription. **Type:** string **Default Value:** (unspecified)	**Ambari Config:** Basic **Config File:** /etc/hst/conf/hst-server.ini	Contact Hortonworks Support to obtain a Flex Subscription ID. When passing the ID, you must also enable flex subscription.
server.storage.dir	Directory used by HST server for storing bundles. **Type:** string **Default Value:** /var/lib/smartsense/hst-server/data	**Ambari Config:** Basic **Config File:** /etc/hst/conf/hst-server.ini	Use a non-root partition for hosting this directory. For reliable operations, we recommend that you have at least 10GB of free space on that partition.
server.tmp.dir	Directory used by HST server for temporary operations. **Type:** string	**Ambari Config:** Basic **Config File:**	Use a non-root partition for hosting this directory. For reliable operations, we recommend that you have

Property Name	Description	Where to Configure	Guidelines
	Default Value: /var/lib/smartsense/hst-server/tmp	/etc/hst/conf/hst-server.ini	at least 10GB of free space on that partition.
server.port	Port to access the HST server web interface and API. **Type:** int **Default Value:** 9000	**Ambari Config:** Operations **Config File:** /etc/hst/conf/hst-server.ini	This port is internally used for HST operations. Change only if port 9000 is already in use or cannot be unblocked, or if SSL needs a different port. This has no impact on SmartSense Ambari View.
server.max.heap	Maximum heap size (in MB) allocated for the HST server process. **Type:** int **Default Value:** 2048	**Ambari Config:** Advanced > Advanced hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Usually 2048 MB is sufficient for clusters up to 500 nodes. Tuning might help if cluster has more than 500 nodes or if you encounter OOM errors on the server side.
agent.request.processing.timeout	Agent request processing timeout (in seconds). This usually indicates the total time for agent capture to finish. **Type:** int **Default Value:** 7200	**Ambari Config:** Operations **Config File:** /etc/hst/conf/hst-server.ini	Increase the capture timeout to more than 120 minutes if you are capturing more than 4-5 services or have huge logs for support bundle captures. You may also want to increase this if captures are timing out.
agent.request.syncup.interval	Interval (in seconds) after submitting data collection request in which all the data collections requests from various agents are treated as part of same bundle. In other words, this determines the maximum time for any agent to sync back with server on capture request. If multiple agents join data collection process within this interval, they will be treated as part of same bundle. If any agent joins data collection after this interval, it will be treated as another bundle. **Type:** int **Default Value:** 180	**Ambari Config:** Operations **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for most clusters. In cases where Ambari server and agent requests are slow and SmartSense bundle collection shows unreported agents in every bundle collection, increasing this interval may help.
client.threadpool.size.max	Server thread pool size to handle bundle requests. **Type:** int **Default Value:** 40	**Ambari Config:** Operations **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for most clusters Consider increasing this property if you see multiple agent upload requests timing out on a large cluster with more than 500 nodes.
gateway.host	Fully qualified domain name of the host where the SmartSense Gateway process has been deployed and is running.	**Ambari Config:** Gateway **Config File:**	Keep the default if your HST server has outbound internet access to reach smartsense.hortonworks.com. Otherwise, set up a

Property Name	Description	Where to Configure	Guidelines
	Type: string **Default Value:** embedded	/etc/hst/conf/hst-server.ini	separate standalone gateway which has outbound access.
gateway.port	Port on which the SmartSense Gateway is listening and through which data is transferred. It is set up with two-way SSL. This port is not applicable for embedded gateway. **Type:** int **Default Value:** (no value)	**Ambari Config:** Gateway **Config File:** /etc/hst/conf/hst-server.ini	This port is used for internal communication between the gateway and HST server. Change this only if this port is already in use or cannot be unblocked. Note that if you change this port, you must update a similar property in the gateway.
gateway.registration.port	Port which is used by clients to register with the gateway. Data is not transferred through this port. It is set up with one-way SSL. This port is not applicable for embedded gateway. **Type:** int **Default Value:** (no value)	**Ambari Config:** Gateway **Config File:** /etc/hst/conf/hst-server.ini	This port is used for internal communication between the gateway and HST server. Change only if this port is already in use or cannot be unblocked. Note that if you change this port, you must update a similar property in the Gateway.
hst_log_dir	Directory where SmartSense log files are created. **Type:** string **Default Value:** /var/log/hst	**Ambari Config:** Advanced > Advanced hst-log4j **Config File:** /etc/hst/conf/log4j.properties	Changing this setting is usually not recommended. If you change it, you must provide read/write/create permissions for this directory to Ambari Agent user.
hst_max_file_size	Maximum size of SmartSense HST log files. **Type:** int **Default Value:** 30	**Ambari Config:** Advanced > Advanced hst-log4j **Config File:** /etc/hst/conf/log4j.properties	Default value is suitable for most clusters. Check available storage capacity before updating this property.
hst_max_backup_index	Maximum number of HST log files. **Type:** int **Default Value:** 10	**Ambari Config:** Advanced > Advanced hst-log4j **Config File:** /etc/hst/conf/log4j.properties	Increase this number to keep the record of older logs. Check available storage capacity before updating this property.
java.home	Path to the JAVA home for HST server. **Type:** string **Default Value:** (no value)	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	This setting is automatically configured from Ambari env settings and usually there is no reason to change it. We recommend that you use the latest 1.7/1.8 JAVA versions with up-to-date security updates. For more security we recommend

