Ignite Technologies
Security and Compliance Solutions

SenSage Administrator's Guide
SenSage Standard 2017

Notices

401 Congress Avenue Suite 2650
Austin, TX 78701
800-248-0027
info@ignitetech.com

Published March 31, 2017

Table of Contents

Preface

For details, see the following topics:

Audience for this Book

This book is directed to system administrators who manage a SenSage AP deployment. Administrators should be familiar with Linux administration including: SSH, managing users and groups, file system permissions and management, network configuration, and using text editors such as VI or emacs.

Administration Guide Organization

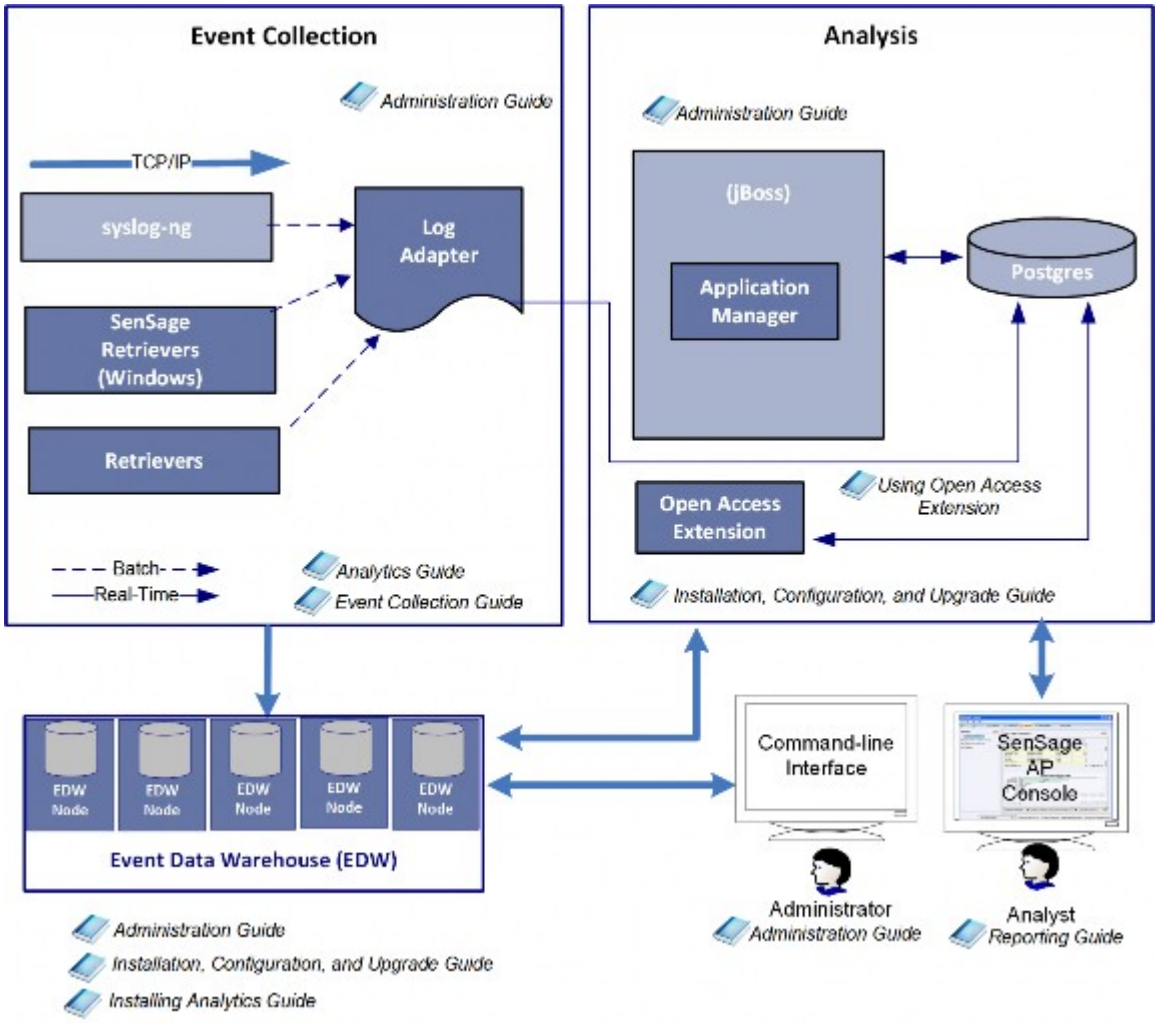
- This book contains the following chapters:
- Introduction on page 19—Provides an overview of SenSage AP system administration tasks and SenSage AP architecture

- Managing Sensage AP on page 25—Describes how to use Deployment Manager to monitor and configure SenSage AP and perform standard operations. Also included are utility tools for general tasks that support SenSage AP configuration and management.
- Sensage Loading, Querying, and Managing the EDW on page 61—Describes the utilities that SenSage AP provides for loading, querying, managing, viewing, and retiring data.
- Administering an EDW Instance on page 123—Describes how to back up and restore EDW tables, restore hosts in an EDW instance, monitor the system, and archive data.
- Replacing a Failed EDW node on page 139—Describes the process for handling a failed EDW node.
- Administering the Collector on page 145—Describes how to perform various administrative tasks for the optimal operation of the Collector, including starting the Collector, monitoring Collector operational logs for errors, handling bad load operations, and backing up processed log files.
- Managing Security (Users and Authentication) on page 159—Describes how to configure authentication and authorization and provides an overview on administering users, roles and permissions.
- Archiving to Nearline Storage on page 181—Describes how to archive data to Nearline storage.
- Troubleshooting on page 193—Describes basic troubleshooting.
- Log Files on page 199—Presents reference material on log files.
- Error Codes on page 205—Presents reference material on error codes.
- Time Zones on page 225—Presents reference material on time zones.

Road Map to SenSage AP Documentation

This document, the Administration Guide, is part of the larger documentation set of your SenSage AP system. Figure P-1 illustrates SenSage AP components and modules in the context of their function within the SenSage AP system.

Figure 1: Figure P-1: Road Map to SenSage AP Documentation



The table below describes all the manuals in the SenSage AP documentation set and the user roles to which they are directed.

Role	Tasks	Documentation
Analyst, Report Developer, System Administrator	• Create and edit reports, charts, and models • Create and edit dashboards • Manage SenSage AP Users • Manage SenSage AP Models • Create and edit data models • Write SQL code and queries	Analyzer Guide
Business Analyst or System Administrator	• Learn about Analytics • Use IntelliSchema views • Learn about the Foundation and Compliance Analytics packages • Learn about additional Analytics packages	Analytics Guide
Developer, Report Developer or Security Analyst	• Use SenSage AP SQL, SenSage AP SQL functions, and libraries to create reports or query the EDW • Access EDW data using open standards as ANSI SQL, ODBC, and JDBC • Create and use Perl code in SenSage AP SQL statements • Use the DBD Driver to query SenSage AP from other locations	Event Data Warehouse Guide
Security System Administrator	• Configure retrievers, receivers, and collectors • Enable/disable log adapters • Configure SenSage Retriever • Create log adapter PTL files	Collector Guide
System Administrator	• Install SenSage AP • Configure SenSage AP and its components • Configure Vmware	Installation, Configuration, and Upgrade Guide

Role	Tasks	Documentation
System Administrator	• Manage the SenSage AP Event Data Warehouse (EDW) • Manage the Collector • Manage users, groups, and permissions • Archive to Nearline storage • Manage assets & monitor security alerts • Monitor log source health • Monitor system health • Troubleshoot • Error Messages	Administration Guide
Legal	Monitor third-party licenses	Third-Party Open Source Licensing

Note: You can access the manuals listed above from the SenSage AP Welcome page. Click on the **Documentation** hyperlink.

Conventions Used in SenSage AP Documentation

This convention...	Indicates...	Example
bold text	Names of user interface items, such as field names, buttons, menu choices, and keystrokes	Click Clear Filter.
italic text	Indicates a variable name or a new term the first time it appears	http://<*host*>:<*port*>/index.mhtml
`Courier text`	Indicates a literal value, such as a command name, file name, information typed by the user, or information displayed by the system	`atquery localhost:8072 myquery.sql`
CAPS	Indicates a key on the computer keyboard	Press ENTER.

This convention...	Indicates...	Example
{ }	In a syntax line, curly braces surround a set of options from which you must choose one and only one. NOTE: Syntax specifications for SELECT statements include curly braces as part of the {INCLUDE_BAD_LOADS] keyword.	{ start | stop | restart }
[ ]	In a syntax line, square brackets surround an optional parameter	atquery [options] <*host*>:<*port*> -
|	In a syntax line, a pipe within square brackets or curly braces separates a choice between mutually exclusive parameters NOTE: Syntax for defining a Nearline Storage Address (NSA) includes a pipe.	{ start | stop | restart } [g|m]
...	In a syntax line, ellipses indicate a repetition of the previous parameter	The following example indicates you can enter multiple, comma-separated options: <*option*>[, <*option*>[…]]
backslash (\)	A backslash in command-line syntax or in a command example behaves as the escape character on Unix. It removes any special meaning from the character immediately following it. In SenSage AP documentation, a backslash nullifies the special meaning of the newline character as a command terminator. Without the backslash, pressing ENTER at the end of the line causes the Unix system to execute the text preceding the ENTER. Without the backslash, you must allow long commands to wrap over multiple lines as a single line.	atquery --user=administrator \ --pass=pass:p@ss localhost:8072\ -e='SELECT * FROM system.users;'

How to Obtain the SenSage AP License

Beginning with SenSage AP 5.0.0, you are required to have a license key to run your cluster. Each EDW cluster will require its own, valid license key. Trying to upgrade to 6.1.1 without a valid license will cause the upgrade to fail.

Your license is contained in a license key file that you must copy to each EDW node in your cluster. A successful query against your EDW cluster confirms that you have correctly installed a valid license key for your cluster.

If you haven't received your license key, contact your IgniteTech representative. When you receive the key, you need to copy it over to the following placeholder files on each EDW node in your cluster.

`<install-root>/etc/sls/instance/<cluster-name>/LICENSE`

When the new license key, LICENSE, is copied to all EDW nodes, you can restart each EDW normally.

Do not alter the license key file in any way. Even a whitespace change will render the file invalid as the embedded digital signature is computed against the exact sequence of bytes in the original license key file.

If there is an issue with the LICENSE file, you will see one of the following errors when submitting queries to the EDW cluster:

Trial License has expired - The license was tied to a trial period, which has expired.

Missing or Invalid License -The LICENSE file specified in the athttpd.conf file on some node is not valid or is missing.

EDW configured for more cores then the license allows - The number of physical cores that the cluster is configured to use is larger than the licensed limit.

Product is operating under a Trial License and evidence of clock tampering has been noted - the system is running under a trial license and there is evidence that the system clock has been pushed back.

Note: "Cluster size" is defined in terms of physical cores. Many Intel CPUs support hyperthreading in which a single physical core appears to the Linux kernel as two logical cores. The `cluster.xml` file defines how many logical cores each EDW node will use. The EDW licensing logic looks at the `cluster.xml` file and the hardware that the EDW is running on to determine the total number of physical cores that the cluster has been configured to use. If the `cluster.xml` file does not specify a number of cores for an EDW node, it is assumed that it will use all cores on that server.

If you need to resolve a licensing issue, contact IgniteTech Support.

How to Upgrade an Existing SenSage AP License

1. Login to the Linux host where the Deployment Manager is installed.
2. Download the correct license to the Deployment Manager host.
3. Reconfigure the license using the hawkeye-deploy script:

```
hawkeye-deploy --action=updateLicense --targetService=hawkeyeAP \
-- licenseFile=<path to license file>
```

Contacting Technical Support

For additional help, call +1 650 631-2810. Also you can log into the IgniteTech Support web page under Services at for SenSage AP documentation, product downloads, and additional information on contacting support to escalate help on issues that impact your production environment.

1

Introduction

This chapter introduces you to the components of a SenSage AP system and contains these sections:

Overview of SenSage AP System Administration

SenSage AP system administrators work mostly in a Linux environment and should be comfortable using Linux in command-line mode. Administrators should also be familiar with SSH, managing users and groups, file system permissions and management, network configuration, logging, and using such text editors as VI or emacs.

If your SenSage AP deployment collects log data from Windows machines, firewalls, routers, or applications, familiarity with those systems is also required.

SenSage AP system administrators perform the following tasks:

- **Installation and configuration**—Installation and initial configuration of your SenSage AP software is discussed in the *Installation, Configuration, and Upgrade Guide*.
- **Upgrading**—Upgrading your SenSage AP deployment to a newer version is discussed in *Example Upgrade Path* in Chapter 4 of the *Installation, Configuration, and Upgrade Guide*.
- **Administration**—Administration tasks discussed in this manual include:
  - Configuring batch collection of event data—After basic installation, you configure SenSage AP to collect batched event data by installing log adapters and configuring the Collector, Retrievers, and Loaders.
  - Managing the EDW data store—As you configure SenSage AP, you load test data and run test queries against that data. You can also create tables, views, and column filters, manage users and permissions for data access, monitor performance, and archive or retire data.
  - Managing the EDW instance—You configure and manage clustered nodes in an EDW instance and configure Nearline storage devices.
  - Manage Near-line storage—you can archive event data stored in the EDW to a variety of Nearline storage devices such as EMC Centera, and Remote NFS (Network) or CIFS (Common Internet) File System.
  - Backing up and restoring event data and raw log files.
  - Managing users, permissions and authentication—Permissions for your SenSage AP deployment and its users must be managed carefully to limit access to information and ensure a secure environment.
  - Managing the report cache—You manage the retention of cached report data.
  - Network configuration—You configure network connections between SenSage AP components and the machines or devices from which you are collecting event data.
  - Monitoring system performance and source health—Your SenSage AP software provides tools to monitor the performance of the SenSage AP deployment. You configure these components to issue alerts when performance is diminished or event data collection is interrupted.

Overview of SenSage AP Components and Processing

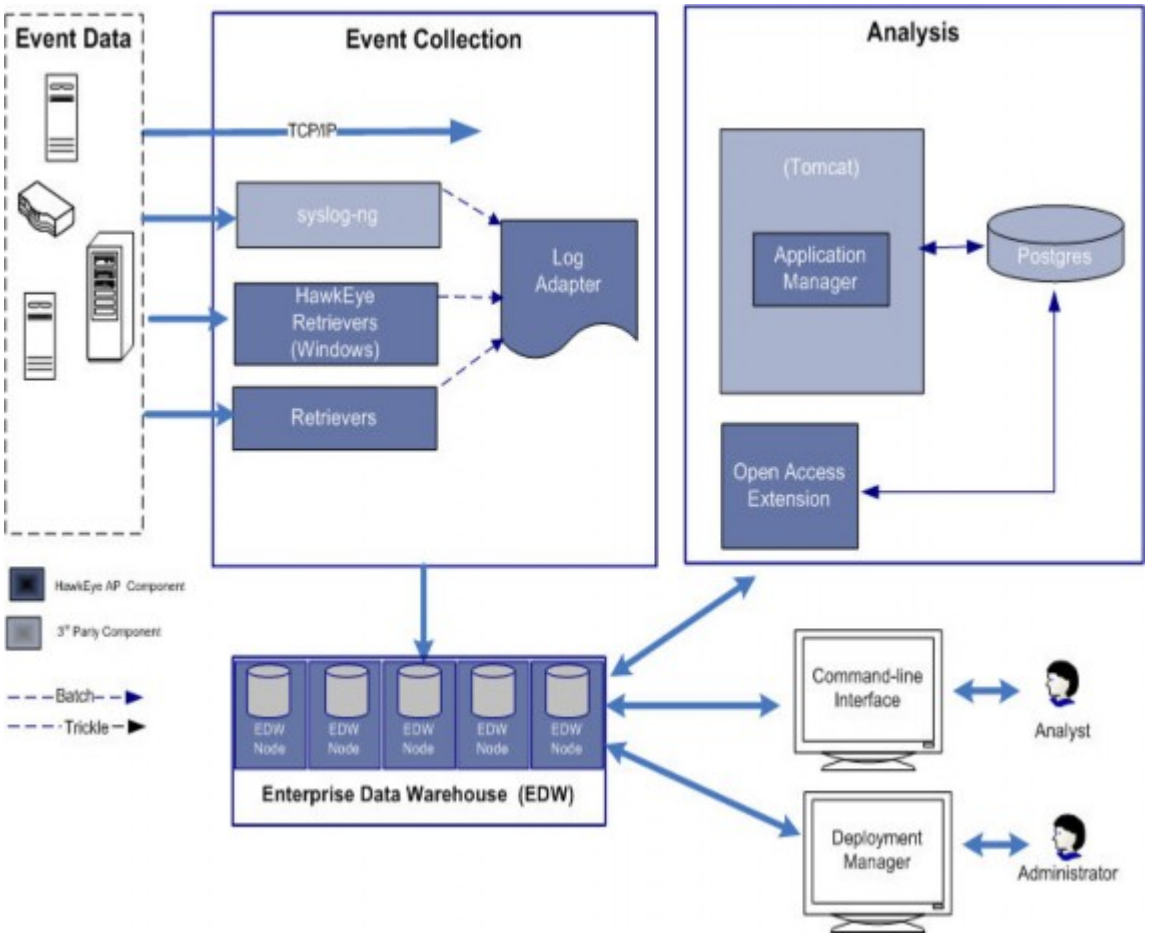
The SenSage AP system supports the industry's most comprehensive event management capabilities. Its patented data model and compression technology facilitate online storage of massive volumes of data across an entire network. The SenSage AP architecture enables rapid querying of and visibility into security threats.

Linear scalability through clustering provides the flexibility needed to support new applications and devices as requirements growtegration capabilities allow SenSage AP to use external enterprise authentication authorities to validate and manage SenSage AP users and external Nearline storage devices to archive SenSage AP data addition, you can forward alerts to enterprise security management products, pagers, phones, and portals.

A SenSage AP system comprises the following core components:

- **Event Data Warehouse (EDW)**—stores event data from multiple sources in a scalable and highly compressed format. Parallel processing enables clustered servers to execute as a single instance, allowing high-speed loading and querying on terabytes of data.
  - **Analyzer**—Command-line utilities communicate with the Analyzer to access stored data and configuration information. The Analyzer lets authorized users view reports, alerts, and dashboards, manage assets, enable and disable Parser rules, run reports, schedule jobs, and manage users and roles. The Application Manager is implemented as a set of services that run within a Tomcat Application Server, which is installed and configured transparently during installation of SenSage AP software.
  - **Open Access Extension (OAE)**—provides access to data stored in the EDW via standard database connectivity tools such as ODBC and JDBC.
  - **LDAP**—provides authentication services for a SenSage AP deployment.
- **Collector**—pulls event data from disparate sources. It uses retrievers that define how the Collector obtains batched event data from event sources and log adapters that define how incoming batched data is parsed and stored in the EDW. The Collector uses a file system retriever to pull data that is stored in syslog-ng, the industry standard in receiving event data from internal and external sources.
- **Syslog-ng**—The industry standard in bifurcating data into the batch modes.

Figure 2: SenSage AP Architecture and Data



The above figure illustrates the complete SenSage AP architecture. As illustrated, events enter the SenSage AP system from any of many external systems, such as network devices and software applications. Events enter in the following way:

- Batched—Events are collected from log files and other event repositories maintained by network devices, operating systems, and software applications.
- The Collector polls a data source or repository to retrieve event data, which it loads into the EDW. The EDW makes the event data available to the Analyzer for reporting. Administration users can access EDW data using the Deployment Manager.

For more information:

- To learn more about configuring retrievers and the Collector, see the chapter **SenSage AP Collector Configuration** in the *Collector Guide*.
- To learn more about administering the Collector, see Administering the Collector on page 145.
- For more information about the EDW and SenSage AP SQL:
  - see, chapters Managing Sensage AP on page 25 and Sensage Loading, Querying, and Managing the EDW on page 61 of this guide.
  - see, chapters **SenSage AP SQL** and **SenSage AP ConsoleSenSage AP SQL Functions** of the *Event Data Warehouse Guide*.

EDW Architecture

About the Event Data Warehouse

The Event Data Warehouse (EDW) is a database built for and dedicated to loading, storing, and analyzing log data. The software uses sophisticated, application-level clustering to perform all load and query tasks fully in parallel across an arbitrary number of hosts. This architecture allows users to load and query massive data volumes in a single, logical database instance without partitioning. The EDW uses a proprietary data model that achieves high levels of compression, while still making all data fully available to query.

Software for the EDW

The EDW software is typically installed across five or more hosts, which make up an EDW instance. All log data is loaded, in parallel, across each host. This ensures that data loads at the fastest possible rate (given the number of hosts in your EDW instance) and provides for high availability and data redundancy.

The EDW instance is known as an *application-level cluster*; that is, any work unit received by one host must be executed across all hosts.

In essence, the EDW is deployed so you can:

1. Load log data into it.
2. Query log data out of it.

The EDW command `atload` performs the loading process in a batch mode from log data that is collected from some other source. To load the data, you need a PTL file that will draw out and reorganize the information from your log files into a format that the EDW is able to understand and load. IgniteTech can make a large number of log adapters available. Please contact your sales representative for details.

For redundancy, two copies of every piece of data is included across the file system—for example, in a five-host EDW instance, data from a single load may appear distributed in the following way:

Load01	Primary	Load01a	Load01b	Load01c	Load01d	Load01e
	Secondary	Load01e	Load01a	Load01b	Load01c	Load01d

After you use `atload` to load data into the EDW, you can use the `atquery` command to retrieve data from this distributed system.

2

Managing Sensage AP

This chapter provides information on how to use the Deployment Manager to monitor, configure, and manage SenSage AP. Also included are utility tools for general tasks that support configuration and management.

- Logging into SenSage AP on page 25, next—Describes how to log into SenSage AP.
- Monitoring SenSage AP on page 26—Describes Deployment Manager Dashboard options that allow you to monitor your SenSage AP system.
- Modifying Installation Configuration Settings on page 40—Describes how to modify current installation settings.
- Configuring a SenSage AP Deployment on page 44—Describes how to configure a SenSage AP deployment using the Deployment Manager.
- Managing SenSage AP Operations on page 45—Describes how to perform basic SenSage AP operations such as start up and shut down.
- Administering Deployment Manager Users on page 46—Describes how to administer users of the Deployment Manager, including adding and deleting existing users and resetting passwords.
- Using Utility Tools on page 48—Describes command-line options common to utility tools and how to use specific tools such as: `cldiff`, `clssh`, `clsync`, `cltop`, and `clhosts`.

Logging into SenSage AP

The Deployment Manager allows you to monitor the state of your SenSage AP deployment clusters and components, view current status, analyze anomalies, and make configuration and deployment changes to the SenSage AP system.

To access the Deployment Manager, enter the URL in your browser; when you see the Sign In Screen, enter your username and password.

Note: The URL to access the Deployment Manager was noted during the installation of the SenSage AP product in the format `http://<Server_name_or_IP_address>:8080`, at the end of the `install.sh` instructions.

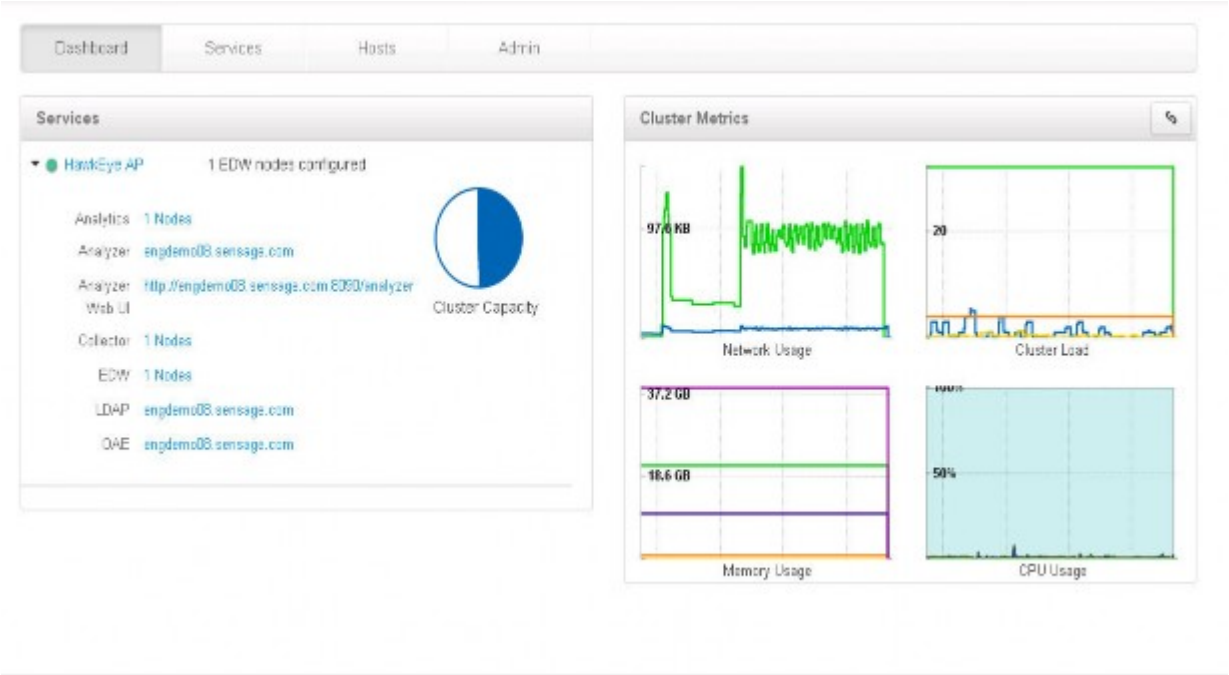
Once your credentials are authenticated, the dashboard is displayed as shown in the figure Figure 3 on page 26.

Monitoring SenSage AP

To the left of the dashboard is a summary of the nodes and the components that make up your deployment. You can start and stop any of the components from the dashboard. You can browse quickly through the display of icons beside each component. If a "Start" icon is displayed beside a component, this can alert you that the component is not in operation.

From this dashboard, you monitor your cluster by viewing the Cluster Metrics graphs displayed to the right, which show the current usage for the network, cluster load, memory and CPU. Note that clicking on the icon in the corner of the Cluster Metrics area pulls up different graph timetables for your selection.

Figure 3: Deployment Manager Dashboard



For details on Cluster Metrics graphs, see Metrics on page 28.

Two tabs that allow you to monitor specific parts of your deployment:

- **Service tab** - Shows the status of the services in your SenSage AP deployment.

Note: In addition, this tab also contains a configuration link lets you change the configuration settings from the initial installation. For details, see Modifying Installation Configuration Settings on page 40.

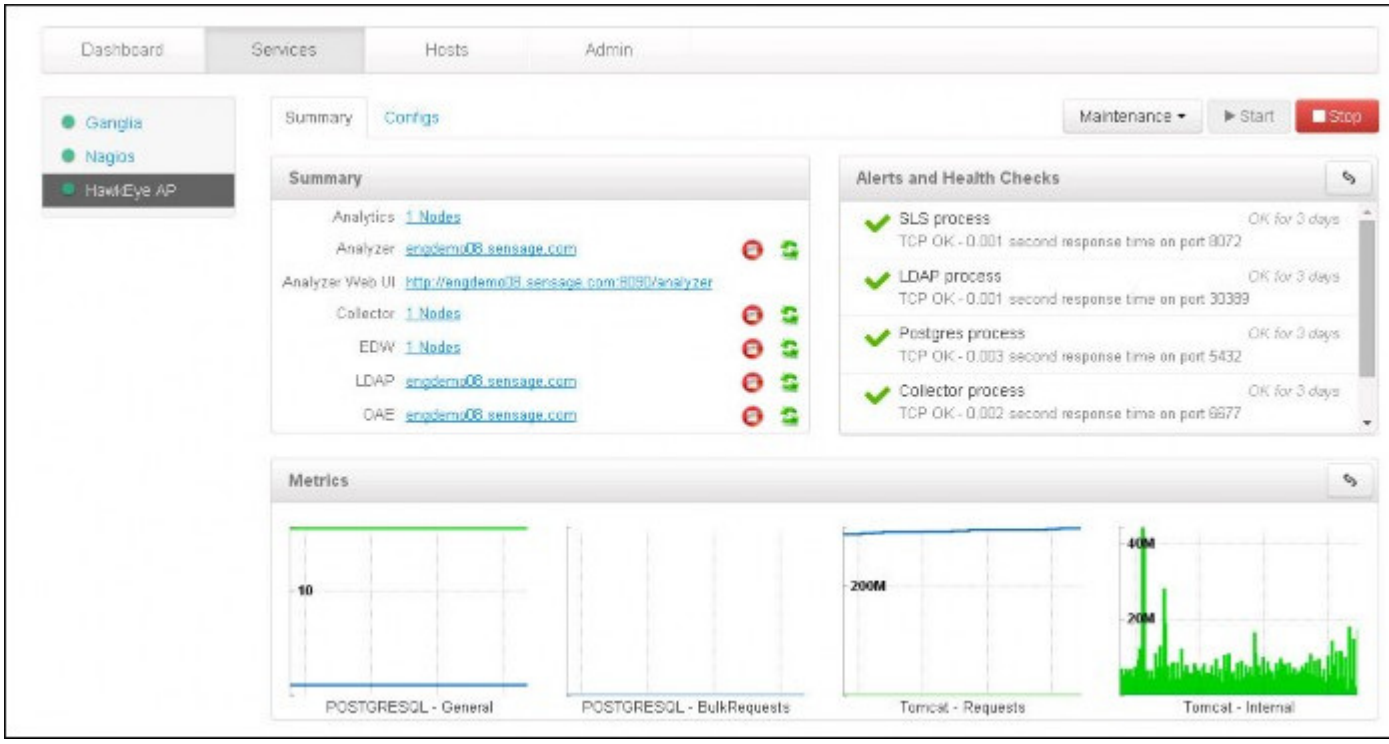
- **Host tab** - Shows the status of components in your SenSage AP deployment on a host-by-host basis.

Each of these tabs and their options are described in the following sections.

Services Tab

Click the **Services** tab to view the metrics for a service in a SenSage AP Cluster, as shown in the screenshot below. The default selection for viewing is "SenSage AP", which is displayed on the menu bar. The two other selections are "Nagios" and "Ganglia", which provide the metrics and monitoring capabilities of the SenSage AP system.

Figure 4: Services Screen



Summary

The Summary area to the left displays the nodes and the components that make up your SenSage AP deployment and the status of each. You can individually start and stop any of the components using the start and stop icons. If any component or service is stopped, the start icon will be displayed. and like the dashboard, this can alert you to a component that is not operating.

Alerts and Health Checks

The Alerts and Health Checks display the status of your deployment. All green checks beside each component indicate a healthy run state; otherwise:

- If a red "X" appears beside any component, this indicates a component that is stopped or failed (in which case your attention is required).

Note: The green check and red "X" does not apply to the Analytics component since it does not have its own process.

- If a yellow triangle appears beside the component, this indicates that a more up-to-date release of the component is available on the support site. You will need to stop the deployment; once you have updated the component, then you can start the deployment.

Be sure to also choose "Nagios" and "Ganglia" on the menu bar to check if your monitoring system is functioning. Again, you will see a red "X" if any component for these services are not running.

Above the Alerts and Health Checks, you can start and stop the entire deployment using the Start and Stop button.

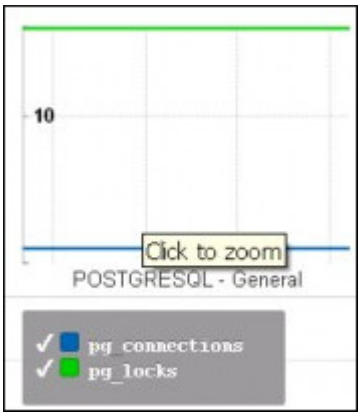
Note: If the deployment is running, you will see both the Restart button (green top and bottom curved lines) with the red circular Stop button; however, if any components in the deployment are stopped, you will see the triangular **Start** button instead of the red circular **Stop** button.

Metrics

The Metrics area on the Service page displays usage graphs for SenSage AP Postgres and Tomcat services described below. When you click Analyzer link in the Summary page, you will more metrics that includes host usage for CPU, Disk, Memory, Network and Processes.

PostgreSQL-General

Figure 5: PostgreSQL-General Sample Graph

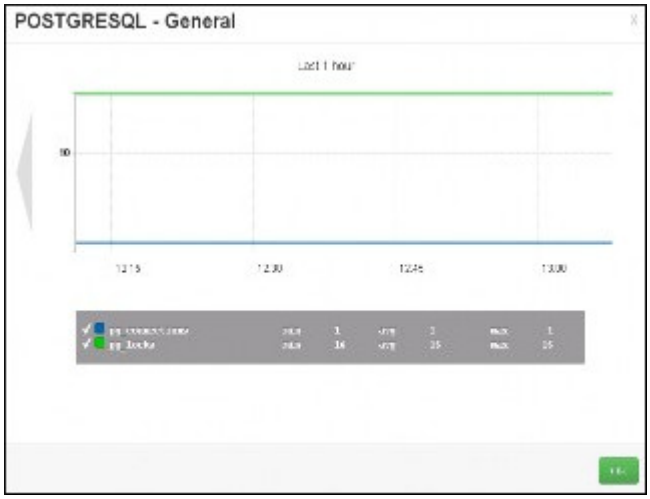


Note: Observe the following:

- Blue graph—Represents the number of connections to PostgreSQL against time.
- Green graph—Represents the number of locks held by connections at that point of time.

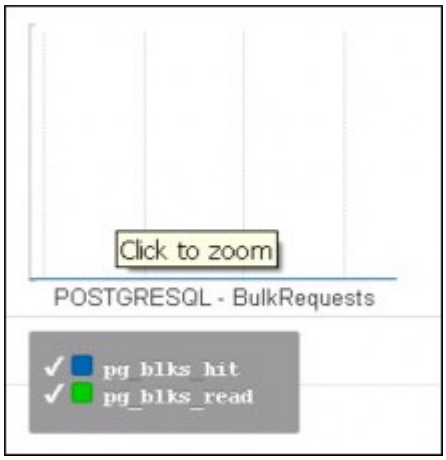
When you click zoom, a more detailed view of this graph is displayed as shown below:

Figure 6: PostgreSQL-General Detail Sample Graph



PostgreSQL-BulkRequests

Figure 7: PostgreSQL-BulkRequests Sample Graph

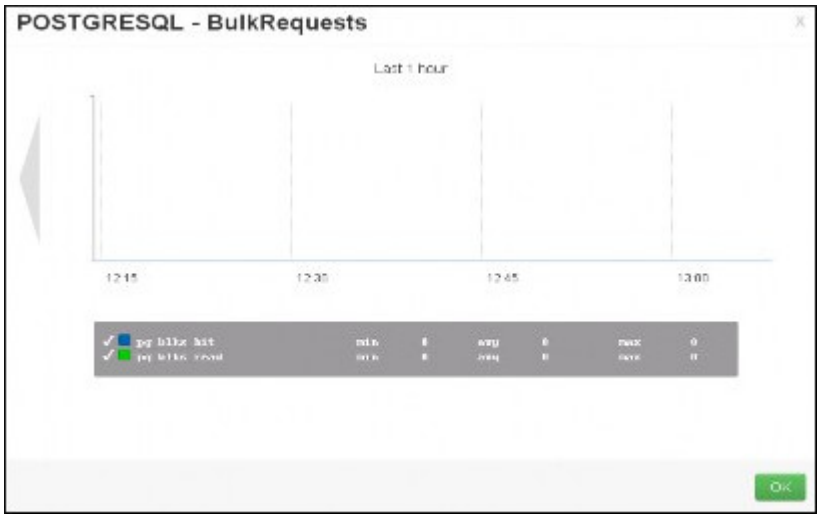


Note: Observe the following:

- Blue graph—represents the number of bulk requests.
- Green graph—represents the number of bulk requests read.

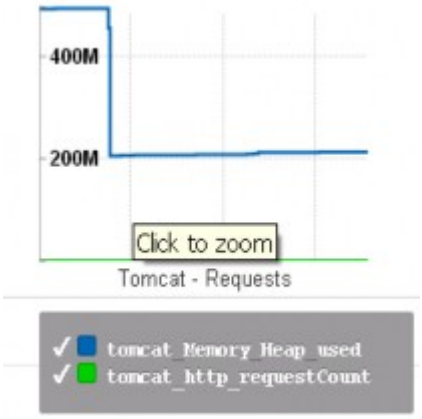
When you click zoom, a more detailed view of this graph is displayed as shown below:

Figure 8: PostgreSQL-General Detail Sample Graph



Tomcat Usage

Figure 9: Tomcat Usage Sample Graph

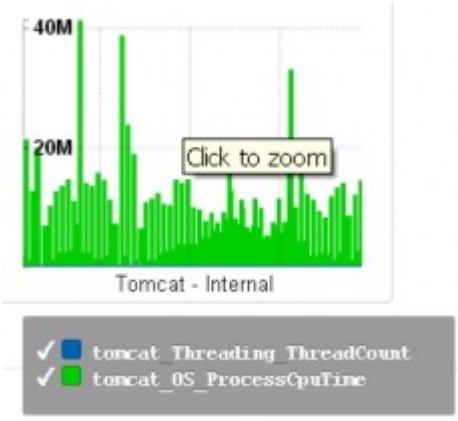


When you click zoom, a more detailed view of this graph is displayed as shown below:

Figure 10: Tomcat Usage Detail Sample Graph

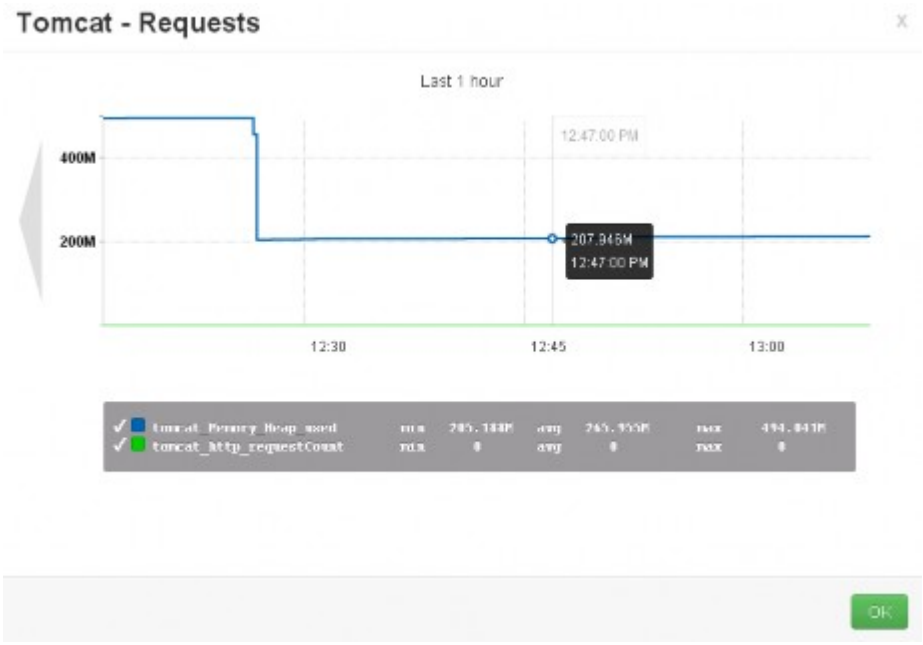
Tomcat Internal

Figure 11: Tomcat Internal Sample Graph



When you click zoom, a more detailed view of this graph is displayed as shown below:

Figure 12: Tomcat Internal Detail Sample Graph



Hosts Tab

Click the Hosts tab to view the status of components in your cluster on a host-by-host basis. Note that the Host screen is displayed depicting either a single host or a multi-host environment. The screenshot below illustrates a multi-host environment.

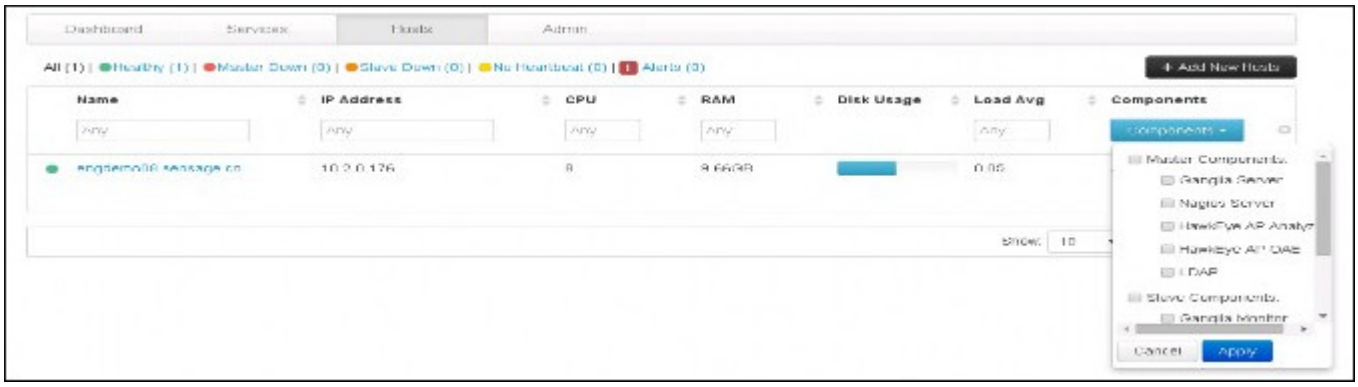
Figure 13: Hosts Screen

On the Host screen the top row displays the number of hosts in the cluster, indicating whether the hosts are healthy, what masters/slaves, if any, are down, if no heartbeat was detected, and if any of the hosts have an alert (meaning that one or more components are not in service.)

Beneath the Components drop-down button, you can mouse over an applicable row that represents the components of the host that you want to display. This is shown in the screenshot below.

When you click the components drop-down button itself, you will see a list of all possible master and slave components in the deployment as shown in the screenshot below. You can place a check beside the component to which you want to add to a specific host and click **Apply**.

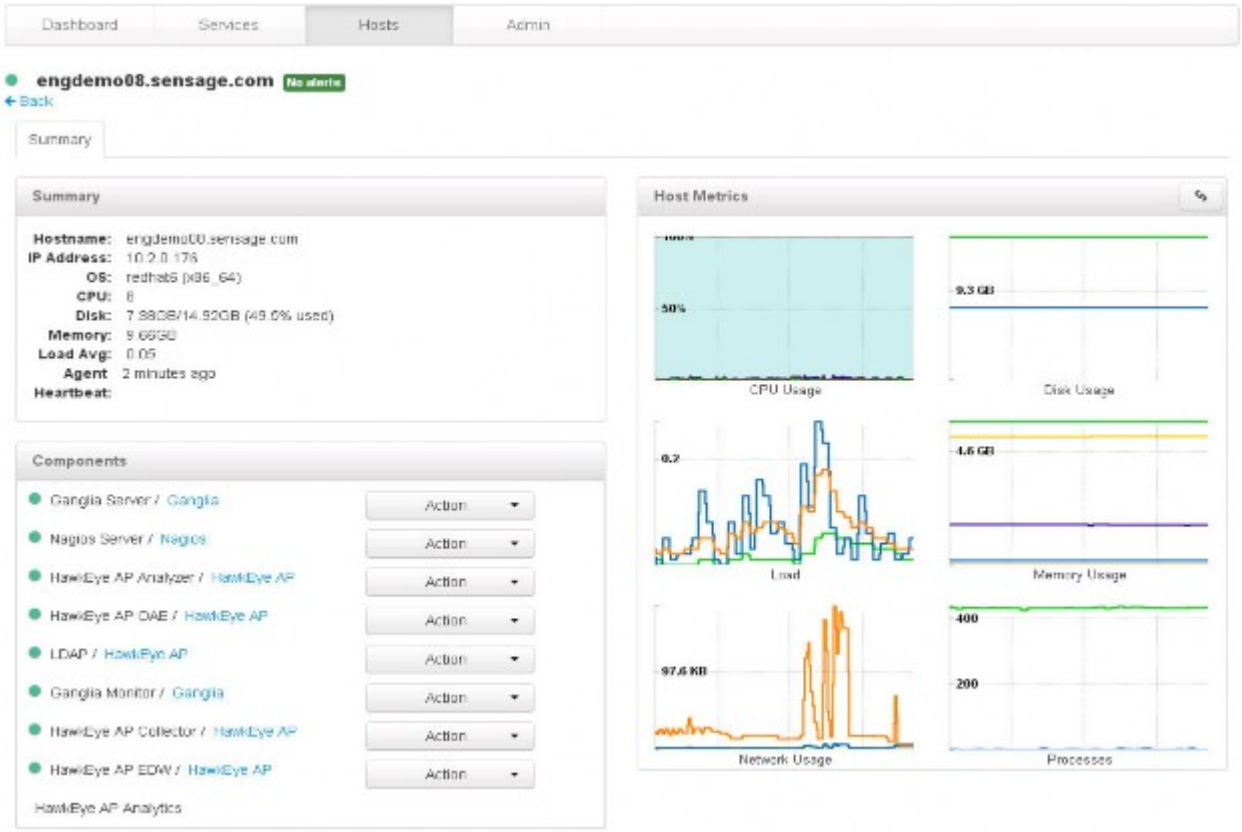
Figure 14: Components Drop-down on Hosts



Host Summary

To display more details on a specific host, click the host's link under Name on the Hosts screen. Details on the host are displayed, including a Summary that contains the hostname, IP address, OS, CPU, Disk, Memory, Load Average, Agent, and Heartbeat. The Agent refers to when the host's metrics was last updated.