Property Name	Description	Where to Configure	Guidelines
			that you have unlimited JCE policy installed.
derby.system.home	Home directory path for Derby database used internally by HST server. **Type:** string **Default Value:** /var/lib/smartsense/hst-server/hstDB	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for most clusters. This property should only be changed during the HST server setup. If you change thus after HST server is already set up, remember to make a backup and move existing data to the new location.
bundle.monitor.interval	Interval (in seconds) determining how often a bundle is checked for completeness. After every interval, data uploaded from agents will be collated into a single bundle. When data from all agents is collected into a bundle, the bundle is marked as completed. **Type:** int **Default Value:** 20	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for most clusters. On very large clusters (with more than 1000 nodes) if bundle collection causes performance issues with the default configuration, this interval can be increased to one minute to minimize file compressions/decompressions.
bundle.alert.progress.timeout.percent	This percentage of bundle processing for which the bundle is failing will raise an alert. **Type:** float **Default Value:** 0.6	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for most clusters. If the bundle is failing during capture or processing, you can adjust the percentage of processing for which alert will be issued.
server.cleanup.task.interval	Time in hours to execute server cleanup tasks (clean up stale/cancelled bundle temp data). **Type:** int **Default Value:** 1	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for all clusters.
security.server.two_way_ssl.port	Port for two-way SSL communication between HST server and HST agents. This port is used internally for HST operations. **Type:** int **Default Value:** 9441	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Change only if port 9441 is already in use or cannot be unblocked. This has no impact on SmartSense Ambari View.
security.server.one_way_ssl.port	Port for one-way SSL communication between HST server and HST agents. This port is usually required during two-way SSL setup. This port is used internally for HST operations. **Type:** int **Default Value:** 9440	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Change only if port 9442 is already in use or cannot be unblocked. This has no impact on SmartSense Ambari View.
security.openssl.digest.algorithms	Permitted algorithms for SSL encryption.	**Ambari Config:**	Not required to modify as sha256,sha512 are

Property Name	Description	Where to Configure	Guidelines
	Type: string **Default Value:** sha256,sha384,sha512,sha,sha1,md5	Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	available and provide strong encryption. Change only if there are very specific security requirements that can not be met by sha256/sha512.
server.connection.max.idle.millis	The maximum period in milliseconds that a connection may be idle before it is closed. **Type:** int **Default Value:** 900000	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Update this if you see too many open threads in idle state on the HST server.
security.server.disabled.ciphers	A comma-separated list of disabled ciphers for SSL. **Type:** string **Default Value:** (no value)	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Weaker ciphers are already disabled. Change only if you have very specific security requirements.
security.server.disabled.protocols	A comma-separated list of disabled protocols for SSL. **Type:** string **Default Value:** (no value)	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Weaker ciphers are already disabled. Change only if you have very specific security requirements.
upload.permits	Agents capture data and upload it to the HST server which assembles it together into a single bundle. This property defines the number of concurrent uploads allowed from agent to server. **Type:** int **Default Value:** 10	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	This property may need to be increased if agent upload requests are timing out on a cluster with more than 500 nodes.
upload.initiate.timeout	Agents capture data and upload it to the HST server which assembles it together into a single bundle. Upload will fail if not initiated within the timeout window (in seconds) defined in this property. **Type:** int **Default Value:** 20	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	This property may need to be increased if agent upload requests are timing out on a cluster with more than 500 nodes.
bundle.keepuploaded	This tells the HST server whether to keep bundles received from agents even after merging. If set to false, the agent bundles are deleted after merging. **Type:** boolean **Default Value:** false	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Set this to TRUE if you have to inspect the agent bundles for debugging purposes. Note that this will require plenty of available disk space.
bundle.purge.enabled	Enables a daemon process to purge old bundles. By default, the daemon process cleans up	**Ambari Config:**	We recommend not to disable this process as it

Property Name	Description	Where to Configure	Guidelines
	old bundles to efficiently use the disk space. **Type:** boolean **Default Value:** true	Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	will require a large amount of additional disk space.
bundle.min.retention.days	Number of days to keep the bundle before soft purging. Bundles will be soft purged after the defined number of retention days: the bundle file will be deleted and the DB entries will be soft deleted. **Type:** int **Default Value:** 30	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Update this if you want to keep bundles for longer time (to keep records) or for shorter time (to reduce storage utilization).
bundle.min.force.purge.retention.days	Number of days to keep the bundle before hard purging. Bundles will be hard purged after the defined number of retention days: the DB entries of bundle data along with associated recommendations will be cleaned up. **Type:** int **Default Value:** 90	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for all clusters.
bundle.purge.threadpool.size	Thread pool used for purging hundreds of bundles. **Type:** int **Default Value:** 1	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for all clusters.
bundle.purge.interval	The frequency (in hours) with which to run the purge process. **Type:** int **Default Value:** 24	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	The default setting (once per day) is sufficient unless you have tens of bundles created daily.
bundle.validity.days	Bundle validity days for retrieving recommendations. After this number of days, a bundle will no longer be considered for retrieving recommendations. **Type:** int **Default Value:** 15	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	We recommend not to increase this beyond default because older bundles might not provide the latest status of the cluster.
recommendation.expiry	Recommendation actions such as "Apply" are not permitted on bundles which are older than this number of days. **Type:** int **Default Value:** 30	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	We recommend that you capture a new bundle and get new recommendations instead of referring to older recommendations.
recommendation.history.expiry	Recommendation history actions are not permitted on bundles older than this number of days.	**Ambari Config:**	We recommend that you capture a new bundle and get new recommendations

Property Name	Description	Where to Configure	Guidelines
	Type: int **Default Value:** 90	Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	instead of referring to older recommendations. Update this value if you have to refer to earlier actions.
recommendation.auto.download.bundle.expiry	If recommendations are not received, HST server will stop trying to retrieve recommendations after this number of days. **Type:** int **Default Value:** 7	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Change this only if you have a very specific requirement and want to stop requesting for recommendations earlier than after 7 days.
recommendation.auto.download.interval	Interval (in seconds) for retrieving recommendations. By default, recommendations are retrieved every 300 seconds. **Type:** int **Default Value:** 300	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Change this only if you have a very specific requirement. Increase this if you want to reduce the frequency of retry attempts.
recommendation.feedback.push.interval	Interval (in seconds) for submitting customer feedback for recommendations. By default, HST server will submit feedback to Hortonworks every 30 minutes if new feedback is available. **Type:** int **Default Value:** 1800	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for most clusters. Change this only if you have a very specific requirement.
recommendation.feedback.push.max.entries	The number of feedback entries submitted in one request. By default, HST server submits a batch of 50 feedback entries in one request. **Type:** int **Default Value:** 50	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for most clusters. Change this only if you have a very specific requirement. Requires tuning only if you submit more than 100 feedback entries on a daily basis.
gateway.enabled	Enables auto upload of bundles after capture. **Type:** boolean **Default Value:** true	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Disable this if you are capturing the bundles for internal review purposes only. We recommend to keep it enabled to receive valuable insights and recommendations for your cluster.
gateway.retry.attempts	Defines how many attempts HST server makes to connect to the SmartSense Gateway. **Type:** int **Default Value:** 10	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:** /etc/hst/conf/hst-server.ini	Default value is suitable for all clusters.
gateway.retry.interval.increment	The amount of time (in milliseconds) to wait before making a subsequent SmartSense Gateway connection attempt. In other words, this is the wait time	**Ambari Config:** Advanced > Custom hst-server-conf **Config File:**	Default value is suitable for all clusters.

Property Name	Description	Where to Configure	Guidelines
	between subsequent connection attempts. **Type:** int **Default Value:** 5000	/etc/hst/conf/hst-server.ini	

3.3.2. HST Agent

The following configuration properties are available for HST Agent:

Table 3.3. HST Agent Configuration Properties

Property Name	Description	Where to Configure	Guidelines
agent.tmp_dir	Temporary directory used by agents to keep local bundles during bundle preparation. **Type:** string **Default Value:** /var/lib/smartsense/hst-agent/data/tmp	**Ambari Config:** Basic **Config File:** /etc/hst/conf/hst-agent.ini	You must have at least 10GB of free space in this directory. This should be set to a different location than the server tmp directory.
security.anonymization.max.heap	The maximum heap allocated (in MB) on every agent for anonymization. **Type:** int **Default Value:** 2048	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	If you experience out of memory exceptions during the anonymization process, increase the heap size gradually depending on availability.
agent.loglevel	Provides ability to change the hst-agent logging level. Possible values are: INFO, DEBUG, WARNING, ERROR, CRITICAL. **Type:** string **Default Value:** INFO	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	To debug issues on the agent, set this to DEBUG.
bundle.logs_to_capture	Patterns of log files to be captured. **Type:** string **Default Value:** (.*).log$,(.*).out$	**Ambari Config:** Data Capture **Config File:** /etc/hst/conf/hst-agent.ini	Be careful when capturing more log files as they may turn out to be large and require extra space on the HST server.
server.url_port	Port for one-way SSL communication between HST server and HST agents. This port is usually required during two-way SSL setup. **Type:** **Default Value:** 9440	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	This should be modified in sync with similar property in HST server configurations.
server.secured_url_port	Port for two-way SSL communication between HST server and HST agents. **Type:** **Default Value:** 9441	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:**	This should be modified in sync with similar property in HST server configurations.