Figure 15: Host Summary Display



Components

The components area displays the Components installed on the host. You can click the Action drop-down to start or stop individual components.

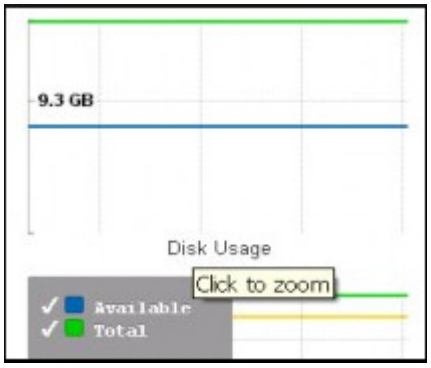
Host Metrics

The Host Metrics area displays information on the SenSage AP host. Each graph is described below:

Disk Usage

This metric is used to monitor disk usage.

Figure 16: Disk Usage Sample Graph



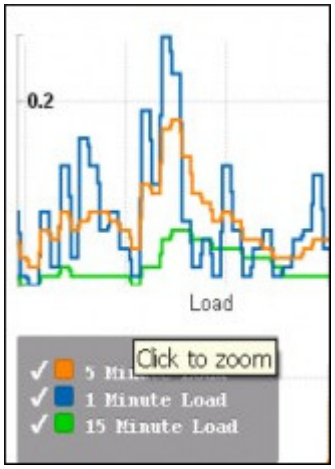
When you click zoom, a more detailed view of this graph is displayed as shown below:

Figure 17: Disk Usage Details Sample Graph

Cluster Load

Cluster Load

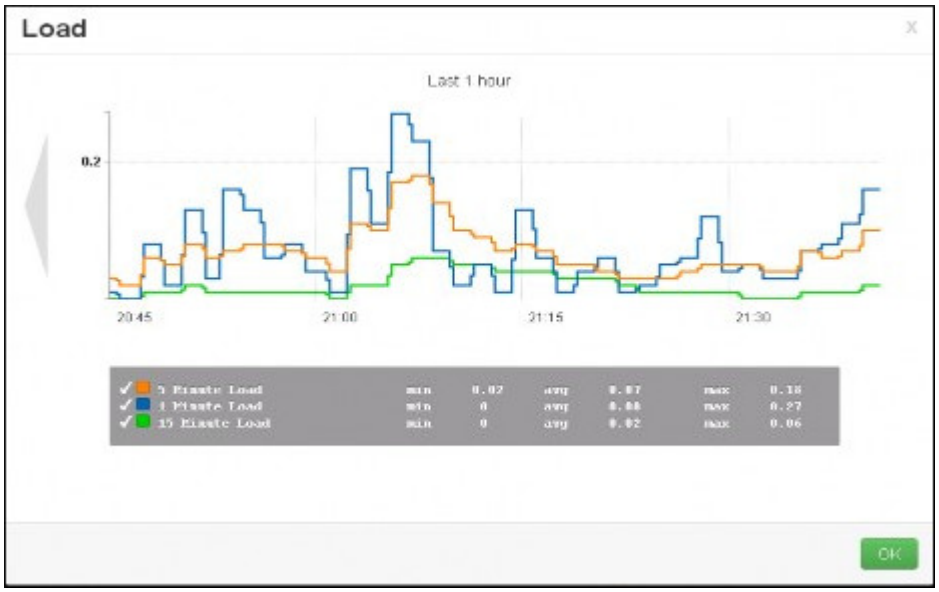
Figure 18: Cluster Load Sample Graph



This metric is used to detect the load on the system.

When you click zoom, a more detailed view of this graph is displayed as shown below:

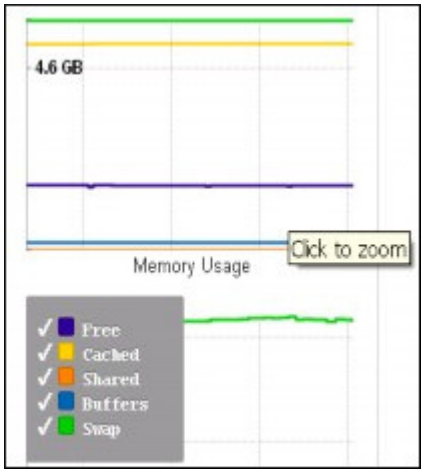
Figure 19: Cluster Load Detail Sample Graph



Memory Usage

This metric is used to represent the different memories and their usage at the point in time.

Figure 20: Memory Usage Sample Graph

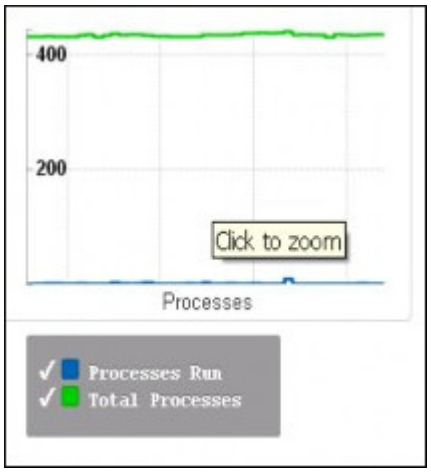


When you click zoom a more detailed view of this graph is displayed as shown below:

Figure 21: Figure 2-19: Memory Usage Detail Sample Graph

Processes Usage

Figure 22: Processes Sample Graph



This metric is used to represent the different processes and their usage at the point in time

Note the following:

- Blue—Represents the current number of processes occurring at this point in time.
- Green—Represents the total number of processes available in the system.

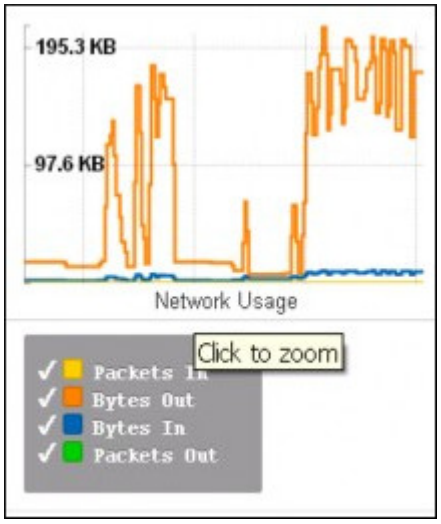
When you click zoom a more detailed view of this graph is displayed as shown below:

Figure 23: Figure 2-21: Processes Detail Sample Graph

Network Usage

This metric is used to represent network traffic through the packets and bytes that are sent and received.

Figure 24: Network Usage Sample Graph



This metric is used to represent network traffic through the packets and bytes that are sent and received.

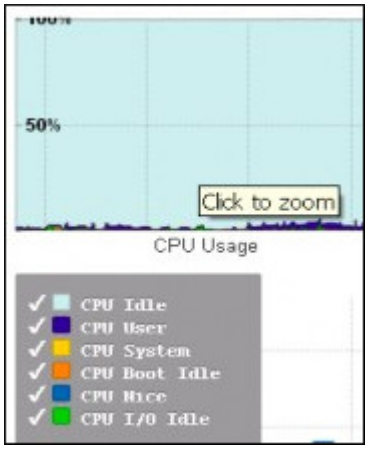
When you click zoom a more detailed view of this graph is displayed as shown below:

Figure 25: Figure 2-23: Processes Detail Sample Graph

CPU Usage

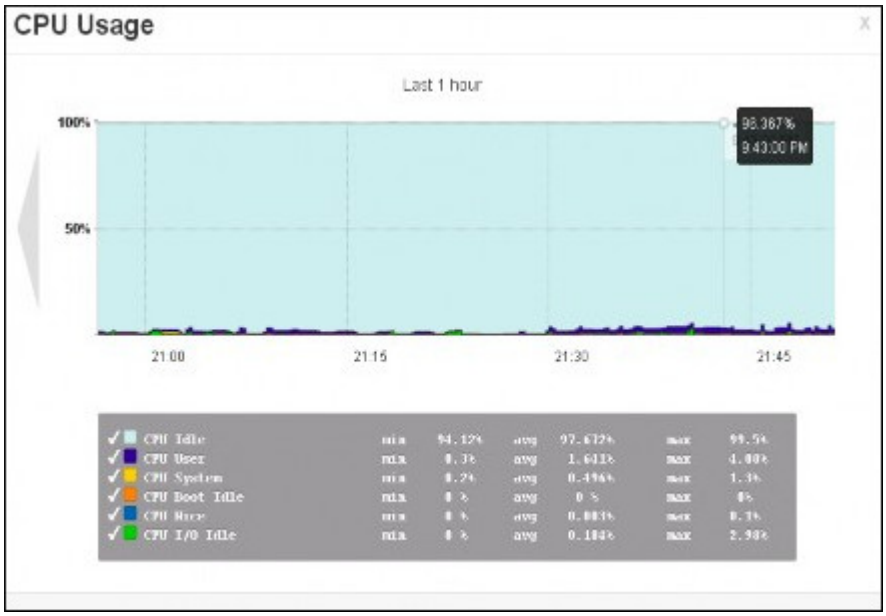
This metric is used to represent the different CPU metrics such as CPU idle time, boot time, etc.

Figure 26: CPU Usage Sample Graph



When you click zoom a more detailed view of this graph is displayed as shown below:

Figure 27: Figure 2-25: CPU Usage Detail Sample Graph



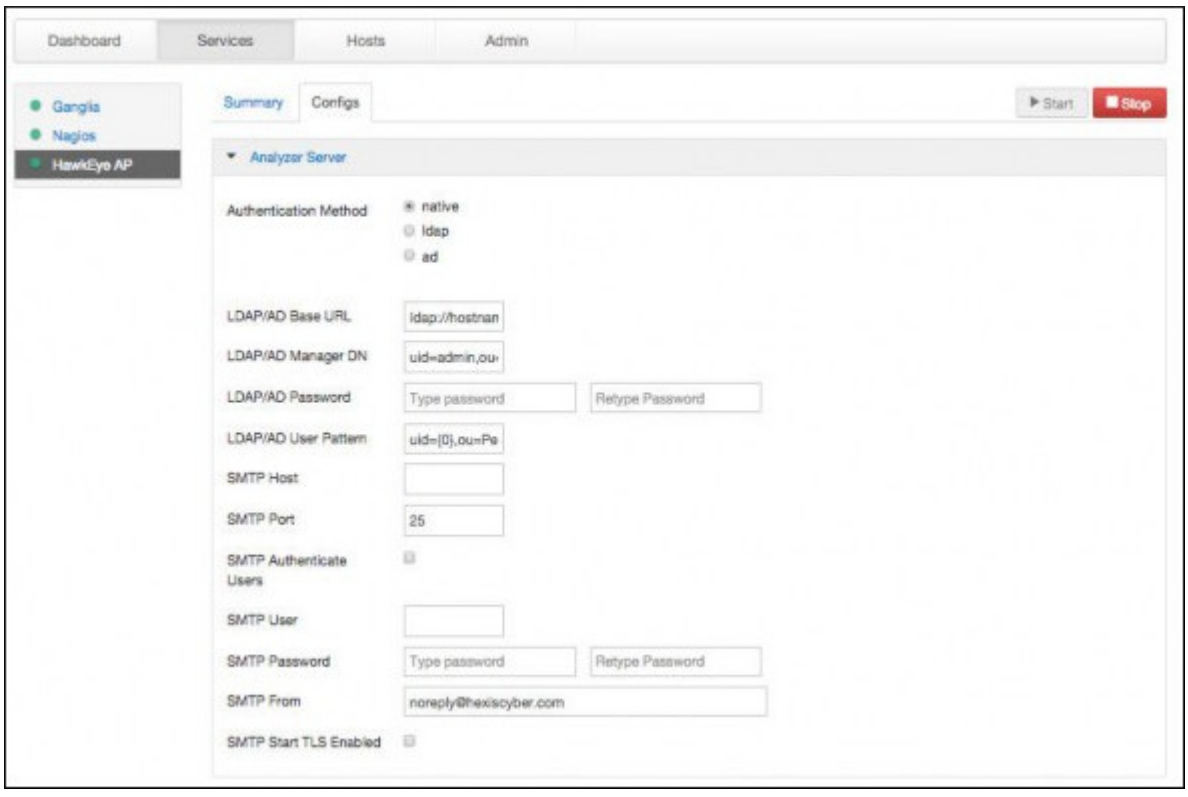
Modifying Installation Configuration Settings

The Config tab on the Services page allows you view existing parameter settings and change those affected by any updates you have made to your deployment. Click the **Configs** link shown below to bring up the Configuration page (Figure 2-27)

Figure 28: Configuration Link (NEW SCREENSHOT)

Scroll through the Configuration page to find the parameter settings that you need to change. Description for the parameters are available when you hover your mouse over each of the parameter fields. Those parameters that you cannot change are grayed out or left blank.

Figure 29: First Section of Configuration Page



The following table describes the Analyzer configuration parameters on the configuration page. These parameters

Configuration Parameter	Description
LDAP/AD Base URL	This is the LDAP/AD Base URL for the Active Directory Server in a single-server environment or the base URL for the first server defined in a multi-server environment (in which the remaining servers are defined directly in the properties file.
LDAP/AD Server URL	This is the LDAP/AD Server UURL for the Active Directory Server in a single-server environment or the base URL for the first server defined in a multi-server environment (in which the remaining servers are defined directly in the properties file.

LDAP/AD Manager DN	This is the distinguished name that uniquely identifes an entry in the LDAP Directory Information Tree and in this case represents the Common Name (CN), Organizational Unit (OU), and Domain Component (DC). This entry cannot be changed. For example:
	`CN=analyzeradmin,OU=hexis,DC=customer,DC=com`
	`CN=analyzeradmin`

LDAP/AD Username	The user name of the Analyzer Administrator.
LDAP/AD Password	The password of the Analyzer Administrator.
LDAP/AD Password to authenticate	The LDAP password to authenticate with LDAP/AD-NEED MORE DETAIL
LDAP/AD User pattern	The required user pattern in the following format: CN=users,DC=dev-ap,DC=local
Test Connection	Used to check whether the credentials (defined above) are accurately configured.
SMTP Port	Port on which the SMTP server listens and is used for emails.
SMTP Authenticate Users	The SMTP login user that is used to send emails.
SMTP Password	The password for the SMTP user.
SMTP From	The originating user for SMTP
SMTP From Email Address	The originating email address for SMTP
SMTP Start TLS Enables	Used to enable secure connections to send emails.

Configuring SSL for the Deployment Manager

Connections to the Deployment Manager are protected by SSL, using a self-signed SSL certificate that is generated automatically during the installation or upgrade process. You can replace the automatically generated self-signed SSL certificate with your own certificate. For more information, see Using a Custom SSL Certificate for Deployment Manager on page 42. The self- signed SSL certificate is valid for one year. You can replace an expired certificate for the Deployment Manager (see Replacing an Expired Self-signed Certificate for Deployment Manager on page 42) and the Analyzer (see Using a Custom Certificate for Analyzer on page 43).

Additionally, the Deployment Manager manages AP components on each host of the AP cluster via SSL connections, using SSL certificates that were generated during installation on each AP host in the cluster. These certificates are set to expire in 3 years. When the certificates expire, the Deployment Manager will be unable to manage the hosts in the AP cluster, although the AP functions will continue to work. To re-enable the Deployment Manager's capability to manage the AP cluster after the SSL certificate expires, follow the procedure described in Replacing Expired SSL Certificates for Ambari Agent Hosts on page 43.

Using a Custom SSL Certificate for Deployment Manager

To use your own SSL certificate on the Deployment Manager, perform the following steps from the Deployment Manager host:

1. Stop the Ambari server:

   ```
   ambari-server stop
   ```

2. Copy the SSL certificate to `/var/lib/ambari-server/resources/ssl.crt` and the private key to `/var/lib/ambari-server/resources/ssl.key`.
3. Go to `/var/lib/ambari-server/keys` and delete the following: `https.crt` and `https.key`.
4. Issue the following command to set up the certificate:

   ```
   ambari-server setup-security
   ```

   Note: If the SSL key was password protected, you are prompted to enter it.

5. Start the Ambari server:

   ```
   ambari-server start
   ```

Replacing an Expired Self-signed Certificate for Deployment Manager

1. Stop the Ambari server:

   ```
   ambari-server stop
   ```

2. Go to `/var/lib/ambari-server/keys` and delete the following: `https.crt` and `https.key`.
3. Go to `/var/lib/ambari-server/resources` and delete the following: `ssl.crt` and `ssl.key`.
4. Issue the following commands:

   ```
   fqdn=’hostname -f’
   openssl req -x509 -nodes -newkey rsa:2048 -days 365 \
   -keyout /var/lib/ambari-server/resources/ssl.key \
   -out /var/lib/ambari-server/resources/ssl.crt \
   -subj "/CN=$fqdn/O=Hexis Cyber Solutions./C=US"
   ```

5. Issue the following command to set up the certificate:

   ```
   ambari-server setup-security
   ```

6. Start the Ambari server:

   ```
   ambari-server start
   ```

Using a Custom Certificate for Analyzer

If you have a private key, an SSL certificate, and a certificate bundle from a Certificate Authority (CA), you can use OpenSSL to create a certificate keystore that Tomcat can use for Analyzer access.

The following components are required to create a keystore for Tomcat:

- OpenSSL
- Private key with a key file extension from CA
- SSL certificate file from CA
- java keytool

Note: keytool is only available in Oracle. keytool is NOT available in open-jdk)

Before you begin, note the assumptions which are made in the procedure that follows:

File Type	File Name
SSL Private Key File name	ssl.key
SSL Cert File name	ssl.cnt

To access Analyzer using custom certificate, follow these steps:

1. Copy the `ssl.crt` and `ssl.key` files to the following directory: `/opt/hexis/hawkeye-ap/etc/customkeys`
2. Restart Analyzer from Ambari web or hawkeye-deploy.
3. Browse Analyzer URL.

   The new SSL certificate is now visible in the browser.

Replacing Expired SSL Certificates for Ambari Agent Hosts

1. On the Deployment Manager host, perform the following steps:
   a. Stop Ambari server:ambari-server stop
   b. Go to `/var/lib/ambari-server/keys`.

      You will find a `.csr` and a `.crt` file for each of the ambari-agent hosts listed.

   c. Delete these files for each host.
   d. Additionally, delete `ca.crt` and `ca.key`.

Note: DO NOT delete `ca.config`.

e. Go to /var/lib/ambari-server/keys/db and issue the following command:

```
cat /dev/null/ > index.txt
```

2. On each host in the AP cluster (which may include the Deployment Manager host), perform the following steps:

   a. Stop Ambari agent:

   ```
   ambari-agent stop
   ```

   b. Go to `/var/lib/ambari-agent/keys` and delete all the files in this folder (except for `ca.config`).

   c. Start Ambari agent:

   ```
   ambari-agent start
   ```

3. On the Deployment Manager host, start Ambari Server:

   ```
   ambari-server start
   ```

Configuring a SenSage AP Deployment

This section describes how to configure a SenSage AP deployment: using the Deployment Manager Dashboard:

- Adding a New Host on page 44

Adding a New Host

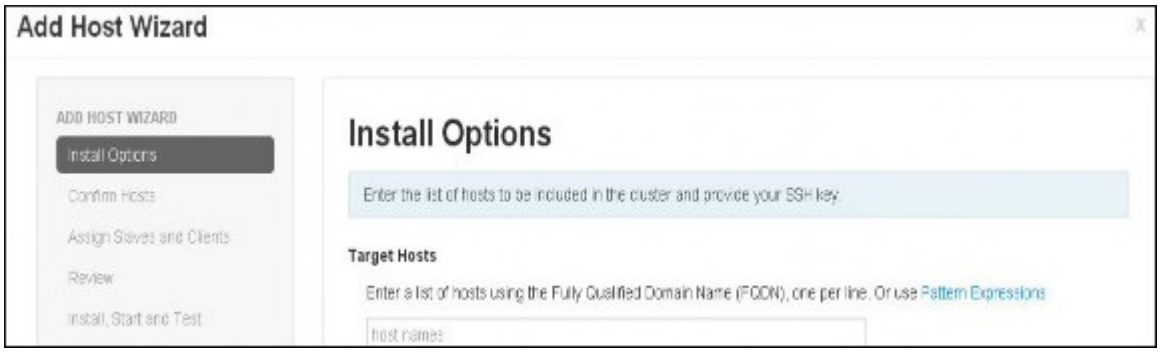
To add a new host:

1. Click the Hosts tab to go the Hosts screen:
2. Click the Add New Hosts button.
3. The Add New Hosts screen is displayed as shown in Figure 2-28.
4. Complete the screen by adding the names of the hosts that you want to add by supplying the FQDN of each host.
5. To allow the Deployment Manager GUI to access the SSH Private Key that you set up (so it can securely access all the hosts in the system), provide the key that is in the file `~/.ssh/id_dsa` on the Deployment server for the root user. Copy and paste the key in the field provided, or click **Choose File** if you want the GUI to open the file and get the contents for you.

Note: If you are using IE 9, the Choose File button may not appear. Use the text box to cut and paste your private key manually.

6. Click **Register and Confirm**.
7. Follow instructions beginning with the *Confirm Hosts* section of the *Installation, Configuration, and Upgrade Guide*.

Figure 30: Add Host Wizard



Managing SenSage AP Operations

This section describes how to perform specific operation tasks:

Starting, Stopping, and Restarting Your SenSage AP Deployment

This procedure stops, starts, or restarts the EDW and the other SenSage AP components on the host from which you are running Deployment Manager.

Note: Deployment Manager manages specific SenSage AP files, which means any edits to these managed files are overwritten (usually when the service is started through Deployment Manager). For a list of files managed by Deployment Manager, see Appendix D.

To start and stop your entire SenSage AP deployment:

1. Log into Deployment Manager.
2. . From the Deployment Manager Dashboard, select the Services tab.
3. Above the Alerts and Health Checks, start or stop the entire deployment by selecting the **Start** or **Stop** button.

Note: If the Deployment Manager is running, you will see both the Restart button (green top and bottom curved lines) with the red circular Stop button; however, if any components in the deployment are stopped, you will see the triangular Start button instead of the red circular Stop button.

Starting, Stopping, and Restarting SenSage AP Components

To start, stop and restart SenSage AP components:

1. Log into Deployment Manager.
2. From the Deployment Manager Dashboard, select the Services tab.
3. In the Summary area of the Services tab, start and stop any of the components using the start and stop icons. If any component or service is stopped, the start icon will be displayed. For details on status icons, see Alerts and Health Checks on page 27.

Listing all EDW Instances

To display configuration details about the EDW and its instances:

1. Log into Deployment Manager.
2. From the Deployment Manager Dashboard, select the Services tab.
3. On the Services screen, find EDW in the Summary section and click the nodes link. The system displays configuration details the EDW and each of its instances.

Disabling or Deactivating an EDW Instance

To disable or deactivate an EDW instance:

1. Log into Deployment Manager.
2. Click the Hosts tab and in the summary section of the screen click the Host name link to which the EDW instance belongs.
3. In the Host Summary Display screen, click the Action drop-down and select Stop to disable or deactivate the EDW instance.

Reactivating

To later reactivate the EDW instance, in the Host Summary Display screen, click the Action drop- down and Select Start to reactivate the EDW instance.

Administering Deployment Manager Users

Under the Deployment Manager Admin tab, you can create a new Deployment Manager user, delete an existing user, or change the username or password of an existing user.

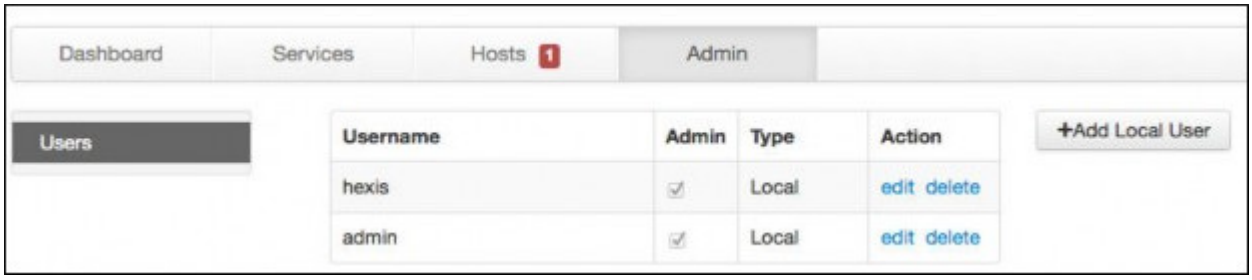
Adding a New User

To add an new user:

1. From the Deployment Manager Dashboard, select the Admin tab.

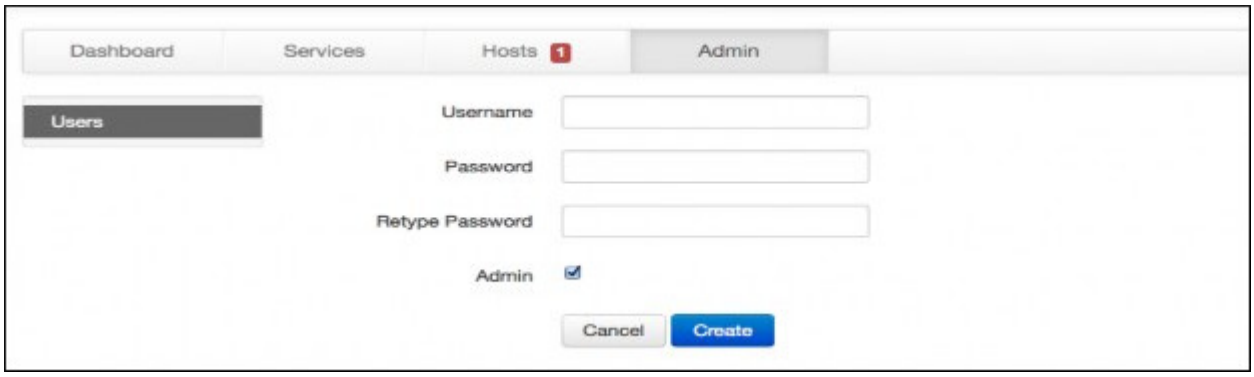
   The current list of users are displayed. To the right, select Add Local User, as shown below.

   Figure 31: Add a User



2. Enter the user information in the fields that are displayed. Check Admin and click Create.

   Figure 32: Username and Password



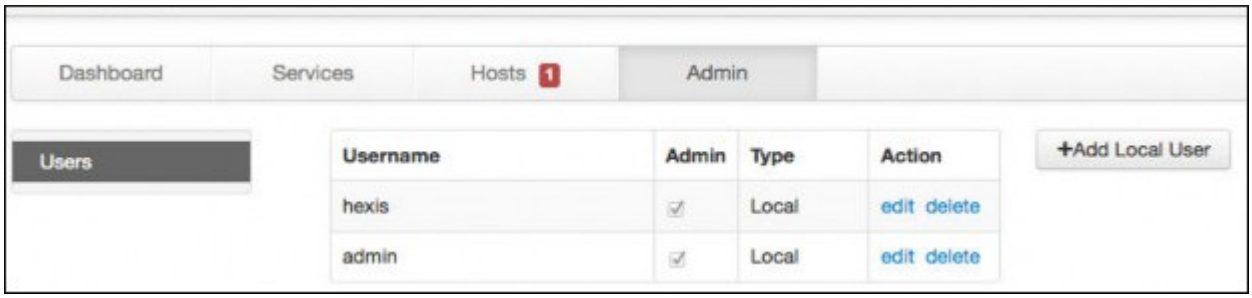
   The user is now added to the Deployment Manager system.

Deleting or Modifying an Existing User

To delete or modify an existing Deployment Manager user:

1. From the Deployment Manager Dashboard, select the Admin tab.

   Figure 33: Edit/Delete a User



2. To delete an existing user, browse through the current list of usernames until you find the user that want to delete. Go to the Action field and click delete.

The user is deleted and can no longer able to access Deployment Manager.

3. To modify an existing user, browse through the current list of usernames until you find the user that you want to modify Go to the Action field and click edit.
4. To change the password of a local user: retype the password; save your changes.

To change the password of an existing user:

1. Login as any admin user.
2. Go to the Admin tab; click edit in the row that lists the username whose password requires the change.
3. In the current password, enter the password of the user you are logged in as.
4. In the new password, enter new password for the target user.
5. Click the Save button when you are done.

Note: A user without Admin privileges cannot change the password for any user, including his or her own.

Using Utility Tools

This section provides details on using utility tools, providing general usage rules and details on specific tools to perform common tasks. It includes the following sections:

- General Syntax for Utility Tools on page 48
- Utility Tool Descriptions on page 51

General Syntax for Utility Tools

Each utility tool described in this section has arguments and options. The options fall into two categories: those generally available to all utilities, and those available only to specific utilities. As shown below, you specify an option by preceding its name with two dashes, for example: `--hosts`. To specify a value for the option, you can precede the value either with a space or an equals sign (=). Therefore, both of the following examples are valid:

```
--hosts=my_machine
--hosts my_machine
```

Note: If you specify the equals sign (=), do not put a space on either side of it.

This section describes the general syntax used with utility tools and contains the following sections:

Options Default Values

Some options have a default value. For example, the value of the --hosts option defaults to localhost. To use the default value, do not specify the flag on the command line. To use a non- default value, include the flag and specify the desired value.

If the option is not explicitly specified for a setting, the default value is used most cases, the online help for the tools will show you what the default value is by displaying the default value in parenthesis.

Getting Help

--help

Displays usage information.

--longhelp

Displays detailed usage information. When you use the ATLOAD command, you can specify this option together with the `--[no]diskusage` flag to specify the computation of the disk used by the load (which can be slow for large installations). When no is specified with this flag, it means do not perform computation of the disk used by the load.

Editing Parameters

--[no]edit

Invokes editor to enter missing or incorrect parameters; default is --edit. If you specify --noedit, the editor is not opened when parameters are missing or are incorrect.

--editor [<editor>]

Uses specified editor to edit missing or erroneous parameter; defaults to vi. Alternatively, you can specify the value in the EDITOR or VISUAL environment variable.

Control of execution

--debug

Displays debug data while the command is running.

--force

Ignores errors.

--parse-only

Parses and validates the command line options, but does not execute.

--preview

Displays actions that would be executed, but stays on local host.

--[no]progress

Displays progress indicators by default. If you specify `--noprogress`, disables display of progress indicators.

--remote-preview

Displays actions that would be executed, including those on remote hosts.

--verbose=<information_level>

Displays increasingly detailed progress information.

--version

Displays version information.

--[no]welcome

Displays welcome banner with change number; defaults to `--welcome`.

Remote Execution

--prompt-password

Always prompt for passwords; does not cache or reuse passwords from other hosts

--serialize

Executes tasks on one host at a time; does not run in parallel

--timeout=<tseconds>

Expiration time in seconds if a task cannot successfully execute.

Shared Mandatory Fields with Default Values

The configuration utility provides default values for all of these fields./options. If you do not change the default value, you do not need to specify these fields on the command line. Note that SenSage AP requires that resources are placed in consistent locations across all servers.

--gnu-dir

Specifies the valid path on all machines for the directory containing GNU utilities (including tar and gunzip).

--java-dir

Specifies the valid path on all machines for the directory that contains a usable Java JDK.

--rsync-dir

Specifies the valid path on all machines for the directory containing the rsync program.

--openssl-dir

Specifies the valid path on all machines for the directory containing the openssl program.

--ssh-dir

Specifies the valid path on all machines for the directory containing the ssh programs.

Utility Tool Descriptions

This section describes utility tools to perform the following tasks:

Comparing Files or Directory-listings Across the Cluster (cldiff)

This section describes the `cldiff` utility. For each file or directory specified compares the specified file with identically named files on each host in the cluster. For directories, cldiff compares directory listings.

This section contains the following topics:

SYNOPSIS

`cldiff [general_options] [specific_options] <source_item> [<source_item> […]]`

ARGUMENTS

`<source_item>`

You must specify at least one source item. Additional source items are optional. Separate multiple items with spaces. You can specify files, directories, or combinations of both, as the following three examples show:

```
cldiff file1 tmp/file2.txt file3
cldiff dir1 dir2
cldiff file1 tmp/file2.txt dir1
```

If you specify a path without a leading slash (/), the utility resolves the path from the current directory. The utility expects to find the files and directories in the same current directory on the other hosts in the cluster. If the corresponding files are in a different location on the other hosts, use the `--target-dir` option to specify the alternate, remote location.

OPTIONS

[*general_options*]

See General Syntax for Utility Tools on page 48.

`[specific_options]`

--hosts

Executes command on hosts specified in the list; defaults to localhost

--sls-instance

Executes command on every host on which the specified instance is running; value can also be set through the SLS_INSTANCEenvironment variable

--target-dir

The target directory on remote hosts where the copies to be compared are located. Use this option if the location of files to compare are in different locations on the remote hosts than on the localhost.If the remote locations on remote hosts differ by host, use the <host>:<path> syntax for specifyinghosts with the --hosts option described above.

--exclude

When it examines directories for files to compare, the cldiff utility ignores files and directories with base names that match the specified pattern. Separate multiple patterns with spaces or use multiple --exclude options. Enclose the patterns within single quotes, for example: “cldiff -exclude=’**/dsroot/**’/opt /sensage/etc”

--diff-command

The cldiff utility uses the "rsync -n (preview)" command to discover files that are different. When it discovers files that are different, cldiff uses the "diff -c" command to report the differences.

You can modify the way in which differences are reported by specifying the options you want the diff command to run with; for example:

`--diff-command` (diff -c, default)

`--diff-command` (diff -c)

EXAMPLE

Following is an example of the cldiff command executed using the `--exclude` option on a two- node EDW instance named mys ls.

```
[dev-01 latest]# cldiff --exclude='*/dsroot/*' --sls-SLSInstance /opt/sensage/
etc/sls/instance/mys ls
cldiff - clsetup-4.7.0, change #73647;

Initializing (ssh) rooted-01.sensage.com
Initializing (ssh) root@dev-02.sensage.com
dev-01:Done | dev-02:Done
All done.

====[dev-01.sensage.com]====
Identical files:
= /opt/sensage/etc/sls/instance/mysls/PERMISSIONS
= /opt/sensage/etc/sls/instance/mysls/add.xml
= /opt/sensage/etc/sls/instance/mysls/athttpd.conf
= /opt/sensage/etc/sls/instance/mysls/cluster.xml
= /opt/sensage/etc/sls/instance/mysls/docroot
= /opt/sensage/etc/sls/instance/mysls/dsroot
= /opt/sensage/etc/sls/instance/mysls/ldap.conf
= /opt/sensage/etc/sls/instance/mysls/md5
= /opt/sensage/etc/sls/instance/mysls/md5.errors
= /opt/sensage/etc/sls/instance/mysls/sensage_sls_mysls
= /opt/sensage/etc/sls/instance/mysls/shared_secret.asc

====[dev-02.sensage.com]====
Only on "dev-02.sensage.com":
> /opt/sensage/etc/sls/instance/mysls/omnisight.ldif

Identical files:
= /opt/sensage/etc/sls/instance/mysls/PERMISSIONS
= /opt/sensage/etc/sls/instance/mysls/add.xml
= /opt/sensage/etc/sls/instance/mysls/athttpd.conf
= /opt/sensage/etc/sls/instance/mysls/cluster.xml
= /opt/sensage/etc/sls/instance/mysls/docroot
= /opt/sensage/etc/sls/instance/mysls/dsroot
= /opt/sensage/etc/sls/instance/mysls/ldap.conf
= /opt/sensage/etc/sls/instance/mysls/md5
= /opt/sensage/etc/sls/instance/mysls/md5.errors
= /opt/sensage/etc/sls/instance/mysls/sensage_sls_mysls
= /opt/sensage/etc/sls/instance/mysls/shared_secret.asc

====[Summary]====
Identical on all hosts:
= /opt/sensage/etc/sls/instance/mysls/PERMISSIONS
= /opt/sensage/etc/sls/instance/mysls/add.xml
= /opt/sensage/etc/sls/instance/mysls/athttpd.conf
= /opt/sensage/etc/sls/instance/mysls/cluster.xml
= /opt/sensage/etc/sls/instance/mysls/docroot
= /opt/sensage/etc/sls/instance/mysls/dsroot
= /opt/sensage/etc/sls/instance/mysls/ldap.conf
= /opt/sensage/etc/sls/instance/mysls/md5
= /opt/sensage/etc/sls/instance/mysls/md5.errors
= /opt/sensage/etc/sls/instance/mysls/sensage_sls_mysls
= /opt/sensage/etc/sls/instance/mysls/shared_secret.asc

None of the specified files are different.

None of the specified files exist only on localhost.

Exist only on remote hosts:
```

```
> /opt/sensage/etc/sls/instance/mysls/omnisight.ldif dev-02.sensage.com

[root@dev-01 latest]#
```

Running a Command on Each Host in the Cluster (clssh)

This section describes the `clssh` utility, which runs the specified command in parallel on a specified set of hosts.

This section contains the following topics:

SYNOPSIS

clssh [*general_options*] [*specific_options*] <*ssh_command*> [*ssh_options*]

ARGUMENTS

<ssh_command>

You must specify a `ssh` command to run.

OPTIONS

`[general_options]`

See General Syntax for Utility Tools on page 48.

`[specific_options]`

--hosts

Executes command on hosts specified in the list; defaults to localhost.

Important: If you use curly braces {} to factor out common prefixes (for example, `host{0-9}`) and you run `clssh` on `csh` or `tcsh`, enclose the braces in quotation marks (for example, "`host{0- 9}`")

--sls-instance

Executes command on every host on which the specified instance is running; value can also be set through the SLS_INSTANCE environment variable. Chapter 2: Managing SenSage AP 54 Administration Guide

--[no]cwd

Uses current working directory on remote hosts; if --nocwd is specified, uses the user's home directory; defaults to --cwd.

--in-dir ([current directory])

Uses specified directory on remote hosts; if is specified, uses the user's home directory; defaults to the user's current directory.

--user

Uses the specified userid for remote execution.

--root

Uses root for remote execution.

--make-root-trust

Generate SSH key pairs for root trust and distribute them. For more information, see **Installing and Configuring SenSage AP** in the *Installation, Configuration, and Upgrade Guide*.

EXAMPLE

```
% bin/clssh ls /home/lms/cg_2002
Welcome to clssh version #8245.
Initializing (ssh) cim@host01.myco.com

cim@host01.myco.com's password:
Initializing (ssh) cim@host01.myco.com
* host01:-Done- | host08:-Done-
All done.

====[host01.myco.com]====
bin
conf
contrib
docroot
dsroot
example_logs
lib
misc

====[host08.myco.com]====
bin
conf
contrib
docroot
dsroot
example_logs
lib
misc
```

RETURNS

The standard output and standard error from executing the command on each host, organized by host.

Copying Files and Directories to Each Host (clsync)

This section describes the clsync utility, which synchronizes local files and directories with files and directories on specified remote hosts.

- For files, ensures that the file is replicated in the same location on each host.
- For directories, this utility recursively synchronizes the contents of the directory. This utility uses the `rsync` program.

This section contains the following topics:

SYNOPSIS

`clsync [general_options] [specific_options] <source_item> [<source_item> […]]`

ARGUMENTS

<source_item>

You must specify at least one source item. Additional source items are optional. Separate multiple items with spaces. You can specify files, directories, or combinations of both, as the following three examples show:

```
clsync file1 tmp/file2.txt file3 clsync dir1 dir2
clsync file1 tmp/file2.txt dir1
```

If you specify a path without a leading slash (/), the utility resolves the path from the current directory. The utility synchronizes the local files and directories with those in the same current directory on the other hosts in the cluster. If the corresponding files are in a different location on the other hosts, use the --target-dir option to specify the alternate, remote location.

OPTIONS

`[general_options]`

See General Syntax for Utility Tools on page 48.

`[specific_options]`

--hosts

Executes command on hosts specified in the list; defaults to localhost

--sls-instance

Executes command on every host on which the specified instance is running; value can also be set through the `SLS_INSTANCE` environment variable.

--target-dir

The target directory on remote hosts where the files and directories to be synchronized are located. Use this option if the location of files to synchronize are in different locations on the remote hosts than on the local host. If the remote locations on remote hosts differ by host, use the `<host>:<path>` syntax for specifying hosts with the `--hosts` option described above.

--exclude

When it examines directories for files to synchronize, the clsync utility ignores files and directories with base names that match the specified pattern. Separate multiple patterns with

spaces or use multiple `--exclude` options. Enclose the patterns within single quotes, for example:

```
clsync --exclude='**/dsroot/**' /opt/sensage/etc
```

--include

Use the --include option to selectively override excluded items. When it ignores files and directories because of the `--exclude` option, the `clsync` utility does not ignore files and directories with base names that match the pattern specified for `--include`. Separate multiple patterns with spaces or use multiple `--include` options. Enclose the patterns within single quotes, for example: clsync --exclude='**/dsroot/**' --include='*.xml' /opt/sensage/etc

RETURNS

A report of what `clsync` did, organized by host.

EXAMPLE

Following is an example of the `clsync` command executed on a two-node cluster: dev-01,dev-02 in this example, a file is missing on dev-02. The `clsync` command identifies the missing file and copies it from dev-01 to dev-02, bringing the two nodes back in sync.

```
[root@dev-01 latest]# clsync --hosts=dev-01,dev-02 /opt/sensage/sql/
clsync - clsetup-4.7.0, change #73647;

Initializing (ssh) root@dev-01.sensage.com
Initializing (ssh) root@dev-02.sensage.com
Pending: 0 Lagging: - last file copied: .../oae/sls_name.sql
Done

====[dev-01.sensage.com]====
/opt/sensage/sql/ -> <same place>: no files copied.
No files copied.

====[dev-02.sensage.com]====
/opt/sensage/sql/oae/sls_name.sql -> <same place>
Total number of files copied: 1

[root@dev-01 latest]#
```

Merging the Results from Top on Each Host in the Cluster (cltop)

This section describes the `cltop` utility, which creates a merged view of the operating system status across an entire cluster of computers other words, it is a cluster-wide version of the `top` program.

This section contains the following topics:

SYNOPSIS

cltop [general_options] <instance_name>

ARGUMENTS

<instance_name>

The instance reference is a required argument that provides the name of the EDW (SLS) instance. You can specify the instance reference in any of the following formats:

`<host>:<port>`

For example: `host1.myco.com:8072`

`<host>:<instance_name>`

For example: `host1.myco.com:my_instance`

`<instance_name>`

For example: `my_instance`

Note: Use the `<instance_name>` argument only for instances that reside on the same host from which you run the command. If you specify only one instance reference, the command includes all hosts that are part of that instance.

Note:

You can use `<host>:<port>` and `<host>:<instance_name>` for instances that reside on the local host or remote hosts.

As a convenience, you can omit this argument if you set the environment variable SLS_INSTANCE. For example: `SLS_INSTANCE=localhost:8072`.

OPTIONS

[general_options]

See General Syntax for Utility Tools on page 48.

EXAMPLE

```
% bin/cltop

Welcome to cltop version #8245.

Initializing (ssh) cimarron@p02.hq.myco.com

cimarron@p02.hq.myco.com's password:
Initializing (ssh) cimarron@p08.hq.myco.com
* p02.hq:-Done- | p08.hq:-Done-
All done.
NODE NODE ..........CPU usage......... ............... RAM usage ............
NUM ROLES idle busy (user+system) total free used shared cache
02 DMNQ 0.0% | 0.6% => 0.5+0.1 1,507MB 4MB 1,502MB 0MB 0MB
08 DMNQ 3.0% | 3.5% => 0.5+3.0 1,004MB 21MB 983MB 0MB 0MB

NODE PROCESS VMEM RAM SHMEM PROC CPU INSTALL
```

```
NUM ID (MB) (MB) (MB) STAT used TIME PORT START NAME ACTION OTHER
08 13277 10.2 10.0 1.8 S N 9.5% 0:29 10 Dec31 10 10 /usr/local/bin/perl
02 17851 2.5 0.3 0.3 S 0.0% 0:00 2002 2:14 cg_2002 init
02 17854 2.1 0.2 0.2 S 0.0% 0:00 2002 2:14 cg_2002 init
08 9522 5.6 5.5 4.8 S 0.0% 0:00 2002 2:14 cg_2002 init
08 31134 5.6 5.6 4.9 S 0.0% 0:00 2005 3:11 cg_2005 init
08 9519 5.0 5.0 4.1 S 0.0% 0:00 2002 2:14 cg_2002 init

KEY:
- PROC STAT:
D uninterruptible sleep (usually IO)
R runnable (on run queue)
S sleeping
T traced or stopped
Z a defunct ('zombie') process
W has no resident pages
< high-priority process
N low-priority task
L has pages locked into memory
```

RETURNS

Prints out of memory usage, CPU usage and processes.

Listing hosts in the cluster (clhosts)

This section describes the `clhosts` utility, which lists the hosts in the cluster in a form suitable for use with other programs.

This section contains the following topics:

SYNOPSIS

`clhosts [general_options] [specific_options] <instance_name>`

ARGUMENTS

<instance_name>

The instance reference is a required argument that provides the name of the EDW (SLS)

instance. You can specify the instance reference in any of the following formats:

<host>:*<port>*

For example: `host1.myco.com:8072`

<host>:<instance_name>

For example: `host1.myco.com:my_instance`

<instance_name>

For example: `my_instance`

Note: Use the <instance_name> argument only for instances that reside on the same host from which you run the command. If you specify only one instance reference, the command includes all hosts that are part of that instance.

You can use <host>:<port> and <host>:<instance_name> for instances that reside on the local host or remote hosts.

As a convenience, you can omit this argument if you set the environment variable SLS_INSTANCE. For example: SLS_INSTANCE=localhost:8072.