Property Name	Description	Where to Configure	Guidelines
		/etc/hst/conf/hst-agent.ini	
server.two_way_ssl	Enables two-way SSL for communication between HST server and HST agents. **Type:** boolean **Default Value:** true	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	We recommend not to change this unless you have a very specific requirement.
server.connection_retry_count	Number of times to retry to connect to server in case of connection failures and timeouts. **Type:** int **Default Value:** 100	**Ambari Config:** Operations **Config File:** /etc/hst/conf/hst-agent.ini	In many cases, the default value (100 retry attempts) is often more than needed. Reduce it if retry connection attempts are keeping the system busy.
server.connection_retry_interval	Defines the interval (in seconds) between retries. **Type:** int **Default Value:** 10	**Ambari Config:** Operations **Config File:** /etc/hst/conf/hst-agent.ini	Default value is suitable for all clusters.
java.home	Path to the JAVA home for HST agents. **Type:** string **Default Value:** (no value)	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	This setting is automatically configured from Ambari env settings and usually there is no reason to change it. We recommend that you use the latest 1.7/1.8 JAVA versions with up-to-date security updates. For more security we also recommend that you have unlimited JCE policy installed.
command.heartbeat_interval	The heartbeat interval (in seconds). During agent capture, this heartbeat helps ensure connectivity with HST server and executes certain commands such as cancel capture. **Type:** int **Default Value:** 30	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	Do not change this unless you experience performance issues.
command.check_commands_retry_count	Number of times to retry check commands. **Type:** int **Default Value:** 10	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	Default value is suitable for all clusters.
command.check_commands_retry_interval	Interval (in seconds) between retries for check commands. **Type:** int	**Ambari Config:** Advanced > Custom hst-agent-conf	Default value is suitable for all clusters.

Property Name	Description	Where to Configure	Guidelines
	Default Value: 10	**Config File:** /etc/hst/conf/hst-agent.ini	
management.updates.dir	Directory to store updates received from HST server. **Type:** string **Default Value:** /var/lib/smartsense/hst-agent/updates	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	We recommend not to change this unless you have a very specific requirement. If changing this, verify that permissions are set accordingly.
management.patch.auto.apply.enabled	Enable automatically downloading and applying updates received from HST server. **Type:** boolean **Default Value:** true	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	Disable only if you do not want the HST server to propagate the agent configuration changes to all agents.
bundle.compress_captured_log_locally	By default this is set to false; i.e the log files are included as they are, without applying compression. **Type:** boolean **Default Value:** false	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	If capture requests are timing out, set this to true to activate log compression. Note that the compressed files will not be anonymized.
upload.retry_count	Number of times the agent will retry to submit its local bundle to server. Note that this is different from uploading the final bundle to Hortonworks. **Type:** int **Default Value:** 100	**Ambari Config:** Operations **Config File:** /etc/hst/conf/hst-agent.ini	Default value is suitable for all clusters.
upload.min_retry_interval	Minimum interval (in seconds) between bundle upload retries made by agents. Random value between min_retry_interval and max_retry_interval will be used. For constant value, use retry_interval=x. **Type:** int **Default Value:** 15	**Ambari Config:** Operations **Config File:** /etc/hst/conf/hst-agent.ini	Default value is suitable for all clusters.
upload.max_retry_interval	Maximum interval (in seconds) between bundle upload retries made by agents. Random value between min_retry_interval and max_retry_interval will be used. For constant value, use retry_interval=x. **Type:** int **Default Value:** 120	**Ambari Config:** Operations **Config File:** /etc/hst/conf/hst-agent.ini	Default value is suitable for all clusters.
handler.largefiles.size	Minimum file size (in MB) for a file to qualify as a large file. Large files are handled	**Ambari Config:**	Update this if you have a specific requirement which

Property Name	Description	Where to Configure	Guidelines
	based on action defined in the `handler.largefiles.action` property. **Type:** int **Default Value:** 512	Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	includes capturing files larger than 512 MB.
handler.largefiles.action	Action to handle a large file. Supported actions are 'allow', 'ignore', 'truncate', 'fail'. **Type:** string **Default Value:** truncate	**Ambari Config:** Advanced > Custom hst-agent-conf **Config File:** /etc/hst/conf/hst-agent.ini	Configure action to handle large files based on your requirements.

3.3.3. SmartSense Gateway

The following configuration properties are available for the SmartSense Gateway:

Table 3.4. SmartSense Gateway Configuration Properties

Property Name	Description	Where to Configure	Guidelines
security.openssl.digest.algorithms	A comma separated list of supported algorithms for SSL. **Type:** string **Default Value:** sha256,sha384,sha512,sha,sha1,md5	**Ambari Config:** N/A **Config File:** /etc/hst/conf/hst-gateway.ini	Typically it there is no need to modify this as sha256,sha512 are available and provide strong encryption. Change this only if you have a very specific requirement.
security.gateway.cert.name	Use this property if you have to use a custom root Certificate Authority for SmartSense Gateway operations. This file must exist at /var/lib/hst-gateway/keys before gateway is started. **Type:** string **Default Value:** ca.crt	**Ambari Config:** N/A **Config File:** /etc/hst/conf/hst-gateway.ini	You can set up your own CA to sign certificates for two-way SSL communication between HST server and SmartSense Gateway. Modify this property to customize the root CA.
gateway.thread.pool.size	Thread pool for the gateway server's API endpoint. Default is automatically calculated based on CPU cores. **Type:** int **Default Value:** (Automatically calculated)	**Ambari Config:** N/A **Config File:** /etc/hst/conf/hst-gateway.ini	Since the count is already dynamic, it is usually not required to change it.
gateway.start.validation.enabled	Enables the outbound connectivity check that SmartSense Gateway performs during startup. By default, the connectivity check in enabled. **Type:** boolean **Default Value:** true	**Ambari Config:** N/A **Config File:** /etc/hst/conf/hst-gateway.ini	Set to false if you use an HTTP proxy for gateway and gateway start command fails with "unable to connect" error. There is bug in SmartSense Gateway versions earlier than 1.3.2 where the socket connectivity test does not use the intended proxy.

Property Name	Description	Where to Configure	Guidelines
gateway.cache.expiry.hours	The frequency with which the SmartSense Gateway cache is refreshed. Gateway caches the outbound connectivity status to report to HST server. By default, this cache is refreshed every two hours and upon gateway startup. **Type:** int **Default Value:** 2	**Ambari Config:** N/A **Config File:** /etc/hst/conf/hst-gateway.ini	Tweak this property to refresh the outbound connectivity status.
gateway.data.transfer.buffer.size	Buffer size for data transfer via SmartSense Gateway. Gateway uses chunked buffers to transfer encrypted data between SmartSense and Hortonworks Datalake. **Type:** int **Default Value:** 4096	**Ambari Config:** N/A **Config File:** /etc/hst/conf/hst-gateway.ini	Tune this property to effectively use the network bandwidth for communication between HST server and Hortonworks.

3.3.4. Activity Analyzer

The following configuration properties are available for Activity Analyzer:

Table 3.5. Activity Analyzer Configuration Properties

Property Name	Description	Where to Configure	Guidelines
phoenix.sink.batch.size	Activities are batched for better storage performance. A batch is persisted when either the batch size becomes equal to phoenix.sink.batch.size or activity.status.update.interval.seconds has elapsed. **Type:** int **Default Value:** 100	**Ambari Config:** Activity Analysis **Config File:** /etc/smartsense-activity/conf/activity.ini	Increasing batch size can lower the load on storage and improve storage performance; however, it can delay the availability of data and increase memory pressure. Reducing batch size can make data available sooner but has negative performance impact on storage layer.
global.activity.processing.parallelism	Number of parallel threads that process each activity type. Controls the threads used for Tez, YARN, MR, and HDFS activity data collection. **Type:** int **Default Value:** 8	**Ambari Config:** Activity Analysis **Config File:** /etc/smartsense-activity/conf/activity.ini	Reduce the number of threads if you encounter out of memory exceptions.
phoenix.sink.flush.interval.seconds	Time after which data will be flushed to Phoenix. A batch is persisted when either the batch size becomes equal to phoenix.sink.batch.size or activity.status.update.interval.seconds has elapsed. **Type:** int	**Ambari Config:** Activity Analysis **Config File:** /etc/smartsense-activity/conf/activity.ini	Increase the time to reduce the number of persist operations to Phoenix only if number of records to be batched together is much less than 100.

Property Name	Description	Where to Configure	Guidelines
	Default Value: 30		
mr_job.activity.watcher.enabled	Enables automatic activity analysis for MapReduce jobs. **Type:** boolean **Default Value:** true	**Ambari Config:** Activity Analysis **Config File:** /etc/smartsense-activity/conf/activity.ini	Disable only if you do not want to analyze MapReduce jobs.
mr_job.max.job.size.mb.for.parallel.execution	Maximum size (in bytes) that a MapReduce job can have in order to be executed in parallel. Some large MapReduce jobs may contain thousands of tasks. Such jobs require a lot of memory and they put memory pressure on JVM, especially in multi-threaded execution. Any job with history size larger than specified in this parameter will be executed in synchronized fashion. This may slow the performance down, but will avoid OOM errors. Any job with history file size smaller than specified in this parameter will be executed in parallel. **Type:** int **Default Value:** 500	**Ambari Config:** Activity Analysis **Config File:** /etc/smartsense-activity/conf/activity.ini	Reduce the parallel execution job size if you encounter OOM errors.
tez_job.activity.watcher.enabled	Enables automatic activity analysis for Tez jobs. **Type:** boolean **Default Value:** true	**Ambari Config:** Activity Analysis **Config File:** /etc/smartsense-activity/conf/activity.ini	Disable only if you do not want to analyze Tez jobs.
tez_job.tmp.dir	Temporary location where Tez job information is downloaded. **Type:** string **Default Value:** /var/lib/smartsense/activity-analyzer/tez/tmp/	**Ambari Config:** Activity Analysis **Config File:** /etc/smartsense-activity/conf/activity.ini	You can symlink it to a non-root partition or change it to use a directory in a non-root partition.
yarn_app.activity.watcher.enabled	Enables automatic activity analysis for YARN apps. **Type:** boolean **Default Value:** true	**Ambari Config:** Activity Analysis **Config File:** /etc/smartsense-activity/conf/activity.ini	Disable only if you do not want to analyze YARN jobs.
hdfs.activity.watcher.enabled	Enables automatic analysis for HDFS files. **Type:** boolean **Default Value:** true	**Ambari Config:** Activity Analysis **Config File:**	Disable only if you do not want to analyze HDFS fsImage.