OPTIONS

[*general_options*]

See General Syntax for Utility Tools on page 48.

[*specific_options*]

--cluster-xml

Displays cluster.xml file

--show-datadir

Outputs <host>:<datadir> for all hosts.

--show-port

Outputs <host>:<port> form of the instance reference.

Note: When you execute clhosts --show-port <host>:<port>, the results display the port of the local machine on which the command executes. You can use this information to convert <name> or <host>:<name> to <host>:<port>, which is the only format that the data-store utilities accept. For more information, see Sensage Loading, Querying, and Managing the EDW on page 61.

Example

```
% clhosts doctests
Welcome to clhosts version #8245.

lms02.myco.com
lms08.myco.com
```

3

Sensage Loading, Querying, and Managing the EDW

This chapter contains the following sections:

Loading Data into the EDW

This chapter contains the following sections:

Synopsis

```
atload [<options>] <cluster_list> <table_name> <ptl_file>
      { [<log_file> [<log_file> [...]] | - }
```

Description

The `atload` utility loads source log events from the specified log file(s) into the specified table on an EDW instance. If the specified table does not exist, it creates the table. The `<ptl_file>` argument specifies a file that describes how the log events are transformed and loaded into an EDW instance.

The `atload` utility supports loading multiple log files with one invocation of the command. If you specify the names of multiple log files on the command line, they are loaded in the order specified. When specifying multiple log files, separate each name with a space addition, you can specify archive files that contain multiple log files.

International Support for Loading Data

The character encoding of <*ptl_file*> must be UTF-8. The character encoding of <*log_file*> can be ISO 8859-1 or one of the character encoding schemes you specify with the `--data-char-encoding` option. If not specified, the character encoding of log files processed by the atload command is assumed to be ISO 8859-1.

To determine the character encodings available in your environment, run the following command:

```
iconv -l
```

A list of supported character encodings displays.

For more information on character encoding, see International Support in the EDW on page 137.

Arguments

Argument	Description
<cluster_list>	`<cluster_list>` represents a comma-separated list of <host>:<port> pairs. The list must be enclosed in quotation marks ("). Specifying a list rather than a single host:port pair allows the utility to run even if the first of the specified EDW nodes is down. Each host:port pair identifies the EDW instance; for example, sls01:8072. The command tries each host:port pair in order (left to right) until it finds one that allows a TCP (Transmission Control Protocol) connection. Failure to establish a TCP connection indicates that the EDW host is down. If the connection succeeds but the command fails, the command does not attempt to establish additional connections. Example: `atload --user=administrator --password=changeme "sls01:8072,sls02:8072,sls03:8072" mytable src.ptl src`
<table_name>	Destination table name (created if it doesn't exist).
<ptl_file>	The file that describes how to parse the log data.
<log_file>	One or more data files to load into the table. Use "-" to read log data from standard input.

Table names can be 1 to 255 characters long and can consist of letters, numbers and underscores (_).

Options

--archive-add-filename

Passes in the name of each file in an archive. When you load multiple log files from archive files, such as a .zip, .tar or .tar.gz files, this option prefixes the archive file name with the names of the interior files as each is processed. This option affects the $_log_filename automatic expression macro that can be used in PTL files to load the source log-file name as a column value in each loaded row.

For more information on $_log_filename, see Automatic Expression Macros on page 71 and Loading Information about the Log File on page 72.

Note: The `--archive-add-filename` option is relevant only when loading batched data, not streamed data.

--archive-tempdir=<directory_name>

Specifies the temporary directory to use for loading operations that involve archive files. Use this option to control where temporary files are stored when loading archive files, such as .zip, .tar, or .tar.gz.

The default is the current working directory.

--batchload

Loads all files in a single operation rather than individually. Batch loads can improve performance, but the rows loaded from all files have the same value in the _uploadid column.

--clocksync

Corrects for client-side clock drift.

Important: Only use this option if the client running atload is also the client on which the log file being loaded was generated. This option attempts to compensate for clock mismatches between the client and the EDW server when determining the timestamp for rows loaded into the table.

--compression={none|low|high}

Specifies the level of compression for storing the new data.

High compression decreases the disk space consumed by the data store of the EDW instance, but it increases the load time due extra processing overhead. The supported values are none, low, and high. The default is high.

--data-char-encoding=<encoding>

Specifies the character encoding of source log files processed by the atload command. Supported values are:

- UTF-8
- UTF-16
- EUC-JP
- Shift-JIS
- UJIS

The character encoding of the log files processed during a single invocation of `atload` must all be the same. For example, assume the following command-line invocation:

```
$> atload --data-char-encoding='UTF-8' host5:8072 mytable \
           MyPtl.ptl log1 log2 log3
```

The log files log1, log2, and log3 must all be written in UTF-8. Regardless of the character encoding you specify for the log files, the character encoding of the PTL file must be UTF-8. The `atload` command converts source log events to UTF-8 before they are evaluated by the regular expression in the PTL file and stored as rows in the EDW.

The default encoding for log files is ISO 8859-1.

--dbgprint-request

Dumps the client-server request, then exits. This option forces atload to print the requests that it would have sent to the EDW and stop without actually loading the data. Use this option for debugging purposes.

--filetype=<type>

Specifies the data type of the files being loaded. Sometimes the files being loaded do not end with the proper extension, and the EDW cannot determine if decompression is required. Allowed values for <type> are `ascii`, `gzip`, `bzip2`, `zip`, `tar`, `targz`, and `tarbz`.

Note: The file you are loading must be of the type you specify.

--finalsummary

Prints a final summary table after completing its operation. Each line in the table reports information about one file-loading operation, including the number or log events loaded, performance statistics, number of retries, and success or failure.

--help, --longhelp

Displays online help for command in short form (`--help`), or with additional detail (`--longhelp`). You can specify `--longhelp` together with the `--[no]diskusage` flag to specify the computation of the disk used by the load (which can be slow for large installations). When no is specified with the diskusage flag,it means do not perform computation of the disk used by the load.

--matchfailures={<filename>|-}

Displays regular-expression match failures in the terminal window, or writes them to a file. The PTL file contains a regular expression that parses source log events from the log file. For debugging purposes it is useful to see the source log lines that the regular expression fails to match. The supported values are:

- `<filename>`—write match failures to the specified file. The file is created if it does not exist; the file is appended to if it does exist.

  Note: Although you can specify the `--matchfailures` option when `atload` runs on a schedule managed by the Collector, the preferred method is to use the <ParseFailures> element in loader definitions and include a <LogFile> element.

- `-`(a dash)—displays match failures in the terminal where `atload` is running. Only specify this option when you run `atload` from the command line.

  Note: Specifying the `--matchfailures` option slows the operation of atload when there are many match failures. This can occur if the incorrect PTL file is specified, because the majority of source log lines will fail to match the regular expression in the PTL file.

This option is overridden by the `--raw` option.

--namespace=<namespace>

The namespace in which to manage items. The default namespace is default. To specify the top level, use "" (empty quotation marks).

Note: When you run `atload`, you can specify a table name explicitly (for example, `ns1.ns2.ns3.mytable`) or use the `--namespace=`*`<namespace>`* option. When you use the namespace option, it specifies the namespace once for the entire command; the utility prefixes the namespace to each table name used in the rest of the request.

--on-error={continue|stop}

Specifies how to process remaining log files after one of them fails to load. This option applies only when you load multiple files with one invocation of `atload`. The supported values are:

- `continue`—proceed to the next log file and try to load it.
- `stop`—do not load any remaining log files.

The default is continue.

--override=<name>:<value>

Specifies values that override the default values declared for expression macros in the SQL section of the PTL file. Repeat this option for each expression macro that you want to override.

For more information, see the description for **Expression Macros**, in the Chapter *SenSage AP SQL*, in the *Event Data Warehouse Guide*.

--preprocregex=<regex>

--preprocschema=<name>:<value>[,<name>:<type>[...]]

Applies a regular expression to each source log line being loaded before it is processed by the indicated PTL file. These options let you strip out information that was prefixed to the log lines during preprocessing and store it in extra columns in the table.

For example, you can prefix each line of log data with the name of the source host that generated the data. The assumption is that a PTL file for processing the original log data already exists and that you want to re-use that PTL file for loading the annotated log data.

The regular expression indicates how to separate the annotations prefixed to log lines from the original log data. One of the matches from the regular expression must be the original line of log data. You can construct the regular expression to also return matches for the various annotations, which allows you to load those into the table along with the standard columns for that data source.

The schema indicates how to name the matches that are derived from the regular expression. One of the matches must be named '_REST_OF_LINE_' and must have a type of varchar. This is the match that contains the original log line of data and is fed to the PTL file. Other matches from the regular expression, if there are any, become additional columns in the target table.

Example

```
atload --preprocregex='([^,]*),(.*)' \
       --preprocschema='UID:VARCHAR,_REST_OF_LINE_:VARCHAR' \
       <host>:<port> <table_name> <filename>.ptl <log_file>
```

--raw

Prints the raw XML load results that the EDW returns, instead of printing formatted results. Use this option for debugging purposes.

This option overrides the `--matchfailures` option.

--retries=<n>

Specifies the number of times to retry loading an aborted load operation. Sometimes errors that cause load operations to stall and eventually abort correct themselves. Aborting stalled

load operations without retrying a few times results in unnecessary and time-consuming recovery procedures.

The default is 3 retries.

--retry-sleep=<n>

Specifies the number of seconds to wait before retrying an aborted load operation. Retrying a load operation immediately after aborting does not give sufficient time for transient errors to correct themselves.

The default is 45 seconds.

--segmentsize=<n>

Specifies the number of rows stored in a new EDW leaf node. (The EDW stores data in a tree structure with leaf nodes.) The default value is 250,000 rows. This value is correct for most applications and should not be changed. It is recommended that you Consult IgniteTech Support before changing this parameter.

--timeout=<n>

Specifies the number of seconds to wait before a stalled load operation aborts. Sometimes transient errors, such as network latency, cause loads to stall. Transient errors usually resolve themselves, and the load operation resumes at a regular pace. Other errors, such as hardware failures or locking contention, can persist indefinitely. Setting a time-out value allows a permanently stalled load to be aborted so that other load operations can occur. Set the value to 0 to wait indefinitely without timing out.

The default value is 0.

--trickle

Specifies that the EDW instance use the trickle feed (or trickle load) option for that LOAD, which means that the log data is loaded into an intermediate storage layout in the EDW. Trickle feed can also be set with the following method in Java EDW API: `LoadRequest::setTrickleFlag`.

To decide whether to use the trickle feed option, weigh user requirements against pros and cons and determine the best trade-off. As log files grow larger (or there is less overlap between them) a tipping point is reached in which performing normal, non-trickle loads makes most sense. On the other hand, when files are larger and there are longer time limits between the time-of-collection and the time-of-availability, the use of the trickle feed option (merge-load) starts to make more sense. Always consider the customer requirements against ideal scenarios to determine whether trickle feed is the appropriate loading solution.

Do use the trickle feed option in these cases:

- You have small files, collected at erratic intervals, no timestamp relationship between them, and short time limits between the time-of-collection and the time-of-availability this case, use the `--trickle` flag to load each file as it is collected. Optionally, you can add the`--trickle` flag to the Collector's `config.xml` file as an ATLOAD option in the appropriate cLoader definition.

  Note: Although there are no hard and fast rules for using the trickle feed option, optimum loading conditions (for example, when there are timestamp relationships between files) make it less likely that using the trickle load option will even be helpful.

- You have a data set being loaded has a small number of rows (less than the run length of a leaf times the number of nodes in the EDW cluster). This type of loading ensures that the collected log data is available within a short period time in the EDW such a case the loader is typically configured to run every few minutes to push whatever data has been collected into the EDW.
- You have data being loaded that is expected to overlap (in terms of the time range of the log data) with other data sets that have been or soon will be loaded. This type of loading is for periodic data collection from a large number of sources in parallel; for example, pulling the last day's log data from 10,000 servers once a day, which results in as many log files that individually cover the same period of time this case, the result is 10,000 log files each of which covered the same 24-hour period.

The latter two use cases cause excessive fragmentation in the EDW datastore, resulting in significant performance issues when the loads are placed directly into the EDW long term storage layout. The trickle load operation directs the data set into an intermediate storage layout which supports efficient appending of out-of-order data. This avoids the load-time performance problems associated with the fragmentation of data in the long-term storage layout.

Both of these use cases cause excessive fragmentation in the EDW datastore, resulting in significant performance issues when the loads are placed directly into the EDW long term storage layout.The trickle load operation directs the data set into an intermediate storage layout which supports efficient appending of out-of-order data. This prevents load-time performance problems associated with the fragmentation of data in the long-term storage layout.

Queries against tables that have trickle-loaded data can show some performance degradation in some cases. Most notably bloom filters do not accelerate access to data in the intermediate storage layout. If you are using trickle loads, provide a schedule for periodically running a compact operation against the target table. Schedule the compact operation to occur once a day during a period of time where there is typically little other activity. The compact operation will retrieve whatever log data has been accumulated in the intermediate storage layout and merge.

When the Collector invokes the EDW COMPACT operation, this allows the EDW data structures to stay small and prevents unnecessary delays when a manual compact operation is invoked after large amount of data is accumulated in TFL data structures.

When performing an in-line COMPACT during a load, you can set the compactquota configuration parameter in the EDW Compact Quota field of the Advanced EDW section of the SenSage AP service Configs tab of the Deployment Manager.

The parameter value is an integer indicating the percentage of RAM that should serve as the load threshold when performing the in-line COMPACT. The threshold wants to guestimate that running N parallel COMPACT operations (where 'N' is the run-queue size) will not use more than the indicated amount of RAM in the system.

Compaction parameters are also defined within a <Loader> section that includes:

- Compaction schedule and /or
- A threshold determined by rows and/or bytes and/or load counts
- Loaders used against a given table. (more than one may be used)
- Compaction threshold statistics stored globally (within a given Collector) on a table-by-table basis.
- Compaction to lock out loads to a table globally (within given Collector).

The COMPACT parameters, when used, have the following attributes that result in a compact operation when the value for the given parameter is reached.

- Compaction schedule and /or
- A threshold determined by rows and/or bytes and/or load counts
- Loaders used against a given table. (more than one may be used)
- Compaction threshold statistics stored globally (within a given Collector) on a table-by-table basis.
- Compaction to lock out loads to a table globally (within given Collector).

If none of the threshold options are chosen, the Collector performs compaction according to the Schedule option defined in the parameter.

Examples:

```
<!-- simple parameterized compact ->
<Compact loadsThreshold="100" rowsThreshold="100000000"
megsThreshold="1000000000"/>

<!-- one parameter only, compact only at size threshold ->
<Compact megsThreshold="1000"/>

<!-- compact only at one minute after midnight - Schedule>
<Compact> <Schedule>1 0 * * *</Schedule> </Compact>

<!-- always compact at one minute after midnight, or after 10 gigs
is loaded since last compact -->
<Compact megsThreshold="10000">
<Schedule>1 0 * * *</Schedule>
</Compact>
```

--verbose=<n>

Where <n> is one of the following verbosity levels.

Verbosity Level	Meaning	Description
1	error	Displays errors
2 (default)	warning	Displays errors and warnings

Verbosity Level	Meaning	Description
3	progress	Displays errors, warnings, progress indication with ellipses (...), and parsing or row-loading errors with exclamation marks (!)
4	everything	Displays, errors, warnings, progress indication, parsing errors, and informational messages.

--version

Displays version of `atload`.

Parameters

Authentication Options

--user=<user_name>

--user=default

Runs the command under the authorization of <*user_name*>, who must be a user defined in the EDW instance. If you specify default, the atload command runs under the authorization of the user specified by the ADDAMARK_USER environment variable.

Note: To log in as the **guest** user, do not specify the `--pass` or `--shared-secret` flags. The default is to run the command under the authorization of the currently logged-in user.

--pass=<password>

--pass=file:<path_and_filename>

Specifies the password that authenticates the user that you specified with the `--user` option. You can specify the password in clear text on the command line. You can place the password in a file and specify its location and file name on the command line. If you omit the `--pass` option, the atload command uses the password specified by the ADDAMARK_PASSWORD environment variable.

For more information, see Setting the ADDAMARK_PASSWORD Environment Variable on page 71.

--shared-secret=default

Specifies the shared secret for the hosts in the EDW instance. You can specify the shared secret in clear text on the command line. You can place the shared secret in a file and specify its location and file name on the command line. If you specify default, the atload command uses the shared secret specified by the ADDAMARK_SHARED_SECRET environment variable.

The value for `--pass` takes precedence over the value for `--shared-secret` if both are present.

--skip-queues

Bypasses normal EDW task queueing.

You must specify the `--shared-secret` option when you use the `--skip-queues` option.

Setting the ADDAMARK_PASSWORD Environment Variable

To set the ADDAMARK_PASSWORD environment variable (for example, in the Bourne shell), submit one of the following commands.

- To set a clear-text password in the environment variable:

  Syntax

  ```
  bash$ export ADDAMARK_PASSWORD=<password>
  ```

  Example

  ```
  bash$ export ADDAMARK_PASSWORD=s0mep@ss
  ```

- To store the location and name of the file that contains a list of valid passwords:

  Syntax

  ```
  bash$ export ADDAMARK_PASSWORD=file:<path_and_filename>
  ```

  Example

  ```
  bash$ export ADDAMARK_PASSWORD=file:/local/home/sls/passwords
  ```

IgniteTech recommends that you use SSH or a similar protocol over a secure session to submit all authentication commands.

Loading Archive Files

When `atload` processes archive files, it uses the appropriate module to read the archive contents and extracts internal files into an individual `.gzip` files. It then uses the regular atload mechanism to load the `.gzip` files.

Large archive files can cause atload to run out of disk space on the client machine. To avoid running out of disk space, use the `--archive-tempdir` option to control where temporary files are stored.

Because the name of each internal file in an archive is lost after the data is combined, use the

`--archive-add-filename` option to prefix every newline-separated record with the archive internal file name. If you use `--archive-add-filename`, you need to change your PTL file to parse this new field.

For example, add ... to the regular expression, and add filename:VARCHAR to the list of parse fields.

Automatic Expression Macros

The atload utility provides these expression macros automatically for your use in the PTL SELECT statements in PTL files:

- $_log_filename on page 72
- $_log_srchost on page 72

For an example of their use, see Loading Information about the Log File on page 72.

$_log_filename

The automatic expression macro $_log_filename carries the filename of the log file currently being loaded. Files in Collector log queues often carry useful information in their filenames. You should parse and store this information as column values in the row being loaded. For example, you should store the entire filename and parse other useful information from it, such as the time the log file was created.

$_log_srchost

The automatic expression macro $_log_srchost carries the name of the machine from which the log file is being loaded. You should capture this information and store it as a column value in the row being loaded. This helps maintain the chain of custody for the log entry.

Loading Information about the Log File

The atload utility allows you to store information about the log files from which rows were loaded in a column of the data. Use the $_log_filename and $log_srchost expression macros in your PTL SELECT statements. The atloadcommand declares the expression macros automatically.

For example, you have log entries from multiple log sources in a large file named:

```
snmp-2002-01-09_04:00:00.gz
```

The filename carries the timestamp when the log file was created—its so-called roll time. The log file is being loaded from a host named h03. The following PTL SELECT statement uses $_log_filename and $log_srchost to store the roll time and host name as column values in the row being loaded.

```
SELECT _strptime(_strmatch($_log_filename, "(\d+-\d+-\d+_\d+:\d+:\d+)\.gz$"),
"%Y-%m-%d_%H:%M:%S") AS _rolltime, $_log_srchost AS _srchost FROM stdin;
```

Note: Follow the convention of using an underscore (_) to begin the names of columns with metadata about the log entry to distinguish them from columns with data parsed from the log entry itself.

Reading Log Data from stdin

The `atload` utility reads the log data from standard input when you specify '–' as the log filename.

When you specify the log data in this manner, the data does not have a file name with an extension that identifies its file type. For example, when you load data as `myload.tgz`, the EDW recognizes the `.tgz` suffix as the tgz file type. IgniteTech recommends that you include the `--filetype` option when you specify '–' as a log file name. If you do not specify a file type, `atload` attempts to guess the type from the stdin input stream by examining the first few bytes.

Tracking Uploads in the EDW

In addition to loading the log data into the specified table(s), atload saves information in the EDW about the load. You can view this information by querying against two system tables:

- `system.upload_info`

This table enables administrators to check on upload status in the system. When the system is healthy, this table returns one row for each upload. When the upload data is not consistent across all the hosts in the instance, this table returns a value of `false` in the CONSISTENT column for this case, the table may return multiple rows with the same value in the `_uploadid` column; all of these rows are marked as inconsistent.

- `system.raw_upload_info`

  This table enables a support person to troubleshoot an inconsistency problem by seeing the raw data distributed across the system

Both of these tables contain a unique identifier for the upload: the value of the _uploadid column. For more information about the _uploadid column, see Using the UPLOADS Command on page 117 and Table Management on page 103.

These tables store additional information that includes: the minimum and maximum timestamps, the number of lines in the source log data, the number of lines successfully parsed by the PTL, the number of rows loaded into the specified EDW table(s), the PTL and client signatures, and whether the load was successful.

Additionally, the `system.upload_info` table returns the consistency data and the

`system.raw_upload_info` returns the logical name of the data's source host.

As it begins loading data into the log tables, the EDW inserts a new row in these system tables to track the uploaditially, the value of the SUCCESSFUL column defaults to false. When the load completes successfully, this column's value changes to true. By default, when you query the data in the log table(s), the EDW returns data only for loads that completed successfully. If you query a log table while it is still receiving data, none of the new data is returned.

Note: To view data from a failed load or from an in-progress load, specify the `{INCLUDE_BAD_UPLOADS}` modifier (enclosed within curly braces) in the `FROM` clause of your query. If a query and a load occur simultaneously and the query continues after the load completes, you may see all, none, or part of the data from the new load.

For more information, see:

- "system.upload_info" and “system.raw_upload_info” in the *Event Data Warehouse Guide*.
- "Including Rows from Bad Loads" in “SenSage AP SQL” of the *Event Data Warehouse Guide*.
- Checking Locking Status and IPC State Information on page 194

Errors and Return Values

The `atload` utility returns an exit code of 0 on success, or non-zero exit code and an error condition message upon error. For a full list, see Errors and Return Values for Data-Store Utilities on page 118.

Parsing errors may occur if records fail to match the regular expression in the PTL file or if a given field cannot be parsed as the requested data type. You can return these match failures to the client by running atload with the `--matchfailures` option.

Querying Data

This section describes the `atquery` utility. It contains the following topics:

Synopsis

```
atquery [options] <cluster_list> <file_name>.sql [<file_name>.sql […]]
atquery [options] <cluster_list> -
```

Note: When you use the second option, in which you enter the query directly from the command line, you must enter CTRL-D on a separate line after the SQL statement to actually run the query.

Description

The `atquery` utility processes:

- Data Manipulation Language (DML) statements, which query (SELECT) data in a table and retire (RETIRE) data from a table in an EDW instance.

  For more information, see Overview of AP SQL SELECT Statements in “SenSage AP SQL” of the *Collector Guide* and Retiring Data on page 115.

- Data Definition Language (DDL) statements, which create tables, column filters, and views within an EDW instance.

  For more information on the DDL statements that the atquery command can process, see Defining Data Objects on page 128.

The `atquery` utility supports processing multiple SQL files at once. If you specify the names of SQL files on the command line, they are processed in the order specified. When specifying multiple files, separate each name with a space.

After processing its arguments and checking the SQL files for obvious errors, the querying process proceeds as follows. For each SQL file, `atquery`:

1. Connects to the specified host and port.
2. Sends the SQL file, wrapped as an XML-RPC "message" (also known as XML data).
3. Listens for errors, progress messages, and result records.
4. On the server, parses the query, sends new queries to each of the hosts in the instance, and combines the results to form the final result.

Note: If the `Primary` directory (which contains the primary copy of log data) is unavailable, the EDW uses the `Secondary` directory (which contains the backup copy of log data). If both of these directories are unavailable, the query fails. If the query uses the Secondary directory, processing is slower than when it uses the `Primary`. For more information about these directories, see EDW Architecture on page 22.

INTERNATIONAL SUPPORT FOR QUERYING DATA

The character encoding of *`<file_name>.sql`* must be UTF-8. ASCII character encoding is a proper subset of UTF-8 character encoding.

For more information on character encoding, see International Support in the EDW on page 137.

Arguments

Argument	Description
<cluster_list>	*<cluster_list>* represents a comma- separated list of <host>:<port> pairs. The list must be enclosed in quotation marks ("). Specifying a list rather than a single host:port pair allows the utility to run even if the first of the specified EDW nodes is down. Each <host>:<port> pair identifies the EDW instance; for example, sls01:8072. The command tries each host:port pair in order (left to right) until it finds one that allows a TCP (Transmission Control Protocol) connection. Failure to establish a TCP connection indicates that the EDW host is down. If the connection succeeds but the command fails, the command does not attempt to establish additional connections. Example: `atquery --user=administrator --password=changeme "sls01:8072,sls02:8072,sls03:8072" myfile.sql`
{<file_name.sql> [<file_name.sql>[…]]|-}	Specify either one or more files that contain the SQL statements to run (one statement in each file) or specify a dash (-) to read from standard input. **Note:** When you use the second option, you are prompted to enter your SQL query. You must enter **CTRL-D** on a separate line after the SQL statement to actually run the query.

Options

--version

Displays version of atquery.

--help, --longhelp

Displays online help for command in short form (`--help`), or with additional detail (`--longhelp`).

--namespace=<namespace>

The namespace in which to find the specified table or view. The default namespace is default. To specify the top level, use "" (empty quotation marks).

When you run atquery, you can specify the namespace as a prefix to the table or view name (for example, ns1.ns2.ns3.mytable) or use the `--namespace=<namespace>` flag to specify the namespace.

Note: When you use the namespace flag, you can specify the namespace once for an entire command; SenSage AP prepends the specified namespace to each table or view name used in the rest of the request.

--verbose=<n>

Where <*n*> represents result set(0), errors(1), warnings(2), progress(3), or everything(4). Default is 2.

Verbose Level	Meaning	Description
	result set only	Displays only the result set of the query. Use the --format option to specify the text format of the result set. For more information, see Scripting Support And Exit Codes on page 82.
1	error	Displays the result set and only errors; ignores all non-errors
2	warning	Displays the result set and warnings and errors; uses dots (...) to indicate progress and exclamation points (!) to indicate parsing errors
3	progress	Shows progress in greater detail than level 2
4	everything	Displays all informational messages

--override=<name>:<value>

Overrides default values for expression macros defined in SQL statements. You can repeat this option to override default values for different expression macros.

For more information, see **Expression Macros** in Chapter 3, “SenSage AP SQL” in the *Event Data Warehouse Guide*.

--ansi

Optionally triggers "ANSI" mode for a given query when using Open Access Extension (OAE). The inet datatype family code has a value of "none" (which means that an empty string was converted to an inet address) or "invalid" (which means a non-empty string that did not contain a valid IPv4/IPv6 address was converted to an inet address) ANSI mode, at query-time when these two special inet values exist they are transformed into "0.0.0.0/0" so that the original inet values, "none" or "invalid" are never displayed.

Note: The actual data persisted to disk and into the Bloom Filters and Index files is not changed. This allows the EDW to behave as expected in "Native Mode".

--tableswap=<name>:<newname>

Substitutes table names in FROM clauses with another table name. You can repeat this option to substitute different tables in different FROM clauses, such as in sub-SELECTs and subqueries.

When building libraries of queries, it is often helpful to replace the tables in the FROM clause with different tables, chosen at the time the query is run. The `--tableswap` option enables this functionality, which operates in the same way as the processing directive WITH <*name*> AS TABLE <*newname*>.

For more information, see **Table-Name Substitutes** in Chapter 3, “SenSage AP SQL” in the Event Data Warehouse Guide.

-expression=<SQL_stmt>

-e=<SQL_stmt> (alias)

Executes the given SQL statement. Both of these options allow you to specify an inline SQL statement as a command-line option. You can specify multiple queries this way.

Tip: Alternatively, you can specify a single hyphen at the end of the line preceding the SQL statement, and on a new line, type the SELECT statement followed by EOF (^D). For example:

```
atquery --user=administrator --pass=pass:s0mep@ss \
localhost:8072 --namespace='system' -
SELECT * FROM users;
```

The statement above returns the same results as:

```
atquery --user=administrator --pass=pass:s0mep@ss localhost:8072 \
--namespace='system' -expression='SELECT * FROM users;'
```

--postproc={<filename>|<perl_code>}

Optionally post-processes the rows in the final result set of a query. Post-processing logic is written in Perl code, which you supply from a file or the command line. When you provide <*perl_code*>, start the value with an open curly brace ({).

Use postprocessing to change the rows in results sets as they come back from the EDW, before the client application receives them. A postprocessor might do the following:

- Recalculate values in particular column, perhaps through lookup
- Aggregate data returned from the EDW in some special way
- Change the set of columns contained in the result set.

For more information, see Query Postprocessor on page 83.

--format=<string>

To format the result rows, <*string*> must be one of the following:

- max_width — pipe-delimited that spaces the columns to the maximum-width value
- tsv — tab-separated values
- csv — comma-separated values
- psv — pipe-symbol-separated values
- dots — a max_width variant meant for two-column results

The default is `max_width`.

Note: Specify only tsv, csv, or psv if you specify --verbose=0.

--timeout=<seconds>

Specifies how long to let queries run before they time out. The default is 0, which indicates no timeout.

Automatically cancels a query that is taking too long. This is useful for scripting atquery when you are concerned that a query may "hang" for some reason.

--[no]raw

This option forces atquery to print the raw XML load results that the EDW returns, instead of the preformatted results that atquery usually shows. This is useful for debugging.

The default is `noraw`.

--[no]dbgprint-request

Dumps the client-server request, then exits. The default is `--nodbgprint-request`.

This option forces `atquery` to print the execution requests that it would have sent to the EDW, and then stops execution rather than proceeding with the query or queries. This is useful for debugging situations and for understanding the EDW API.

--progress-type=<type>

For verbose level 3 and above, this option controls what type of progress indicators to return; they include:

- `dot` — prints only periods, which indicate that the query is running, but does not provide any hints about when it will finish. This indicator is useful when you do not want to see the query details, but want to know that the query is working and has not hung.
- `all` — the default, this indicator provides instance-wide statistics, which roll up progress indicators sent by each host in the instance.
- `unfmt` — prints details in a more machine-readable format; for example, suitable for incorporating progress feedback into applications.
- `host` — prints very verbose per-host details.

The example below illustrates the format of the **all** indicator:

```
- 97.0s elapsed, on select 1/3, pass 2/3, during 1/2, about 32% done
 (some at
select 2/3)
- selecting 1970-01-01 00:00:00 GMT from L0307060456
- time range 1970-01-01 00:00:00 ------|------------------ 1970-01-01
 00:00:03
- scanned 17M records, 227K rps ( 180K rps average)
- aggregated 591K records, 7,306 rps ( 6,101 rps average) into 6,669
 groups
- RAM free 2,521MB of 4,747MB total (53%), min is perf02: 579MB (36%)
```

--detailedprogress

In addition, displays per-query or per-host progress.

--skip-queues

Ignores task priority queuing. For more information, see Task Priority and Queuing on page 123.

Parameters

Authentication Options

--user=<username>

This parameter authenticates the specified name as an EDW user. If no user is specified, atquery reads the environment variable ADDAMARK_USER. If it cannot find this environment variable, it uses the operating system to get the name of the user who is logged in.

Note: To log in as the guest user, do not specify the `--pass` or `--shared-secret` flags.

--pass=<pwmethod>

This parameter provides several methods to specify the password. You can specify the password in clear text, place the password in a file and specify the location of the file, or set it in the ADDAMARK_PASSWORD environment variable.

The syntax is:

`--pass=<password>`

`--pass=pass:<password>`

`--pass=file:<filename>`

`--pass=default`

When you specify default as the password, atload looks in the ADDAMARK_PASSWORD environment variable. For more information, see Setting the ADDAMARK_PASSWORD Environment Variable on page 71.

--shared-secret=<secret>

This parameter provides several methods to specify the shared secret. You can specify the secret in clear text, place the secret in a file and specify the location of the file, or set it in the `ADDAMARK_SHARED_SECRET` environment variable. This parameter provides two methods to specify the shared secret. You can either specify the secret in clear text or place the secret in a file and specify the location of the file.

The syntax is:

`--shared-secret=<secret>`

`--shared-secret=file:</full_path/filename>`

`--shared-secret=default`

When you specify `default` as the password, `atmanage` looks in the `ADDAMARK_SHARED_SECRET`

environment variable.

Note: The value for `--pass` takes precedence over the value for `--shared-secret`, if both are present.

--assume-role=role<role_to_be_assumed>

This parameter causes the operation to be run as if the user had assumed the specified EDW role. The assumed role must have **admin** privileges.

--skip-queues

This parameter bypasses normal EDW task queueing. Shared secret authentication is required when you use this parameter.

Return Codes And Error Messages

See Errors and Return Values for Data-Store Utilities on page 118.

Replay Support

When diagnosing performance issues, it is often helpful to save the output from the EDW particular, `atquery` supports a command (replay) that lets you simulate re-running the query that was previously run.

To save the exact bytes sent over the network from the EDW host, run atquery `--raw` and save the results to a file. To replay the results, run `atquery replay <filename>` . Specifying "-" (dash) instead of the filename causes the replay to get data from standard input.

The following example connects the live query to replay:

```
# runs myfile.sql and saves the raw XML messages to savefile.rwt
# and simultaneously visualizes them using "replay"
atquery host:port myfile.sql --raw | tee savefile.rwt | atquery replay -
# replay them (doesn't re-run the query)
atquery replay savefile.rwt
```

Note: You will need to `grep` out the "delay" lines to clean up the output.

Scripting Support And Exit Codes

The `atquery` utility returns an exit code of `0` on success, or a non-zero exit code upon error. Use these exit codes to control branching logic in scripts. For example, the following code example controls sending email under various conditions.

```
# these code snippets have been tested with bourne shell (sh), bash,
# tcsh and ksh. They assume that there's an SLS running on port
# 8072 on the local host, and that there's a table named "syslog"
# that has been loaded.
echo 'select count(*) from syslog during all' > myquery.sql
echo 'select * from nosuchtable during all' > badquery.sql

# send an email if the EDW is unavailable, e.g. not running
atquery nosuchlms:8072 myquery.sql || \
(mail -s "myquery.sql failed" root < myquery.sql)

# send an email if there's a syntax error
atquery localhost:8072 badquery.sql || \
(mail -s "myquery.sql failed" root < badquery.sql)

# send an email if/when the query succeeds
atquery localhost:8072 myquery.sql && \
(mail -s "myquery.sql succeeded" root < myquery.sql)

# send an email either way -- sh/bash version
if ( ./atquery3 localhost:8072 myquery.sql ); then
mail -s "sql succeeded" asah < /etc/hosts
else
mail -s "sql failed" asah < /etc/hosts
fi
```

For a full list of errors, see Errors and Return Values for Data-Store Utilities on page 118.

In addition to exit codes, the `atquery` utility prints the query result, progress information, and error messages to stdout. You control the verbosity of progress information and error messages with the `--verbose` option. The `atquery` utility always prints the result set to `stdout`, regardless of the `--verbose` option.

The results are set off from other text with the following separation lines:

```
==== BEGIN: Results for SQL file
<result_set>
==== END_DATA ====
```

If you specify `--verbose=0`, all messages and separation lines are suppressed, and only the result set is printed. Use the `--format` option to specify the text format of the result set. When `--verbose=0`, valid values for `--format` are `tsv` (tab-separated values), `csv` (comma-separated values), and `psv` (pipe-separated values).

Turning .sql Files into Unix Shell Scripts

For Unix users who would like their SQL files to be runnable, `atquery` knows to ignore the first line of SQL files that start with "`#!`". To support this capability, you must create a wrapper shell script like the following:

```
#!/bin/sh
# this will launch the real atquery program with the given .sql file
# note: please adjust INSTALLNAME, HOST and PORT to reflect your installation
/home/lms/INSTALLNAME/atquery HOST:PORT $*
```

Then, start runnable `.sql` files like the following:

```
#!/usr/local/bin/runatquery
select count(*) from mytable during all
```

Note: You must `chmod` both shell scripts, so they become executable; for example, `chmod 755 runatquery`. Also, many shells have a limit on the length of the `#!` line; often, it's limited to 32 characters, and fails silently.

Query Postprocessor

This section describes the client-side atquery postprocessor feature. It contains the following topics:

What is a Postprocessor?

A postprocessor:

- is a set of functions in Perl.
- accepts input row/column data returned from the EDW in response to a query.
- accepts input data that conforms to an "incoming schema" (that is, a set of column names and SQL data types).
- generates output row/column data, which is the data eventually displayed by the `atquery` program.
- generates that output data that conforms to an "outgoing schema," which may or may not be the same as the "incoming schema."

A postprocessor is typically used for the two following scenarios:

- To slightly modify one or more column values before return to the client application

  The postprocessor takes incoming rows, makes a slight change to each row, and passes on the values as output rows

- To do specialized aggregation or data formatting

  The postprocessor takes incoming rows and generates outgoing rows that do not directly correspond one-for-one to input rows.

Postprocessors do not chain or cascade. You cannot feed the output of one postprocessor as input to a second postprocessor.

Query Postprocessor API

Perl Functions that a Postprocessor Provides

The first aspect of the postprocessor API a developer must understand is the set of Perl functions the postprocessor must implement. Some of these functions are optional, as indicated below.

- `postprocInit` — postprocessor initialization [optional function]

  The `atquery` utility calls the `postprocInit` function before any input rows from the EDW results are submitted to the `postprocRow()` function. The `atquery` utility invokes `postprocInit` once before proceeding with results from each individual query, and that a single postprocessor (and its local variables) may be employed for multiple queries.

```
sub postprocInit {
 # The postprocInit function is passed:
 #
 # * a "response" object, representing the response from the SLS --
 # this "response" handles both the "incoming" and "outgoing" schema and
 # rows for query postprocessing
 #
 my ($response) = @_;
 # ... Postprocessor initialization code follows -- implementation-specific
 ...
 }
```

  A postprocessor typically uses the initialization function to set up the “outgoing” schema for the postprocessed query, to initialize Perl variables required to perform aggregations, and so on.

- `postprocRow` — a call invoked per row of input arriving from the EDW [required function]

  Do not assume that `postprocRow` is called at least once because, at times, the EDW may return no rows for a given query.

```
sub postprocRow {
 # The postprocRow function is passed:
 #
 # * a "response" object
 # * an "input row" object, which holds the column values for a row of data

 # incoming from the SLS server
 #
 my ($response, $inputRow) = @_;
 # ... Postprocessor per-row code follows -- implementation-specific ...
 }
```

  A postprocessor typically uses the per-row function to lookup and modify the value of one or more columns for each input row, or to do data aggregation, finding a particular input row's effects on the aggregated values.

- `postprocFinal` — postprocessor finalization [optional function]

  The `atquery` utility calls the postprocFinal function after all input rows from the EDW results are submitted to the postprocRow function.

```
sub postprocFinal {
 # The postprocFinal function is passed:
 #
 # * a "response" object
 #
 my ($response) = @_;
 # ... Postprocessor finalization code follows -- implementation-specific
...
 }
```

  A postprocessor typically uses the finalization function to put out aggregated values.

The `atquery` utility does not pass errors directly to any postprocessor function. If a fatal error occurs in transmitting the query to the EDW, while processing the query within the EDW, or while receiving results from the EDW, postprocessing stops and `atquery` handles the error as it would for any other query.

Perls Functions that a Postprocessor Calls

To compute results, a postprocessor can use local Perl variables, subroutines, or other features. At some point, however, the postprocessor must interact with the `$response` and `$inputRow` objects. Those interactions are described below.

- `$response->getIncomingSchema()`

  Returns an array containing the column names and SQL data types for the postprocessor's incoming schema. An example return value might be:

  ```
  [ "ts:timestamp", "v1:varchar", "v2:int32", "v3:int64", "v4:bool",
  "v5:timestamp" ]
  ```

  Note: The returned values can be either in upper or lower case.

- `$response->setMetadata(schema => @outputSchema)`

  Sets the schema for the output displayed by atquery.

  Passes in an array of the same form as that returned by the `$response->getIncomingSchema()` function.

- `$response->getSchema()`

  Returns an array containing the column names and SQL data types for the postprocessor's outgoing schema (same format as the `$response->getIncomingSchema()` function).

- `$response->addRowData(rowdata => { "col1" => "value1", "col2" => "value2" })`

  Adds a row of data to the response. The row conforms to the postprocessor's outgoing schema and any number of column names and values can be specified. Any unspecified columns take on the value "" (empty string), and any extra columns that don't conform to the output schema are ignored.

- `$inputRow->getColumnValue("col1")`

  Returns the value for a particular column of data within the input row.

- `my $hashRef = {}; $response->copyValuesForOutputSchema($inputRow, $hashRef)`

  Convenience function that copies column values from an input row of data to a Perl hash.

  Only those values from the input row whose column names match the outgoing schema are copied to the hash. The hash can subsequently be used (perhaps after further modification) to submit a row of data, with `$response->addRowData(rowdata => $hashRef)`.

  Note: You are not required to use the function `$response->setMetadata(schema =>@outputSchema)` if the outgoing schema matches the incoming schema. The `atquery` utility does this for you implicitly.

Query Postprocessor Examples

This topic provides examples of two postprocessor scripts. One example does a simple data lookup to replace values in one column of the incoming data. The other example aggregates and formats data (although the aggregation itself could be done on the EDW, the example uses it for illustrative purposes). The examples cover all postprocessor API functions documented above in Perls Functions that a Postprocessor Calls on page 86.

This topic provides a script that loads the required data and runs the postprocessors. The major steps are outlined below.

EXAMPLE DATA SET

Both postprocessor examples execute against the following data set, which is a simple weblog of presidents shopping for cars:

```
+---------------------------+-----------+--------------------------------------+
|            ts             | user_name |                  url                 |
|        (timestamp)        | (varchar) |                (varchar)             |
+---------------------------+-----------+--------------------------------------*
|2002-02-28T14:23:57.000000Z|gwashington|www.shop.com/home.htm
|2002-02-28T14:37:15.000000Z|gwashington|www.shop.com/catalog.htm
|2002-02-28T14:39:23.000000Z|gwashington|www.shop.com/prodinfo/infiniti.htm
|2002-02-28T14:49:27.000000Z|gwashington|www.shop.com/other_page.htm
|2002-02-28T14:29:30.000000Z|jadams     |www.shop.com/home.htm
|2002-02-28T14:41:51.000000Z|jadams     |www.shop.com/specials.htm
|2002-02-28T14:46:25.000000Z|jadams     |www.shop.com/specials/audi_europe.htm
|2002-02-28T14:46:29.000000Z|jadams     |www.shop.com/purchase.htm
|2002-02-28T14:47:35.000000Z|jadams     |www.shop.com/transaction_complete.htm
|2002-02-28T14:28:23.000000Z|ztaylor    |www.shop.com/home.htm
|2002-02-28T14:44:13.000000Z|ztaylor    |www.shop.com/catalog.htm
|2002-02-28T14:46:22.000000Z|ztaylor    |www.shop.com/prodinfo/cadillac.htm
|2002-02-28T14:48:31.000000Z|ztaylor    |www.shop.com/purchase.htm
|2002-02-28T14:50:01.000000Z|ztaylor    |www.shop.com/catalog.htm
---------------------------+-----------+--------------------------------------+
```

SIMPLE LOOKUP EXAMPLE

The URLs are not as descriptive as they could be, so for presentation purposes, the example substitutes the URL with a more descriptive string. The postprocessor script:

- contains only the postprocRow function, with no initialization or finalization function
- uses an outgoing schema that is identical to the incoming schema

The postprocessor script shown below uses an internal Perl hash lookup to map URL values to more user friendly page descriptions, and allows for pages not found in the hash.

```
# ---------- FILE: 01_lookup.pproc_in
my %easyUrlMap = (
  "home",                       "Home Page",
  "catalog",                    "Product Catalog",
  "prodinfo/infiniti",          "Car Info Page -- INFINITI",
  "prodinfo/cadillac",          "Car Info Page -- CADILLAC",
  "prodinfo/audi",              "Car Info Page -- AUDI",
  "prodinfo/saturn",            "Car Info Page -- SATURN",
  "specials",                   "Special Offers Central",
  "specials/audi_europe",       "Special Offer--Purchase AUDI in Europe",
  "purchase",                   "CASH REGISTER",
  "transaction_complete",       "-SALE-"
);
sub postprocRow
  my ($response, $inputRow) = @_;
  # Create a copy of the input row, for output
  my $outputRow = {};
  $response->copyValuesForOutputSchema($inputRow, $outputRow);
  # Get the URL, prepare default friendly URL (which isn't very friendly)
  my $rawUrl = $inputRow->getColumnValue("url");
  my $resultUrl = "... $rawUrl ...";
  # Try to find a more friendly URL
  my $url2;
  if (($url2) = $rawUrl =~ /^www.shop.com\/(.*)\.htm$/) {
     my $url3 = $easyUrlMap{$url2};
     if (defined($url3)) {
       $resultUrl = $url3;
     }
  }
  # Replace the URL in the output row
  $outputRow->{"url"} = $resultUrl;
  # Add the modified row to the response
  $response->addRowData(rowdata => $outputRow);
# ---------- END OF FILE: 01_lookup.pproc_in
```

The following command runs the example:

```
echo "SELECT ts, user_name, url FROM test ORDER BY 2, 1 DURING ALL;" | \
 atquery lmshost:8072 --namespace=myNamespace.pproc_example -
 --postproc=01_lookup.pproc_in
```

The query results are illustrated below:

```
+---------------------------+-----------+--------------------------------+
|     ts                    | user_name |             url                |
| (timestamp)               | (varchar) |           (varchar)            |
+---------------------------+-----------+--------------------------------+
|2002-02-28T14:23:57.000000Z|gwashington|Home Page                       |
|2002-02-28T14:37:15.000000Z|gwashington|Product Catalog                 |
|2002-02-28T14:39:23.000000Z|gwashington|Car Info Page -- INFINITI       |
|2002-02-28T14:49:27.000000Z|gwashington|... www.shop.com/other_page.htm .|
|2002-02-28T14:29:30.000000Z|jadams     |Home Page                       |
|2002-02-28T14:41:51.000000Z|jadams     |Special Offers Central          |
|2002-02-28T14:46:25.000000Z|jadams   |Special Offer--Purchase AUDI in Europe|
|2002-02-28T14:46:29.000000Z|jadams     |CASH REGISTER                   |
|2002-02-28T14:47:35.000000Z|jadams     |-SALE-                          |
|2002-02-28T14:28:23.000000Z|ztaylor    |Home Page                       |
|2002-02-28T14:44:13.000000Z|ztaylor    |Product Catalog                 |
|2002-02-28T14:46:22.000000Z|ztaylor    |Car Info Page -- CADILLAC       |
|2002-02-28T14:48:31.000000Z|ztaylor    |CASH REGISTER                   |
|2002-02-28T14:50:01.000000Z|ztaylor    |Product Catalog                 |
+---------------------------+-----------+--------------------------------+
```

MORE COMPLEX AGGREGATION EXAMPLE

The second example uses a postprocessor to aggregate some per-page metrics directly from the query results. The metrics are “hits” on any given URL, and “unique users” that saw the given URL. This example also formats the results with more white space than would typically display in query results.

Additionally, this example puts within the query results both the query's incoming schema from the EDW and the output schema the postprocessor defines for the results.

The postprocessor script for this more complex aggregation example:

```
# ---------- FILE: 01_lookup.pproc_in
my %easyUrlMap = (
  "home",                       "Home Page",
  "catalog",                    "Product Catalog",
  "prodinfo/infiniti",          "Car Info Page -- INFINITI",
  "prodinfo/cadillac",          "Car Info Page -- CADILLAC",
  "prodinfo/audi",              "Car Info Page -- AUDI",
  "prodinfo/saturn",            "Car Info Page -- SATURN",
  "specials",                   "Special Offers Central",
  "specials/audi_europe",       "Special Offer--Purchase AUDI in Europe",
  "purchase",                   "CASH REGISTER",
  "transaction_complete",       "-SALE-"
);

sub mapUrl
   my ($rawUrl) = @_;
   # Prepare default friendly URL (which isn't very friendly)
   my $resultUrl = "... $rawUrl ...";
   # Try to find a more friendly URL
   my $url2;
   if (($url2) = $rawUrl =~ /^www.shop.com\/(.*)\.htm$/) {
       my $url3 = $easyUrlMap{$url2};
     if (defined($url3)) {
         $resultUrl = $url3;
       }
    }
   # Return the mapped URL
   return $resultUrl;

sub pushSchemaInfoIntoTable
   my ($response, $message, $schema) = @_;
```

```
    # For each element of the schema, push out the column name/value
    my $schemaElement;
    foreach $schemaElement (@{$schema}) {
        # Find this column's name and type
     my $columnName;
     my $columnType;
     ($columnName, $columnType) = $schemaElement =~ /^(.*):(.*)$/;
     # Prepare a row
     my $rowData = {
      "direction" => $message,
      "column_name" => $columnName,
      "column_type" => $columnType,
      };
      # Push the row
      $response->addRowData(rowdata => $rowData);
     }
     my %pageMetrics;


sub postprocInit
 my ($response) = @_;
 # Initialize aggregation metrics -- we MUST do this each time
 # postprocInit is called, because we can have multiple queries
 # running within one 'atquery' session, and we want fresh results
 # each time we run a query.
 %pageMetrics = ();
 # Set the output schema, which will be different from the input schema
 my $outputSchema = [
    "section:varchar",
    "direction:varchar",
    "column_name:varchar",
    "column_type:varchar",
    "page:varchar",
    "page_metric:varchar",
    "metric_value:int32",
 ];
 $response->setMetadata(schema => $outputSchema);
 # To help illustrate input and output schemas, let's print out the
 # input/output schemas in the result set itself
 my $emptyRow = {};
 $response->addRowData(rowdata => $emptyRow);
 my $sectionRow = { section => "input/output schema" };
 $response->addRowData(rowdata => $sectionRow);
 $response->addRowData(rowdata => $emptyRow);
 my @inputSchema = $response->getIncomingSchema();
 my @outputSchema2 = $response->getSchema();
 pushSchemaInfoIntoTable($response, "input", \@inputSchema);
 $response->addRowData(rowdata => $emptyRow);
 pushSchemaInfoIntoTable($response, "output", \@outputSchema2);
 $response->addRowData(rowdata => $emptyRow);

sub postprocRow
 my ($response, $inputRow) = @_;
 # Get relevant fields out of the row
 my $userName = $inputRow->getColumnValue("user_name");
 my $rawUrl = $inputRow->getColumnValue("url");
 # Translate URL to a more friendly value -- we assume
 # the return value is unique
 my $page = &mapUrl($rawUrl);
 # We're doing per-page aggregation, make sure we have space
 # for this page (url)
 if (!defined($pageMetrics{$page})) {
    $pageMetrics{$page} = {
       numHits => 0,
```

```
       uniqueUsers => {},
    };
 }
 # Now update the per-page aggregates
 $pageMetrics{$page}->{numHits}++;
 $pageMetrics{$page}->{uniqueUsers}->{$userName}++;


sub postprocFinal
 my ($response) = @_;
 # Start a new section
 my $emptyRow = {};
 $response->addRowData(rowdata => $emptyRow);
 my $sectionRow = { section => "page metrics" };
 $response->addRowData(rowdata => $sectionRow);
 $response->addRowData(rowdata => $emptyRow);
 # Present each page and its metrics
 my $page;
 foreach $page (sort(keys(%pageMetrics))) {
    # Add a row for page hits
    my $row = {
       page => $page,
       page_metric => "hits",
       metric_value => $pageMetrics{$page}->{numHits}
    };
    $response->addRowData(rowdata => $row);
    # Add a row for number of unique users
    my @uniqueUsers = keys(%{$pageMetrics{$page}->{uniqueUsers}});
    my $row = {
       page_metric => "unique users",
       metric_value => ($#uniqueUsers + 1)
    };
    $response->addRowData(rowdata => $row);
    # Add empty row
    $response->addRowData(rowdata => $emptyRow);
 }
# ---------- END OF FILE: 02_aggregate.pproc_in
```

The following command runs the example:

```
echo "SELECT ts, user_name, url FROM test DURING ALL;" | \
      atquery lmshost:8072 --namespace=myNamespace.pproc_example -
      --postproc=02_aggregate.pproc_in
```

The query results are illustrated below:

section (varchar)	direction (varchar)	column_name (varchar)	column_type (varchar)	page (varchar)	page_metric (varchar)
input/output schema					
	input	ts	timestamp		
	input	user_name	varchar		
	input	url	varchar		
	output	section	varchar		
	output	direction	varchar		
	output	column_name	varchar		
	output	column_type	varchar		
	output	page	varchar		
	output	page_metric	varchar		
	output	metric_value	int32		
page metrics					
				-SALE-	hits unique user
				... www.shop.com/other_page.htm ...	hits unique user
				CASH REGISTER	hits unique user
				Car Info Page -- CADILLAC	hits unique user
				Car Info Page -- INFINITI	hits unique user
				Home Page	hits unique user
				Product Catalog	hits unique user
				Special Offer--Purchase AUDI in Europe	hits unique user
				Special Offers Central	hits unique user

Query Postprocessor Sample Script

This topic documents a query postprocessor example script that creates and runs multiple postprocessors against an EDW. This file is not itself the postprocessor script, but rather, it does the following:

- loads some data into a table
- queries the un-postprocessed data to verify the information is in the table
- creates a simple postprocessor script that does a simple lookup, and runs that script against the original query
- creates a more complex postprocessor script that does some client-side data aggregation, and runs that script against the original query

For more information, see Query Postprocessor Examples on page 87.

To use the sample query postprocessor script

1. Copy the script file to an executable file on a machine that supports the `sh` or `bash` scripting environment (for example, a Unix machine).
2. Ensure the values of the following variables are set appropriately:
   - `_HOST, LMS_PORT`
   - `LMS_CLIENT_TOOL_DIR`

3. Execute the file against the live EDW server.
4. Examine the client-side files and output returned; refer to the explanation in Query Postprocessor Examples on page 87.

```
#!/bin/bash
#-----------------------------------------------------------------------
# pproc_example.bash -- a set of example for Query Postprocessing
#-----------------------------------------------------------------------
#-----------------------------------------------------------------------
# To run, please set the following values to point to the correct
# SLS instance, and to the directory containing the
# tools 'atload', 'atquery', 'atmanage', and 'atview'.
LMS_HOST=localhost
LMS_PORT=7010
LMS_CLIENT_TOOL_DIR=/home/nwatson/mytools_nfw/eng_nfw/mill/bin

#-----------------------------------------------------------------------
# Miscellaneous
SUFFIX="3"
VISIBLE="=+=+=+=+=+=+=+=+=+="
NAMESPACE="myNamespace.pproc_example"
TNAME=test
FULLTNAME="${NAMESPACE}.${TNAME}"
ATVIEW0="${LMS_CLIENT_TOOL_DIR}/atview${SUFFIX}"
ATVIEW="${ATVIEW0} ${LMS_HOST}:${LMS_PORT} --namespace=${NAMESPACE}"
ATMANAGE0="${LMS_CLIENT_TOOL_DIR}/atmanage${SUFFIX}"
ATMANAGE="${ATMANAGE0} ${LMS_HOST}:${LMS_PORT} --namespace=${NAMESPACE}"
ATLOAD0="${LMS_CLIENT_TOOL_DIR}/atload${SUFFIX}"
ATLOAD="${ATLOAD0} ${LMS_HOST}:${LMS_PORT} --namespace=${NAMESPACE}"
ATQUERY0="${LMS_CLIENT_TOOL_DIR}/atquery${SUFFIX}"
ATQUERY="${ATQUERY0} ${LMS_HOST}:${LMS_PORT} --namespace=${NAMESPACE}"

#-----------------------------------------------------------------------
# Make sure client-side tools exist
if [ ! -x ${ATVIEW0} -o \
   ! -x ${ATMANAGE0} -o \
   ! -x ${ATLOAD0} -o \
   ! -x ${ATQUERY0} ]; then
   echo
   echo "ERROR: Client side tools not found!"
   echo
   exit -1
fi

#-----------------------------------------------------------------------
# Make sure SLS (or at least master node) is running
echo "SELECT * FROM properties;" | \
   ${ATQUERY} --namespace=system - > /dev/null 2>/dev/null
if [ $? != 0 ]; then
    echo
    echo "ERROR: Cannot contact SLS!"
    echo
    exit -1
fi

#-----------------------------------------------------------------------
# Create log data file
echo
echo ${VISIBLE} "Creating log file './logdata.pproc_in'" ${VISIBLE}
/bin/rm -f ./logdata.pproc_in 2>/dev/null
cat > ./logdata.pproc_in << EOF
Feb 28 14:23:57 2002,gwashington,http://www.shop.com/home.htm
Feb 28 14:28:23 2002,ztaylor,http://www.shop.com/home.htm
```

```
Feb 28 14:37:15 2002,gwashington,http://www.shop.com/catalog.htm
Feb 28 14:29:30 2002,jadams,http://www.shop.com/home.htm
Feb 28 14:39:23 2002,gwashington,http://www.shop.com/prodinfo/infiniti.htm
Feb 28 14:41:51 2002,jadams,http://www.shop.com/specials.htm
Feb 28 14:44:13 2002,ztaylor,http://www.shop.com/catalog.htm
Feb 28 14:46:22 2002,ztaylor,http://www.shop.com/prodinfo/cadillac.htm
Feb 28 14:46:25 2002,jadams,http://www.shop.com/specials/audi_europe.htm
Feb 28 14:46:29 2002,jadams,http://www.shop.com/purchase.htm
Feb 28 14:47:35 2002,jadams,http://www.shop.com/transaction_complete.htm
Feb 28 14:48:31 2002,ztaylor,http://www.shop.com/purchase.htm
Feb 28 14:49:27 2002,gwashington,http://www.shop.com/other_page.htm
Feb 28 14:50:01 2002,ztaylor,http://www.shop.com/catalog.htm
EOF

#-----------------------------------------------------------------------
# Create PTL script file
echo
echo ${VISIBLE} "Creating PTL file './ptl.pproc_in'" ${VISIBLE}
/bin/rm -f ./ptl.pproc_in 2>/dev/null
cat > ./ptl.pproc_in << EOF
^([^,]*),([^,]*),\s*http://([^,]*)$
TimeStr:VARCHAR,UserName:VARCHAR,Url:VARCHAR
SELECT _timestamp(TimeStr) as ts,
    UserName as user_name,
    Url as url
FROM stdin;
EOF

#-----------------------------------------------------------------------
# Drop example table if present
echo
echo ${VISIBLE} "Dropping table ${FULLTNAME}" ${VISIBLE}
${ATMANAGE} droptbl ${TNAME} > /dev/null 2>/dev/null

#-----------------------------------------------------------------------
# Load data into table
echo
echo ${VISIBLE} "Loading data into table ${FULLTNAME}" ${VISIBLE}
echo
${ATLOAD} ${TNAME} ./ptl.pproc_in ./logdata.pproc_in

#-----------------------------------------------------------------------
# Pull all the raw data
echo
echo ${VISIBLE} "Querying raw data in table ${FULLTNAME}" ${VISIBLE}
echo
echo "SELECT ts, user_name, url FROM ${TNAME} ORDER BY 2, 1 DURING ALL;" |
${ATQUERY} -

#-----------------------------------------------------------------------
# Use a simple postproc to replace URL values with more "user-friendly" values
cat > 01_lookup.pproc_in << EOF

# ---------- FILE: 01_lookup.pproc_in
my %easyUrlMap = (
   "home", "Home Page",
   "catalog", "Product Catalog",
   "prodinfo/infiniti", "Car Info Page -- INFINITI",
   "prodinfo/cadillac", "Car Info Page -- CADILLAC",
   "prodinfo/audi", "Car Info Page -- AUDI",
   "prodinfo/saturn", "Car Info Page -- SATURN",
   "specials", "Special Offers Central",
   "specials/audi_europe", "Special Offer -- Purchase AUDI in Europe",
   "purchase", "CASH REGISTER",
```

```
   "transaction_complete", "-SALE-"
);
sub postprocRow
my (\$response, \$inputRow) = @_;
# Create a copy of the input row, for output
my \$outputRow = {};
\$response->copyValuesForOutputSchema(\$inputRow, \$outputRow);
# Get the URL, prepare default friendly URL (which isn't very friendly)
my \$rawUrl = \$inputRow->getColumnValue("url");
my \$resultUrl = "... \$rawUrl ...";
# Try to find a more friendly URL
my \$url2;
if ((\$url2) = \$rawUrl =~ /^www.shop.com\\/(.*)\\.htm$/) {
my \$url3 = \$easyUrlMap{\$url2};
if (defined(\$url3)) {
\$resultUrl = \$url3;
}
}
# Replace the URL in the output row
\$outputRow->{"url"} = \$resultUrl;
# Add the modified row to the response
\$response->addRowData(rowdata => \$outputRow);
# ---------- END OF FILE: 01_lookup.pproc_in
EOF
echo
echo ${VISIBLE} "Querying data in table ${FULLTNAME}, replacing URLs" ${VISIBLE}
echo
echo "SELECT ts, user_name, url FROM ${TNAME} ORDER BY 2, 1 DURING ALL;" | \
${ATQUERY} --postproc=01_lookup.pproc_in -

#-----------------------------------------------------------------------
# Use a more complex postproc to do client-side aggregation/presentation
cat > 02_aggregate.pproc_in << EOF

# ---------- FILE: 02_aggregate.pproc_in
my %easyUrlMap = (
    "home",                         "Home Page",
    "catalog",                      "Product Catalog",
    "prodinfo/infiniti",            "Car Info Page -- INFINITI",
    "prodinfo/cadillac",            "Car Info Page -- CADILLAC",
    "prodinfo/audi",                "Car Info Page -- AUDI",
    "prodinfo/saturn",              "Car Info Page -- SATURN",
    "specials",                     "Special Offers Central",
    "specials/audi_europe",        "Special Offer -- Purchase AUDI in Europe",

    "purchase",                     "CASH REGISTER",
    "transaction_complete",         "-SALE-"
);


sub mapUrl
    my (\$rawUrl) = @_;
    # Prepare default friendly URL (which isn't very friendly)
    my \$resultUrl = "... \$rawUrl ...";
    # Try to find a more friendly URL
    my \$url2;
    if ((\$url2) = \$rawUrl =~ /^www.shop.com\\/(.*)\\.htm$/) {
        my \$url3 = \$easyUrlMap{\$url2};
        if (defined(\$url3)) {
            \$resultUrl = \$url3;
        }
    }
    # Return the mapped URL
    return \$resultUrl;
```

```
sub pushSchemaInfoIntoTable
    my (\$response, \$message, \$schema) = @_;
    # For each element of the schema, push out the column name/value
    my \$schemaElement;
    foreach \$schemaElement (@{\$schema}) {
        # Find this column's name and type
        my \$columnName;
        my \$columnType;
        (\$columnName, \$columnType) = \$schemaElement =~ /^(.*):(.*)$/;
        # Prepare a row
            my \$rowData = {
                "direction" => \$message,
                "column_name" => \$columnName,
                "column_type" => \$columnType,
            };
        # Push the row
        \$response->addRowData(rowdata => \$rowData);
    }
    my %pageMetrics;


sub postprocInit
    my (\$response) = @_;
    # Initialize aggregation metrics -- we MUST do this each time
    # postprocInit is called, because we can have multiple queries
    # running within one 'atquery' session, and we want fresh results
    # each time we run a query.
    %pageMetrics = ();
    # Set the output schema, which will be different from the input schema
    my \$outputSchema = [
        "section:varchar",
        "direction:varchar",
        "column_name:varchar",
        "column_type:varchar",
        "page:varchar",
        "page_metric:varchar",
        "metric_value:int32",
    ];
    \$response->setMetadata(schema => \$outputSchema);
    # To help illustrate input and output schemas, let's print out the
    # input/output schemas in the result set itself
    my \$emptyRow = {};
    \$response->addRowData(rowdata => \$emptyRow);
    my \$sectionRow = { section => "input/output schema" };
    \$response->addRowData(rowdata => \$sectionRow);
    \$response->addRowData(rowdata => \$emptyRow);
    my @inputSchema = \$response->getIncomingSchema();
    my @outputSchema2 = \$response->getSchema();
    pushSchemaInfoIntoTable(\$response, "input", \\@inputSchema);
    \$response->addRowData(rowdata => \$emptyRow);
    pushSchemaInfoIntoTable(\$response, "output", \\@outputSchema2);
    \$response->addRowData(rowdata => \$emptyRow);

sub postprocRow
    my (\$response, \$inputRow) = @_;
    # Get relevant fields out of the row
    my \$userName = \$inputRow->getColumnValue("user_name");
    my \$rawUrl = \$inputRow->getColumnValue("url");
    # Translate URL to a more friendly value -- we assume
    # the return value is unique
    my \$page = &mapUrl(\$rawUrl);
    # We're doing per-page aggregation, make sure we have space
```

```
    # for this page (url)
    if (!defined(\$pageMetrics{\$page})) {
        \$pageMetrics{\$page} = {
            numHits => 0,
            uniqueUsers => {},
        };
    }
    # Now update the per-page aggregates
    \$pageMetrics{\$page}->{numHits}++;
    \$pageMetrics{\$page}->{uniqueUsers}->{\$userName}++;


sub postprocFinal
    my (\$response) = @_;
    # Start a new section
    my \$emptyRow = {};
    \$response->addRowData(rowdata => \$emptyRow);
    my \$sectionRow = { section => "page metrics" };
    \$response->addRowData(rowdata => \$sectionRow);
    \$response->addRowData(rowdata => \$emptyRow);
    # Present each page and its metrics
    my \$page;
    foreach \$page (sort(keys(%pageMetrics))) {
        # Add a row for page hits
        my \$row = {
            page => \$page,
            page_metric => "hits",
            metric_value => \$pageMetrics{\$page}->{numHits}
        };
        \$response->addRowData(rowdata => \$row);
        # Add a row for number of unique users
        my @uniqueUsers = keys(%{\$pageMetrics{\$page}->{uniqueUsers}});
        my \$row = {
            page_metric => "unique users",
            metric_value => (\$#uniqueUsers + 1)
        };
        \$response->addRowData(rowdata => \$row);
        # Add empty row
        \$response->addRowData(rowdata => \$emptyRow);
    }

# ---------- END OF FILE: 02_aggregate.pproc_in
EOF

echo
echo ${VISIBLE} "Querying data in table ${FULLTNAME}, aggregating data"
${VISIBLE}
echo
echo "SELECT ts, user_name, url FROM ${TNAME} DURING ALL;" | \
${ATQUERY} --postproc=02_aggregate.pproc_in -

#-----------------------------------------------------------------------
# Done
echo
exit 0
#----------
# End of file 'pproc_example.bash'
#----------
```

Managing an EDW Data Store

This section describes the `atmanage` utility. It contains the following topics:

Synopsis

```
atmanage [options] <cluster_list> <action>
```

Description

The `atmanage` utility enables management of various aspects of the EDW, including managing users, roles, and permissions; creating, deleting, and renaming tables; and stopping tasks.

Because atmanage requests may alter the state of the EDW, use this utility carefully. To examine the current state of the EDW, use the `atview` command.

Options

Information about atmanage

--version

Displays version of atmanage.

--help, --longhelp

Displays online help in short form (`--help`), or with additional detail (`--longhelp`).

Specify a Namespace

--namespace=<name>

The table namespace in which to manage items. The default namespace is default. To specify the top level, use "" (empty quotation marks).

Note: When you run `atmanage`, you can specify a table name explicitly (for example, `ns1.ns2.ns3.mytable`) or use the `--namespace=<namespace>` flag to specify the

namespace. When you use the namespace flag, you can specify the namespace once for an entire command; SenSage AP prepends the specified namespace to each table name used in the rest of the request.

Authentication Options

--user=<username>

This parameter authenticates the specified name as an EDW user. If no user is specified, atmanage reads the environment variable ADDAMARK_USER. If it cannot find this environment variable, atmanage uses the operating system to get the name of the user who is logged in.

Note: To log in as the guest user, do not specify the `--pass` or `--shared-secret` flags.

--pass=default

This parameter provides several methods to specify the password. You can specify the password in clear text, place the password in a file and specify the location of the file, or set it in the `ADDAMARK_PASSWORD` environment variable. The syntax is:

--pass=<password>

--pass=pass:<password>

--pass=file:<filename>

--pass=default

When you specify `default` as the password, `atmanage` looks in the `ADDAMARK_PASSWORD` environment variable. For more information, see Setting the ADDAMARK_PASSWORD Environment Variable on page 71.

--shared-secret=<secret>

This parameter provides several methods to specify the shared secret. You can specify the secret in clear text, place the secret in a file and specify the location of the file, or set it in the `ADDAMARK_SHARED_SECRET` environment variable. This parameter provides two methods to specify the shared secret. You can either specify the secret in clear text or place the secret in a file and specify the location of the file. The syntax is:

--shared-secret=<secret>

--shared-secret=file:</full_path/filename>

--shared-secret=default

When you specify default as the password, atmanage looks in the `ADDAMARK_SHARED_SECRET` environment variable.

Note: The value for `--pass` takes precedence over the value for `--shared-secret`, if both are present.

--assume-role=role<role_to_be_assumed>

This parameter causes the operation to be run as if the user had assumed the specified EDW role. The assumed role must have admin privileges. For more information about managing users and roles, see Authentication Management on page 103“.

Debugging

--[no]dbgprint-request

Dumps client-server request, and then exits. The default is `--nodbgprint-request`.

This option forces atmanage to print the execution requests that it would have sent to the EDW, and then stops execution rather than proceeding with the action. This is useful for debugging situations and for understanding the EDW API.

--[no]raw

This options forces atmanage to print the raw XML results that the EDW returns, instead of the interpreted results atmanage usually shows.

The default is `--noraw`.

--[no]show-responses

When reading the response from the EDW server, `atmanage` filters out certain messages, reformats others, and attempts to present a simpler response suitable for most cases. The `--show-responses` flag forces `atmanageto` show the uninterpreted responses from the EDW in addition to normal output. The default is `--noshow-responses`.

Arguments

The `atmanage` utility takes as its first argument the *<cluster_list>* for the EDW that is to be managed. The *<cluster_list>* represents a comma-separated list of *<host>:<port>* pairs. The list must be enclosed in quotation marks (").

Specifying a list rather than a single host:port pair allows the utility to run even if the first of the specified EDW nodes is down. Each host:port pair identifies the EDW instance; for example, `sls01:8072`.

The command tries each host:port pair in order (left to right) until it finds one that allows a TCP (Transmission Control Protocol) connection. Failure to establish a TCP connection indicates that the EDW host is down. If the connection succeeds but the command fails, the command does not attempt to establish additional connections.

Example:

```
atmanage --user=administrator --password=changeme
  "sls01:8072,sls02:8072,sls03:8072" adduser joe_cool
```

The command then expects an *<action>* argument, which indicates how the program should act on the specified EDW. Depending on the *<action>*, there are a number of additional arguments, as indicated in the table below.

Important: : These commands must be submitted by a user with administrator permission.

Action/Admin Action	Additional Arguments	Description
Authentication Management		
`adduser`	*<username> <password>*	Add a user
`deleteuser`	*<username>*	Delete a user
`changepassword`	*<username> <new_password>*	Add or change a user password
`addrole`	*<rolename>*	Add a role
`deleterole`	*<rolename>*	Delete a role
`addpermission`	*<permission_name> <type> <admin_value>*	Create a permission that can later be associated with a role; *<type>* is either 'token' or 'property'
`deletepermission`	*<permission_name>*	Delete a permission object
`addrolepermission`	*<rolename> <permission_name> <value>*	Associate a permission with a role
	<rolename> sls.namespace <namespace>	Grant a role permission to access a namespace
`deleterolepermission`	*<rolename> <permission_name> <value>*	Dissociate a permission from a role
	<rolename> sls.namespace <namespace>	Revoke a role's permission to access a namespace
`adduserrole`	*<username> <rolename>*	Associate a user with a role
`deleteuserrole`	*<username> <rolename>*	Dissociate a user from a role
`setuserstate`	*<username> [enable|disable]*	Enable or disable a user
`setrolestate`	*<rolename> [enable|disable]*	Enable or disable a role
Table management		
`createtbl`	<table_name> <column1>:<type> [<column2>:<type>[...]]	Create a table with the given columns **Note:** This option enables creation of tables but not views.
`renametbl`	<old_tablename> <new_tablename>	Rename a table and/or move it between namespaces **Note:** This option enables renaming and moving tables but not views.

Action/Admin Action	Additional Arguments	Description
droptbl	<table_name>	Delete a table **Note:** This option enables deletion of tables but not views.
quiesce		Set the data store to read-only mode
resume		Restore the data store to its nominal mode
backuptbl	*start <table_name>*	Backup a table
	commit <table_name>	Commit the backup
	rollback <table_name>	Rollback a table backup to its original state
restoretbl	*start <table_name>*	Create a sandbox into which you restore a previously backed up table
	commit <table_name>	Commit the sandbox into the data store
	rollback <table_name>	Rollback a table restore to its original state
archivetbl	*<table_name> <nsi> [end_time] [start_time]*	Archive the requested (ISO) time range from the requests table to the specified NSI
setnsi	*[nsi] [nsa]*	Add or change NSI/NSA association
deletensi	*[nsi]*	Delete an NSI/NSA association
listnsi		List NSI/NSA associations
updateuploadinfo	*<table_name> <CSV_file>*	Update upload information for the table from CSV file
adduploadinfo	*<table_name> <CSV_file>*	Add upload information for the table from CSV file
deleteuploadinfo	*<table_name> <CSV_file>*	Delete upload information for associated IDs
canceltask	*<request_id>*	Kill the given task (auto-detects internal task ID rather than application-defined task ID).

Action/Admin Action	Additional Arguments	Description
`compacttbl`	*<table_name> [<seconds_to_run>] [<enable_compact_archived_data>][<number>d/m/h]*	Compact all fragmented portions of a table. Optionally, can facilitate scheduled compaction in the Loader by specifying: • number of seconds to run; compacting stops when the specified number is exceeded. • whether to compact archived data and the recent number_of_days, number_of_minutes, or number_of_hours. For more information, see:compacttbl on page 108 and *Defining Loaders* in Chapter 2, “SenSage AP Collector Configuration” in the *Collector Guide*
`force-upgrade`	*<table_name>*	Manually upgrade the data store Upgrades the table's storage format to the latest version. **Important:** May be very slow, and may cause the table to be unreadable by previous versions of the software. Consult IgniteTech Support before using.

Authentication Management

Authentication Management is provided through the Analyzer tool that enables administrators to manage users, roles, and passwords. For conceptual information, see SenSage AP Security Model and Postgres on page 162. For details on using the Analyzer, see the SenSage AP Analyzer documentation.

Table Management

This topic describes utilities provided by the EDW that enable administrators to manage tables and other database objects.

Important:

- These commands must be submitted by a user with administrator permission.
- These commands affect only the EDW instance in which they are run; in other words, creating a table in one instance does not create the table in other instances.

This topic describes the following utilities:

createtbl

createtbl <table_name> <column1>:<type> [<column2>:<type> [...]]

The `atload` utility automatically creates a table at load time if the specified table does not exist. However, there are times you want to create an empty table in a given namespace, with a given name and set of columns. The `atmanage` utility provides the `createtbl` action for this purpose. For example, the following command creates a table named quotes in the external namespace and defines six columns:

```
atmanage --namespace=external slshost:8072 createtbl quotes \
         ts:TIMESTAMP col1:VARCHAR col2:INT32 col3:INT64
col4:TIMESTAMP col5:FLOAT
```

Note: When you create a table, keep the following in mind:

- You must always define a column named **ts**, with SQL data type **TIMESTAMP**.
- The EDW creates automatically an extra column named _uploadid, with SQL data type varchar. The column holds the ID of the upload operation that inserted each row into the table.
- You can define other columns.

---

renametbl

renametbl <old_tablename> <new_table_name>

The `renametbl` action allows you to:

- correct spelling mistakes
- roll out ("swap in") new data sets
- establish or change namespaces
- establish table-naming conventions.

The following example moves the quotes table from the external namespace to the external.websvcs namespace, and changes the table's name from quotes to stock_quote_log:

```
atmanage slshost:8072 --namespace=external renametbl quotes \
          websvcs.stock_quote_log
```

The following example illustrates two commands. The first creates a table in the default namespace (default). The second moves the table from the default namespace to a namespace named new_namespace:

```
atmanage slshost:8072 createtbl my_table ts:TIMESTAMP val:VARCHAR
          atmanage slshost:8072 --namespace '' renametbl \
          myNamespace.my_table new_namespace.my_table
```

Before specifying the old and new table names, the example above uses an empty string to identify the top namespace. If this flag were missing from the command, the EDW would assume the old table was myNamespace.myNamespace.my_table and would rename the table to myNamespace.new_namespace.my_table.

droptbl

droptbl <table_name>

The droptbl action allows you to delete a specified table if none of its data is archived to a Nearline storage device. If any of its data has been archived, use the retire command to remove the non- archived data. Only after all of its data has been removed from the Nearline storage device can you drop the table. For more information, see Retiring Data on page 115.

Important: :

- Because typical EDW tables are gigantic in size, no undo is possible. Therefore, IgniteTech recommends that you use this action with extreme caution and that you back up your tables first.
- When you issue `droptbl` against a table that has any data under retention on a Nearline storage device, the droptbl command fails. Its error message provides the date in the future when the Nearline storage device will retire the data and advises you to use retire to remove local and non-retained data.
- When you issue droptbl against a table that has data archived to a Nearline storage device but the data is not under retention or the retention date has past, the droptbl command fails. Its error message informs you to use retire to remove the archived data before running droptbl.

For more information, see **Installing Perl Modules** in "Perl Subroutines" in the *Event Data Warehouse Guide* and Retiring Data on page 115.

The following example deletes the table created and renamed in renametbl above:

```
atmanage slshost:8072 --namespace=external.websvcs droptbl
stock_quote_log
```

backuptbl

backuptbl [start|commit|rollback] <table_name>

The backuptbl action allows you to back up a table. You must use this action in conjunction with a standard backup tool and the atquery utility. For more information, see Backing Up and Restoring an EDW Table on page 125.

```
backuptbl [start|commit|rollback] <table_name>
```

This action has three arguments:

- `start`—begins the backup session
- `commit`—commits the changes
- `rollback`—returns the table data to its pre-backup state

restoretbl

restoretbl [start|commit|rollback] <table_name>

The `restoretbl` action allows you to restore a table from a backup. You must use this action in conjunction with a standard backup tool. For more information, see Backing Up and Restoring an EDW Table on page 125.

This action has three arguments:

- `start`—creates a sandbox directory to which the table can be restored; you can use the sandbox to double-check the data before you run a restore on your EDW instance's data store
- `commit`—commits the sandbox back into the data store
- `rollback`—returns the table data to its pre-restored state

Important: You cannot use the `restoretbl` option to upgrade a datastore from an older version of SenSage AP to a newer version. You can only use `restoretbl` to restore tables created using the same version of SenSage AP.

```
restoretbl [start|commit|rollback] <table_name>
```

canceltask

canceltask <request_id>

The `canceltask` action allows you to cancel a task currently running on the EDW.

Note:

- Because the "atview tasks" command illustrated above returns the request ID as well as the task ID, you can use either value to cancel the task.
- You can use the "atview tasks" command to obtain the request ID and task ID of tasks run from the SenSage AP Console.

REQUEST IDS

Client programs such as atview, atmanage, atquery, and atload interact with the EDW to send requests for various actions (for example, requests to load data into tables or query the data in tables), and wait for results. Requests have an associated request ID, which the client program defines and prints. For example, atload might print the following information in response to running a load:

```
request id: [app=atload,u=harry,h=perf02.hq.myco.com,id=e4f5e41f]
```

To cancel the request's operation in the EDW, take the value that follows id= above and use it in a command like the one below:

```
atmanage slshost:8072 canceltask e4f5e41f"
```

Killing a task is generally safe. However, IgniteTech recommends that, if possible, you use the client-side utility that emits the request to perform the kill. For example, press CTRL-C during a load with atload. The atmanage canceltask command was designed for system administrators who detect "runaway" jobs submitted by remote users.

TASK IDS

A client program may not define an appropriate, unique, and readily usable request ID for the requests it processes. For any request running on the EDW, there is always a unique task ID that the EDW server itself defines. Each request sent to the EDW can involve multiple processes that run on multiple hosts in the instance. The EDW treats all these processes as a single "task".

If a client program does not define a usable request ID, an administrator can use the atmanage utility to cancel a task by specifying this task ID. This option ensures that, no matter what shortcomings a client program may have, there is always a way to cancel a running task. To find the internal task IDs, run "atview tasks" to list the tasks. The graphic below illustrates running this command and its output.

Locate the internal_task_id for the desired request and specify its value as the *request_id* when you run `canceltask`.

For example:

```
atmanage slshost:8072 canceltask CAC6B255FBAF505935DAF3BAC2AD0B5B
```

Note:

- Because the "`atview tasks`" command illustrated above returns the request ID as well as the task ID, you can use either value to cancel the task.
- You can use the "`atview tasks`" command to obtain the request ID and task ID of tasks run from the SenSage AP Console.

compacttbl

compacttbl <table_name> [<num_seconds>]
[<enable_compact_archived_data>][<number>d/m/h]

By default, the EDW loads new data into separately indexed fragments to avoid the overhead associated with combining data into more efficient files for long-term archival. If you have many loads in the same time range (especially small loads under 10,000 records), query performance and/or compression can suffer.

Optionally you can use this utility to compact archived data as well as local data. To do so, set *<enable_compact_archived_data>* to `1`. You can also set the number of seconds compacting occurs before it stops. The following example compacts local and archived data for **myTable**; compacting stops after 10 minutes (600 seconds).

```
compacttbl myTable 600 1
```

Set the duration to run compact on recent data by providing the last days, minutes, and hours. Specify the duration using the following syntax:

```
[<number>d/m/h]
```

The following example compacts local and archived data for myTable for the last 2 days; compacting stops after 10 minutes (600 seconds).

```
compacttbl myTable 600 1 2d
```

Assuming that the current date and time is April 25th, 2016 11:40:00, in this example, the manual compact tool will compact leaves with timestamp from April 23rd, 2016 00:00:00 till the current timestamp the case of days, a day is always calculated from the start of the specified day (00:00 hours) till the current date and is not calculated based on current hours and minutes.

Similarly, the following example compacts local and archived data for the last 30 minutes; compacting stops after 10 minutes (600 seconds).

```
compacttbl myTable 600 1 30m
```

Assuming that the current date and time is April 25th, 2016 11:40:00, in this example, the manual compact tool will compact leaves with timestamp from April 25th, 2016 11:10:00 till the current timestamp.

The following example compacts local and archived data for the last 4 hours.

```
compacttbl myTable 600 1 4h
```

Assuming that the current date and time is April 25th, 2016 11:40:00, in this example, the manual compact tool will compact leaves from April 25th, 2016 7:40:00 till the current timestamp.

Note: One day is not calculated as 24 hours, make sure that you set the compact duration appropriately.

Important: This command cannot compact any node that stores data archived to a Nearline storage device. Only after the retention period has ended can the Nearline storage data be compacted. For more information, see:

- Installing Perl Modules in Chapter 8, “Perl Subroutines” in the Event Data Warehouse Guide
- Compact in Chapter 2, “SenSage AP Collector Configuration” in the Collector Guide

Important: : Consult IgniteTech Support before using this command, because it can be very slow and often yields little improvement.

force-upgrade

force-upgrade <table_name>

To avoid modifying terabytes of data during upgrades, the EDW only upgrades the file format of its data files as new data is loaded. Over multiple upgrades, this can result in a data store with many versions of the files some rare cases, newer versions of the data store provide better performance or other features, and it is advantageous to manually upgrade production tables to the latest version.

Important: Consult IgniteTech Support before using this command, because it can be very slow and results in the table being unreadable by older versions of the EDW.

Errors and Return Values

The atmanage utility prints an exit code of 0 on success, or a non-zero exit code and an error condition message upon error. For a full list, see Errors and Return Values for Data-Store Utilities on page 118.

Examining the State of an EDW Data Store

This section describes the atview utility. It contains the following topics:

Synopsis

```
atview [options] <cluster_list> <item> [<item> […]]
```

Description

The atview utility examines the state of an EDW instance, such as its metadata or status.

Note: Requests from atview do not alter the state of the EDW. For information on changing an EDW instance, see “Managing an EDW Data Store on page 98.

Options

Information About atview

--version

Displays version of atview.

--help, --longhelp

Displays online help for command in short form (--help), or with additional detail (-- longhelp).

--timestamps

This parameter specifies that the output includes the minimum and maximum timestamp values of the data in each table. If the table is empty, the display shows "no records". When views are displayed in the output, those rows do not include the minimum and maximum timestamp values. The command executes more quickly when run without this parameter.

Namespace Identification

--namespace=<name>

Identifies the namespace from which to query information. The default is default. To specify the top level, use "" (empty quotation marks).

If you run the `atview` command without specifying a table name, the output includes all tables in the specified namespace. For example, if you specify `--namespace=hq.west`, the command returns the requested information for every table in the hq.west namespace.

If you specify a table when you run the atview command, the output includes the requested information only for the specified table. For example, if you specify `--namespace=hq.west columns mytbl`, the output is specific to `hq.west.mytbl`.

Note: When you run atview, you can specify a table name explicitly (for example, ns1.ns2.ns3.mytable) or use the `--namespace=<namespace>` flag to specify the namespace. When you use the namespace flag, you can specify the namespace once for an entire

command; SenSage AP prepends the specified namespace to each table name used in the rest of the request.

The command will display an 'error' column and include error messages in the column when there are inconsistencies across hosts in the cluster. For example, error type TABLE_OPEN_FAILURE displays in the 'error' column when the owner of a view is deleted from the system and the view no longer carries the permissions associated with the former owner.

Authentication Options

--user=<username>

This parameter authenticates the specified name as an EDW user. If no user is specified, atview reads the environment variable `ADDAMARK_USER`. If it cannot find this environment variable, it uses the operating system to get the name of the user who is logged in.

Note: To log in as the **guest** user, do not specify the `--pass` or `--shared-secret` flags.

--pass=<pwmethod>

This parameter provides several methods to specify the password. You can specify the password in clear text, place the password in a file and specify the location of the file, or set it in the `ADDAMARK_PASSWORD` environment variable. The syntax is:

--pass=<password>

--pass=pass:<password>

--pass=file:<filename>

--pass=default

When you specify default as the password, `atview` looks in the ADDAMARK_PASSWORD environment variable. For more information, see Setting the ADDAMARK_PASSWORD Environment Variable on page 71.

--shared-secret=<secret>

This parameter provides two methods to specify the shared secret. You can either specify the secret in clear text or place the secret in a file and specify the location of the file. The syntax is:

--shared-secret=<secret>

--shared-secret=file:</full_path/filename>

--shared-secret=default

When you specify default as the password, atmanage looks in the `ADDAMARK_SHARED_SECRET` environment variable.

Note: The value for `--pass` takes precedence over the value for `--shared-secret`, if both are present.

--assume-role=role<role_to_be_assumed>

This parameter causes the operation to be run as if the user had assumed the specified EDW role. The assumed role must have admin privileges. For more information about managing users and roles, see Authentication Management on page 103.

--timestamps

This parameter specifies that the output includes the minimum and maximum timestamp values of the data in each table. If the table is empty, the display shows "no records". When views are displayed in the output, those rows do not include the minimum and maximum timestamp values.

Debugging

--[no]dbgprint-request

Dumps client-server request, and then exits. The default is `--nodbgprint-request`.

--[no]raw

This option forces atview to print the execution requests that it would have sent to the EDW, and then stops execution rather than proceeding with the action. This is useful for debugging situations and for understanding the EDW API.

--[no]show-responses

When reading the response from the EDW server, atview filters out certain messages, reformats others, and attempts to present a simpler response suitable for most cases. The `--show-responses` flag forces `atview` to show the uninterpreted responses from the EDW in addition to normal output. The default is `--noshow-responses`.

Arguments

The `atview` utility takes as its first argument the *<cluster_list>* for the EDW that is to be examined. The *<cluster_list>* represents a comma-separated list of *<host>:<port>* pairs. The list must be enclosed in quotation marks (").

Specifying a list rather than a single `host:port` pair allows the utility to run even if the first of the specified EDW nodes is down. Each `host:port` pair identifies the EDW instance; for example, **sls01:8072**.

The command tries each host:port pair in order (left to right) until it finds one that allows a TCP (Transmission Control Protocol) connection. Failure to establish a TCP connection indicates that the EDW host is down. If the connection succeeds but the command fails, the command does not attempt to establish additional connections.

Example:

```
atview --user=administrator --password=changeme
"sls01:8072,sls02:8072,sls03:8072" --namespace=s0meNS tables
```

The command then expects an <*item*> argument, which indicates the EDW object, state, or task to examine. Depending on the <*item*>, there are a number of additional arguments, as indicated in the table below.

Item	Additional Arguments	Description
tables		view tables and views in a namespace
namespaces		view namespaces (within a namespace)
columns		view columns for all tables in a namespace
columns	<*table_name*>	view columns for one table
tasks		show tasks currently running across the instance
disk		show disk capacity, use, and free space
tabledisk	<*table_name*>	show disk use for the specified table, not including disk used for directories

Usage and Examples

tables: Viewing Tables and Views in a Namespace

Lists the tables and views in a given namespace. If you specify the `--timestamps` option, the output includes the start and end timestamps, which are useful to create `DURING` clauses.

Example:

```
atview --user=administrator --pass=s0mep@ss lmshost:8072 \
--namespace=external.websvcs --timestamps tables
```

Note:

Views are listed always with "(no records)" in the columns for minimum and maximum timestamps. The command will display an 'error' column and include error messages in the column when there are inconsistencies across hosts in the cluster. Otherwise the 'error' column is empty.

```
|---------------+---------------+---------------+-------------+-----------|
|namespace      |table_name     |min_time       |max_time     |error      |
|---------------+---------------+---------------+-------------+-----------|
|default        |events         |(no records)   |(no records) |           |
|default        |sys_alerts     |(no records)   |(no records) |           |
|---------------+---------------+---------------+-------------+-----------|
```

namespaces: Viewing Namespaces Within a Namespace

Lists the namespaces within a given namespace. The output includes each namespace, the number of tables within it, and the cumulative number of tables for all namespaces within the namespace.

Example:

```
atview --user=administrator --pass=pass:s0mep@ss lmshost:8072 \
--namespace=external.websvcs namespaces
```

columns: Viewing Columns for All Tables in a Namespace

Lists the columns for every table in the given namespace, including each column's data type, and the start and end timestamps for the table. Timestamps are stored in the ts column.