Property Name	Description	Where to Configure	Guidelines
		/etc/smartsense-activity/conf/activity.ini	
global.activity.analyzer.user	Defines the user used to read activity data from HDFS and YARN. This user must have read access to all activity data from HDFS/YARN/ATS, and so on. **Type:** string **Default Value:** activity_explorer	**Ambari Config:** Advanced > Advanced activity-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Default value is suitable for all clusters.
activity.explorer.user	Defines the user used to read pre-analyzed data. This user does not need access to HDFS and YARN. **Type:** string **Default Value:** activity_explorer	**Ambari Config:** Advanced > Advanced activity-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Default value is suitable for all clusters.
analyzer_jvm_opts	Allows you to specify multiple jvm options separated by space. **Type:** string **Default Value:** -Xms128m	**Ambari Config:** Advanced > Advanced activity-env **Config File:** /etc/smartsense-activity/conf/activity-env.sh	This parameter allows you to add any additional jvm options for executing activity analyzers, for example for GC tuning.
analyzer_jvm_heap	Maximum heap space (in MB) allocated for Activity Analyzer process. **Type:** int **Default Value:** 8192	**Ambari Config:** Advanced > Advanced activity-env **Config File:** /etc/smartsense-activity/conf/activity-env.sh	Usually 8192 MB is sufficient, but it can be increased if you encounter OOM errors.
activity_log_dir	Directory where activity log files are created. **Type:** string **Default Value:** var/log/smartsense-activity	**Ambari Config:** Advanced > Advanced activity-log4j **Config File:** /etc/smartsense-activity/conf/log4j.properties	Default value is suitable for most clusters. If you change this directory, you must provide read/write/create permissions on the new directory to activity_analyzer user.
activity_max_file_size	Maximum size (in MB) for SmartSense activity log files. **Type:** int **Default Value:** 30	**Ambari Config:** Advanced > Advanced activity-log4j **Config File:** /etc/smartsense-activity/conf/log4j.properties	Default value is suitable for most clusters. Check available storage capacity before updating this property.
activity_max_backup_index	Maximum number of SmartSense activity log files. **Type:** int **Default Value:** 10	**Ambari Config:** Advanced > Advanced activity-log4j **Config File:** /etc/smartsense-activity/conf/log4j.properties	You can increase this number to keep the record of older logs. Check available storage capacity before updating this property.

Property Name	Description	Where to Configure	Guidelines
global.date.format	Format in which dates are converted to strings and sometimes persisted. **Type:** string **Default Value:** "YYYY-mm-DD"	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Default value is suitable for all clusters.
global.activity.status.update.interval.seconds	Interval (in seconds) after which status of processed/failed/in process activities is updated in DB. **Type:** int **Default Value:** 30	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Default value is suitable for all clusters.
activity.batch.interval.seconds	Interval for batching activities. Activities are batched for better storage performance. A batch is persisted when either the batch size becomes equal to phoenix.sink.batch.size or activity.status.update.interval.seconds is elapsed. **Type:** int **Default Value:** 60	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Increasing the batch interval can lower the load on storage and improve storage performance; however, it can also delay the availability of data and increase memory pressure. Reducing the interval size can make data available sooner, but has negative performance impact on storage layer.
activity.watcher.enabled	Enables regular collection of job data for analysis. **Type:** boolean **Default Value:** true	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Disable this only if you want to temporarily turn off data collection.
activity.history.max.back.track.days	The number of days of history to retrieve job information. **Type:** int **Default Value:** 7	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Increase this number if you have to refer to older jobs. Note that older jobs should have data available in AMS. This is used only during first run after installation.
phoenix.setup.continue.on.error	During initial setup, errors in DB setup may occur. This parameter indicates whether to continue if any error occurs. **Type:** boolean **Default Value:** false	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Default value is suitable for all clusters.
phoenix.setup.drop.existing.tables	During initial setup matching tables may be found in the DB (typically from previous install attempts). This parameter determines whether they should be dropped and recreated. By	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:**	Default value is suitable for all clusters.