Example:

```
atview --user=administrator --pass=pass:s0mep@ss lmshost:8072 \
--namespace=external.websvcs columns
```

columns <*table_name*>: Viewing Columns for One Table

Lists the columns for one table.

Example:

```
atview --user=administrator --pass=pass:s0mep@ss lmshost:8072 \
--namespace=external.websvcs columns stock_quote_log
```

tasks: Showing Tasks Currently Running Across the EDW instance

Lists tasks currently running in the EDW instance.

Example:

```
atview --user=administrator --pass=pass:s0mep@ss lmshost:8072 tasks
```

disk: Showing Disk Capacity, Used Disk Space, and Free Disk Space

Lists the total disk capacity, amount in use, and the amount of free space across the instance. The amount of free disk is a conservative, weighted estimate, which attempts to account for the uneven filling of disk space as new log records get loaded.

Example:

```
atview --user=administrator --pass=pass:s0mep@ss lmshost:8072 disk
```

tabledisk <table_name>: Showing Disk Use for the Specified Table

Lists the amount of disk space consumed for the given table, across the EDW instance, not including disk space consumed by directories.

Example:

```
atview --user=administrator --pass=pass:s0mep@ss lmshost:8072 \
--namespace=external.websvcs tabledisk stock_quote_log
```

Errors and Return Values

See Errors and Return Values for Data-Store Utilities on page 118.

Retiring Data

This section contains the following topics:

Synopsis

```
RETIRE FROM <table_name> <command> [FORCE]
```

Description

The retire command is a SenSage AP SQL extension rather than a command-line tool and, therefore, is unlike the utilities documented in this chapter. You do not run retire from the command line as you do the utilities. Instead, you use the `atquery` utility to run it.

The retire command replaces the deprecated `atretire` utility as the tool for retiring data from an EDW instance. The retire command provides significantly enhanced performance over `atretire`.

Moreover, the syntax of the `retire` command is easier and more comprehensive than that of the `atretire` command. While the `atretire` command required you to specify the cutoff time in the predefined syntax of a timestamp, the retire command allows you to specify the cutoff time as any SenSage AP SQL expression that evaluates to a timestamp data type.

The retire command permanently deletes data from the specified table in the EDW instance. Use this command to:

- Reclaim Space

  Although the combination of EDW compression and ever-increasing disk capacity has reduced the urgency of deleting older data, there are still applications that require this regular maintenance. You can use `retire` to delete old data, retiring on a range of upload IDs or timestamps.

Important: Retiring data from an EDW table removes only data that is not stored under retention on a Nearline storage device. The command skips archived data that is under retention and returns a message that provides the date in the future when the Nearline storage device will retire the data. That date is the earliest you can retire the data. The command does remove archived data that is not under retention or its retention period has passed. On some Nearline storage devices, however, in order to ensure retire time specifications are enforced, the command may be required to restore some data not under retention and let it remain online even though it falls outside the retire date. For more information on Nearline storage, see Archiving to Nearline Storage on page 181.

- Undo Duplicate Loads

  Occasionally, the same log file is accidentally loaded twice. You can use retire to remove one of the copies. You delete based on upload ID.

- Undo Loads Containing Bad Data

  Occasionally, the source data itself is incorrect, and you want to fix the data in the long-term archives. If the problem is limited to one log file, IgniteTech recommends backing out this one file, fixing the source data, then reloading. However, if the corruption occurred over the course of weeks, it may be preferable to use retire to delete selected records.

Options

See the options documented for `atquery`. For more information, see Querying Data on page 74.

Parameters

See the parameters for `atquery`. For more information, see Querying Data on page 74.

Arguments

Argument	Additional Arguments	Description
<tablename>	target table name	
<command>	`UPLOADS <upload_id> [, <upload_id>[...]][DURING <timestamp>, <timestamp>]` Deletes the data associated with specified load(s)— specified by upload ID.BEFORE <timestamp_expression> Deletesdatawhosetimestamp('ts')value is prior to the specified timestamp expression.	One of two deletion commands to perform on the specified table For more information, see Specifying the Data for Deletion on page 117.

Using the FORCE Keyword

The FORCE keyword causes the RETIRE command to remove corrupted data. If you run RETIRE without this keyword and the EDW detects any corruption in the data to be retired, it removes only the non-corrupted data specified by the command and returns an error about the corrupted data.

Hexis Cyber Solutions recommends that you run RETIRE without the FORCE keyword when you first retire old or bad data. Running without forcing the retirement allows the EDW to inform you of corrupted data and provides the opportunity for you to examine that data. After you have taken the necessary steps to prevent similar corruption in the future, run RETIRE with the FORCE keyword to remove the corrupted data.

Specifying the Data for Deletion

There are two ways to specify which data to delete.

- Using the UPLOADS Command on page 117
- Using the "BEFORE <timestamp_expression>" Command on page 117

Using the UPLOADS Command

To permanently delete the records associated with a given set of upload IDs, provide a comma- separated list of upload IDs. The syntax is:

```
UPLOADS <upload_id> [, <upload_id>[...]][DURING <timestamp>, <timestamp>]
```

where the DURING causes the RETIRE command to delete only the data in the specified time and date range. You can choose the optional DURING clause when deleting an upload ID that encompasses a very wide time and date range. Using the DURING clause can help avoid possible negative impacts on performance by breaking the deletion job into a series of smaller operations addressing shorter time ranges within the larger min and max timestamp of the load.

Example:

```
atquery localhost:8072 -e RETIRE FROM syslog
 UPLOADS ’31AEAF94551AA4366519F4)EB3F9A4FA’, ’F8OC8F2003D655A8444A8B4EF172EC7a’
```

Example With DURING clause:

```
atquery localhost: 8072 -e "RETIRE FROM syslog
 UPLOADS ’31AEAF94551AA4366519F4)EB3F9A4FA’, ’F8OC8F2003D655A8444A8B4EF172EC7a’

 DURING time(’FEB 01 00:00:00 2011’), time(’May 31 23:60:00 2011’)"
 OR DURING time (’Mar 01 00:00:00 2011’), time (’May 31 23:60:00 2011’)
```

Example With FORCE Keyword:

```
atquery localhost:8072 -e "RETIRE FROM syslog
 UPLOADS '31AEAF94551AA4366519F40EB3F9A4FA', 'F80C8F2003D655A8444A8B4EF172EC7A'

 FORCE"
```

Using the "BEFORE <timestamp_expression>" Command

To permanently delete the records whose date and time precede a specified timestamp expression, follow the BEFORE key word with a timestamp expression. The syntax is:

```
BEFORE <timestamp_expression>
```

where *<timestamp_expression>* can use any SenSage AP SQL expression that evaluates to a timestamp data type. For example:

```
atquery localhost:8072 -e "RETIRE FROM syslog
 BEFORE time('Aug 26 11:02:34 2005')"
```

Example With FORCE Keyword:

```
atquery localhost:8072 -e "RETIRE FROM syslog
 BEFORE time('Aug 26 11:02:34 2005') FORCE"
```

For more information, see **Time Functions** in “SenSage AP ConsoleSenSage AP SQL Functions” of the *Event Data Warehouse Guide*.

Permissions Required to Retire Data

To run this command, you must have access to the namespace where the table exists and your user role must have sls.retire permission. For more information, see SenSage AP Security Model and Postgres on page 162

Important: IgniteTech recommends that you give `sls.retire` permission to only a few trusted users.

Interrupting Deletions

The retire command retires all specified data or none of the specified data. Therefore, if a retire operation is interrupted, the EDW rolls back all changes made by the retire request.

Errors and Return Values for Data-Store Utilities

Error Code or Return Value	Command	Description
0	`all`	Success. Command completed successfully or `--dbgprint-request` or `--version` or `--raw` was also specified. For "atview tasks", at least one task was found running.
2	`atmanage droptbl`	The table did not exist.
	`atview tasks`	No relevant tasks found.
	`atview [namespaces,tables,columns]`	No namespaces, tables, or columns/types found.
	`atview tabledisk`	Specified table does not exist.
3	`atmanage`	Could not fulfill command against the table.
	`atview`	Some problem retrieving the specified information (tasks, disk space information, meta-data).

Error Code or Return Value	Command	Description
10	all except `atload`	Interrupt (for example, SIGTERM, SIGINT) caught once, and then caught again before the program finishes recognizing the first interrupt.
11	`all`	The `--help` or `--longhelp` options are specified, or there are command-line parsing problems, or any other condition that leads to the program displaying its usage information.
12	`atload`	Load failure due to interrupt or failed load.
	`atquery`	Errors happen during the query, and no interrupts happen; or `--raw` is specified and the program was not able to connect to the specified port on the specified host.
	`atmanage`	The upgrade process was interrupted (twice), not giving full opportunity for this utility to clean up any ongoing upgrade it had started.
13	`atmanage`, `atview`	`--raw` is specified.
	`atquery`	Query interrupted (and cancellation ran to completion).
14	`atquery`	Problems with `--postproc`.
15	`atquery`	Interrupt happens while query is not running.
20	`all`	There are problems with the supplied command line, consistency and compatibility in the command line arguments, or problems with the specified files (for atload, PTL files; for atquery, SQL or Perl file).
21	`atmanage force-upgrade`	The upgrade was cancelled before any effects took place on the table.
22	`atmanage force-upgrade`	The upgrade was cancelled, but only after some effect took place on the table.
23	`atmanage force-upgrade`	The upgrade failed before any effects took place on the table.
24	`atmanage force-upgrade`	The upgrade failed, but only after some effects possibly took place on the table.
98	`all`	Problems connecting to the EDW server.
99	`all`	Unhandled internal errors.

Example Event-Log Data

Your SenSage AP solution includes example log events from a variety of log-source types. The examples include the source log events, corresponding PTL files for loading the events into the EDW, and corresponding `.sql` files with queries that can be run against the loaded events. The event data comes from real production systems; to protect privacy, some values have been substituted in a 1-for-1 mapping to preserve statistical relevance.

This section includes these topics:

Note: SenSage AP compression technology works more efficiently than gzip for very large volumes of data. Due to file-system overhead, benefits are not seen generally below about 10- 50MB of log data.

Tuxedo Data

The example source log file contains Tuxedo “STDERR” log events. You can find the log and related files in this location:

`/opt/hexis/hawkeye-ap/example_logs/tuxedo`

records	500,000
size	36,368,971 (uncompressed) / 2,692,070 (gzip -9 ==> 13.5:1)
start time	997080780 (2001/08/05-11:53:00 PDT 2001/08/06-06:53:00 GMT)
end time	997085724 (2001/08/06-01:15:24 PDT 2001/08/06-08:15:24 GMT)
bytes/record	72.74
fields	6
bytes/field	12.12
SenSage AP size	683,264 bytes (53:1)

Websrv Data

The example source log file contains Web site events recorded by a Microsoft IIS web server. You can find the log and related files in this location:

`/opt/hexis/hawkeye-ap/example_logs/websrv`

records	60,191
min timestamp	Wed Jan 30 07:44:37 2002 GMT (1012376677)
max timestamp	Sun Mar 17 18:08:45 2002 GMT (1016388525
columns	12
data files	`websrv_log.gz` (495,653 bytes) `websrv_log_100.gz` (1734 bytes) `websrv_log2_100.gz` (1260 bytes)
parse failures	88 records in `websrv_log.gz`

Windows Data

The example source log file contains events recorded by Windows systems and collected by the SenSage AP Windows Event Retriever. You can find the log and related files in this location:

```
/opt/hexis/hawkeye-ap/example_logs/sensage_win_evt
```

records	29,940
size	8,040,491 (uncompressed) / 218,638 (gzip-9 ==> 37:1)
start time	1161808842 (2006-11-25 20:40:42 GMT)
end time	1164652992 (2006-11-27 18:43:12 GMT)
bytes/record	268.56
fields	15
bytes/field	17.90
SenSage AP size	3,036,256 bytes (2.6:1)

SenSage AP compression technology works more efficiently than gzip for very large volumes of data. Due to file-system overhead, benefits are not seen generally below about 10-50MB of log data.

4

Administering an EDW Instance

This chapter contains the following sections:

Task Priority and Queuing

This section contains the following subsections:

Also see: Querying Data on page 74.

How the Event Data Warehouse (EDW) Queues Tasks

The EDW has three queues: one for Normal Priority processes, one for Critical Priority processes, and one for Urgent Priority processes. The first two of these queues are defined in:

`/opt/hexis/hawkeye-ap/etc/sls/instance/<instance_name>/athttpd.conf`

as:

- `normalrunqueuequota` (default `3`)
- `criticalrunqueuequota` (default `10`)

Note: These queues are set in the Deployment Manager's Advanced EDW section (under the HawEye AP service Configs tab) in the fields "EDW normal run queue quota" and "EDW critical run queue quota".

The third process, `UrgentRunQueueQuota`, cannot be configured. This process has an infinite depth, which means it can run an infinite number of jobs. The EDW immediately runs everything on this queue)

The command "`atmanage canceltask`" has urgent priority by default, all other tasks have normal priority. Therefore the `--skip-queues` option is not necessary for cancels to work on a loaded system.

The Normal and Critical queues independently allow a set number of requests to run simultaneously. (This number can be changed for these queues, see Backup and Restore on page 125).

If there are more requests of a given priority than can be run simultaneously, the later requests will be blocked until an earlier request completes. Blocked tasks are marked as `EVENT_WAIT` in the state column of the `system.task_list` table.

Monitoring Task Priority

The EDW returns progress indicators for requests waiting in the queue. Use the "atview tasks" command to see task priority. For example:

```
atview --user=administrator --pass=pass:s0mep@ss lmshost:8072 tasks
```

To view more specific information about task priority, query the system.task_list table with the `--skip-queues` flag to push your query to the top of the tasks. For example:

```
atquery --skip-queues --user=administrator \
--shared-secret=file:shared_secret.asc "sls01:8072,sls02:8072,sls03:8072"\
--namespace='system' -expression='SELECT * FROM task_list;'
```

Important: The `--skip-queues` flag works only with a shared-secret, not a password. See Authentication Options on page 80.

The `system.task_list` table contains the following information about running tasks:

Column Name	Data Type	Description
`row_type`	`varchar`	Type of information being returned: `NODE` or `TASK`
`node`	`varchar`	The host that is being returned
`port`	`int32`	The port number (installation)
`_systaskid`	`varchar`	A 32-character hex string denoting the internal task ID

Column Name	Data Type	Description
method_name	varchar	The name of the internal method being run
_extid	varchar	The "external" ID to which the client program given this task
at_top	bool	If true, the process running as the "master" for this operation
start_time	varchar	The start time of this task
finish_time	timestamp	The end time of this task (flushes every few minutes)
state	timestamp	The state of the task, such as RUNNING. The wait/blocked state for task priority queuing is displayed as EVENT_WAIT.

Cancelling Tasks

To cancel a task, use "atmanage canceltask *<request_id>* " as shown in the following example:

```
atmanage --user=administrator --pass=pass:s0mep@ss \
lmshost:8072 canceltask 673FA7200BC6F109955E227201BCEA73255
```

To find the internal task IDs, run "atview tasks".

Backup and Restore

This section describes the backup and restore process.

For information on recovering failed EDW node, see Replacing a Failed EDW node on page 139.

Backing Up and Restoring an EDW Table

This topic provides an overview of the backup and restore process for EDW tables. IgniteTech strongly recommends that you perform these processes only in consultation with Technical Support.

Overview

Backup and restore operations occur at the table level and are performed using the atmanage and atquery commands. The two sections below document the operations these commands perform on a table in the datastore.

BACKUP OVERVIEW

- User submits the following command, which sets the table to read-only (quiesce) and verifies the table's integrity; if the table is already read-only, saves the state:

  ```
  atmanage --user=administrator --pass=<password> \
  --namespace=<namespace> <cluster_list> backuptbl start <tablename>
  ```

- User submits atquery against virtual table (*<tablename>*.storage.dsminfo)
- EDW uses returned data from dsminfo to pass to file-system backup tool

- User submits the following command, which returns the table to its original read-only / read- write state before the "`atmanage backuptbl start`" command was submitted; if the table was already in a read-only state, the following command retains that read-only setting:

```
atmanage --user=administrator --pass=<password> \
--namespace=<namespace> <cluster_list> backuptbl commit <tablename>"
```

- User submits the following command, which returns the read-only / read-write state to previous setting; there is nothing to roll back because your backup tool performs the actual backup:

```
atmanage --user=administrator --pass=<password> \
--namespace=<namespace> <cluster_list> backuptbl rollback <tablename>"
```

Note: In addition to backing up your tables, you should backup your LDAP server. For more information, see “Backing Up Your LDAP Server”, on page 122.

RESTORE OVERVIEW

- User submits the following command, which returns a sandbox (directory) on each host of the EDW instance to which a dsroot tree is to be restored:

```
atmanage --user=administrator --pass=<password> \
--namespace=<namespace> <cluster_list> restoretbl start <tablename>"
```

- User submits the following command, which verifies the sandbox on each host, places the sandbox into the tree on each host, and returns the read-only / read-write state to previous setting:

```
atmanage --user=administrator --pass=<password> --namespace=<namespace> \
<cluster_list> restoretbl commit <tablename>
```

- User submits the following command, which removes the sandbox and returns the read-only / read-write state to previous setting:

```
atmanage --user=administrator --pass=<password> \
--namespace=<namespace> <cluster_list> restoretbl rollback <tablename>"
```

Note:

- Incremental backups are by date and may encompass more than you would expect due to tree reorganization.
- In addition to restoring your tables, you must restore your LDAP server. For more information, see Restoring Your LDAP Server on page 127.

Backing Up Your LDAP Server

To back up your LDAP server, run the following commands as the hexis user on the host running the LDAP Server:

```
service sensage_atslapd stop
/opt/hexis/hawkeye-ap/sbin/slapcat -f /opt/hexis/hawkeye-ap/etc/\
atslapd/slapd.conf | grep -v '^creat' | grep -v '^modif' > /opt/data/backups/\
backup.ldif
service sensage_atslapd start
```

Restoring Your LDAP Server

To restore your LDAP server, run the following commands as the hexis user on the host running the LDAP Server:

1. Stop atslapd

```
service sensage_atslapd stop
```

2. Remove the corrupted atslapd data directory. The data directory can be determined by running:

```
grep "^directory" /opt/hexis/hawkeye-ap/etc/atslapd/slapd.conf
```

3. Create a new atslapd data directory:

```
mkdir <data_dir>
chown -R <hexis_user>:<hexis_user><data_dir>
```

4. Restore:

```
/opt/hexis/hawkeye-ap/sbin/slapadd -f /opt/hexis/hawkeye-ap/etc/atslapd/\
slapd.conf -l /opt/data/backups/backup.ldif
```

5. Start:

```
service sensage_atslapd start
```

Backing Up Your PostgreSQL Server

To back up your PostgreSQL database, login to the Postgres host and run the following command:

```
su <hexis_user> -c '<bin_path>/pg_dump -C -f <controller_backup_file>
controller'
```

The *<hexis_user>* for upgrades from SenSage AP 5.0.1 is usually lms or hexis for fresh SenSage AP 6.x.x installs.

The *<bin_path>* is *<install_prefix>*`/latest/bin` for upgrades from SenSage AP 5.0.1 or

`/opt/hexis/hawkeye-ap/bin` for fresh SenSage AP 6.x.x installs. If the new Analyzer is installed, additionally run:

```
su <hexis_user> -c '<bin_path>/pg_dump -C -f <analytics_backup_file> analytics'
```

Restoring Your PostgreSQL Server

To restore your PostgreSQL server:

1. Execute the following commands on the OAE host as the Hexis user:

```
service sensage_atpgsql stop
rm -rf <OAE_data_directory>/*
/opt/hexis/hawkeye-ap/bin/initdb -E UTF8 -D <OAE_data_directory>
rm <OAE_data_directory>/pg_hba.conf
rm <OAE_data_directory>/postgresql.conf
ln -s /opt/hexis/hawkeye-ap/etc/atpgsql/pg_hba.conf <OAE_data_directory>/
pg_hba.conf
ln -s /opt/hexis/hawkeye-ap/etc/atpgsql/postgresql.conf <OAE_data_directory>/
postgresql.conf
```

2. Use the Deployment Manager to start OAE.
3. Create a Postgres user by executing the following command and entering at the password prompt, controller.

```
/opt/hexis/hawkeye-ap/bin/createuser -U hexis --pwprompt --createdb
--nocreaterole
controller
```

4. Issue the following psql commands:

```
/opt/hexis/hawkeye-ap/bin/psql -U hexis -d template1 -c "set autocommit to
on; ALTER ROLE hexis PASSWORD 'changeme';
create role sls noinherit connection limit 4 password 'sls';"

/opt/hexis/hawkeye-ap/bin/psql -U hexis -d template1 -f
<controller_backup_file>
```

Defining Data Objects

In addition to using the atquery utility to query the database, you can also use it to create, rename, and drop database tables, column filters, and views. You can either specify these actions in the `.sql` files that `atquery` processes or you can use the dash option to specify the data manipulation statement directly from the command line.

You can use `atquery` to create and manipulate the following database objects in an EDW instance:

- **Tables**—tabular objects that store loaded log events

  or more information, see “Defining Tables with SenSage AP SQL (AP SQL)” in the *Event Data Warehouse Guide*.

- **Column Filters**—special objects attached to table columns to improve query performance For more information, see “Defining Column Filters with SenSage AP SQL”\ of the *Event Data Warehouse Guide*.
- **Views**—tabular objects through which log events in tables can be viewed in different ways.

  For more information, see “Defining Views with SenSage AP SQL” of the *Event Data Warehouse Guide*.

Listing, Deleting, and Renaming Tables

In this topic:

Note: Before performing these activities, make sure you back up your data store. See Backing Up and Restoring an EDW Table on page 125.

Listing Tables

To list the tables in a given installation, use

```
atview <cluster_list> tables --namespace= --user=administrator --pass=<password>
```

The `--namespace= argument` tells `atview` to list all tables in all namespaces, including the default namespace, `default`.

Also see Examining the State of an EDW Data Store on page 109.

Deleting Tables

To delete tables, use

```
atmanage <cluster_list> droptbl <tablename>
```

Once deleted, a table's data is permanently removed.

Also see “Defining Views with SenSage AP SQL” in the *Event Data Warehouse Guide*.

Renaming Tables

To rename a table, use

```
atmanage <cluster_list> renametbl <old_tablename> <new_tablename>
```

Also see “Defining Views with SenSage AP SQL” in the *Event Data Warehouse Guide*.

`<temporary-column>` a valid column name that does not exist in the table.

Modifying Table Schema Using the ALTER TABLE Tool

The ALTER TABLE tool provides the following SQL statements that you can execute to modify the schema of an EDW database table.

- ADD: To add columns
- RENAME: To rename existing columns
- MODIFY: To modify the data type of an existing column in a table
- SPLIT: To split and create multiple columns

When you change the schema of a table using the ALTER TABLE tool, the table is changed across the cluster. The ALTER TABLE tool can be used directly or invoked on the terminal to start a console session the session you can start one or more ALTER TABLE SQL statements.

Alternatively, the ALTER TABLE can also be used as an interpreter where you can write the required ALTER TABLE SQL statements in an executable script file and run it.

Note: The time taken to alter a table schema depends on various parameters such as number of EDW nodes, volume of data in the table, network latencies etc.

Accessing the ALTER TABLE Tool

By default, the ALTER TABLE tool is located in the bin folder of the SenSage AP installation. If the SenSage AP product is installed in the default folder, following is the full path of the tool:

```
/opt/hexis/hawkeye-ap/bin/altertbl
```

In order to run the tool, you must be an EDW user. Also, you must launch the program from a host that is trusted by the rest of the hosts in the cluster.

Adding Columns

Use the following SQL statement to add multiple columns to a table:

```
ALTER TABLE ADD <schema qualified table name> [COLUMN]
(
<new column name> <EDW data type> [DEFAULT <text string in double quotes>] ,
.....
<new column name> <EDW data type> [DEFAULT <text string in double quotes>]
);
```

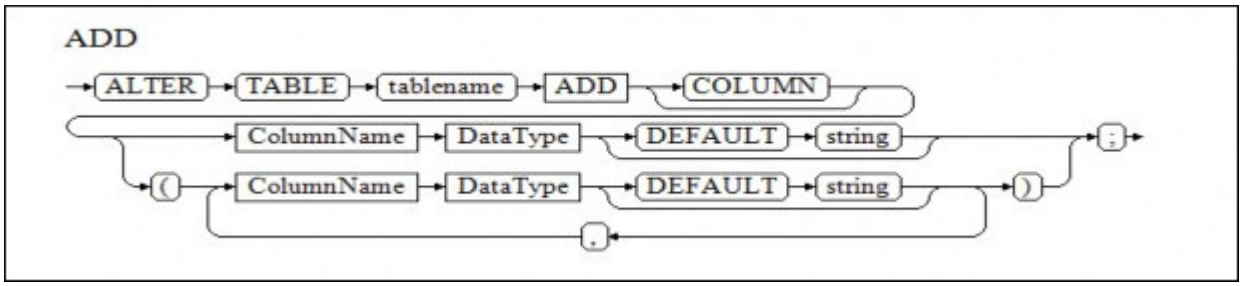
Note: If you omit the DEFAULT clause, new columns will be initialized with the following values:

- False for the bool type
- ’0’for arithmetic types
- ’0’seconds-since-epoch for the timestamp type

- 127.0.0.1 for the inet type
- A single blank space for the varchar type.

The following diagram represents the syntax diagram for the ALTER TABLE ADD statement:

Figure 34: Syntax diagram for adding columns



Examples

```
SQL> alter table default.cdr add column xyz int32 ;
SQL> alter table default.cdr add column xyz int32 default “12345” ;
SQL> alter table default.cdr add column ( xyz int32 default “1234” , pqr int64
default “123456789012345” ) ;
SQL> alter table default.cdr add pqr varchar ;
```

Renaming Existing Columns

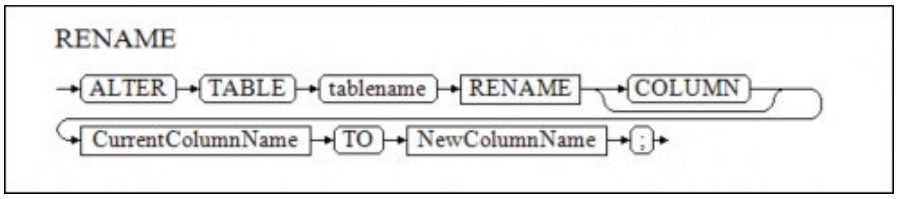
Use the following SQL statement to rename an existing column in a table:

```
ALTER TABLE <schema qualified table name> RENAME [COLUMN] <existing column
name>
TO <new name of the column>;
```

Note: You can rename only one column in a given RENAME statement. To rename multiple columns, write each rename operation in a separate SQL statement.

The following diagram represents the syntax diagram for the ALTER TABLE MODIFY statement:

Figure 35: Syntax diagram for renaming existing columns



Note: Any bloom filters associated with the column are dropped.

Examples

```
SQL> alter table default.cdr rename column exchg_id to exchange_id ;
SQL> alter table default.cdr rename call_ref to call_reference ;
```

Modifying Data Types of Existing Columns

Use the following SQL statement to modify the data type of an existing column in a table:

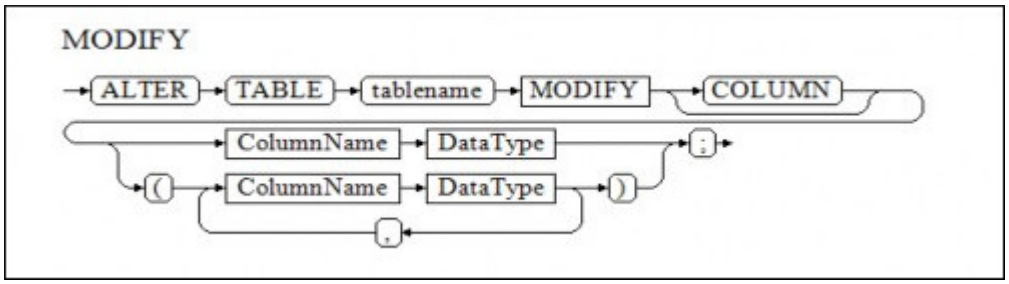
```
ALTER TABLE <schema qualified table name> MODIFY [COLUMN] <existing column
name>
<new data type>;
```

To modify data types of multiple columns in a single statement, use the following syntax:

```
ALTER TABLE <schema qualified table name> MODIFY [COLUMN] (
<existing column name> <new data type>,
.............. <existing column name> <new data type>
);
```

The following diagram represents the syntax diagram for the ALTER TABLE MODIFY statement:

Figure 36: Syntax diagram for modifying data type of existing columns



- Consider the following when using the MODIFY statement:
- The current and the targeted data types must be different.
- A column of data type VARCHAR is convertible to any type and vice versa,
- A column of arithmetic type is convertible to a higher arithmetic type or a timestamp type.
- A column of bool type is convertible to an arithmetic type.

The following table depicts the valid conversions:

Figure 37: Valid conversions of data types

	bool	int32	int64	float	timestamp	varchar	inet
bool	no	YES	YES	YES	no	YES	no
int32	no	no	YES	YES	YES	YES	no
int64	no	no	no	YES	YES	YES	no
float	no	no	no	no	YES	YES	no
timestamp	no	no	no	no	no	YES	no
varchar	YES	YES	YES	YES	YES	no	YES
inet	no	no	no	no	no	YES	no

Examples

```
SQL> alter table default.cdr modify called_number int64;
SQL> alter table default.cdr modify column calling_imei int64;
SQL> alter table default.cdr modify ( called_number int64, calling_imei int64);
```

Note: Any bloom filters associated with the column are dropped.

Splitting a Column

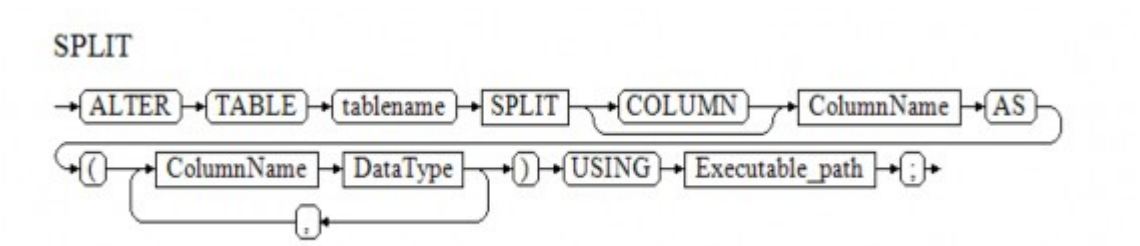
You can split the contents of an existing column to create new columns. Use the following SQL statement to split a source column in the table:

```
ALTER TABLE <schema qualified table name> SPLIT [COLUMN]
<existing column name> AS (
<output column name> <data type> ,
.......
<output column name> <data type>
USING <absolute path of the executable in double quotes>;
```

Note: The split operation does not remove the source column; only new columns are created as specified.

The following diagram represents the syntax diagram for the ALTER TABLE SPLIT statement:

Figure 38: Syntax diagram for splitting columns



Consider the following when using the SPLIT statement:

- You must provide the absolute path of the executable (script or binary) program which reads lines from the standard input (stdin) and writes a list of comma-separated values to the standard output (stdout) for each line read.
- Each output line must contain the same number of values in the comma-separated list. Also, the number of values in each comma-separated list must be same as the number of output columns mentioned in the SPLIT statement.
- The values from the comma-separated list are assigned to the output columns in the respective order.

Example

```
SQL> alter table default.cdr split column exchg_id as ( c1 varchar , c2 int32
 )
using "/tmp/split.sh";
```

and the shell script /tmp/split.sh is created as

```
#!/usr/bin/env bash
  while true
    do
    read var
    [[ ! -z $var ]] || exit 0
    echo $var | tr '-' ','
   done
```

Exiting ALTER TABLE Tool Session

To quit the ALTER TABLE session, use either of the following exit statements:

- EXIT
- QUIT

Batch Execution of ALTER TABLE Statements

You execute multiple ALTER TABLE SQL statements in an executable script file. The following example shows two statements which are run in a script file. Note the first line in this example which must precede the two ALTER TABLE statements.

```
#!/opt/hexis/hawkeye-ap/bin/altertbl
alter table tt.table1 add column column1 int64;
alter table tt.table2 add column column2 int64;
```

General Considerations for Using ALTER TABLE Tool

While executing the ALTER TABLE SQL statements, note the following:

- Before executing any ALTER TABLE SQL statements, ensure that the EDW is shut down.
- The tables must not contain buckets (which results due to trickle loads). If any table has trickle loads, you must compact the table before altering it.
- Do not run ALTER TABLE SQL statements on archived tables.
- While an ALTER TABLE SQL statement is running, let the session complete before terminating the statement.

NearLine Storage: Considerations

When adding empty columns to a table archived in NearLine Storage, consider the following:

- Empty columns you add are stored locally, regardless of where the original leaf node is archived. Because columns are empty, their footprint on disk is not large.
- Columns added using the INPUT or OUTPUT tags, the new, populated columns, are stored locally. They cannot be archived because the leaf node is already archived.
- If you delete columns, the `altertbl` command still retains those columns in NearLine storage, although the data store no longer references the columns. You won't be able to query the data, but it is stored in that location until the leaf node is retired.

Monitoring an EDW Instance

The following topics provide general guidelines for monitoring your EDW cluster:

- Monitoring CPU and Memory Usage on page 135
- Verify Hosts Are Up and Running on page 137

Note: Many of the messages reported by the EDW are informational and require no action.

Monitoring CPU and Memory Usage

CPU usage on each machine is a good indicator of the health of the cluster. Use `cltop` to monitor CPU usage particular:

- Check that the average CPU is below 50% during loads or queries. Don't worry if you see only high CPU usage—this is due to the EDW process of compressing and decompressing the log data.
- Check for high CPU usage and no running tasks. This is an indication of poor CPU usage.

Memory usage is another excellent indicator on the state of your cluster. However, the `top` command will not correctly report the amount of free RAM. Instead, use the `free` command to return a more accurate number for available memory. To see if the cluster is running out of memory, examine the progress indicators returned from a query. If a progress indicator is consistently visible, with no more than a one-second pause, this typically means the cluster has sufficient memory.

Note: You must examine each host separately to see if any one host is running low on memory and slowing the cluster down.

```
# free
total used free shared buffers cached
Mem: 2074928 1784060 290868 0 164632 1256520
-/+ buffers/cache: 362908 1712020
Swap: 2031608 148 2031460
cltop Pubs_Instance
cltop - cltop 4.1, change #42464;
Initializing (ssh) root@piltdown.hq.myco.com
Initializing (ssh) root@poweredge04.hq.myco.com
Initializing (ssh) root@proliant05.hq.myco.com
* piltdown.hq:-Done- | poweredge04.hq:-Done- | proliant05.hq:-Done-
All done.
NODE NODE ..........CPU usage......... ............... RAM usage ............
NUM ROLES idle busy (user+system) total free used shared cache
NODE PROCESS VMEM RAM SHMEM PROC CPU INSTALL
NUM ID (MB) (MB) (MB) STAT used TIME PORT START NAME ACTION OTHER
04 28154 608.0 605.0 8.3 R N 45.1% 0:28 8072 10:19 1178817571 run 1
/opt/sensage/
etc/sls/instance/Pubs_Instance/sensage_sls_Pubs_Instance r=ETL%2DEngine
e=RS%5FRUN t=BA07101495ACF887728D635C89885C69
05 21401 141.0 141.0 8.3 R N 24.9% 0:29 8072 10:21 1178817711 run 0
/opt/sensage/
etc/sls/instance/Pubs_Instance/sensage_sls_Pubs_Instance r=ETL%2DEngine
e=RS%5FRUN t=BA07101495ACF887728D635C89885C69
04 28150 217.0 216.0 7.0 R N 7.7% 0:08 8072 10:19 1178817571 run 1 /opt/sensage/
etc/sls/instance/Pubs_Instance/sensage_sls_Pubs_Instance
r=Addamark%2ELoad%2EFarm t=BA07101495ACF887728D635C89885C69
04 28143 20.2 14.0 9.2 S 2.4% 0:02 14 Dec31 14 14 0 /opt/sensage/java/bin/java
 -
cp /opt/sensage/../3.7/lib/java/cli/cli-3.7.jar com.sensage.cli.Ssiload --
namespace=blue poweredge04:8072 blue /tmp/blue_websense.ptl /tmp/
SG_WA_Sensage__100922070000.log
04 28149 14.1 13.0 6.7 S N 0.9% 0:00 8072 10:19 1178817571 run 0 /opt/sensage/
etc/sls/instance/Pubs_Instance/sensage_sls_Pubs_Instance r=StdinContent
t=BA07101495ACF887728D635C89885C69
04 28148 14.0 12.0 6.7 S N 0.4% 0:00 8072 10:19 1178817571 run 0 /opt/sensage/
etc/sls/instance/Pubs_Instance/sensage_sls_Pubs_Instance r=StdinTrans
t=BA07101495ACF887728D635C89885C69
04 28145 25.6 25.0 4.9 S 0.4% 0:00 8072 May 1178578523 init 1 /opt/sensage/etc/
sls/instance/Pubs_Instance/sensage_sls_Pubs_Instance
04 28146 16.6 16.0 8.4 S N 0.4% 0:00 8072 10:19 1178817571execute 1
/opt/sensage/
etc/sls/instance/Pubs_Instance/sensage_sls_Pubs_Instance r=Addamark%2ELoad
e=RS%5FRUN t=BA07101495ACF887728D635C89885C69
04 28144 20.2 14.0 9.2 S 0.4% 0:00 14 Dec31 14 14 0 /opt/sensage/java/bin/java
 -
cp /opt/sensage/../3.7/lib/java/cli/cli-3.7.jar com.sensage.cli.Ssiload --
namespace=blue poweredge04:8072 blue /tmp/blue_websense.ptl /tmp/
SG_WA_Sensage__100922070000.log
KEY:
- PROC STAT:
D uninterruptible sleep (usually IO)
R runnable (on run queue)
```

```
S sleeping
T traced or stopped
Z a defunct ('zombie') process
W has no resident pages
< high-priority process
N low-priority task
L has pages locked into memory
```

Verify Hosts Are Up and Running

To see if all the hosts in your EDW instance are up and running, submit an atquery command to display the `cluster_properties` table in the system namespace of your EDW:

```
atquery [machine name]:[port] -e "select * from system.cluster_properties"
        --namespace= | grep NODE_INFO
```

The `-e` option tells `atquery` to read the SQL statement that follows it.

In the results, look for the `NODE_INFO` lines. Each host in your EDW instance should have an entry. For example, if you have a 10 hosts in your EDW instance, you should expect 10 records back from this query.

Monitoring Disk Usage

You can monitor disk usage in the EDW by running one of the EDW System reports, available as part of the Foundation Analytics report package. For more information, see following reports in the *Analytics Guide*.

- EDW System Disk Space Usage Alerts in “Foundation Analytics Report Package”
- EDW System Disk Usage Per Node in “Foundation Analytics Report Package”
- EDW Percent Disk Space Usage in “Foundation Analytics Report Package” For details on running these reports, see the SenSage AP Analyzer documentation.

For details on running these reports, see the *SenSage AP Analyzer documentation*.

International Support in the EDW

The EDW supports the storage of *international characters*, which are characters other than those in the U.S. English alphabet and some of its punctuation marks. For example, the Spanish word

“España” contains the international character “ñ”. The EDW stores all text values with UTF-8 character encoding to ensure that international characters are stored and queried successfully.

For more details on character sets, encoding schemes, and using computer fonts with encoding schemes, see. “SenSageUsing the SenSage Event Data Warehouse (EDW)” in the *Event Data Warehouse Guide*.

System Tables

You can query the EDW to get information about your SenSage AP system such as the state of an EDW instance or all defined roles and which users have been assigned to them. When you query for this information, the EDW dynamically formats the result data as a standard table.

These tables return system information and are known as *system tables*. Because the data returned is not saved on disk but is dynamically retrieved and formatted as a standard table, they are also referred to as *virtual tables*. For a detailed description of each system table, refer to *Appendix D: SenSage APSystem Tables* of the *Event Data Warehouse Guide*.

The virtual nature of these tables is apparent when you query the EDW for all tables in your instance by running the atview command; for example:

```
atview --user=administrator --pass=<passwd> --namespace='' <cluster_list> tables
```

The above command does not return any system tables.

Because the system tables are virtual, you cannot insert data directly into them. For example, you cannot run `atload` to load a new user into `system.users`.

To explore the state of your system, run the `atview` command; for more information, see Examining the State of an EDW Data Store on page 109 and Monitoring an EDW Instance on page 135.

To explore authorization and authentication data, run the `atquery` command against specific system tables.

5

Replacing a Failed EDW node

This section describes the process for handling a failed EDW node and includes the following topics:

Overview

If an EDW node in your SenSage AP instance fails, you can restore functionality using the procedures in this section. When one of the EDW nodes fails, you can configure your SenSage AP instance to operate in a degraded mode, without the failed EDW node this mode of operation, you can continue to load data into the EDW and run queries against the data, but you cannot perform meta-data operations such as adding, or dropping tables or views.

After configuring this limited mode, you can then restore the failed EDW node to restore the SenSage AP instance to full functionality. Alternately, if an EDW node fails, you can stop operations on the entire SenSage AP instance and then restore or replace the failed EDW node. Because each EDW node has a duplicate copy of its data store located on another node, no data is lost.

Note:

A small amount of data may have been loaded to the destination table. If it has, all data uniquely identified by the load's `upload_id` is marked as unsuccessful. Use `atretire` to remove this data; for more information, see Retiring Data on page 115.

Replacing a Failed EDW Node

To configure a SenSage AP instance to operate with a failed EDW node:

1. Stop the EDW component on all nodes.
2. Get the state of the suspected failed host by running the following command:

```
curl -k --user admin:<ambari_admin_password>
https://<deployment_manager_ip>:8443/api/v1/clusters/<cluster_name>/hosts/
<bad_host>?fields=Hosts/host_state
```

Note: If state = HEARTBEAT_LOST, skip to Step 4.

3. Stop all components on the failed host.

```
# GANGLIA_MONITOR
# GANGLIA_SERVER
# NAGIOS_SERVER
# SENSAGE_ANALYTICS
# SENSAGE_APPSERVER
# SENSAGE_COLLECTOR
# SENSAGE_POSTGRES
# SENSAGE_SLS

curl -k --user admin:<ambari_admin_password> -X 'PUT' -d
'{"HostRoles":{"state":"INSTALLED"}}'
https://<deployment_manager_host>:8443/api/v1/clusters/<cluster_name>/hosts/
<bad_host>/host_components/<component>
```

4. Delete all components on the failed host.

```
# GANGLIA_MONITOR
# GANGLIA_SERVER
# NAGIOS_SERVER
# SENSAGE_ANALYTICS
# SENSAGE_APPSERVER
# SENSAGE_COLLECTOR
# SENSAGE_POSTGRES
# SENSAGE_SLS

curl -k --user admin:<ambari_admin_password> -X 'DELETE'
https://<deployment_manager_host>:8443/api/v1/clusters/<cluster_name>/hosts/
<bad_host>/host_components/<component>
```

5. If the bad host still shows up in the hosts tab of the Deployment Manager, run the following command to remove it:

```
curl -k --user admin:<ambari_admin_password> -X 'DELETE' https://
<deployment_manager_host>:8443/api/v1/clusters/<cluster_name>/hosts/<bad_host>
```

6. Update the existing configuration and replace all occurrences of the bad host's name:
   a. Get the list of configurations on the server:

```
# Get a list of configurations
curl -k -H -i -u admin:<ambari_admin_password> https://
```

```
<deployment_manager_host>:8443/api/v1/clusters/<cluster_name>/
configurations?type=global
```

b. Use the highest tag number from the output above to retrieve the configuration values from just before adding the new host:

```
# Get a list of configurations
curl -k -H -i -u admin:<ambari_admin_password> https://
<deployment_manager_host>:8443/api/v1/clusters/<cluster_name>/
configuations?type=global\&tag=<second_highest_tag>
```

Note: You only need to copy the configurations for these two parameters: `sensage_sls_head_node` and `sensage_sls_instances_data`.

---

c. Create an XML file named `reconfig_edw.xml` and add the two parameters pulled from above to it. Replace all occurrences of the old hostname with the new hostname. For example:

```
<?xml version="1.0"?>
<ambari>
<sensage_sls_instances_data>{\"instance_8072:8072\":{\"cmmdev02.se
nsage.com\":\"cmmdev02.sensage.com\",\"cmmdev04.sensage.com\":\"cmmdev
04.sensage.com\"}}</sensage_sls_instances_data>
<sensage_sls_head_node>cmmdev02.sensage.com</sensage_sls_head_node>
</ambari>
```

d. Edit `/opt/hexis/hawkeye-ap/hawkeye-deploy/componentInstall.py` and comment out lines 136 and 137 as shown below. If your lines don't look like this, you have a different version of hawkeye-deploy and it isn't necessary to comment out these lines.

```
# if property == "sensage_sls_instances_data":
# self.configDict[property] = properties[property].replace('"','\\"')
```

e. Update the configurations on the server with the new values:

```
hawkeye-deploy -u admin -p <ambari_admin_password> --host
<deployment_manager_host> --config=reconfig_edw.xml --targetService
hawkeyeAP
--targetComponent=edw --action reconfigure
```

7. Stop EDW on all hosts:

```
hawkeye-deploy -u admin -p <ambari_admin_password> --host
<deployment_manager_host> --targetService hawkeyeAP --targetComponent=edw
--action=stop
```

8. Add a new host. and select to install EDW on it. Ignore warnings about data replication and expansion as this is not necessary since the data will be overwritten.