Property Name	Description	Where to Configure	Guidelines
	default, the existing entries are kept. **Type:** boolean **Default Value:** false	/etc/smartsense-activity/conf/activity.ini	
phoenix.activity.analyzer.jdbc.url	JDBC URL used by Activity Analyzer to store its data. **Type:** string **Default Value:** (no value)	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Do not change it. It is auto configured based on the cluster setup.
ams.jdbc.url	JDBC URL used by Activity Analyzer to fetch data from AMS. **Type:** string **Default Value:** (no value)	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Do not change it. It is auto configured based on the cluster setup.
global.store.job.configs	Enables storing job-specific configs in AMS after analysis. **Type:** boolean **Default Value:** true	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Do not disable it. Keeping it on helps in debugging.
global.store.tasks	Enables persisting task-level data in AMS after analysis. **Type:** boolean **Default Value:** false	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Task-level data can be huge and may overwhelm AMS, so keep it disabled unless absolutely needed. If enabling, disable again later.
global.store.task.counters	Enables storing task counter data in the AMS after analysis. **Type:** boolean **Default Value:** false	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	All task counters can be huge and may overwhelm AMS, so keep it disabled unless absolutely needed. If enabling, disable again later.
global.activity.fetch.retry.interval.seconds	Interval (in seconds) between retry attempts to fetch the activity details. **Type:** int **Default Value:** 5	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	Default value is suitable for all clusters.
global.activity.fetch.retry.attempts	Number of tries to fetch activities before giving up. **Type:** int **Default Value:** 5	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:**	Default value is suitable for all clusters.

Property Name	Description	Where to Configure	Guidelines
		/etc/smartsense-activity/conf/activity.ini	
global.tmp.dir	Temporary directory used by activity-analyzer for internal purposes. **Type:** string **Default Value:** /var/lib/smartsense/activity-analyzer/tmp/	**Ambari Config:** Advanced > Custom activity-analyzer-conf **Config File:** /etc/smartsense-activity/conf/activity.ini	We do not recommended to change this unless you have a very specific requirement. If using a different directory than the default, verify that permissions are set accordingly.

3.3.5. Activity Explorer

The following configuration properties are available for Activity Explorer:

Table 3.6. Activity Explorer Configuration Properties

Property Name	Description	Where to Configure	Guidelines
users.admin	Password for Activity Explorer's admin user when using local authentication. **Type:** string **Default Value:** (no value)	**Ambari Config:** Activity Analysis **Config File:** /etc/zeppelin/conf/shiro.ini	This should be updated only during installation. It requires uninstall and re-install if you have to update the password for admin access.
main.sessionManager	The SessionManager, as its name might imply, manages sessions for all subjects in an application: session creation, deletion, inactivity, validation, and so on. Like other core architectural components in Apache Shiro, the SessionManager is a top-level component maintained by the SecurityManager. The default SecurityManager implementation uses a DefaultSessionManager out of the box. The DefaultSessionManager implementation provides enterprise-grade session management features (such as session validation and orphan cleanup) needed for an application. **Type:** string **Default Value:** org.apache.shiro.web.session.mgt.DefaultWebSessionManager	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/shiro.ini	Refer to Apache Shiro documentation.
main.securityManager.sessionManager	The default value applies the value set in main.sessionManager to this property. If needed, you can set this to a value specific for security manager. **Type:** string	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:**	Refer to Apache Shiro documentation.

Property Name	Description	Where to Configure	Guidelines
	Default Value: $sessionManager	/etc/zeppelin/conf/ shiro.ini	
securityManager.sessionManager.globalSessionTimeout	Sets the default timeout value (in milliseconds) for all newly created sessions. Changing this property will automatically apply the new value to all sessions. **Type:** long **Default Value:** 86400000	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Shiro documentation.
zeppelin.server.addr	Binding address for Zeppelin Activity Explorer. **Type:** string **Default Value:** 0.0.0.0	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.server.port	Port on which Zeppelin UI is available. **Type:** int **Default Value:** 9060	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.server.context.path	Context path of the web application. **Type:** string **Default Value:** /	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.war.tempdir	Location of Jetty temporary directory. **Type:** string **Default Value:** /var/lib/smartsense/activity-explorer/webapp	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.notebook.dir	Path or URI for notebook persist. **Type:** string **Default Value:** /var/lib/smartsense/activity-explorer/notebook	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.notebook.homescreenhide	When set to true, hides home screen notebook from list. **Type:** boolean **Default Value:** false	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.

Property Name	Description	Where to Configure	Guidelines
zeppelin.notebook.storage	Notebook persistence layer implementation. **Type:** string **Default Value:** org.apache.zeppelin.notebook.repo.VFS NotebookRepo	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.interpreter.dir	Interpreter implementation base directory. **Type:** string **Default Value:** /usr/hdp/share/hst/activity-explorer/interpreter	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.interpreters	A comma separated list of interpreter configurations. First interpreter becomes default. **Type:** string **Default Value:** org.apache.zeppelin.phoenix.Phoenix Interpreter	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.interpreter.connect.timeout	Interpreter process connect timeout in milliseconds. **Type:** int **Default Value:** 30000	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.ssl	Enables using SSL for the servers. **Type:** boolean **Default Value:** false	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.ssl.client.auth	Enables client authentication for SSL connections. **Type:** boolean **Default Value:** false	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.ssl.keystore.path	Path to keystore relative to the Activity Explorer configuration directory. **Type:** stringzeppelin.ssl.truststore.type **Default Value:**	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/ shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.

Property Name	Description	Where to Configure	Guidelines
	/var/lib/smartsense/activity-explorer/keystore		
zeppelin.ssl.keystore.type	The format of the given keystore (for example JKS or PKCS12). **Type:** string **Default Value:** JKS	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.ssl.keystore.password	Keystore password. It can be obfuscated using the Jetty password tool. **Type:** string **Default Value:** admin	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.ssl.key.manager.password	Key Manager password. Defaults to keystore password. It can be obfuscated. **Type:** string **Default Value:** admin	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.ssl.truststore.path	Path to truststore relative to Activity Explorer configuration directory. Defaults to the keystore path. **Type:** string **Default Value:** /var/lib/smartsense/activity-explorer/truststore	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.ssl.truststore.type	The format of the given truststore (for example JKS or PKCS12). Defaults to the same type as the keystore type. **Type:** string **Default Value:** JKS	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.ssl.truststore.password	Truststore password. Can be obfuscated using the Jetty password tool. Defaults to the keystore password. **Type:** string **Default Value:** admin	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.server.allowed.origins	Allowed sources for REST and WebSocket requests (i.e. http://onehost:8080,http://otherhost.com).	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:**	Refer to Apache Zeppelin 0.6.2 documentation.

Property Name	Description	Where to Configure	Guidelines
	If you change from * you are vulnerable to the issue described in ZEPPELIN-173. **Type:** string **Default Value:** *	/etc/zeppelin/conf/shiro.ini	
zeppelin.anonymous.allowed	Enables access by anonymous user. **Type:** boolean **Default Value:** false	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.
zeppelin.websocket.max.text.message.size	Size in characters of the maximum text message to be received by WebSocket. **Type:** long **Default Value:** 1024000	**Ambari Config:** Advanced > Advanced activity-zeppelin-shiro **Config File:** /etc/zeppelin/conf/shiro.ini	Refer to Apache Zeppelin 0.6.2 documentation.

4. SmartSense Performance Tuning

This section contains tips for achieving optimal performance for your cluster.

4.1. Tuning the JVM Memory Settings

To achieve optimal performance for your cluster size, you may need to increase the JVM memory settings.

The default setting, 2048 MB, is appropriate for a cluster with up to 100 nodes. For each additional 100 nodes, increase this setting by 0.5 GB to improve performance.

To adjust the setting, in the Ambari Web UI, navigate to the SmartSense service's **Config** section > **Advanced** > **Advanced hst-server-conf** where you will find the **Server max heap size** configuration property.

4.2. Cleaning Up Old Bundles

HST server has a background process which periodically deletes bundles older than 30 days. Additionally, bundle "purge" commands can be used to trigger this process for purging bundles older than specified number of days or for purging a particular bundle.

There are two ways to purge bundles:

- Purge: The bundle file is removed from the storage but associated records such as recommendations from the HST DB are retained.
- Hard purge: All bundle data and its associated records such as recommendations from the HST DB are completely removed.

When using **hst purge**, soft purging is used unless the hard purging option is specified.

Syntax

```
# hst purge -h
Usage: hst purge [-r][-b][-H][-q] arg
Triggers bundle purge job

Options:
-h, --help show this help message and exit
-r RETENTIONDAYS, --retentionDays=RETENTIONDAYS number of days to retain a
 bundle before purging
-H, --hard flag to indicate hard purge
-b BUNDLEID, --bundleId=BUNDLEID purge a particular bundle Id
-q, --quiet flag to purge quietly
```

Examples

1. Purge bundles older than 5 days:

```
# hst purge -r 5
Do you want to continue purging bundles older than 5 days ? y/n (default: n):
 y
Bundles purge job triggered successfully.
```

2. Hard purge bundles older than 20 days:

```
# hst purge -r 20 -H
Do you want to continue hard purging bundles older than 20 days ? y/n
 (default: n): y
Bundles purge job triggered successfully.
```

3. Manually trigger default purge process:

```
# hst purge
Do you want to continue purging bundles older than 30 days ? y/n (default: n):
 n
```

4. Hard purge a particular bundle:

```
# hst purge -b a-xxxxxxxx-c-xxxxxxxx_c6nr_0_2017-05-07_02-00-02 -H
Do you want to continue hard purging bundle : a-xxxxxxxx-c-
xxxxxxxx_c6nr_0_2017-05-07_02-00-02 ? y/n (default: n): y
Bundle purged successfully.
```

5. Purge a particular bundle:

```
# hst purge -b a-xxxxxxxx-c-xxxxxxxx_c6nr_0_2017-05-05_08-32-09
Do you want to continue purging bundle : a-xxxxxxxx-c-
xxxxxxxx_c6nr_0_2017-05-05_08-32-09 ? y/n (default: n): y
Bundle purged successfully.
```