Note: This action may fail with an error while installing Analytics. Just leave it as-is and proceed to step 9.

If EDW is configured for no data mirroring, skip to step 12.

---

9. For each EDW data instance, determine where to copy the data stores:
   a. Look at the the sensage_sls_instances_data element in the `reconfig_edw.xml` file created in step 6c, the hosts are grouped into host sibling pairs.
   b. Find the new host (\"<new_host>\":\"<sibling_host>\") and note its sibling host, which will be <sibling_host> in the `rsync` command noted in step 11.
   c. Find the new host (\"<reverse_sibling_host>\":\"<new_host>\" and note its reverse sibling host, which will be <reverse_sibling_host> from which to copy the primary datastore to the new host's secondary datastore.
   d. On <sibling_host> note the number appended to the Secondary datastore name in the EDW instance data directory:Secondary-#.d.
   e. On <reverse_sibling_host> note the number appended to the Primary datastore name in the EDW instance data directory: Primary-#.d.

10. On the new host, ensure EDW is stopped:

```
service sensage_edw stop
```

11. With the information you have gathered in Step 8, issue the following commands on the new host for each EDW instance:

```
rsync -e ssh -rpogv <sibling_host>:<data_dir>/sls/<EDW_instance>/dsroot/
Secondary-<secondary_number>.d/* <data_dir>/sls/<EDW_instance>/dsroot/Primary-
<secondary_number>.d

rsync -e ssh -rpogv <reverse_sibling_host>:<data_dir>/sls/<EDW_instance>/
dsroot/Primary-<primary_number>.d/* <data_dir>/sls/<EDW_instance>/dsroot/
Secondary-<secondary_number>.d
```

12. For each EDW instance, update the top level `NODE.dat` by issuing the following commands: a
   a. `cd <data_dir>/sls/<EDW_instance>/dsroot`
   b. `zcat NODE.dat > new.dat`
   c. `vi new.dat`
   d. Set the *<PhysicalPath>* element appropriately for the Primary and Secondary data stores based on the numbers noted in Steps 8d and 8e.
   e. `gzip -n new.dat`
   f. `mv new.dat.gz NODE.dat`
   g. `chown <HawkEye-AP_user>:<HawkEye-AP_group> NODE.dat`
   h. `chmod 600 NODE.dat`

13. If you have configured Nearline storage, copy the following NSS ID list file from any other EDW node to the node you are restoring.

Note: The path to the NSS ID list file was configured during installation in the 'NSS ID list file' field in the Advanced EDW section. The default is `/opt/data/nss/nss_idlistfile.dat`.

---

14. Restart ambari-server on the Deployment Manager

15. Start the EDW component.
16. Restart Nagios Server.

Recovering from multiple failed EDW Nodes

The procedure for recovering from multiple failed EDW nodes is exactly the same as replacing a single node; the only exception is that you can expect data loss when you are recovering multiple failed nodes that constitute the primary and secondary data stores of EDW.

6

Administering the Collector

This chapter contains these sections:

Preparing Log Data for Loading

The Collector does not rotate logs. Therefore, you need to provide your own means for rotating logs on your log hosts in a way that is compatible with the Collector. Specifically:

- provide a log file that is not actively being written to
- ensure that the logs maintain unique names
- include an archive sub-directory for SFTP, FTP, and HTTP

The Collector package includes a script called `log-rotate.exe` (for Windows) or `log-rotate` (for Linux) that provides an example of how you can write a script to achieve this task. Run the script with no options to see its usage.

To use the script, you must specify:

- a sourceFile
- a targetDir
- whether you want hourly or daily naming

The script creates an `archive/` directory (under the targetDir), then compresses and renames the files.

The script also has an option that enables you to specify a hierarchy of directories for archived sub-directories. Because the Collector typically does not use this option, you should turn it off.

Starting, Stopping, and Restarting the Collector

You can start, stop, and restart SenSage AP components, including the Collector, from the Deployment Manager. For details, see Starting, Stopping, and Restarting SenSage AP Components on page 46.

Monitoring Logs

This section describes these topic:

About Collector logs

You can configure the Collector to log to three log files:

- **transaction.log**—Only the start and end of loads and the retrieval of loads are recorded here. This is a pipe-delineated file used by PTLs.
- **activity.log**—All Collector activity logs to this file.
- **error.log**—Error messages generated from `activity.log` appear in this file. The mapping of error codes to messages is stored in the following file:

  `/opt/hexis/hawkeye-ap/share/locale/<locale>/collector/errors.xml`

  The above file is loaded at startup and used whenever an error code is specified. You can also specify how Collector alerts users of errors in this file (using email or pagers, for example).

To specify whether the Collector logs to files, and to specify the logging levels and file location, edit the logging section of the `config.xml` file for editing. This file is located in:

`/opt/hexis/hawkeye-ap/etc/collector/`

The logging section is:

```
<!-- Log level 3 is most verbose -->
<Logging level="1" type="syslog"/>
<!-- Uncomment below for file logging -->
<!--
<Logging level="1" type="file" file="/opt/hawkeye ap/var/log/collector/
error"/>
<Logging level="2" type="file" file="/opt/hawkeye ap/var/log/collector/
activity"/>
<Transaction>/opt/hawkeye ap/var/log/collector/transaction</Transaction>
-->
```

Logging to syslog

To log to syslog, the only change you should make is to the log level. Then restart the Collector.

To set File-Based Logging

To use file-based logging, comment out the syslog line and uncomment the remaining lines. If you are not using the default SenSage AP home directory, you must also modify the path.

Setting Both syslog and File-Base Logging

Make both changes described above other words, set the logging level for syslog and uncomment the lines for file-based logging. Modify the path if necessary and restart the Collector. For more information, see Starting, Stopping, and Restarting the Collector on page 146.

Log Formats for Transaction Logs

The following fields appear in `transaction.log`:

```
timestamp
type
name
fileSource
startTimestamp
endTimestamp
bytesCopied
errorCode
errorMsg
minTimestamp
maxTimestamp
uploadid
recordsLoaded
matchFailures
atloadCommand
PTLPath
atloadVersion
atloadExitStatus
```

Log Formats for Activity and Error logs

The following fields appear in `activity.log`, `error.log`, and associated alerts:

```
Timestamp (in ISO time format: YYYY-MM-DDTHH:mm:ss)
Process ID
Log level of message (1 for major activity, 2 for notices, 3 for verbose)
Error code, if any (defaults to 0 for no error)
Class Name
Line number
Message
```

Setting the Logging Level

Set the logging level in `config.xml` with the `level` attribute in the `<Logging>` element. You can specify multiple `<Logging>` tags, which allows you to specify different logging parameters for each level. For example, the following configuration specifies that all logs (level 3) are logged to a file while *errors* (level 2) are logged to syslog:

```
<Logging level=”2”> type=”syslog”/>
<Logging level="3"> type=”file” file=”tmp/activity”/>
```

Specify one of these logging levels.

Level	Meaning
1	Least verbose. Only critical errors or severe warnings are logged.
2	Moderately verbose. Event history is logged, as well as critical errors and sever warnings. Set logging to level 2 if you intend to tail log files. For more information, see Tailing Logs Files on page 149.
3	Most verbose. IgniteTech recommends that you set logging to level 3 during installation, initial configuration, and troubleshooting.

Tailing Logs Files

You can view the log files using any text editor, or you can view the most recent entries using the `tail` command.

To view data logged to a file

```
tail -f /opt/hexis/hawkeye-ap/var/log/collector/<logfile>
```

where <*logfile*> is `activity`, `error`, or `transaction`.

To view data logged to syslog

```
tail -f /var/log/messages
```

Things to watch out for

As you monitor logs, watch out for the following:

- Failed Retrievers on page 149
- Failed Loaders on page 149
- Failed Loads on page 150
- Parse Failures on page 150

Failed Retrievers

A Retriever may go down if any of its processes fail (for example, `BackupDir` for backing up raw log data) most cases, you need to restart the Collector after fixing the problem. When a retriever is configured to use a preprocessor and the preprocessor fails to generate an output file, the Collector will rename the files to `.noload` in the spool directory.

To determine if this is the problem, run the preprocessor (with and files) outside of the Collector. If you do not see your postprocessed data in the file, fix the preprocessor script (rename the `.noload` files to `.raw`). Then fix the preprocessor in the Collector (rename the `.noload` files to `.raw` in the spool directory).

Failed Loaders

Restart the Collector if a loader goes down (time out if `atload` fails). Some other issues you may see include:

- Zero-byte log files.
- Incorrect permissions on directories and files—generates critical error.
- Low or no free space on disk—causes Log Queues to become disabled (restart to fix).
- Unmounted file systems—do not use NFS; it will throw error on Retrievers.
- Files being unlinked during processing—might not throw error if no metafile gets created. Clean out the queues and start over. Delete plugin files. Remove MD5 lines for those files (still could get duplicate loads).

Failed Loads

On failed loads, you need to get the unloaded files back into the system. Loads can fail in two locations:

- In the main queue directory; here, rename log files with the `.noload` extension to `.log` and restart the Collector.
- In the spool directory (means that preprocessing failed); rename the `.noload` files to `.raw`, then rerun the preprocessor.

To trace back to the failure, set the logging level to 3 and investigate `activity.log`.

For more information, see Setting the Logging Level on page 148 and Handling Unsuccessful Loads on page 150.

Parse Failures

Parse failures mean that there is a problem in the PTL file. These failures may also occur if the log format changes, or if log files become corrupt.

Also see the ParseFailure setting described in *Defining Loaders* in “SenSage AP Collector Configuration” of the *Collector Guide*. You can have multiple Loaders mapped to the same parse failure location.

Handling Unsuccessful Loads

This section describes these topics:

About Unsuccessful Loads

Log data does not always load correctly into the EDW instance. Incorrectly loaded data can include loads that partially completed or loads that were aborted and reloaded. The EDW tracks every upload to the system, storing the unique identifier of each load in the `_uploadid` column of each table. Loads that fail upload are tagged as inconsistent. By default, the data in an inconsistent load does not display when you query the EDW.

Note: A successful load does not necessarily insert every row from the source log file. Typically log files contain data that cannot be parsed and loaded; however, bad data does not prevent these files from loading successfully. A failed load is caused by a protocol error, such as not receiving all the expected data or a load cancellation.

Viewing Data from Unsuccessful Loads

Each row in EDW tables tracks the ID of the upload operation inserted the row. Rows inserted by an unsuccessful load are excluded from queries by default.

If you want to view all data in a table, including data from unsuccessful loads or data currently being loaded, you must specify the `{INCLUDE_BAD_UPLOADS}` modifier on the table in the `FROM` clause. Include the curly braces as part of the modifier.

Note: If a query and a load occur simultaneously and the query continues after the load completes, you may see all, none, or part of the data from the new load.

Tracking Uploads in the EDW

In addition to loading log data into the specified tables, `atload` saves information in the EDW

about the load. You can view this load information by querying these system tables:

- `system.upload_info`

  This table enables administrators to check on upload status in the system. When the system is healthy, this table returns one row for each upload. When the upload data is not consistent across all the nodes in the cluster, this table returns a value of `false` in the CONSISTENT column. In this case, the table may return multiple rows with the same value in the UPLOADID column; all of these rows are marked as inconsistent.

- `system.raw_upload_info`

  This table enables a support person to see the raw data that is distributed across the system in order to troubleshoot an inconsistency problem revealed in the `system.upload_info` data.

Both of these system tables contain a unique identifier for the upload: the value of the `_uploadid` column. These tables store additional information for each load operation, which includes: the minimum and maximum timestamps, the number of lines in the source log data, the number of lines successfully parsed by the PTL file, the number of rows loaded into the specified EDW table, the PTL file and client signatures, and whether the load was successful.

Additionally, the `system.upload_info` table returns the consistency data and the `system.raw_upload_info` returns the logical name of the source node for the log data.

As it begins loading data into the log tables, the EDW inserts a new row in these system tables to track the upload. Initially, the value of the SUCCESSFUL column defaults to `false`. When the load completes successfully, this column's value changes to true.

IgniteTech recommends that you regularly query the `system.upload_info` table for loads with inconsistent data (the value of the CONSISTENT column is `false`). Inconsistent data was not correctly replicated across the cluster. Whenever you discover an inconsistent load, query the `system.raw_upload_info` table and capture the results in a file, contact IgniteTech Support: send email to info@ignitetech.com or call +1 800-248-0027.

Viewing Transaction Recovery Data

You can selectively enable detailed logging on the transaction recovery logic in the EDW. To do this, set the configuration parameter `recovery_audit` in the `thttpd.conf` file to **true** to enable detailed logging of the transaction recovery steps.

The existing detailed logging, which is enabled by setting the trace level to 1 (INFO) or higher is not affected by the `recovery_audit` setting.

Running Multiple Collectors

If a situation requires that you set up multiple Collectors (on different hosts), they should be configured to each manage a subset of log queues, and the subsets should not overlap. You can achieve this setup by using a daisy-chain configuration. For more information, see *Creating Daisy Chains* in "SenSage AP Collector Configuration" of the *Collector Guide*.

Scheduling Retrievers and Loaders

When configuring Retrievers and Loaders, you define when retrieving and loading occurs using one of two methods:

- **The PollInterval and Period Elements**—Use the `<PollInterval>...</PollInterval>` element to specify how often loading and retrieving occurs. For example, you can specify that loads happen every hour. This is the most common method of specifying when loading and retrieving occur. You can combine the `<PollInterval>` element with the optional `<Period>...</Period>` element to specify when loading or retrieval begins.

  Use this method when configuring Merge loaders and when configuring Retrievers that log into other systems to see if there are files to be retrieved. These Retrievers include:

  - FTP Retrievers
  - SFTP Retrievers
  - HTTP Get Retrievers
  - LEA Retrievers
  - Script Retrievers.

- **The Schedule Element**—Use the `<Schedule> ... </Schedule>` element to specify a *specific time* for Retrievers and Loaders to run. For example, you can specify that loading occurs at 8:00 and 14:00 every day.

Important: Do not combine the above methods. Use only the `<PollInterval>` with the optional `<Period>` element or the `<Schedule>` element.

For additional information, see *Defining Retrievers* and *Defining Retrievers* in SenSage AP Collector Configuration of the *Collector Guide*.

The Schedule Element

The `<Schedule>` element has the following syntax:

`<Schedule>`*`minute hour day_of_month month day_of_week`*`</Schedule>`

Use the `<Schedule>` element to define an event that should occur at a specific time. At the end of every event, the Collector tests the event conditions again to determine whether the event should run again. If a given event runs over the time when the next event of the same type would be scheduled, the second (and subsequent) missed events are not queued, they are ignored.

FIELDS IN THE VALUES OF SCHEDULE ELEMENTS

The value of a `<Schedule>` element has these positional fields.

Positional Field	Allowed Numeric Values
minute	0-59
hour	0-23
day_of_month	1-31
month	1-12
day_of_week	0-7 (0 and 7 represent Sunday)

Separate the fields with spaces. You must specify a value for each field. If you are unsure what specify for a positional field, specify an asterisk (*) but *do not specify an asterisk for all five fields*. If you are unsure of what to specify for all fields, do not specify the `<Schedule>` element and SenSage AP will use the default behavior of once every minute.

USAGE NOTES

- A field can contain an asterisk (*), which functions identically as listing every possible value for that field. For example, an asterisk in the `day_of_month` field specifies events occur every day.
- Ranges of numbers are allowed. Ranges are two numbers separated by a hyphen. The specified range is inclusive. For example, `8-11` for in the `hour` field specifies execution at hours 8, 9, 10 and 11.
- Lists are allowed. A list is a set of numbers or number ranges separated by commas. For example, `1,2,5,9,0-4,8-12`.
- Names for the `month` and `day_of_week` fields are allowed. Use the first three letters of the particular day or month. Case does not matter. Number ranges and lists of names are not allowed.
- The day of a command's execution can be specified by two fields: `day_of_month` and `day_of_week`. If both fields have a schedule specification, the command runs when the current time matches the specification in either field matches. For example, the following schedule specification causes a command to be run at 4:30 AM on the 1st and 15th of each month, plus every Friday.

  ```
  <Schedule>30 4 1,15 * 5</Schedule>
  ```

- Do not specify an asterisk for all five fields. If you are unsure of what to specify for all fields, do not specify the `<Schedule>` element and SenSage AP will use the default behavior of once every minute.

EXAMPLES SCHEDULE SPECIFICATIONS

To run once an hour, on the first minute of the hour. (i.e. 12:00):

```
<Schedule>0 * * * *</Schedule>
```

To run every five minutes of every hour of every day:

```
<Schedule>0,5,10,15,20,25,30,35,40,45,50,55 * * * *</Schedule>
```

To run every second hour, starting at midnight:

```
<Schedule>0 0,2,4,6,8,10,12,14,16,18,20,22 * * *</Schedule>
```

To run once an hour, Monday through Friday.

```
<Schedule>0 * * * 1-5</Schedule>
```

The PollInterval and Period Elements

The <PollInterval> and <Period> elements have the following syntax:

```
<PollInterval>Interval</PollInterval>

<Period>Minute Hour DayOfMonth Month DayOfWeek</Period>
```

The <PollInterval> Interval defines how often, in seconds, the event in question will be attempted during the defined <Period>. Poll intervals of less than 60 seconds are not recommended. If the event runs longer than the poll interval or runs into the next period, the event is allowed to complete. (You can also specify <PollInterval> by adding hour, hours, minute, minutes, second, or seconds. For example, the following are all equivalent: <PollInterval>1 hour</PollInterval>, <PollInterval>3600</PollInterval>, <PollInterval>3600 Seconds</PollInterval>.)

The syntax for values in the <Period> element is the same as for the <Schedule> element. For a complete explanation of the syntax, see The Schedule Element on page 152.

EXAMPLES POLL INTERVAL AND PERIOD SPECIFICATION

To specify that an event runs every 5 minutes (300 seconds) from 1:00 AM through 5:59 AM:

```
<PollInterval>300</PollInterval>
<Period>* 1-5 * * *</Period>
```

The specification above is equivalent to the following <Schedule> specification:

```
<Schedule>0,5,10,15,20,25,30,35,40,45,50,55 1-5 * * *</Schedule>
```

To specify that an event runs every hour continuously, starting at the half-hour:

```
<PollInterval>3600</PollInterval>
<Period>30 * * * *</Period>
```

The specification above is equivalent to the following <Schedule> specification:

```
<Schedule>30 * * * * *</Schedule>
```

Scheduling and the Unchanged For Attribute

Regardless of which element you use to set the polling frequency, you can also set how long source log files should remain unchanged before the retriever transfers them for processing. In other words, you separately specify how frequently the Collector polls for events and how long the Retriever keeps an event log file before it releases it for processing.

To set the length of time the Retriever keeps an event log file before processing, you set the <UnchangedFor> attribute of the <Plugin> element. For more information about this element and attribute, see Plugin in “SenSage AP Collector Configuration” of the *Collector Guide*.

The Collector determines when to take a file for processing based upon:

- Polling cycle

  The frequency set either for the <Schedule> element or the <PollInterval> and <Period> elements

  or

- If no values have been set for <Schedule> or <PollInterval> and <Period>, the frequency specified for <CycleDelay> (See CycleDelay in “SenSage AP Collector Configuration” of the *Collector Guide*)

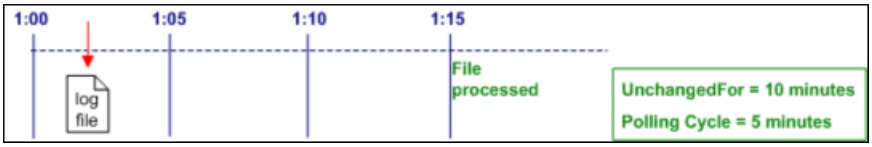
  and

- The value specified for the <UnchangedFor> attribute of the <Plugin> element.

  Each time it polls, the Collector uses the <UnchangedFor> setting to determine whether a file has remained unchanged long enough to be ready for processing.

Figure below illustrates how the Collector processes a file when the <UnchangedFor> attribute has been set to 10 minutes and the polling cycle has been set to 5 minutes.

Figure 39: Illustrating the Relationship of Schedule Frequency & UnchangedFor Attribute



As illustrated above, a log file arrives at 1:02, which is 2 minutes after the last Collector poll. At 1:05 and again at 1:10, the Collector checks whether the log file has changed. Because the file has not changed and 10 minutes has not elapsed since the file was written, the Collector ignores the file at these times. However, at the 1:15 poll, because 10 minutes has elapsed since the file was written, the Collector takes the file for processing.

Note: The value of <CycleDelay> in config.xml is irrelevant if you set either the <Schedule> element or the <PollInterval> and <Period> elements.

Backing Up and Restoring System and Log Data

Files needed to back up for restoring the system are:

- /opt/hexis/hawkeye-ap/etc/collector/config.xml
- All MD5 data in the state directory, as specified by the <StateRoot> element in config.xml. For more information, see *Root Directories Used by the SenSage AP Collector* in “SenSage AP Collector Configuration” of the *Collector Guide*.

BACKING UP RAW LOG DATA

You can configure retrievers to make backups of raw, downloaded logs before beginning preprocessing. Add a <BackupDir> element to the retriever configuration in <SenSage_Home>/etc/collector/config.xml, as the next example shows:

```
<Retriever name="filesystem" type="filesystem" enabled="1" process="files1"
method="hardlink" deleteOriginal="1" >
   ... other elements ...
```

```
    <BackupDir>file:///nfs1/backups</BackupDir>

    <!-- Copies file with scp to example.com using fred's account
    Note: this requires public/private keypairs. -->
    <BackupDir>scp://fred@example.com/nfs2/backups</BackupDir>

    <!-- Copies file with scp using -C for stream compression. -->
    <BackupDir>scpc://fred@example.com/nfs2/backups</BackupDir>
</Retriever>
```

Log files are backed up after completing an atomic file write; that is, file system copy or ftp copy. You can also use scp, scpc, or an arbitrary user-specified command to do log backups.

Important: All commands are executed in parallel. If any command fails, the entire backup fails, and the retriever shuts down to prevent buildup of files that have not been backed up.

Description of Log File Names

Source log files are renamed to have a zero-padded, 3-digit priority number at the beginning of the filename. Non-prioritized files are marked with `000`. To prioritize a log, change the number to 1 higher than the highest number in the log queue. Loaders will look for the file with the highest number when considering which file to load.

Log files also have an ISO timestamp added to their name showing the time at which they were gathered, a sequence ID in the event that two logs are gathered during the same timestamp, and a string identifying the retriever that retrieved them.

The configuration file `config.xml` defines the log-queue directories where log files are stored. For more information, see *Configuring Log Queues* in “SenSage AP Collector Configuration” of the *Collector Guide*.

Syntax for Log File Names

The names of event-log files that are processed by the Collector have this syntax:

```
PPP-YYYYMMDDtHHMMSS-SEQ-RETR.EXT
```

Filename Field	Meaning
PPP	Three-digit priority
YYYMMDDtHHMMSS	Date and time
SEQ	Numeric sequence number, normally 0 unless more than one file is generated per second
RETR	Retriever class name that generated this file
EXT	File extension

Extensions for Log File Names

The names of event-log files that are processed by the Collector these extensions, which indicate the processing states of log files.

Filename Extension	Meaning
`.out`	An output file in the process of transfer or preprocessing; All `.out` files with modification dates over one day old are deleted automatically.
`.in`	An intermediate file in an ongoing preprocess stage; All `.in` files with modification dates over one day old are deleted automatically.
`.pmd`	A file with no MD5 yet computed. At same time, a metadata (`.meta`) file is created.
`.meta`	An XML file containing meta data regarding a log file with the same name.
`.org`	A file needing backup before it can be renamed to `.raw`. Occurs if MD5 doesn't match.
`.noload`	A file whose preprocessor or loader has crashed. Rename to `.raw` to attempt preprocessing and loading again.
`.raw`	A file needing preprocessing. A file may only be preprocessed by the retriever name which created it.
`.log`	A file ready for a loader to process.
`.wrk`	For daisy-chain configurations, a file created in the process of converting to a `.tar` file.

Example Log File Name

```
000-20040101t010101-2-ftp1.log
```

Enabling/Disabling Log Adapters

To enable or disable log adapters, you can use the CollectorAdmin tool that has the following syntax:

```
/opt/hexis/hawkeye-ap/bin/collectorAdmin enable | disable <adapter_name>
```

For example, the following command:

```
/opt/hexis/hawkeye-ap/bin/collectorAdmin enable unix_sshd2_syslogng
```

Reprioritizing Log Files

You can prioritize log files within `config.xml`. Assign the `id` attribute to an integer, where “1” processes first, “2” next, and so on. The id attribute appears in the `<PTL>`, `<Preprocessor>`, and `<Plugin>` elements.

Prioritizing Log Files Manually

You can prioritize the order of log files in log queues manually. Edit the numerical prefix in a filename to place the log file at the appropriate spot in the log queue. Loaders search for the log file with the highest priority when selecting the next file to load.

Note: IgniteTech recommends that you prioritize log files with the `id` attribute in the `config.xml` file. Only use the manual procedure described below for troubleshooting or similar purposes.

To prioritize a log file by changing the priority field in the filename:

1. Create a hard link to the metadata file (found in `<LogQueue>/spool`).

```
# ls 000-20040315T101023-000-cpixrt.ext 000-20040315T101023-000-cpixrt.meta
# ln 000-20040315T101023-000-cpixrt.meta 001-20040315T101023-000-cpixrt.meta
```

2. Rename the log file, incrementing the priority prefix as you rename it. For example:

```
# mv 000-20040315T101023-000-cpixrt.log 001-20040315T101023-000-cpixrt.log
```

3. Remove the original metadata file.

```
# rm 000-20040315T101023-000-cpixrt.meta
```

About metadata files

Each gathered log file is paired with an XML-based metadata file of the same name. The metadata file contains information about the source of the log data and its current status. The metadata file is created when the log file is fully transferred into the Collector (that is, when the `.pmd` file is created). The extension of a log file changes as it moves through the system, but its corresponding metadata file keep its `.meta` extension throughout.

For more information about log file names and extensions, see Description of Log File Names on page 156.

Cleaning Up Processed Log Files

Once a log file has been processed, it is put into the `<LogQueue>/done` subdirectory. The system administrator must archive or remove these files from the system. SenSage AP software includes a cleaner script with the Collector as an example of the type of script you should write to schedule the automatic deletion of processed log files. Run the script or program from the command line, and specify the path to your LogRoot as an option.

On Linux, run this script:

```
/opt/hexis/hawkeye-ap/bin/collector-extras/cleaner.pl
```

To learn more about LogRoot, see *Root Directories Used by the SenSage AP Collector* in “SenSage AP Collector Configuration” of the *Collector Guide*.

7

Managing Security (Users and Authentication)

This chapter contains the following sections:

Important: Commands and references in this chapter are for example purposes only. Copying, pasting, and running commands contained in this chapter may not work depending on your installation setup.

Security Management Overview

SenSage AP manages security through authentication and authorization of security processes, providing two main verification checks, one that authenticates both users and groups and the other that determines the functions they are authorized to perform. SenSage AP Account Administrators manage these security processes through the SenSage AP Analyzer, which provides a dashboard of account and security management tasks.

Each area of Security Management is briefly described below.

User Authentication

The user authentication process identifies and validates users through their user name and password.

- SenSage AP supports the following user authentication features:
- Local authentication and single sign-on.
- Disable and enable users.
- Track failed log-on attempts and disable users when attempts exceed limits.
- Handle forgotten passwords, as well as support password strength requirements.
- LDAP authentication with the ability to synchronize users from LDAP.
- AD/LDAP authentication with the ability to synchronize users from AD/LDAP.

User Authentication Modes

The SenSage AP Analyzer has three modes of authentication:

- *Native* authentication in which users are created and managed solely in the Analyzer database. Passwords are stored and encrypted in the database.
- *LDAP* authentication in which usernames and passwords are stored in LDAP and users are authenticated against LDAP at login time.
- *LDAP/AD* authentication in which usernames and passwords are stored in AD/LDAP and SenSage AP Analyzer delegates user authentication to AD at login time.

The Analyzer's authentication mode is configured during SenSage AP installation through user- selection in the Deployment Manager installation wizard. A property 'auth.method' present in the `/opt/hexis/hawkeye-ap/analyzer/config/app.install.properties` file controls which authentication method is used. If the file contains **auth.method=ldap**, then other properties are read to determine the LDAP or AD/LDAP server location, credentials and search pattern to use.

If set to any of the three modes (native, LDAP, or AD/LDAP), a default Analyzer "Admin" account will be created during installation. After the initial Analyzer "Admin" has logged in, additional users may have a role enabled to their account that provides "User Administrator" privilege. Regardless of the authentication mode, AP Analyzer manages a user's authorizations via roles and groups.

For details on LDAP set up and configuration through the Deployment Manager, see Configuring LDAP Integration for User Authentication on page 165. For details on AD/LDAP set up and configuration, see Configuring Single-AD Integration on page 174 and Configuring Multiple-AD Integration on page 175.

User Authorization

User Authorization locates and verifies the user's roles and permissions from the authentication authority.

There are four types of security enforced through User Authorization:

- **Resource Service Security** with the ability (through the use of user roles) to restrict access to a resource service and restrict the creation of a new resource such as a table, file, etc., Note that each resource has an owner and a type; the type defines the resource's properties and available actions.
- **Resource Security** with the ability to restrict actions (such as Read, Write, Delete, etc.) on a resource instance. For details on Resource Instance, see. "User Authorization", on page 154. Included in Resource Security is the ability to support resource security maintenance, by providing actions the current user

can perform on a secured resource. Resource Security is cot rolled through the object owner and each object's definition provides details on what users and groups are allowed to perform specific action(s).

- **Data Security** with the ability to restrict a user to schema(s) and table(s). Currently, the schema(s) and table(s) a user can view are maintained by Postgres Security (outside the SenSage AP system) through the user's postgresUser attribute and using a tool such as pgadmin. For more details, read "SenSage AP Security Model and Postgres", on page 156.
- **Security Inheritance** with the ability for users to inherit security from their groups and their groups sub-groups. For example, Specific resources can inherit security from a pre-defined resource hierarchy and can disable portions of the user interface. Other examples of inheritance is when a saved result can inherit its security from the report that created it or when a dashboard pod can inherit its security from the dashboard where it resides.

User Administration

Through SenSage AP Analyzer, support for user, role, and group maintenance is provided. For information on using the Analyzer to manage users, roles and permissions, see the *SenSage AP Analyzer Guide*.

Security Object Relationships

The following items provide a summary of the security object relationships in the SenSage AP Security Model:

- When each SenSage AP user is defined, roles and groups are associated with each user. A user's roles are the union of all their direct roles.
- A Resource Instance such as a report, schedule, etc., is defined with an action(s) that a user and/or a group may perform on the instance.
- Roles have a list of privileges.
- Privileges control what actions a user may perform.
- Users and groups control what objects a user may view and what actions users and groups can perform on that object.

SenSage AP Authorization Model

Resource instance actions (view, edit, delete, run) act on a specific resource instance, either a report or dashboard. These actions are fixed and cannot be changed at run time:

- **View**—Allows for viewing the resource only
- **Edit** —Allows for updates to the resource
- **Delete** —Allows for deletion of the resource
- **Run**—Allows for execution of the resource

The table below lists the resource instance actions that are pre-defined on each SenSage AP resource instance.

Resource Instance	VIEW	EDIT	DELETE	RUN
Reports [1]	X	X	X	X
Dashboards	X	X	X	X

Service actions act on a resource service such as Schedules, Users, Groups, et. Each resource service has at two actions; it provides access and creates the resource instance for the service. These actions are fixed and cannot be changed at run time.

Resource Instance	VIEW	EDIT	PUBLIC	PRIVATE
Models			X	X
Schedules	X	X		
Users	X	X		
Groups	X	X		
Roles	X	X		

SenSage AP Security Model and Postgres

SenSage AP security leverages Postgres' powerful and complex security model. By creating Postgres users and granting them access to schemas(s) and table(s) required for user selection, you allow SenSage AP to secure access to log data schemas associated with specific AP functionality.

Securing SenSage AP consists of the following areas:

- Standard Web Application Security
- Resource Security
- Data Security

Note: Postgres security provides in depth and robust security capabilities. The scope of this overview is limited only to the fundamentals required to achieve security through access to AP Event Data Warehouse (EDW) schemas, as managed through the SenSage AP Analyzer. For further details on Postgres' vast capabilities, see the Postgres documentation. For specifics on Analyzer usage, see the *SenSage AP Analyzer Guide*.

Postgres and Log Data Security

The SenSage EDW (Postgres) and the SenSage AP Analyzer authenticate and authorize data access by three discreet security features that control:

- SenSage AP features a user may perform
- SenSage AP Application objects a user may view and act on (that is, View, Edit, and Run permissions)
- SenSage AP owners of log data to ensure security and appropriate access of the data.

[1] Reports also has the following two resource instance actions: Export PDF and Export CSV.

Note that no separation exists between a defined Postgres user and role--the user and role is synonymous in the SenSage AP (Postgres) Security Model. The separation of security into the three separate areas noted above, allows, for example, an Analyzer Administrator not to have access to a specific user's (role's) log data.

Predefined SenSage AP (Postgres) Users

The SenSage AP system comes predefined with the following users and should not be modified.

- Hexis—Identifies the SenSage AP Superuser.
- Controller—Identifies the Database Owner.
- EDW—Identifies the OAE System User.

Setting up Analyzer Users with Postgres

The Postgres users (roles) and their associated EDW actions and schema (namespace) permissions specify what areas of the EDW, SenSage AP Analyzer, and event-log data are accessible to a given SenSage AP user and what actions the user may perform.

You initially create users through a tool like **pgadmin** and maintain users through the SenSage AP Analyzer interface. When creating a new Analyzer User, you need to determine which log data a specific Postgres user is allowed to access, that is, which log schemas the user can view. You will need to consider what Postgres users already exist and what schemas these users can access. Note that multiple Analyzer users can use the same Postgres user and an application user can be mapped to a Postgres user by the value of the user's postgresUser attribute.

To create a new Postgres user, follow the example steps below:

1. Log into the Analyzer.
2. Create a Postgres user (role); in the example below the user is named abc:

```
psql -h localhost -p5432 -d controller -U hexis -c "CREATE ROLE abc
NOSUPERUSER
NOCREATEDB NOCREATEROLE INHERIT NOLOGIN;"
```

Note: NOLOGIN prevents direct login from outside the application.

3. Grant access to the schema(s) this user is allowed to access. as noted in the following sample commands below.
   a. This command allows access to schema **oae** to allow user **abc** Postgres extension over the EDW:

```
psql -h localhost -p5432 -d controller -U hexis -c "GRANT USAGE ON SCHEMA
 oae TO abc";
```

   b. This command allows user **abc** access to schema **default_analytics**:

```
psql -h localhost -p5432 -d controller -U hexis -c "GRANT USAGE ON SCHEMA
 default_analytics TO abc";
```

   c. This command allows user **abc** access to query all tables in schema **default_analytics**:

```
psql -h localhost -p5432 -d controller -U hexis -c "GRANT SELECT ON ALL
TABLES IN SCHEMA default_analytics TO abc";
```

4. Update the Analyzer Application user's Postgres user setting.
   a. Logon as **admin**.
   b. Go to **Administration** tab and **Manager Users** screen.
   c. Edit the user(s) whose schema security you want to change. Do not grant access to the **saved_result** schema. Only an Administrator should have access to the **siem** schema.
   d. Change the user's Postgres User setting to the new Postgres user/role created in step 1.
   e. Verify that only the expected schemas exist.
   f. Log into the Analyzer application as the user you edited in step 3.
   g. Create a new report.
   h. Verify the correct schemas show up in the list of available schemas.
   i. Run a report that the user has privilege to access.
   j. Run a report in which the user has no privilege to access.

5. To allow a user to query to all tables from SenSage AP 5.0.0 to 6.0.0 (instead of issuing the command in example 2C above) perform the following steps on the SenSage AP 5.0.0 system:
   a. Issue the follow command; in the example below, the user is named **abc**:

```
psql -h localhost -p54321 -d controller -U hexis -c "SELECT 'GRANT SELECT
 ON default_analytics.' || relname || 'TO abc 'FROM pg_class JOIN
pg_namespace ON pg_namespace.oid=pg_class.relanmespace WHERE nspname =
'mysls_analytics" AND relkind IN ('r', 'v');"
```

   b. Save the output and create a file `mySavedGrants.sql`:

```
psql -h localhost -p5432 -d controller -U lms -c mySavedGrants.sql
```

After you have created a user that has the correct access, run the AP Analyzer application and use the Administration function to create or edit the user's Postgres user attribute and select the Postgres user you just created. For details, see the *SenSage AP Analyzer Guide*.

Revoking SenSage AP (Postgres) User Permissions

To revoke a SenSage AP (Postgres) user's permission, follow the example steps below:

Remove the user's access to the applicable schema. For example, to revoke all privileges on schema default_analytics for user abc, issue the following command:

```
psql -h localhost -p54321 -d controller -U hexis -c "REVOKE ALL PRIVILEGES ON
 SCHEMA default_analytics FROM abc;"
```

Configuring Native Authentication

In *Native* authentication users are created and managed solely in the Analyzer database in which passwords are stored and encrypted. All information for authorizing access to functionality or data in the application is stored in the Analyzer database. The default "Native" mode is selected in the Deployment Manager console at the time of installation; you can ignore all LDAP/AD fields, which will be displayed as read-only when native mode is select.

Native First Time Login

After a fresh installation of SenSage AP Analyzer, no users in the database are guaranteed to be present. When native authentication is in use and the first user logs into the system, if that user is successfully authenticated in AP Analyzer, the user will become a designated "Admin" user with the ability to create other users.

Configuring LDAP Integration for User Authentication

LDAP only manages authentication into the system, not authorizations. Users who are authorized to access AP Analyzer will need to be added as an "Analyzer User" before they can authenticate through LDAP.

Note: The Analyzer application will either be fully native, fully LDAP, or fully AD/LDAP; no mixed or hybrid mode is allowed. If LDAP authentication is in use and some users in the Analyzer database do not currently exist in LDAP, you cannot log in with those users, nor edit them. Note that the user list provides no indication LDAP versus native users.

At installation time, you must select "LDAP" mode in the Deployment Manager console and fill in the LDAP fields.

Designating LDAP user authentication one-time (for all users)

To designate LDAP user authentication one-time for all users, your enterprise must have Active Directory (AD), a Windows-based authentication authority that manages users and groups. If both AD and the SenSage AP cluster is installed and operational, to authenticate all users at once, see Configuring Active Directory/LDAP Integration on page 169.

Roles and Permissions in the Analyzer Database

All information for authorizing access to functionality or data in the application is stored in the Analyzer database. Therefore, in order for a user in LDAP to be able to use the Analyzer application, the user must be created in the Analyzer database and given appropriate permissions. When an administrator creates a new user and LDAP authentication is in use, the administrator must pick from a list of users that are present in LDAP.

Note: In AP Analyzer, you can add or modify a user's first, last name, or email address and synchronize this information with LDAP. For details, see the *SenSage AP Analyzer Guide*.

LDAP First Time Login

After a fresh installation of SenSage AP Analyzer, no users in the database are guaranteed to be present in LDAP. When LDAP authentication is in use and the first user logs into the system, if that user is successfully authenticated against LDAP, the user will become a designated "Admin User" with the ability to create other users. The "Admin" User can designate LDAP user authentication one-time for all users using the procedure as noted in Designating LDAP user authentication one-time (for all users) on page 165.

Analyzer Database and LDAP

A system_state table exists in the Analyzer database. This table contains key/value pairs. If no LDAP admin has been created (initial install with no logins), there will be one row in the database as in:

id	key	value
1	ldapAdmin	null

After a user logs in and is authenticated against LDAP, the database is updated with the username of the LDAP Admin, as in:

id	key	value
1	ldapAdmin	ldapuser1

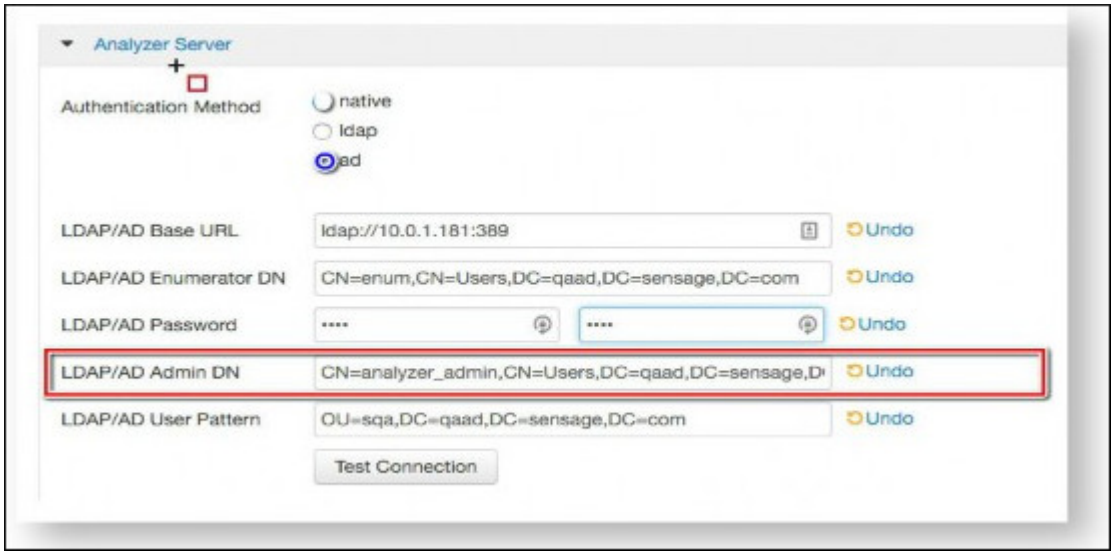
Important: The application currently does not support switching from LDAP or AD/ LDAP authentication back to Native. If LDAP or AD/LDAP authentication is in use and some users in the Analyzer database do not currently exist in LDAP, you cannot log in with those users, nor edit them. Note that the user list provides no indication of LDAP versus native users.

Setting Up the Analyzer Administration Account

The following are set up instructions for setting LDAP for SenSage AP Analyzer:

1. Install Analyzer with LDAP authentication via the Deployment Manager.
2. Configure the following settings, as shown in .
   a. LDAP/AD Manager DN: The distinguished name for the account used to control the LDAP instance, or an account with search access.
   b. LDAP/AD Password: The password for the Manager DN used to connect to the LDAP instance.
   c. LDAP/AD User Pattern: The pattern representing the query to find a user by user identifier.The pattern should include the distinguished name of the entity containing users and the attribute where the user ID is stored.

Figure 40: Configuring the LDAP Server



3. The Analyzer "Admin" logs in.
4. The Analyzer "Admin" creates user accounts with appropriate group and role permissions that match the username of LDAP user.
5. The LDAP user can then login using their LDAP password to Analyzer.

Useful SQL Scripts for LDAP Integration

```
-- Backup the user, user_role and user_group tables and then clear them. Used
 for
testing a clean install with ldap integration where the first user to log in
becomes the admin.
CREATE TABLE siem.user_bkup AS (select * from siem.user);
CREATE TABLE siem.user_role_bkup AS (select * from siem.user_role);
CREATE TABLE siem.user_group_bkup AS (select * from siem.user_group);
DELETE FROM siem.user_role;
DELETE FROM siem.user_group;
DELETE FROM siem.user;
UPDATE system_state SET value = null where key = 'ldapAdmin';
-- Restore the user, user_role and user_group tables back to what they were
before running the backup scripts.
```

```
DELETE FROM siem.user_role;
DELETE FROM siem.user_group;
DELETE FROM siem.user;
INSERT INTO siem.user (SELECT * from siem.user_bkup);
INSERT INTO siem.user_group (SELECT * from siem.user_group_bkup);
INSERT INTO siem.user_role (SELECT * from siem.user_role_bkup);
DELETE FROM siem.user_role_bkup;
DELETE FROM siem.user_group_bkup;
DELETE FROM siem.user_bkup;
```

FAQs

1. Assuming there is a report definition foo, who is able to view it?

   User must be a member of role that has the ReportServiceAccess action. This could be a users role or a user's group's role.

   You need the VIEW action on the resource, you can get that in one of these ways:

   - User is the owner of report foo
   - User has VIEW action to foo
   - User is a member of group that has the VIEW action to foo

2. **Who can run Report foo?**

   Same as above except the action checked is the RUN action. To actually get data a user's Postgres user must also have been granted access to the schema/table the report queries

3. **Who can view saved results?**

   Saved results are an exception; their security is controlled by the report that created it.

   You need VIEW action on the report and your Postgres user needs access to the schema/tables used in that report.

4. **Who can change a resource's security?**

   EDIT action is required to set security for a resource. EDIT action is achieved in one of these ways:

   - The owner can have all permissions on the resource
   - User has EDIT access to the resource
   - Any user that is a member of a group that has EDIT action to the resource

5. **What happens to the owner attribute when the user is removed from the system?**

   When deleting a user, one can specify the new owner. If the owner is removed, the report is "orphaned" and you will need to apply a manual fix in the database.

6. **When I create a new resource who has access?**

   Only the owner has access.

8

Configuring Active Directory/LDAP Integration

This chapter describes how to integrate Active Directory with a SenSage AP deployment and contains the following sections:

Overview

To authenticate and authorize users in SenSage AP, you can do one of the following:

- Specify “native” users who are stored in Postgres for authentication and authorization. Note that this is the default when you install SenSage AP.
- Use LDAP (Light-weight Directory Access Protocol) for authentication which you have the option to select when you install SenSage AP. With LDAP authentication user names and passwords are stored in LDAP and users are authenticated against LDAP at login time.
- Use Active Directory (AD) to delegate SenSage AP Analyzer user authentication to AD. You use Analyzer to synchronize users and groups with the ones in AD. Those users and groups that are members of the AD domain specified in Analyzer are automatically mapped to SenSage AP.

Note: An Analyzer "admin" account is created in Analyzer during product installation. This account has the privilege to create a user account that matches the user name of an LDAP user and provides that user with appropriate group and role permissions.

Using Active Directory with LDAP

The Active Directory authority authenticates each user's name and password at login. After you configure Active Directory integration, SenSage AP Analyzer logs into the Active Directory server for enumeration of groups and users using on-demand and scheduled synchronization.

This solution provides the following advantages:

- allows enterprise administrators to manage users, groups, and passwords using a familiar mechanism that already provides these services for other systems and applications.
- delegates authentication from AP Analyzer to an external authority system (such as AD/LDAP)
- facilitates logging
- facilitates the management of password rotation, enforces password complexity policies and recovery of users accounts from lockout.

What you Need to Know before Integrating Active Directory/LDAP

You can configure AD/LDAP integration on a new SenSage AP deployment or migrate an existing SenSage AP deployment to use AD integration. Whether you are using a new deployment or migrating to an existing one, the process is the same.

Following is a brief overview of the AD/LDAP preparation, configuration, and integration process. The actual configuration and integration steps are described in detail in Setting Up Active Directory.

1. To prepare for AD/LDAP integration:
   a. The AD Administrator creates a AD/LDAP Enumerator user account; a default Analyzer "Admin" that is already in the Analyzer system will be mapped to the AD (enumerator) user.

Important: : User names are case sensitive in SenSage AP and Active Directory will preserve the case of the name as entered by the AD Administrator into the SAM field, from which AP user names are derived.

b. For a new SenSage AP deployment, an administrator is required to identify all users, roles and permissions in the organization. For an existing deployment, an administrator is required to identify all existing roles and users prior to beginning the integration. For details, see the following section, *Mapping Users and Groups*.

c. In the Active Directory Console, the AD Administrator adds a **hexis** OU and adds any desired users and groups to the OU (directory or via OU Association).

2. To perform set up tasks such as user sync or group sync, the Analyzer "Admin" is required to log into the Analyzer for the first time with the AD Enumerator DN password.
3. In Deployment Manager, the SenSage AP Service Configuration screen is used to configure a single AD server as noted in Configuring Single-AD Integration on page 174. Additional servers must be configured in the Analyzer Host as noted in Configuring Multiple-AD Integration on page 175.
4. In AP Analyzer, the Analyzer "Admin" can populate users by synchronizing with AD immediately, or by creating a schedule to synchronize periodically. The Analyzer "Admin" can also edit the populated user accounts and add additional accounts manually through mapping AD server mapping details to each user account.
5. In AP Analyzer, the Analyzer "Admin" can create a group and synchronize the group with the selected AD/LDAP server mapping details.

Note: When groups are synchronized from Analyzer, all *existing* users (if migrating to AD/LDAP from an existing SenSage AP deployment) under the specified DN pattern will be updated in Analyzer. This means that the membership group in Analyzer will match the membership of the group in AD.

Consideration for Mapping User and Groups

his section discusses considerations for configuring AD/LDAP users and groups and how these objects map into a SenSage AP deployment.

Before beginning Active Directory Integration for a new SenSage AP deployment, be sure to identify all of the users, roles, and permissions that are required. (For more information, see Appendix A, "SenSage AP Permission Requirements" in the *SenSage AP Analyzer Guide*.

If you are integrating an existing deployment, note all of the roles and users prior to beginning the integration. When each SenSage AP user is defined, roles and groups are associated with each user. A user's role are the union of all their direct roles.

Consider the following when mapping users and roles:

- Roles are local to Analyzer only and are not mapped in AD/LDAP. You configure SenSage AP authorization in the Role tab of SenSage AP Analyzer by granting permissions to roles. These permissions authorize SenSage AP users to access specific features and data in Analyzer.
- .When you configure AD/LDAP integration, AD groups and users that are members of the specified AD domain are automatically mapped to SenSage AP when users and groups are synchronized in Analyzer.

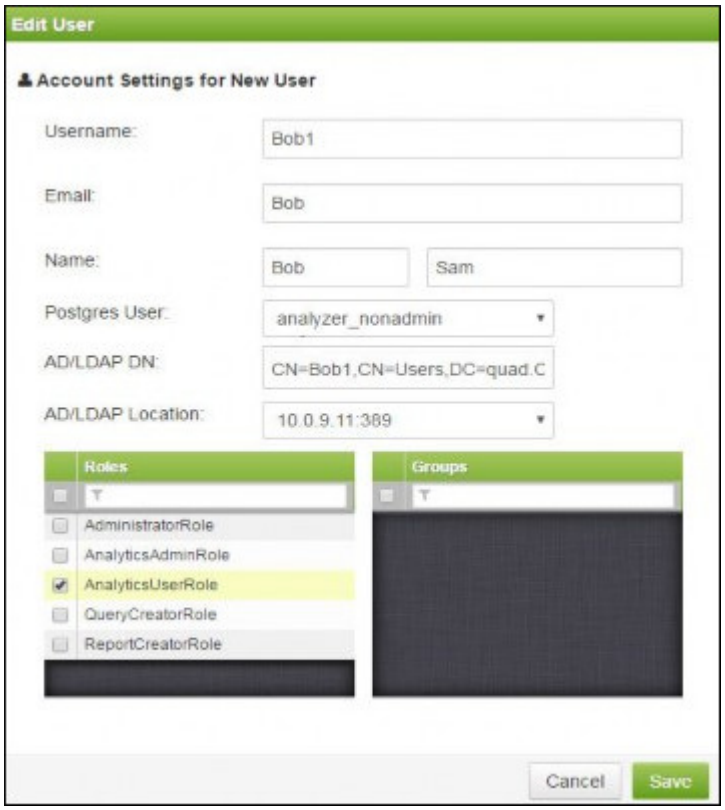
Note: Users are synchronized from configured AD servers to SenSage AP on demand (for a single-server only) or on schedule (all AD-servers). A LDAP scheduling screen is provided in Analyzer to set up a

schedule and a button is provided to update users from LDAP on demand. For details on using this feature, see the *SenSage AP Analyzer Guide.*

- It is possible that users from multiple domains have the same "SAM Account Name". In this case, you should map all of the Active Directory users to different Analyzer users. For example, if there were two users with SAM Account Names of **Bob** in **domain1** and **Bob** in **domain2**, the duplicate user is skipped.

  Since a user already gets created with the username **bob** after synchronization, for the second user, you need to create a user account with the username as **Bob1** in Analyzer (as shown below) and map the user to the appropriate AD. Users will then be able to log into Analyzer with unique names. Analyzer will map these unique names to the proper user account that is identified with the account's DN in the specific AD server where that account is defined. The user is then authenticated by the AD server according to the DN selected.

Figure 41: Mapping SAM Account Names



Setting up Active Directory Integration

Note: Before you begin this procedure, be sure you have SenSage AP cluster(s) installed that are currently operable.

To integrate your deployment with Active Directory, follow each section of this procedure in the order presented.

Setting up the Analyzer Administration Account

In SenSage AP, the AD Administrator can set up an Analyzer administration login account and if desired, map that account to a desired user as noted in steps 1 and 2 below.

1. Create an enumerator user that will be used to connect to AD/LDAP. This user requires READ- ONLY access to view all other AD/LDAP users who will be synchronized with AP Analyzer.

   Note: By default, this enumerator user belongs to the Analyzer “Admin” account. Unlike all other AD/LDAP users specified in AP Analyzer, the Analyzer "Admin" user is given full administrative rights to AP Analyzer.

   Important: : When issuing a username and password for the enumerator user, be sure it is unique and not used by any other AD/LDAP user. The enumerator user will be mapped to the Analyzer "Admin" user until the mapping is changed.

2. If you want to designate the Analyzer "Admin" user as the SenSage AP Administrator, create an appropriate Active Directory account for the SenSage AP Administrator, for example, **AdminUser**. Then follow these steps:

   a. Log into Analyzer and go to **Administration** > **Users** .

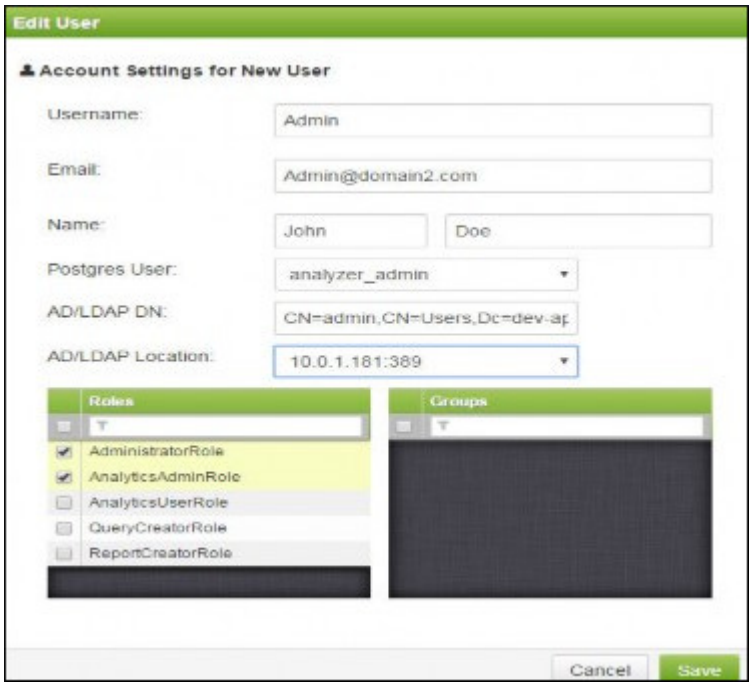
   b. Select **Admin** and click **Edit**.

   c. Change the DN mapping in the LDAP/AD Admin DN field to point to the **AdminUser** account in AD.

   Note: The Analyzer "Admin" can only be mapped to an AD SAM account name. The system will make the necessary changes to the Analyzer "Admin" account (default is "Admin") so that the username of the Analyzer "Admin" user becomes the SAM account name of the entered DN.

The Analyzer "Admin" will now be able to log into SenSage AP Analyzer with his or her own AD credentials with the username as the SAM Account name and the password the same as the one used in AD/LDAP.

Figure 42: Mapping Analyzer Admin to another Administrator Account



Configuring Objects in Active Directory

Active Directory arranges objects in a hierarchy that comprises Domains, OUs (organizational units), groups, and users. The Active Directory framework also includes higher-level collections of objects called forests and trees. For integration with SenSage AP, only domains, OUs, and groups are relevant.

Recommendations

- Provide at least the “enumerator” user that is mapped by default to the Analyzer “Admin” when setting up AD/LDAP integration.
- Enter at least the set of required groups that map to required SenSage AP roles.
- Enter any additional users and groups needed for the deployment.
- Assign existing users who require access to SenSage AP to the appropriate Active Directory groups.

See your Active Directory administrator for details on configuring objects

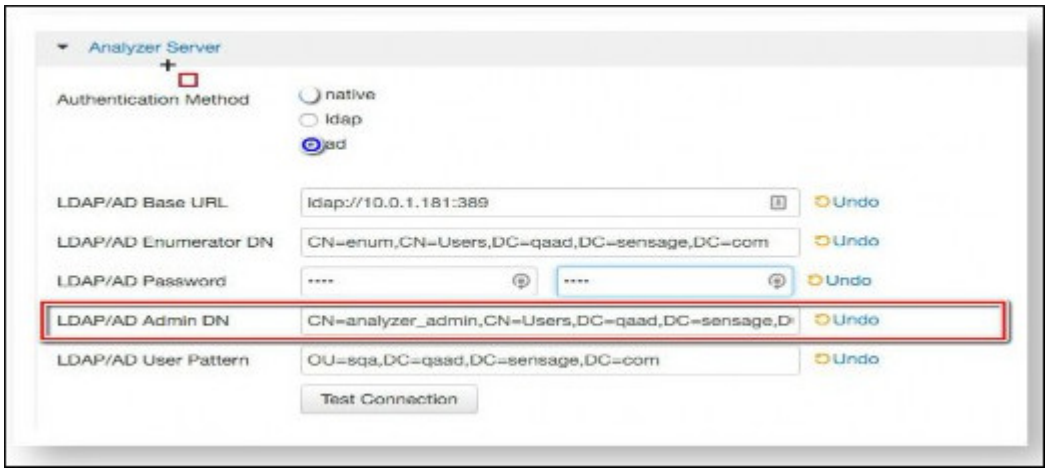
Configuring Single-AD Integration

Follow these steps if you have a single-AD server environment. Otherwise, go to the next section to configure a multi-AD server environment.

1. From the Deployment Manager console, go to the Services tab and select the SenSage AP cluster link.
2. Stop the SenSage AP cluster.

3. Under the Analyzer Server section of the Configs tab, select the “ad” option and provide the following settings as shown in see Figure 8-3.

Figure 43: Configuring a Single-AD Server



4. In the LDAP/AD Base URL field, provide the URL connection string for your first or only AD. For example:

```
ldap://ad_host.customer.com:389
```

5. In the LDAP/AD Enumerator DN field, provide the enumerator account details. For example:

```
CN=analyzeradmin,OU=org,DC=customer,DC=com
```

6. In the LDAP/AD Password field, provide the password for the AD enumerator (Enumerator DN).
7. In the LDAP/AD Admin DN field, provide the fully-distinguished name of the System Administrator in AD. For example:

```
CN=analyzer admin,OU=org,DC=customer,DC=com
```

Note: All LDAP elements (like CN and OU) must be in uppercase.

8. In the LDAP/AD User Pattern, enter the required AD user pattern. For example:

```
CN=Users,DC=dev-ap,DC=local
```

9. Save the changes and start the SenSage AP cluster.
10. Go to Updating/Scheduling an AD/LDAP synchronization on page 176 to synchronize users to the AD server.

Configuring Multiple-AD Integration

This section describes the process of configuring multiple AD servers for integration with Analyzer.

1. Go to the Deployment Manager console and make sure that you have configured your first AD server as noted in the previous section.

Note: In Deployment Manager, the required value that you enter in the LDAP/AD Admin DN field is the fully- distinguished name of a user with a valid SAM Account namein AD. The system will look up

your user-entry in the first server. Once it finds the Sam Account name of the user, it makes the necessary changes to the Analyzer administrative account (which by default is "admin") so that the username of the administrative user becomes the Sam Account name of the entered DN.

2. From the Deployment Manager console, go to the Services tab and select the SenSage AP cluster link.
3. Stop the SenSage AP cluster.
4. In the Analyzer host, locate the following path:

   ```
   /opt/hexis/hawkeye-ap/analyzer/config
   ```

5. Using the vi editor, open the `ldap.properties` file and add an entry for each additional AD server. Specify these details in the following format:

   ```
   auth.ldap.url.1=ldap:<LDAP_Host_Name_1>
   auth.ldap.manager.dn.1=CN=administrator,CN=Users,DC=dev1-ap,DC=local
   auth.ldap.manager.password.1=passwordvalue1
   auth.ldap.user.dn.pattern.1=OU=companyname,DC=dev1-ap,DC=local
   auth.ldap.url.2=ldap:<LDAP_Host_Name_2>
   auth.ldap.manager.dn.2=CN=administrator,CN=Users,DC=dev2-ap,DC=local
   auth.ldap.manager.password.2=passwordvalue2
   auth.ldap.user.dn.pattern.2=OU=companyname,DC=dev2-ap,DC=local
   ```

6. Start the SenSage AP cluster.
7. Go to the next section to synchronize users' AD servers.

Updating/Scheduling an AD/LDAP synchronization

To schedule an AD/LDAP synchronization for an AD server(s):

1. To log into Analyzer, provide **admin** as the username and provide the enumerator password for the first server that was created while setting up AD integration.
2. To set up scheduled sync (which syncs all AD servers) in Analyzer, go to **Administration** > **LDAP** , set a synchronization interval (15 minutes is sufficient), and click **Save Schedule**.

Figure 44: Configuring LDAP Synchronization

3. To synchronize users on demand, select the AD server to be synced from and then click **Update users from LDAP**.

   This is mandatory step because unless the users are synchronized, they cannot login using their LDAP credentials.

4. To verify the updated users, go to the **Administration** > **Users** container and look for the users added from AD.
5. Click the **Update users from LDAP** button. The users are synchronized from the selected LDAP server immediately. This is a mandatory step because unless the users are synced, they cannot login using their LDAP credentials.
6. To verify the updated users, go to **Administration** > **Users** and look for the users added from AD/LDAP.

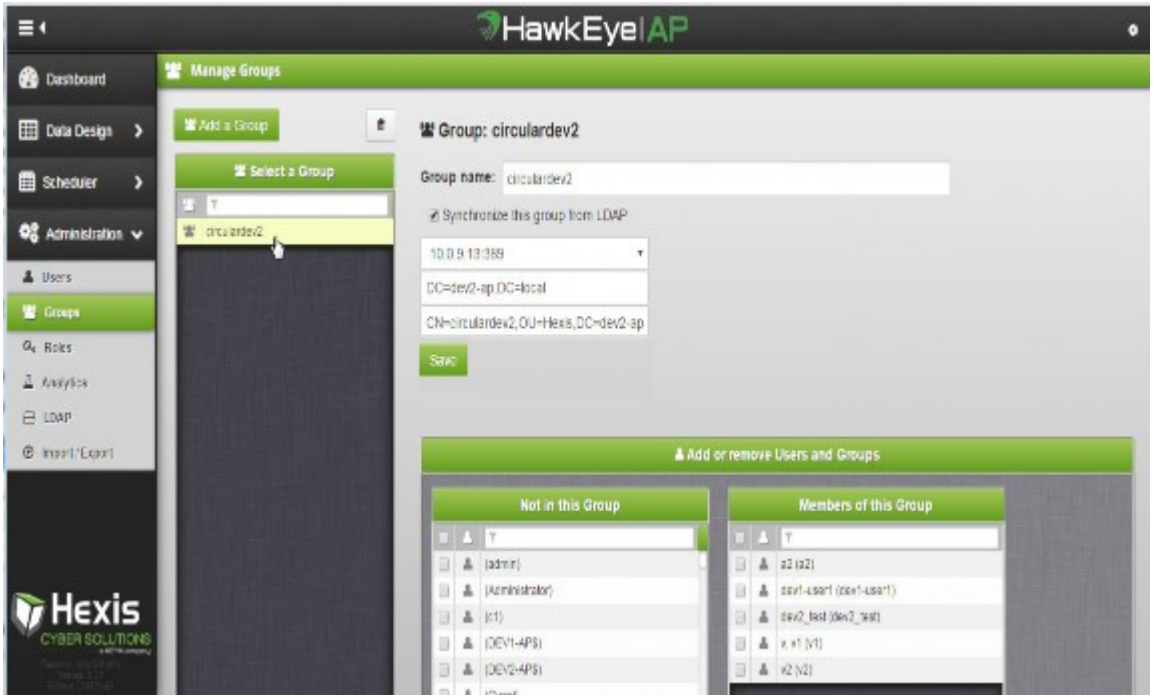
Synchronizing Groups from AD/LDAP

NOTE: Before you synchronize groups from LDAP, make sure you have updated users from LDAP (step 4 of the section above).

To synchronize groups from LDAP:

1. To synchronize groups, go to **Administration** > **Groups** .
2. Add a group.
3. Select **Synchronize this group from LDAP**, provide this group DN information, and click **Save**.

   Figure 45: Synchronizing a Group from LDAP NEW SCREENSHOT NEEDED



Once saved the members of the Analyzer group will be modified to reflect the members of the AD group specified by the Group DN, provided the users are already Analyzer users. Management of the group membership should no longer be performed manually in the Analyzer.

Maintaining Active Directory Integration

After integrating Active Directory into a SenSage AP deployment, you use Active Directory tools for all management of users and groups. You use Active Directory to enter new users and to assign new and existing users to the appropriate Active Directory groups that authorize the user to access SenSage AP.

For more information on using SenSage AP roles to authorize users, see the *SenSage AP Analyzer Guide*.

Troubleshooting

I am not able to synchronize the AD users with the Analyzer Host?

You can verify the connectivity from the Analyzer to the AD server using the telnet command. For example:

```
telnet ad_server 389
```

How do I make sure that enumerator credentials are correct to get all users and groups in the hexis OU returned?

Run the **ldapsearch** command in the host terminal as shown in the syntax below; in the command specify the LDAP container that you want to verify.

Syntax:

```
/opt/hexis/hawkeye-ap/bin/ldapsearch -D "<bind DN>" -w enum -p <port on LDAP
server> -h <LDAP server address> -b "<base dn for search>" -s sub "<search
scope>"
/opt/hexis/hawkeye-ap/bin/ldapsearch -D CN=enum, CN=Users, DC=qaad,
DC=sensage, DC=com" -w enum -p 389 -h 10.0.1.181 -b "OU=sensage, DC=qaad,
DC=sensage, DC=com" -s sub "(objectclass=*)"
```

Example:

```
/opt/hexis/hawkeye-ap/bin/ldapsearch -D
"CN=enum,CN=Users,DC=qaad,DC=sensage,DC=com" -w enum -p 389 -h 10.0.1.181 -b
"OU=sensage,DC=qaad,DC=sensage,DC=com" -s sub"(objectclass=*)"
```

Where can I look for error logs in case any problems with AD/LDAP configuration occurs?

You can refer to the error messages in the **Analyzer.log** located in the following location:

```
/opt/tomcat/logs
```

The following log is one of the example error messages:

```
[Thu Aug 20 18:57:35 UTC 2015] SEVERE:
com.hexiscyber.analyzer.web.security.SiemUsernamePasswordAuthenticationFilter
doFilter - An internal error occurred while trying to authenticate the user.
org.springframework.security.authentication.InternalAuthenticationServiceExcep
tion: [LDAP: error code 49 - 80090308: LdapErr: DSID-0C090334, comment:
AcceptSecurityContext error, data 525, vece]; nested exception is
javax.naming.AuthenticationException: [LDAP: error code 49 - 80090308:LdapErr:
DSID-0C090334, comment: AcceptSecurityContext error, data 525, vece]
```

FAQs

What if I want to use the AD Administrator account to log in to AP Analyzer? Do I need to have the AD Administrator added to the Hexis OU?

No, any user is supported on any Domains and OUs and is not required to be part of the same pattern. However, if the account is not part of the pattern it will not be synchronized whenever the synchronization process is run; if it is part of the pattern, it will be synchronized. To use the AD

Administrator account to log into AP Analyzer, if the SAM account name of the AD Administrator matches with the default SAM account of Analyzer Admin, upon sync, the account is updated and you can log in using the AD Administrator account. In case of a different SAM account (other than Admin), you need to only assign admin roles to this account (using Edit User screen).

Can I selectively change the settings of a user from the Analyzer GUI?

Yes, you can, but these changes will be over-written with settings from AD whenever the user database is synchronized with AD. The AD synchronization updates all fields of all users, whose name matches the SAM account field in AD, and if the account was originally created from synchronizing with the same AD. An Analyzer user account that has no corresponding SAM account in the AD is not updated. An Analyzer user account that is synchronized with one AD server already will not be updated by synchronizing any other AD servers.

How do I delete a user?

To delete a user, you must first remove the user account in AD. Then, remove the user account in Analyzer. This is because the AD synchronization will not cause any user to be deleted in the Analyzer.

If I delete a user in the Analyzer, will the user re-appear whenever AD synchronization happens next?

Yes, it will appear as long as the user is not deleted from AD.

How does AD synchronization work in a multi-realm environment?

If there are multiple accounts with the same SAM account field, then all of them except for the first one should be given a unique name in the Analyzer and separately synced to the respective AD servers. For example, if there are two accounts with **Bob** in the SAM account field in two different AD servers, then the first one is synced and an Analyzer account named **Bob** will be mapped to it. None of the other accounts with the SAM name **Bob** will be mapped to the Analyzer when it is synchronized with them. The Analyzer administrator must create **Bob2** in the Analyzer and manually map that user to the second AD server.

9

Archiving to Nearline Storage

This chapter contains the following sections:

Nearline Storage: Overview

This topic describes:

Models for Managing Local Storage Space

The default configuration of an EDW instance uses local disk space for its storage space. With local storage space, an EDW instance stores and retrieves data very quickly. However, as the amount of stored historical data grows, local disk space can run out.

Note: SenSage AP does not support de-duplicating identical data in a Nearline storage because the files are laid out differently on disk.

The EDW component supports two methods for expanding EDW (Event Data Warehouse Server) storage capacity:

- **Archiving data**

  Archiving data requires configuring the EDW component to communicate with Nearline storage devices. The EDW then can move old data from local storage onto Nearline storage. Old data on local storage is replaced with references to its location on Nearline storage. These references let you reclaim local storage space for new data, because references to archived data are much smaller than the data itself.

  Archiving data allows you to reserve local storage for recent data. Queries of recent data execute quickly because the data is on local storage. Historical data remains accessible on Nearline storage, but queries execute more slowly. When a person queries the EDW for archived data, it automatically and transparently retrieves the data regardless of whether it is in local or Nearline storage.

- **Retiring data**

  Retiring data deletes it from the storage space of EDW instances and reclaims the storage space that it consumed. After you retire data, it is completely gone and is not accessible to queries. Retire data instead of archiving it only if you do not need access to the historical data that you retire.

  Important: Retiring data from an EDW table removes only data that is not stored under retention on a Nearline storage device. The command skips archived data that is under retention and logs a message that provides the date in the future when the Nearline storage device will allow you to retire the data. That date is the earliest you can retire the data. The command does remove archived data that is not under retention or its retention period has passed. On some Nearline storage devices, however, in order to ensure retire time specifications are enforced, the command may be required to restore some data not under retention and let it remain online even though it falls outside the retire date. For more information, see Retiring Data on page 115.

Types of Nearline Storage Space

You can use the SenSage AP Nearline Storage Server (NSS) to archive data to the following Nearline storage devices:

- **EMC® Centera™ Nearline Storage Driver**

  This module allows your organization to take advantage of the unique compliance features that Centera provides. The EDW uses the proprietary Centera API to move data to and from the Centera device.

- **Remote NFS (Network) or CIFS (Common Internet) File System**

  Remote file systems allow your organization to archive historical data to regular storage devices on remote hosts. The remote file systems are mounted locally on the hosts where the EDW instance runs. The local mount points on all EDW hosts must be the same, but each mount point can reference a different remote file system.

In summary, Nearline storage provides:

- Easier management of storage space. Scheduling the archiving of data to Nearline storage helps you maintain sufficient local storage space for EDW instances.
- Easier expansion of storage space. If you require more space, add more storage to your SenSage AP-supported Nearline storage device.

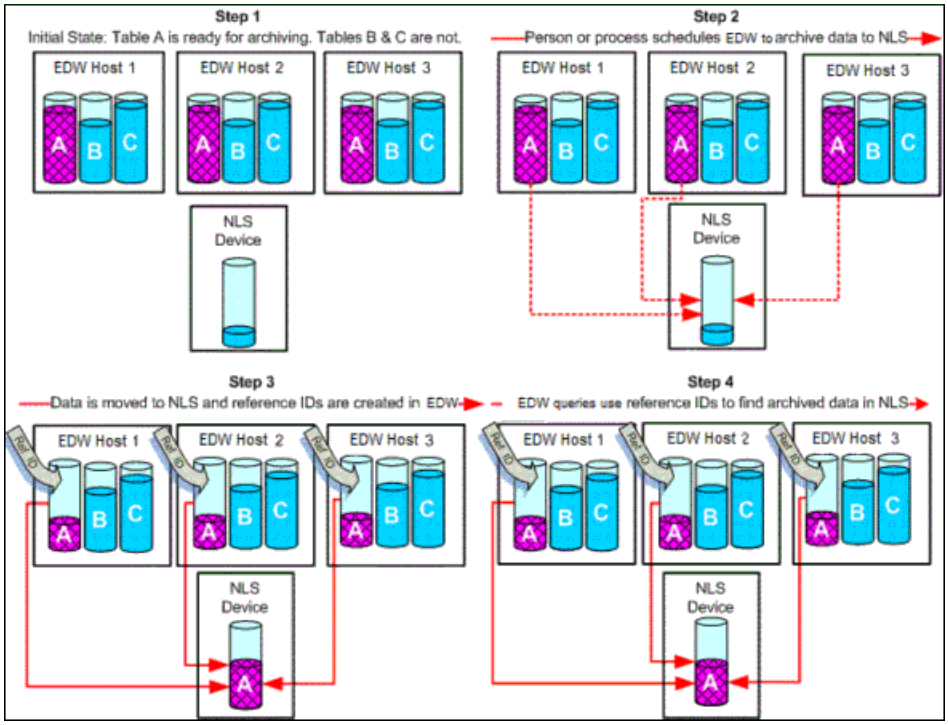
- More evenly balanced data loads on EDW hosts. When you add hosts to an EDW instance, the newer hosts initially have less data on them than the older, existing hosts. Because older data is archived before newer data, older hosts in the EDW instance are generally the first to reclaim local storage space.
- Automatic retrieval of archived data. When people run queries, they do not need to know whether the are retrieved from local or Nearline storage.

How Nearline Storage Works

The figure below illustrates the states of a SenSage AP Nearline-storage archive process.

- **Step 1: Initial State**—The data store for the EDW instance resides on the local hosts, such as the three hosts shown in the figure below.
- **Step 2: Archive Process**—An outside agent (a person or automated process) schedules archiving of EDW data onto the Nearline storage device.
- **Step 3: Archive Complete**—After archiving completes, a BLOB (binary large object) that contains the user's data and other associated data is created on the NLS device. A reference ID to the blob is created in each data store of the EDW instance.
- **Step 4: Active**—When a query that requires access to the archived data on the Nearline storage device is invoked on the EDW, the EDW uses the reference IDs to locate the data for retrieval.

Figure 46: Archiving to Nearline Storage



Managing the Archiving to Nearline Storage

You can manage the archiving of local data in an EDW instance manually or on a scheduled basis. SenSage AP provides a command-line utility for manually archiving local data on demand.

Before people can manage archiving to Nearline storage, a system administrator performs some initial configuration tasks. Thereafter, other people can manage archiving manually or on a schedule.

For more information on the command-line utility, see Archiving of Local Data to Nearline Storage on page 191.

Initially Configuring Nearline Storage

Before people can manage the archiving of local data to Nearline storage, a system administrator performs these initial configuration tasks described in the following sections:

- Defining and Managing Nearline Targets: Overview on page 184
- Specifying NSAs for Centera Systems on page 185

Defining and Managing Nearline Targets: Overview

A *Nearline* target is a specific Centera device or remote file system. You define a Nearline target for the EDW component with two pieces of information:

- **NSI (Nearline Storage Identifier)**—a unique text identifier for a Nearline storage target. The format of an NSI is the same regardless of the type of Nearline storage.
- **NSA (Nearline Storage Address)**— a connection specifier for a Nearline storage target. The format of an NSA differs according to the type of Nearline storage device.

After you define Nearline targets, people can use them when they manually archive local data from a table or schedule the archiving of local table data.

To define a Nearline target, a user with administrator privilege (`sls.admin` permission) uses the `atmanage` utility to specify the Nearline storage identifier and network storage address for the target. An administrator also uses the `atmanage` utility to delete Nearline targets and to list them. The syntax has three forms:

- Adding a Nearline target:

  Syntax

  ```
  atmanage <cluster_list> --user=<name> --pass=<pass> setnsi <nsi> <nsa>
  ```

  Examples

  ```
  atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
  --password=changeme setnsi centera-1 centera://centera1.cascommunity.org
  ```

- Removing a Nearline target:

  Syntax

  ```
  atmanage <cluster_list> --user=<name> --pass=<pass> deletensi <nsi>
  ```

  Important: Running atquery against data archived to a deleted NSI generates errors.

  Examples

  ```
  atmanage "sls01:8072,edw02:8072,edw03:8072" --user=administrator \
   --password=changeme deletensi centera-1
  ```

- Listing Nearline targets:

  Syntax

  ```
  atmanage <cluster_list> --user=<name> --pass=<pass> listnsi
  ```

  Example

  ```
  atmanage "sls01:8072,sls02:8072,sls03:8072" --user=administrator \
   --password=changeme listnsi
  ```

Specifying NSAs for Centera Systems

When specifying the NSA for a Centera unit, precede the network storage address with the string:

```
centera://
```

The high-level NSA syntax for a Centera unit is:

```
centera://<connection_specification>[|<retention_specification>]
```

Note: In the syntax above, the pipe symbol (|) is a character in the syntax; it is not the meta- syntax character that separates mutually exclusive options.

The network storage address for a Centera unit has two parts:

- *<connection_specification>*—includes parameters legal in Centera for FPPool_Open. For more information, see Specifying the Connection Specification on page 185, next
- *<retention_specification>*—specifies how long the data should be retained. For more information, see Specifying Data Retention in Centera NSAs on page 187.

Specifying the Connection Specification

The connection specification includes parameters legal in Centera for `FPPool_Open`. These parameters include:

- Host Identifiers—for more information, see Specifying Host Identifiers in Centera NSAs on page 185.
- PEA file location—for more information, see Specifying the PEA File in Centera NSAs on page 186.
- The username/secret for the PAI module to be used by the application—for more information, see Specifying the Username and Secret for the PAI Module in Centera NSAs on page 186.

The full syntax for the connection specification is one of the following:

```
<IP_Address>[:<port>]|<Host_Name>[:<port>]?<path_to_PEA_file_on_SLS_hosts>
```

or

```
<IP_Address>[:<port>]|<Host_Name>[:<port>]?name=<username>,secret=<password>
```

Specifying Host Identifiers in Centera NSAs

The simplest host identifier is an IP address or host name.

Syntax for specifying the host identifier of a Centera unit:

```
<IP_Address>[:<port>]|<Host_Name>[:<port>]
```

Note: In the syntax above, the pipe symbol (|) is the meta-syntax character that separates mutually exclusive options.

Examples

```
centera://centera1.cascommunity.org
centera://10.2.3.4
```

You can specify multiple Centera host identifiers in a single NSA. Specifying multiple host identifiers allows use, even when one or more Centera systems are unavailable.

Minimal syntax for a Centera NSA:

```
centera://<Host_Identifier>[,<Host_Identifier>[,<Host_Identifier>[...]]]
```

Example

```
centera://10.2.3.4,10.1.7.6
```

Specifying the PEA File in Centera NSAs

In addition to specifying the host identifier, you can specify the location of the Centera Pool Entry Authorization (PEA) file on your EDW hosts. If using authorized access (application profiles) on the Centera unit, you must:

- Copy the PEA file to all hosts of your EDW instance; use the same path structure on each host.
- Specify the path to the PEA file in your NSA definition; precede the path with a question mark (?).
- Enclose the entire NSA specification within quotation marks.

  Syntax for specifying the location of the Centera PEA File:

  "centera://<*Host_Identifier*>?<*path_to_PEA_file_on_EDW_hosts*>"

Examples

```
atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
--password=changeme setnsi \
"centera://10.2.3.4,10.006.07.8?/opt/sensage/etc/nss/myProfile.pea"

atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
--password=changeme setnsi\
"centera://centera1.cascommunity.org?/opt/sensage/etc/nss/myProfile.pea"
```

Specifying the Username and Secret for the PAI Module in Centera NSAs

To specify username/secret for the PAI module, you must:

- Include name/value pair specifications for the username and password; separate them with a comma.
- Precede the name/value pair specification with a question mark (?).
- Enclose the entire NSA specification within quotation marks. Syntax for specifying the username and secret for a Centera unit:

  "centera://<*Host_Identifier*>?name=<*username*>,secret=<*password*>"

Note: EMC may request a user name and password for the PEA file. SenSage AP does not require a specific user name or password. Use relevant names and passwords for the Centera device.

Examples

```
atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
--password=changeme \
setnsi "centera://10.2.3.4?name=s0m3User,secret=s0m3Passwd"

atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
--password=changeme \
setnsi "centera://centera1.cascommunity.org?name=s0m3User,secret=s0m3Passwd"
```

Note: You also can assign multiple profiles on a connection string to access one or more clusters. For more information on PAI modules and the syntax of connection strings, refer to specifying parameters to `FPPool_Open` in the *Centera Programmer's Guide*.

Specifying Data Retention in Centera NSAs

You can configure Centera NSAs with a retention period. Setting a retention period requires SenSage AP to retain references to the archived data in the EDW and the data itself on the Centera unit until the retention period passes and you retire the data from the EDW. Only after the retention period has passed can you retire data from the EDW and the NLS device. You must run atquery with the retire command on the EDW table to remove the archived data from the EDW and the Centera unit. For more information, see Retiring Data on page 115.

Note: When you retire archived data that is not under retention, SenSage AP removes the references to the data from the EDW and removes the data itself from the Centera unit.

There are two ways to specify the retention period: retentionPeriod and retentionClass. Each of these is preceded by a pipe symbol (|).

- retentionPeriod—basic syntax

  `|retentionPeriod=<num_seconds>`

  Syntax for specifying the retention period for a Centera unit:

  `"centera://<Host_Identifier>|retentionPeriod=<num_seconds>"`

  Examples

  ```
  atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
  --password=changeme \
  setnsi "centera://centera1.cascommunity.org|retentionPeriod=31536000"

  atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
  --password=changeme \
  setnsi "centera://10.2.3.4,10.006.07.8|retentionPeriod=31536000"
  ```

Note:

- When you specify a retention period of some specific number of seconds, you ensure that data archived with that NSI is under a fixed length of retention. The data will be available for deletion when that number of seconds has passed.
- After data has been archived with a retention period of a fixed number of seconds, the retention period cannot be changed on the Centera. You can only change the retention period by deleting and re- defining the target. Use `deletensi` to delete the existing one and `setnsi` to define a new one. For more information, see Defining and Managing Nearline Targets: Overview on page 184.
- The number of seconds you specify will be a very large number. For example, 86400 seconds represent a single day.

- retentionClass—basic syntax

  `|retentionClass=<class_name>`

  Syntax for specifying a retention class for a Centera unit:

  `"centera://<Host_Identifier>]|retentionClass=<class_name>"`

  Examples

```
atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
--password=changeme \
setnsi "centera://centera1.cascommunity.org|retentionClass=IgniteTech"

atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
--password=changeme \
setnsi "centera://10.2.3.4,10.006.07.8|retentionClass=IgniteTech"
```

Usage

- The value you specify for `<class_name>` is provided by a Centera administrator. The class name specifies the retention period as defined by the administrator.
- When you specify a retention class on the Centera, the archived data is associated with a named retention class on the device. Depending on the model of the Centera unit (as illustrated in the table below), a retention class can be modified to represent a longer or shorter period of time. The changes are retroactive, so that a change to a retention class affects not only new data to be archived, but existing data archived with that retention class.

Note: You must restart the nss for a change to retention class to become effective.

- The Event Data Warehouse (EDW) sets the expiration timestamp in the datastore at the time of archive, based upon either the retention class value returned or the retention period specified. The EDW is unaware of any retrospective changes to the underlying Centera retention class. If the retention period in the original class is shortened, then the retire operation will still fail since ithis operation uses the expiration timestamp calculated at archive time. Similarly, if the retention period in the class was extended and the EDW attempts to delete the data from Centera, the delete operation will fail and the data remains in storage.
- The value you specify for `<class_name>` cannot be changed. The Centera administrator can change the underlying retention period that the class defines. If you need to change the class name instead,

delete and re-define the target. Use `deletensi` to delete the existing one and `setnsi` to define a new one. For more information, see Defining and Managing Nearline Targets: Overview on page 184.

- When you specify data retention in a Centera NSA, enclose the entire NSA within quotation marks.
- The value you specify for a Centera NSA depends on the configuration of your system for failover, replication, and compliance mode, and your profile definitions. To determine how to specify the Centera NSA, consult the EMC Centera documentation that describes the parameters for the `FPPool_Open` call used by their API.
- If you require different retention periods for different data sources, create a separate NSA for each desired retention period.

Important: Always specify `retentionPeriod` or `retentionClass` on Centera. Do not specify the default. Doing so, for exmaple, could cause undesired retention of test data. Although certain models of Centera usually or always define a default retention period., if you archive data without specifying a retention class or period, the data may or may not already be under some form of retention when you archive it. Therefore, SenSage AP recommends that you ***always*** specify either a retention period or retention class (even if the class specifies no retention period).

The table below illustrates the various Centera models and how they interpret retention classes and default retention:

Centera Package Installed	Default Retention Period Required?	Default Retention Period	Allowed Changes to Retention Class
Basic			Increase or Decrease
Governance			Increase or Decrease
Compliance Edition +			Only Increase

Configuring a Centera Unit for Single-Instance Storage

The Centera provides multiple ways to generate unique IDs (references) that identify the event-log data archived to it. For optimal storage capacity, IgniteTech recommends that you configure your Centera unit for single-instance storage; EDW saves two copies of all event-log data (primary and secondary), using single-instance storage cuts data storage in half.

Note: Configuring your SenSage AP installation such that multiple inserts to the same table occur at the same time reduces the benefit of single-instance storage for data loaded simultaneously to the same table.

Specifying NSAs for Remote File Systems

When you specify the NSA for a remote file system, precede the network storage address with:

`directory://`

The complete NSA syntax for a remote file system is:

`directory://< local-NFS-mount-point >`

Substitute *<local-NFS-mount-point>* with the local mount point that you will use on all hosts in the EDW instance.

Example

```
directory:///nss
```

Generally, you set up one remote host as the Nearline target for all EDW hosts when you use remote file systems for Nearline storage.

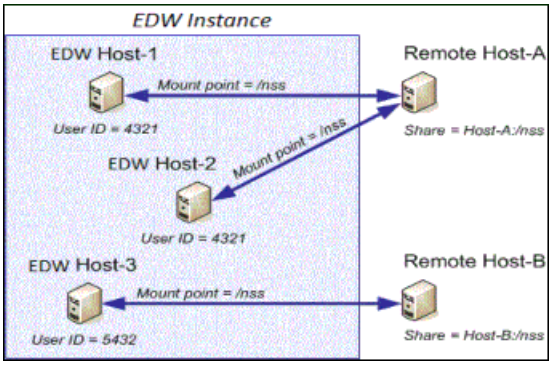
Note: Before you mount your NFS mountpoint as a hard or soft mount, consider the following:

- **hard mount**—If the connection to the remote file system breaks, the NFS server hangs indefinitely while it waits for the IO request to complete. When the NFS server hangs, the archive process also hangs while it waits for the remote file system to return online. This situation does not cause other operations to hang but it prevents the archive operation from completing and logging an error.
- **soft mount**—If the connection to the remote file system fails, the NFS server fails all IO requests, which causes the EDW to fail the archive operation.

CONFIGURING NFS NEARLINE STORAGE FOR SENSAGE AP SYSTEM USERS WITH DIFFERENT IDS

The SenSage AP system user is known on remote file systems by the numeric user ID, not by the alphanumeric user name. This can cause a problem for Nearline storage if the numeric user IDs for the SenSage AP system user are different on different EDW hosts. In this situation, you must create a different NFS share for each of the user IDs assigned to the SenSage AP system user on the EDW hosts.

For example, assume you have a 3-host EDW instance. The SenSage AP system user has "`4321`" as the user ID on two of the hosts; the user ID is "`5432`" on the other host.



All three hosts use the same local mount point, "`/nss`". The two hosts where the user ID is "`4321`" mount the share "`Host-A:/nss`"; the other host where the user ID is "`5432`" mounts the share "`Host-B:/nss`". Alternate shares do not need to be on separate hosts; you can export and share different directories from the same remote host.

As an alternative, you can change the user IDs for the SenSage AP system user to be the same on all EDW hosts.

Archiving of Local Data to Nearline Storage

You can archive the local data in a table to Nearline storage with the `archivetbl` subcommand of the `atmanage` utility. You can archive the entire table, or you can limit the archiving to a specific time frame. You will need to run this command every time you want to archive data. You may want to write a script that calculates the maximum and minimum timestamp parameters required by the command, and then schedule that script to run on a regular schedule using a scheduling utility such as the Linux `cron` utility.

Syntax

```
atmanage <cluster_list> --user=<name> --password=<pass> \
--namespace=<namespace> archivetbl <table_name> <NSI> \
[<max_timestamp> [<min_timestamp>]]
```

where:

- No time stamps—specifies archiving the entire table.
- One time stamp—specifies archiving data that is older than the time stamp.
- Two time stamps—specifies a time frame; the first time stamp represents the latest or maximum timestamp, followed by the earliest or minimum time stamp.
- Timestamps should be in ISO8601 date format (YYYY-MM-DDTHH:MM:SS)

Note: Some data older than the cutoff timestamp may not be archived immediately, due to the how the EDW stores data. A record of the archived data displays after the archive command has completed.

Tip: To determine the names of tables to archive, run the following command to list tables in the EDW:

```
atview <host:port> tables --namespace=<namespace> --user=<user> --
pass=<password>
```

For more information, see Listing Tables on page 129 .

Examples

The following examples pertain to an EMC Centera device, but apply to all SenSage AP Nearline storage devices.

The following command archives the "`Firewall`" table to the NSI identified by "`centera-1`" up to January 1, 2007, midnight.

```
atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
--password=changeme --namespace=documentation \
archivetbl Firewall centera-1 2007-01-01T00:00:00
```

The following command archives the "Firewall" table to the NSI identified by "centera1" between January 1, 2006, midnight and January 1, 2007, midnight.

```
atmanage "edw01:8072,edw02:8072,edw03:8072" --user=administrator \
  --password=changeme --namespace=documentation \
  archivetbl Firewall centera-1 2007-01-01T00:00:00 2006-01-01T00:00:00
```

It is possible that not all data will be archived immediately. SenSage AP moves data to Nearline storage only when all data in the same directory meets the age requirements.

10

Troubleshooting

This chapter includes these sections:

Checking Space for the Data Store

To determine the space and inode (i-number) usage for data store, use the `ds_stat` utility. Run the `ds_stat` command to make estimates and projections of disk space usage on the EDW.

`ds_stat { <file> | <directory> }`

The `ds_stat` utility takes a single parameter, which is the name of a directory or the name of a file. If the argument is a file, the utility generates disk usage statistics for that file. If the argument is a directory, utility generates a summary of the disk usage statistics for the entire directory tree.

The statistics that `ds_stat` generates are:

- Total inodes used
- Total file size
- Total disk blocks used

- Total disk space used
- Ration of disk space to inodes

Checking Locking Status and IPC State Information

Run the showlocks command to display a list of all ACTIVE and PENDING locks plus additional IPC-related state information associated with the indicated data store.

```
showlocks { --delete-ipc-data } [dsroot]
```

Use the `--delete-ipc-data` switch only if it is determined that a process has crashed and left the IPC-state information in a corrupted state. This switch is meant for use as a last-resort solution.

Monitoring for Inconsistent Loads

IgniteTech recommends that you regularly query the `system.upload_info` table for loads with inconsistent data. Data is inconsistent when it is not replicated correctly across all hosts in an EDW instance. Rows with the value `false` in the CONSISTENT column identify loads with inconsistent data.

When you discover loads with inconsistent data through a query of `system.upload_info`, run a separate query of the `system.raw_upload_info` table and capture the results in a file. Then, contact IgniteTech Support for assistance.

For more information on the system tables, see "system.upload_info" and “system.raw_upload_info”, in the *Event Data Warehouse Guide*.

Handling Rendezvous Timeouts

If you see a “rendezvous timeout” error, contact Technical Support for assistance.

Improving Query Performance

This section documents three options for improving query performance:

- Generating a SQL Query Plan for Query Tuning on page 195
- Enhancing Performance for Specific TOP Queries on Large Clusters on page 196
- Using Best Practices when Writing AP SQL Queries on page 197

Generating a SQL Query Plan for Query Tuning

A query plan, or query execution plan, describes how the EDW Query Engine plans to run a given query. This information is useful primarily to trouble-shoot a query that is running too slowly. A knowledgeable database administrator can use the plan to tune query performance.

Important: This section describes how to generate a query plan. However, because interpreting a query plan and using it to tune performance requires sophisticated EDW Query Engine knowledge, IgniteTech does not expect its system administrators to interpret these plans directly. The instructions in this section are provided only to make it easier for you to generate a plan that you can deliver to Technical Support when query performance would benefit from improvement.

SenSage AP SQL provides the EXPLAIN extension to enable you to create a query plan. The syntax for this extension is:

```
EXPLAIN <any_select_statement>
```

When you precede the SELECT keyword with the EXPLAIN keyword for any valid query, the EDW Query Engine generates a query plan instead of actually running the query. You can generate the plan for a query that you run on the command line or from a file.

The following query example selects four columns from a table named test between May, 2008 and June, 2009. It returns only the first 40 rows:

```
SELECT top 40 Host, ClientIP, Get, Code
  FROM test
  WHERE ClientIP = '167.29.33.430'
  DURING _time('2008-05-01T10:11:00'),_time('2009-06-01T10:11:00');
```

To return the query plan rather than the data for this query, you would modify the query as follows:

```
EXPLAIN SELECT top 40 Host, ClientIP, Get, Code
  FROM test
  WHERE ClientIP = '167.29.33.430'
  DURING _time('2008-05-01T10:11:00'),_time('2009-06-01T10:11:00');
```

The query above highlights the single change in **boldface** type.

Assume you run this query directly from the command line with a command like the following:

```
atquery --user=administrator --pass=changeme --namespace=default localhost:8072\
-e "EXPLAIN SELECT top 40 Host, ClientIP, Get, Code FROM test\
WHERE ClientIP = '167.29.33.430'\
DURING _time('2008-05-01T10:11:00'),_time('2009-06-01T10:11:00');
```

The EDW engine returns the query plan as a single varchar column. The output includes rows of dashed lines. To return only the query plan without the dashed lines, add the following `atquery` flags:

```
--format=tsv --verbose=0
```

The `--verbose=0` option forces the EDWengine to return only the query plan. The `--format=tsv` option forces the varchar field to display as tab-separated values. The varchar column contains several new lines, all represented by the `\n` character.

Note: The tsv format is preferable to the csv (comma-separated value) or psv (pipe-separated value) formats. The varchar output that represents the query plan contains no tabs, but does contain commas and could contain pipes. Therefore, csv-formatted output would include escaped commas and psv-formatted output could include escaped pipes. The most easily readable format is tsv.

To make the query plan more readable, you can run the EDW output through the Unix sed (stream editor) utility. This utility reads the text input, applies formatting that you specify, and outputs the formatted text. Add the following formatting to the `sed` utility to display the query plan in text appropriately broken into separate lines:

```
sed 's/\\n/\n/g'
```

As a final step, redirect the query plan into a text file that you can share with Technical Support.

In other words, to create a readable query plan in its own text file for the example query above, you could run the following:

```
atquery --user=administrator --pass=changeme --namespace=default --format=tsv\
--verbose=0 localhost:8072 -e "EXPLAIN SELECT top 40 Host, ClientIP, Get, Code\
FROM test WHERE ClientIP = '167.29.33.430'
DURING _time('2008-05-01T10:11:00'),_time('2009-06-01T10:11:00');"
| sed 's/\\n/\n/g' > plan.out
```

Enhancing Performance for Specific TOP Queries on Large Clusters

When a query that runs on a large cluster includes both the `TOP` or `FIRST` clause and an `ORDER BY`, `GROUP BY`, or `DISTINCT` clause, the EDW Query Engine must retrieve all relevant records from all hosts in the cluster before it can sort the data and return the specified top records.

However, run the same query without an `ORDER BY`, `GROUP BY`, or `DISTINCT` clause and performance improves significantly. Because such a query requires no sorting, the EDW Query Engine stops scanning nodes and terminates the query after it retrieves the number of records specified in the `TOP` or FIRST clause.

Important:

- The value you specify in the `TOP` or `FIRST` clause should exceed the number of rows returned.

  In other words, a query that specifies `TOP 1000` but returns fewer than 1000 rows does not benefit from this performance recommendation.

- Performance benefits are greatest when the query runs on a multi-node cluster. Do not expect performance enhancement if you run the query on a 1-node cluster.

Setting the TopQueryFastExit Flag

Above and beyond the performance improvement gained by running a `TOP` query without sorting the data, you may also be able to improve query performance by enabling the "EDW top query fast exist" field in the Advanced EDW section of the SenSage AP Service Configs tab of Deployment Manager.

Although the EDW Query Engine automatically stops processing an unordered `TOP` query as soon as it retrieves the number of rows specified in the `TOP` clause; enabling the `TopQueryFastExit` may further improve performance.

Set `TopQueryFastExit` to 1 to cause an unordered `TOP` query to stop processing as soon as the EDW retrieves the number of rows specified in the `TOP` clause. You must set this field in the `athttpd.conf` file on every node in the cluster and restart the EDW for the change to take effect.

By default the flag is set to 0 and the feature is not enabled.

Note: Although the EDW log file contains broken pipe error messages when you enable `TopQueryFastExit`, these messages indicate no actual problem. The only issue is that they clutter the log file. If running the query without enabling this flag performs sufficiently, there is no need to enable this flag as well.

Using Best Practices when Writing AP SQL Queries

Follow these guidelines for enhanced query performance when writing AP SQL queries and subqueries:

- In a target list when returning columns, select a subset of columns rather than all columns. Selecting fewer columns increases performance.
- In data processing expressions, instead of the LIKE operator use the _strstr() function as in, _strstr(<COL>,<VAL>) instead of <COL> like %<VAL>%, when specifying values.

  The following WHERE clause example, uses the _strstr() function instead of the LIKE operator to maximize performance:

  ```
  WHERE
  _strstr(log_type,$EVENT_SOURCE) >-1
  and
  event_description = $EVENT_DESCRIP
  ```

  For more information, see the section on **_strstr()** in the *Event Data Warehouse Guide*.
- For performance and memory usage reasons, don't use the ORDER BY clause in the first subquery when writing queries that include more than one subquery. Also, use ORDER BY in a separate subquery after any UNION ALL queries or subqueries.

A

Log Files

The following sections document log files generated by SenSage AP components. The files are grouped alphabetically by components. This section documents the following log file components:

Analyzer

Log File Name	Category	Purpose	Log Rotate	PTL	Notes
`/var/log/tomcat/analyzer.log`	Debug, Monitoring	ERR MON	N		
`/var/log/tomcat/audit.log`	Debug, Monitoring	AUDIT	N		
`/var/log/tomcat/catalina.out`	Debug, Monitoring	ERR MON	N		

Log File Name	Category	Purpose	Log Rotate	PTL	Notes
/opt/hexis/tomcat/host-manager.log	Debug, Monitoring	ERR MON	N		
/opt/hexis/tomcat/localhost.log					
/opt/hexis/tomcat/localhost_access.log					
/opt/hexis/tomcat/manager.log					

Analytics Installer

Log File Name	Category	Purpose	Log Rotate	PTL	Notes
/opt/hexis/hawkeye_ap/var/log/analytics/analytics_installer.log	Debug, Monitoring	ERR MON	N	N	log file for the Analytics installer

Atpgsql

Log File Name	Category	Purpose	Log Rotate	PTL	Notes
/opt/hexis/hawkeye-ap/var/log/atpgsql/run.log	Debug, Monitoring		N		loading the database

Atslapd

Log File Name	Category	Purpose	Log Rotate	PTL	Notes
/opt/hexis/hawkeye-ap/var/log/atslapd/atslapd-<date>.log	Debug, Monitoring	ERR MON	Y		ATSLAPD message **Note:** Rotated by syslog-ng; (see /etc/logrotate.d/syslog-ng)

Collector

Log File Name	Category	Purpose	Log Rotate	PTL	Notes
/opt/hexis/hawkeye-ap/var/log/collector/collector.output		Debug, Monitoring	N		
	Debug, Monitoring	ERR MON	N		Stderror prints from the activity log write process
	Debug, Monitoring		N		Deprecated

Log File Name	Category	Purpose	Log Rotate	PTL	Notes
/opt/hexis/hawkeye-ap/var/log/collector/transaction.log	Monitoring	AUDIT	N	Y	All collector transactions can be loaded into the EDW for self audit
	Debug, Monitoring	ERR MON	N		Parse failers for loaders. These are defined in the config.xml
/opt/hexis/hawkeye-ap/var/log/collector/windowretriever*. log	Debug, Monitoring	ERR MON	Not by default; can be set to rotate		Error and debugging information for the windows retriever

Nearline Storage

Log File Name	Category	Purpose	Log Rotate	PTL	Notes
/opt/hexis/hawkeye-ap/var/log/nss/init.log	Debug, Monitoring	ERR MON	N		
/opt/hexis/hawkeye-ap/var/log/nss/nss_activity.log	Debug, Monitoring	AUDIT	N		Activity log for the Nearline storage server
/opt/hexis/hawkeye-ap/var/log/nss/nss_error.log	Debug, Monitoring	ERR MON	N		nss error log
/opt/hexis/hawkeye-ap/var/log/nss/nss_stderr.log	Debug, Monitoring	ERR MON	N		nss stderr log

SenSage AP

There are no logs for SenSage AP. See the Analyzer for SenSage AP logging.

EDW

Log File Name	Category	Purpose	Log Rotate	PTL	Notes
`/var/log/messages`	Debug, Monitoring	ERR MON	Y		"EDW Errors (default location of RedHat syslog data) **Note:** • To capture all EDW data (debug and info), change the syslog configuration file to load all Local 0 facility data to a named file. • Rotated by logrotate; see /etc/logrotate.d/syslog-ng
`/opt/hexis/hawkeye-ap/var/log/sls/<instance_name>/athttpd_log`	Debug, Monitoring	AUDIT	N		Log of httpd communications. Mostly POST traffic
`/opt/hexis/hawkeye-ap/var/log/sls/<instance_name>/stdout_log`	Debug, Monitoring		N		stdout statements
`/opt/hexis/hawkeye-ap/var/log/sls/<instance_name>/error_log`	Debug, Monitoring	ERR MON	N		errors recorded by the EDW instance

B

Error Codes

This appendix documents error messages raised when you use SenSage AP. The messages are grouped by component and, within each component, by category and code number. This chapter documents error messages for the following components:

Collector

Error Code	Message
B1000	Unable to find configuration file %s (%s)
B1001	Unable to read configuration file %s (%s)
B1002	Unable to create file '%s' (%s)
B1003	Unable to delete file '%s' (%s)
B1004	Unable to rename file '%s' to '%s' (%s)
B1005	Unable to read file '%s' (%s)
B1006	Unable to append to file '%s' (%s)
B1007	Unable to link '%s' to '%s' (%s)
B1008	Unable to read directory '%s' (%s)

Error Code	Message
B1009	Unable to fork child process (%s)
B1010	Exec of '%s' failed (%s)
B1011	Unable to create directory '%s' (%s)
B1012	Error writing stream '%s' (%s)
B1013	Error reading stream '%s' (%s)
B1014	Socketpair failed (%s)
B1015	Program exiting due to thrown exception
B1016	Can't create socket '%s'->'%s' (%s)
B1017	Can't open SMTP server '%s' (%s)
B1018	SMTP server fail on '%s'
B1019	Unable to find configuration DTD file in directory %s
B1200	Found %s nodes matching xpath='%s' expected exactly 1
B1201	DTD parse failed file='%s' error=[%s]
B1202	XML parse failed file='%s' error=[%s], backup config file may be available in '%s'
B1203	DTD validation failed during load xml='%s' dtd='%s' error=[%s], backup config file may be available in '%s'
B1204	DTD validation failed during save xml='%s' dtd='%s' error=[%s]
B1205	XML Node not found xpath='%s'
B1206	XML string parse failed error=[%s] string=[%s]
B1207	XML empty element '%s', backup config file may be available in '%s'
B1208	Expected DTD validation to fail on save, it didn't
B1209	Expected DTD validation to fail on load, it didn't
B1300	Time range Bad value '%s'
B1301	Time range expected 0-%s but got %s
B1501	Bad token '%s' as array delimiter
B1502	Bad hash name/value seperator expected ':' got '%s'
B1503	Bad token '%s' as hash delimiter
B1504	Unexpected token '%s'
B1505	Residuals tokens %s != %s
B1506	Test failed, unexpected result
B1507	Can't set more binary '%s'
B1601	Test of '%s' failed, expected throw of error code '%s' got no error
B1700	Failed to create SSL_CTX (%s) [%s]
B1701	Failed to set CTX options [%s]
B1702	Failed to create SSL (%s) [%s]

Error Code	Message
B1703	Failed to connect SSL [%s]
B1704	Failed to read private key from '%s' (%s)
B1705	Failed to read certificate from '%s' (%s)
B1706	Failed to accept SSL [%s]
B1800	Unmapped log field name(s) '%s'
B1900	Months and/or years not supported in interval (%s)
B1901	Interval %s seconds is less than %s in (%s)
B1902	Interval %s seconds is greater than %s in (%s)
C1	Unexpected Exception (%s) in file %s at line %s
C1020	Unable to create PID file %s
C1021	Unable to delete PID file %s
C1022	Unable to daemonize Collector. Exiting.
C1023	I/O Error during Controller startup (%s)
C1024	Could not stop running Collector (PID %s) during startup
C1025	Collector did not remove PID file %s upon exit
C1026	Could not find PID file %s
C1027	Could not reap PID %s after trying for %s seconds
C1100	Failed to backout load from SLS instance='%s' uploadId='%s' exitCode='%s'
C1101	Preprocess failure prog='%s' file='%s' (%s)
C1102	Load failed prior to uploadId instance='%s' file='%s'
C1103	Load failed after uploadId instance='%s' file='%s' exitValue='%s' uploadId='%s'
C1104	Parse Failure instance='%s' file='%s' uploadId='%s' line=[%s]
C1105	Unable to parse time range '%s'
C1106	Loader stopped due to fatal error text=%s
C1107	Timeout waiting for sub-process, terminating
C1108	No PTL found which matchs file name '%s'
C1109	Load failed before 'final status' instance='%s' file='%s' uploadId='%s'
C1110	Load failed instance='%s' file='%s' uploadId='%s' with 'final status... %s'
C1111	Load failed broken pipe instance='%s' file='%s' uploadId='%s'
C1112	Load failed bad exit status instance='%s' file='%s' uploadId='%s' status='%s'
C1113	Load failed exit status 10 (double TERM or INT) instance='%s' file='%s' uploadId='%s' status='%s'
C1114	Load failed exit status 15 (insufficient space on server) instance='%s' file='%s' uploadId='%s' status='%s'

Error Code	Message
C1115	Load failed exit status 20 (bad command line) instance='%s' file='%s' uploadId='%s' status='%s'
C1116	Load failed exit status 98 (failed to contact SLS) instance='%s' file='%s' uploadId='%s' status='%s'
C1117	Load failed exit status 99 (internal error) instance='%s' file='%s' uploadId='%s' status='%s'
C1118	Tar for daisy chain returned bad exit value '%s'
C1119	Webui instance '%s' has no config.dat file
C1120	Bad or missing meta data file '%s'
C1121	Loader '%s' already running ($!)
C1400	Retriever stopped due to fatal error name=%s text=%s
C1401	Preprocessor exited with bad status on preprocessor='%s' file='%s' status='%s'
C1402	No retriever type named '%s'
C1403	Backup to '%s' failed status (%s), will retry in '%s' seconds.
C1404	Bad backup url '%s'
C1405	Can't load .meta file '%s' (%s)
C1406	Retriever name must contain alpha numeric and '_' '-' characters only '%s'
C1407	Bad exit status for md5sum command (%s)
C1408	No output from md5sum command
C1409	Bad output from md5sum command (%s)
C1410	Interval '%s' is less than 1 second
C1411	Months and years not supported in interval '%s'
C1412	FTP connection to host '%s' failed to put file '%s'
C1413	Size mismatch after download of '%s', %s != %s
C1414	No LogQueue specified for retriever '%s'
C1415	Retriever '%s' already running ($!)
C1416	Preprocessor failed to generate output file preprocessor='%s' file='%s'
C1417	Gap in queue probability coverage at '%s' in retriever '%s'
C1418	Unexpected failure to place file '%s' in queue
C1419	Low and high attribute on LogQueue must be from 0-99 in retriever '%s'
C1420	Illegal retriever name '%s'
C1500	FTP connection to host '%s' failed with message '%s'
C1501	FTP connection to host '%s' failed on login to '%s'
C1502	FTP connection to host '%s' failed on cwd to '%s'
C1503	FTP connection to host '%s' failed on data transfer
C1504	FTP connection to host '%s' failed to delete file '%s'

Error Code	Message
C1505	FTP connection to host '%s' failed to create file '%s'
C1506	FTP connection to host '%s' failed on nlist
C1507	FTP connection to host '%s' failed on pwd
C1508	FTP connection to host '%s' failed on get '%s'
C1509	FTP connection to host '%s' failed to set mode binary
C1600	SFTP connection to '%s' failed (%s)
C1601	SFTP ls of '%s' failed (%s)
C1602	SFTP get of '%s' failed (%s)
C1603	SFTP rm of '%s' failed (%s)
C1604	SFTP create of '%s' failed (%s)
C1605	SFTP connection lost (%s)
C1606	SFTP bad server version (%s)
C1607	SFTP sftp client exit pid: %s, exit code: %s
C1608	SFTP put of '%s' failed (%s)
C1609	SFTP unparseable line from ls '%s'
C1610	SFTP unable to find ssh in path
C1611	SFTP unable to cd to '%s'
C1612	SFTP unable to parse directory line '%s'
C1700	Bad request '%s' only PUT supported
C1701	Bad request '%s' expect HTTP/1.x
C1702	Can't parse request '%s'
C1703	Can't listen on '%s'
C1704	Missing Content-Length: header on '%s'
C1705	Oversized body %s > %s
C1706	Premature socket EOF or error '%s' in %s
C1707	Can't parse file record '%s', on '%s'
C1708	User/passwd required and not provided
C1801	Can't parse server status line '%s'
C1802	Unexpected EOF in server data
C1803	Server reports error (%s %s %s)
C1804	Unable to parse XML reply from server (%s)
C1805	Server reports error detail (%s)
C1806	Failed to load SSL libraries (%s)
C2000	Could not start Controller

Error Code	Message
C2001	Could not create Log Queue at %s
C2002	Could not read Log Queue %s. Removing from rotation.
C2003	Could not write Log Queue %s. Removing from rotation.
C2004	Could not execute Log Queue %s. Removing from rotation.
C2005	%s MB free space on %s Log Queue is below minimum %s MB. Disabling %s.
C2006	Could not disable Log Queue %s.
C2007	Could not enable Log Queue %s
C2008	Could not find PID of group %s in state. Could not stop it.
C2009	Could not start Loader %s because it is using the same Log Queue %s as Loader %s
C2010	User %s does not exist. Exiting.
C2011	No group id for user %s. Exiting.
C2012	Failed to change UID to %s (now: %s, %s)
C2013	Failed to change GID to %s (now: %s, %s)
C2014	Log Queue Root dir %s does not exist
C2015	Collector/FileRoot is not defined
C2016	FileRoot is not configured correctly. %s does not exist.
C2017	Collector/StateRoot is not defined
C2018	StateRoot is not configured correctly. %s does not exist.
C2019	A Shepherd has disappeared and will not be respawned. May have exited on error or been killed. (name: %s)
C2020	A Loader has disappeared and will not be respawned. May have exited on error or been killed. (name: %s)
C2100	Could not open dashboard file %s (%s)
C2200	Could not create state directory %s
C2201	General error when creating state dir (%s)
C2202	Could not instantiate DB File %s
C2203	Could not optimize statefile %s
C2301	SCP No SourceFile entries specified for '%s'
C2302	SCP non zero exit value from scp downloading '%s'
C2303	SCP no files retrieved '%s'
C2304	SCP STDERR: %s
C2400	WWW IP for host '%s' not found
C2401	WWW Failed to retrieve '%s'
C2402	WWW Body size mismatch Content-Length: %s != downloaded: %s
C2403	WWW No content length in headers

Error Code	Message
C2404	WWW Can't load SSLStream, open SSL possibly not installed? (%s)
C2405	WWW Unparseable status line from server (%s)
C2406	WWW bad status from server (%s)
C2501	Can't use Schedule and Period at the same time in '%s'
C2601	Host Glob of SourceHost '%s' results in 0 results
C2701	Daisy Chain untar of '%s' to '%s' failed exit code '%s'
C2702	Daisy Chain meta and logfile pair are bad, log: '%s' meta: '%s'
C2703	Daisy Chain Got '%s' files in tar file, expected 2
C2801	LEA Can't parse log line (%s) from '%s'
C2802	LEA return code: %s
C2803	LEA debug output: %s
C2804	LEA unsafe cma name '%s'
C2805	LEA Can't parse log line (%s)
C2806	LEA bad mode (%s)
C2900	Unable to find template variable '%s' in replacement hash
C2901	No such test '%s'
C3000	Unable to connect to '%s' ('%s')
C3001	Query failed: %s (SQL: %s)
C3002	Database fetch failed: %s
C3003	No state value for column '%s', and no default exists
C3004	No StateCol definition for parameter '%s'
C3005	No SQL query defined
C3006	State Column '%s' is missing from the SQL query
C3007	Received result type '%d', expected 4040

EDW

The errors listed below represent approximately one quarter of the exceptions issued by the EDW.

Error Code	Message
0010004	Shim not implemented
0010012	FILE* is NULL
0010014	Parameter name is NULL
0010019	FILE* for output is NULL

Error Code	Message
0010020	FILE* for output is NULL
0010022	malformed bucket chain in Tcl_DeleteHashEntry
0010023	called Tcl_FindHashEntry on deleted table
0010024	Name is NULL
0010025	Value is NULL
0010026	called Tcl_CreateHashEntry on deleted table
0010042	Attempting to process data with no schema
0010043	Column Index out of bounds
0010053	Unknown P_UNIT_T
0010057	Unknown Data Type: $1
0010058	Internal Logic Error: Please contact the Development Team
0010062	HTTP data ended before complete content was received
0010063	Stream has received data of an unknown type
0010064	HTTP data ended before complete HTTP header was received
0010066	Errors while reading file
0010067	Unknown Data Type: $1
0010071	Problems compiling statement
0010072	pullDataSource should never be called for CPushedValueStream
0010073	‘processHttpData’ called after ‘getLeftOverData’
0010074	Shim not implemented
0010088	EOS has already been sent
0010089	EOS has already been sent
0010091	EOS has already been sent
0010093	EOS has already been sent
0010094	EOS has already been sent
0010098	Shim not implemented
0010102	Child Process Died
0010124	Internal Error: Unknown parsing state
0010134	Failed to write all the Sort Data
0010140	Multiple Schema Specifiers in Stream
0010141	Incorrect Data Type Requested
0010142	Unknown Internal State
0010158	NULL Byte found in InitialData
0010159	NULL Byte found in InitialData

Error Code	Message
0010168	Unexpected error code: $1
0010176	Unknown STATE Code
0010178	Invalid !DOCTYPE block
0010179	Invalid !ELEMENT block
0010184	Unknown Parameter Type in Table
0010186	Argument is not an XMLRPC Fault Response
0010187	Argument is not a standardized XMLRPC Fault Response
0010210	No Such Channel: $1
0010216	Invalid length on hexadecimal string: $1
0010217	Non-hexadecimal digit: $1
0010230	Non-hexadecimal digit: $1
0010241	Too many fields in input data
0010242	Too few fields in input data
0010243	Multiple !DOCTYPE's in document
0010252	Corrupted BZIP2 data stream
0010260	Unknown Data Type Code
0010261	XMLRPC_BINARY Not Implemented Yet
0010262	No Regex found
0010265	Shim not implemented
0010268	Shim not implemented
0010274	Couldn't find chunk-size in stdin stream
0010284	Invalid Regex Type: $1
0010286	All Network Communication Processes have died
0010290	Internal Error: Received incomplete rows
0010292	'parseData' called after 'getLeftOverData'
0010333	Method should never be called
0010338	'start' called on an uninitialized object
0010339	'init' called on a non-empty object
0010340	Shim not implemented
0010347	Internal Error: Stream Iter is Empty
0010353	Stream Iter is Empty
0010355	Stream Iter is Empty
0010361	Shim not implemented
0010375	Invalid Regex Type: $1

Error Code	Message
0010383	Shim not implemented
0010385	Shim not implemented
0010386	Empty Passwords are not allowed
0010398	Unknown Permission Type: $1
0010399	The ‘$1’ role cannot be deleted
0010400	The ‘$1’ user cannot be deleted
0010422	Index out of bounds
0010427	The ‘$1’ user cannot be deleted
0010453	Read did not fetch an integral number of records
0010455	Encountered EOF while reading Data File
0010456	Unable to read complete row from Data File
0010459	Internal Logic Error: Please contact the Development Team
0010467	The ‘$1’ user cannot be disabled
0010468	The ‘$1’ role cannot be disabled
0010469	The ‘$1’ user cannot be removed from the ‘$2’ role
0010470	The ‘$1’ role cannot be deleted
0010478	Internal Error: Child sent a METADATA packet
0010479	Internal Error: Child sent a SCHEMA packet
0010497	The ‘$1’ permission cannot be removed from the ‘$2’ role
0010498	The ‘$1’ permission cannot be removed from the ‘$2’ role
0010499	Unable to send WAIT message to Application Server
0010526	Internal Logic Error: Please contact the Development Team
0010528	Problems compiling statement
0010532	Invalid SKIP-QUEUE Credentials supplied
0010547	Invalid language/country specifier for locale: $1/$2
0010548	Invalid UTF-8 Encoding in ‘$1’ at position $2
0010549	Incomplete UTF-8 Encoding in ‘$1’ at position $2
0010550	Invalid UTF-8 Encoding in ‘$1’ at position $2
0010551	Incomplete UTF-8 Encoding in ‘$1’ at position $2
0010556	Invalid UTF-8 Encoding in ‘$1’ at position $2
0010557	Incomplete UTF-8 Encoding in ‘$1’ at position $2
0010558	Invalid UTF-8 Encoding in ‘$1’ at position $2
0010561	Incomplete UTF-8 Encoding in ‘$1’ at position $2
0010562	Invalid UTF-8 Encoding in ‘$1’ at position $2

Error Code	Message
0010563	Incomplete UTF-8 Encoding in ‘$1’ at position $2
0010564	Index Out of Bounds: $1 < 0
0010565	Invalid UTF-8 Encoding in ‘$1’ at position $2
0010566	Incomplete UTF-8 Encoding in ‘$1’ at position $2
0010567	Index Out of Bounds: $1 > $2
0010568	Invalid UTF-8 Encoding in ‘$1’ at position $2
0010569	Invalid UTF-8 Encoding in ‘$1’ at position $2
0010570	Incomplete UTF-8 Encoding in ‘$1’ at position $2
0010571	Internal Error: Incomplete ROLES Definition
0010574	Invalid UTF-8 Encoding in ‘$1’ at position $2
0010575	Incomplete UTF-8 Encoding in ‘$1’ at position $2
0010587	Internal Error: Incomplete row
0010619	Parameter $1 is not a constant string
0010708	Failure while reading from ‘$1’
0010713	Invalid Regex Type: $1
0010715	Invalid Regex Type: $1
0010717	Could not open shared secret file
0020033	Invalid table name
0020050	columns array requires at least one column
0020053	Problems compiling statement
0020057	internal error: identifier not compiled
0020069	Must have either/or sql/plan request
0020095	Not a create-table command
0020103	Incorrect Data Type Requested
0020108	Expected non-zero-length field name
0020109	Expected non-zero-length field name (2)
0020118	Expected XMLRPC Structure
0020129	Expected non-zero-length field name
0020130	Expected non-zero-length field name
0020135	Query plan not parsable
0020140	Bad partition column node
0020141	Expected partition column name
0020143	Execution plan is NULL
0020144	For now we must have partition column

Error Code	Message
0020146	Didn’t find any incoming links
0020147	Don’t know muxer type
0020154	Execution plan is NULL
0020173	OOB isn’t a failure OOB
0020183	Unexpected out-of-space
0020222	Expected to catch exception
0020224	Expected to catch exception (2)
0020225	Too many value streams
0020226	Bad data type
0020241	Missing +/- on sorting column
0020251	Unexpected domain
0020255	Non-addamark exception caught
0020256	Cannot use function in this case
0020260	Cannot use function in this case
0020268	Bad state
0020273	Can’t send OOB onto stream
0020277	Can’t shutdown output
0020302	Muxer not currently sending data
0020322	Setting up slave nodes again
0020345	Pushing column without values
0020347	Metadata not sent yet
0020365	No clients
0020366	Index out of range
0020367	Request already set
0020368	Index out of range
0020369	Index out of range
0020370	Muxer already set
0020371	Index out of range
0020372	Muxer not currently sending data
0020376	Expecting one client only
0020377	Expecting one client only
0020394	Problems compiling statement
0020903	Expected to catch exception (2)
0020908	Expected to catch exception

Error Code	Message
0020981	Expecting one client only
0020983	Expecting one client only
0020989	Expecting one client only
0025001	Non-null fatalFunc
0030001	Index out of bounds
0030007	Perl tried to die!
0030010	Invalid data found in INT32 field
0030011	Integer overflow/underflow
0030012	Invalid data found in DOUBLE field
0030013	Floating Point overflow/underflow
0030014	Invalid data found in BOOLEAN field
0030015	BASE64 not implemented
0030016	CDATA found in non-scalar data type
0030017	Unknown XML-RPC Datatype
0030018	Incorrect Data Type Requested
0030019	Incorrect Data Type Requested
0030020	Incorrect Data Type Requested
0030021	Incorrect Data Type Requested
0030023	Incorrect Data Type Requested
0030024	Incorrect Data Type Requested
0030026	Incorrect Data Type Requested
0030028	The ETL ‘SELECT’ Statement must be FROM ‘stdin’
0030046	Internal Error: Unknown XMLRPC Parsing State
0030048	Bad XMLRPC: No
0030049	Internal Error: Unexpected State Stack
0030050	Bad XMLRPC: No
0030051	Internal Error: Unknown XMLRPC Parsing State
0030052	Bad XMLRPC: Unexpected CDATA found
0030058	Internal Error: State Stack is empty
0030059	Internal Error: Index out of bounds
0030067	Unexpected XML tag (was expecting ‘param’)
0030068	Unexpected XML tag (was expecting ‘value’)
0030070	Unexpected XML tag (was expecting ‘member’)
0030074	Unexpected XML tag (was expecting ‘data’)

Error Code	Message
0030075	Found two ‘data’ tags for the same array
0030076	Unexpected XML tag (was expecting ‘value’)
0030077	Unexpected XML tag (was expecting ‘value’)
0030078	Unexpected XML tag (wasn’t expecting any tags)
0030079	Unknown XMLRPC Parsing State
0030080	Unexpected end-tag
0030082	Unexpected State Stack
0030083	Missing required subparts for ‘member’ block
0030084	Unknown XMLRPC Parsing State
0030085	Unexpected CDATA found
0030091	State Stack is empty
0030092	Index out of bounds
0030097	!IS_OP
0030108	Regex not compiled
0030118	!IS_CONST
0030128	Bad XMLRPC: No the
0030129	Bad XMLRPC: No the
0030136	Ran out of data
0030143	Unknown DataType
0030149	!IS_OP
0030150	!IS_CONST
0030159	Perl interpreter not initialized
0030171	Unknown DataType
0030173	Cannot open file
0030216	Invalid ORIENTATION code
0030218	Unknown FORMAT code
0030220	Unknown FORMAT code
0030221	XML Encoding not implemented for COLUMN orientation
0030223	Unknown FORMAT code
0030224	BINARY Encoding not implemented for ROW orientation
0030225	Invalid FORMAT code
0030226	Invalid SQLTYP code
0030228	GZIP Compression not implemented
0030229	Unknown COMPRESSION code

Error Code	Message
0030450	Unimplemented Data Type
0030464	invalid type code
0030563	Attempt to consume more data than exists
0030585	Bad state, cannot send message
0030593	Application server is NULL
0030650	Content type is NULL
0030668	Internal Error: Unable to read MemTotal from meminfo
0030669	Internal Error: Unable to read MemFree from meminfo
0030670	Internal Error: Unable to read Cached from meminfo
0030671	Internal Error: Unable to read Buffers from meminfo
0030707	No manager
0030708	No manager
0030709	No manager
0030710	No manager
0030711	Consuming too much or negative
0030712	Weird state
0030713	Unconsuming negative bytes
0030719	Not first vs. first
0030720	Should not be called when no buffer
0030721	Already managed
0030984	!IS_OP
0030985	!IS_CONST
0030993	Only select from stdin allowed
0030994	Problems compiling statement
0031012	!IS_OP
0031014	!IS_CONST
0031050	name must not be null
0031051	value must not be null
0031067	pullDataSource should never be called
0031102	order info may not be NULL
0031160	No Channels
0031341	val == NULL
0031413	Incorrect Data Type Requested
0031417	Regex not compiled

Error Code	Message
0031418	Regex not compiled
0031471	ORDER by not allowed
0031473	UNION not allowed
0031586	Internal Error: Incomplete row
0031587	Internal Error: Incomplete row
0040001	first argument must be a constant
0060012	Bad channel index
0070011	Empty UserNames are Invalid
0070024	Could not open file $1 for reading
0070025	Out of memory
0070026	Guest role entry not found
0070036	Administrator role entry not found
0070049	Empty role names are invalid
0070051	Empty Permission Names are invalid
0070067	Could not open shared secret file
0070072	Out of memory
0070091	Administrator user entry not found
0070092	Guest user entry not found
0070110	Cannot delete built-in permission ‘$1’
0070203	Out of memory
0080002	Unix socket file name too long: $1
0080004	No near line storage socket specified
0080005	Bad reply from near line server
0080006	Got reply from server before Hello
0080008	Bad hello response from server
0080009	Bad StartTime
0080010	Bad EndTime
0080013	No matching pending store request
0080014	Internal Logic Error: Please contact the Development Team
0080018	Problems compiling statement
0080021	Connection to Near Line Server lost during setNSI command
0080023	Connection to Nearline storage server lost
0080024	Failed to complete near line storage server operations
0080036	The ETL ‘SELECT’ Statement must be FROM ‘stdin’

Error Code	Message
0080038	Only select from stdin allowed
0080039	Problems compiling statement
0080186	No table specified
0080199	Bad keytype
0080207	Bad string format
0080221	Attempt to write read only DB file ‘$1’
0080228	Attempt to write read only DB file ‘$1’
0080229	Bad operation ‘$1’
0080233	UPLOADID must be first CSV column
0080234	gzopen failure
0080237	No table specified
0080238	Auth type mismatch $1 vs. $2
0080239	$1 was invoked on multiple requests
0080242	Shim not implemented
0080243	Shim not implemented
0080244	Shim not implemented
0080245	Shim not implemented
0080246	No active master node(s) found for query
0080253	Unexpected EOF talking to Atalla at ‘$1’
0080267	Bad response code to LA GUID request ‘$1’
0080268	LA GUID cound mismatch in reply ‘$1’
0080277	Attempt to sign digest failed
0080278	Attempt to get binary value of NULL
0080286	Can’t find host ‘$1’
0080296	Can’t find upload ID entry for $1
0080297	TASCMConfig should be of the form LAIP:PORT/MGRIP:PORT
0090022	Error querying system.upload_info table
0090023	For table ‘$1’ upload id not found: ‘$2’

SenSage AP Retriever

SenSage AP Retriever displays messages without error codes. The table below lists the full set of messages.

Message
error: Auto-discovery encountered an exception: {0}
fatal: Required config property not set: {0}
error: Unable to construct reader: {0}. {1}
warn: No logging level specified. Defaulting to: {0}
warn: No logging file specified. Defaulting to: {0}
error: Unknown log type {0} specified for Windows host: {1}
error: Simple config property: {0} requires {1}
error: Required config property not set: {0}
warn: Optional config property not set: {0}, using default: {1}
error: Required config property not set: {0}
warn: Auto-discovery of event sources disabled, missing config property: {0}
error: Required config property not set: {0} , no event types will be read from discovered sources.
warn: Config property not set: {0}
info: Auto-discovered host: {0}
warn: Config property not set: {0} , using default: {1}
warn: Error recovering stale log files in {0}
error: Required config property not set: {0}
warn: Config property not set: {0} , using default: {1}
warn: Config property not set: {0} , using default: {0}
error: Unable to open a TCP socket connection to {0}:{1}
error: Missing required config property: {0}, no records will be filtered by ID.
warn: Config property not set: {0} , using default: {1}
error: Unable to initialize StateManager database, no state will be persisted {0}
error: Error closing StateManager database. {0}
error: Unable to load the state of pipeline[{0}
error: Unable to save the state of pipeline[{0}
debug: EventPipelineState::writeObject
debug: EventPipeline::writeObject pipelineDetails: {0}
info: EventPipeline:readObject started, version {0}
info: EventPipeline:readObject::Upgrading state for key = {0}
info: EventPipeline:readObject::During upgrading state, offset null for {0} using zero.
error: EventPipeline:readObject::Unknown version of EventPipelineState: {0}
info: EventPipeline:readObject pipelineDetails: {0}
info: EventPipeline:readObject this.toString: {0}

Message
error: Unknown version of EventPipelineStateDetails: {0}
info: SMBEventReader:init:: eventTypeStr = {0}
debug: SMBEventReader:isRemoveLogReset:: pipeline = {0}
warn: Pipeline {0}
trace: SMBEventReader:read:: handleName = {0} , readOffset = {1}
info: Using skipAheadMap to populate handle {0} with offset {1}
info: SMBEventReader:read:: For pipe = {0}. Using oldest record number = {1}
trace: SMBEventReader:read:: records.readOffset = {0}
trace: SMBEventReader:read:: {0}
trace: SMBEventReader:read:: skipping recordNumber {0}
warn: SMBEventReader:read:: record number was -1 for handle = {0}
warn: SMBEventReader:prepareForRestart:: skipAheadMap already in use!
info: SMBEventReader:prepareForRestart:: handleName = {0} , readOffset = {1}
error: SMBEventReader:prepareForRestart:: skipping due to null log handles
error: SMBEventReader:close:: null logHandles
error: SMBEventReader:close:: null adminSession
info: SMBEventReader:setDebug:: adminSession is null. enableDebug = {0}.
warn: Config property not set: {0} , using default: {1}
warn: Config property not set: {0} , defaulting to:
info: Remote shell server listening on port: {0}
info: Remote shell accepting localhost connections only: {0}
error: Unable to start a remote shell running on port: {0}, e
info: Received client connection: {0}
info: Refusing non-localhost connection from: {0}
info: Verified localhost connection.
info: Remote client disconnected: {0}
fatal: Uncaught exception in thread ‘ {0}
info: RTNet enabled: {0}
error: Required config property not set: {0} , using default: {1}
debug: Initializing RTNet: {0} = {1} , {2} = {3} ,
fatal: Unable to initialize RTNet: {0}
error: Unable to create pipeline [{0}
error: Uncaught exception in Remote Shell thread.{0}
error: Attempting to restart Remote Shell thread in {0} seconds.

Message
error: Uncaught exception in Pipeline Monitor thread.{0}
error: Attempting to restart Pipeline Monitor thread in {0} seconds.
error: Unable to load config file: {0}
trace: getPipelineThread threw = {0}
error: Pipeline[{0}] has error with reader: {1}
error: Unable to establish pipeline[{0}] due to bad readers.
debug: Pipeline[{0}]: assigning filter: {1}
error: Pipeline[{0}]: Unable to construct filter.{1}
warn: Polling interval property not set: {0} , using default: {1}
warn: Invalid polling interval: {0} , using default: {1}
warn: Invalid block size: {0} , using default: {1}
warn: Delete Check property not set: {0} , using default: {1}
warn: Invalid delete check: {0} , using default: {1}
debug: Unable to write records {0}
info: Pipeline Restarter: attempting to start broken pipeline[{0}]
info: Pipeline Restarter: stopping pipeline thread[{0}]
info: Pipeline Restarter: waiting for pipeline to stop[{0}]
info: Pipeline Restarter: pipeline[{0}] is not stopped yet
warn: Pipeline Restarter: Can’t restart pipeline[{0}]. Pipeline would not stop.
info: Pipeline Restarter: preparing reader for restart[{0}
info: Pipeline Restarter: resetting state on pipeline[{0}]
info: Pipeline Restarter: State reset for pipeline[{0}]
warn: Pipeline Restarter: Pipeline not found: pipeline[{0}]
info: Pipeline Monitor: checking for broken pipelines every {0} seconds.
info: Pipeline Monitor: pipeline[{0}] appears to be broken, will attempt restart.
info: Pipeline Monitor: all pipelines appear to be working.
fatal: RTNet Native Implementation: winretriever stub was not loaded, unknown operating system
warn: Missing config property: {0} , using default: {1}

C

Time Zones

This appendix lists every time zone supported by the EDW and the SenSage AP Analyzer. When you enter a time value in SenSage AP Analyzer, use one of the time-zone formats listed below in Supported Time Zones on page 226.

Because the EDW handles raw data from logs, which does not always use the supported time zones, it must accommodate some shortenings of time zone indicators found in that data (such as PDT for Pacific Daylight Time and CST for Central Standard Time). For more information, see Time-Zone Conversion on page 225, next.

Time-Zone Conversion

SenSage AP SQL provides functions that recognize specific time-zone shortenings and maps them to standard time-zone strings, as shown in the table below.

Shortened Time Zone	Standard Time Zone	Description
PST	PST8PDT	Pacific Standard Time and Pacific Daylight Time are interpreted as either standard time or daylight savings depending on the time of year of the actual date.
PDT	PST8PDT	
MDT	MST7MDT	Mountain Daylight Time is interpreted as either standard time or daylight savings depending on the time of year of the actual date.

Shortened Time Zone	Standard Time Zone	Description
CST	CST6CDT	Central Standard Time and Central Daylight Time are interpreted as either standard time or daylight savings depending on the time of year of the actual date.
CDT	CST6CDT	
EDT	EST5EDT	Eastern Daylight Time is interpreted as either standard time or daylight savings depending on the time of year of the actual date.

Note:

- Because Arizona does not support MDT, Mountain Standard Time is NOT converted to Mountain Daylight Time.
- Because Indiana does not consistently support EDT, Eastern Standard Time is NOT converted to Eastern Daylight Time.

Supported Time Zones

```
Africa/Addis_Ababa
Africa/Algiers
Africa/Asmera
Africa/Bangui
Africa/Blantyre
Africa/Brazzaville
Africa/Bujumbura
Africa/Cairo
Africa/Ceuta
Africa/Dar_es_Salaam
Africa/Djibouti
Africa/Douala
Africa/Gaborone
Africa/Harare
Africa/Johannesburg
Africa/Kampala
Africa/Khartoum
Africa/Kigali
Africa/Kinshasa
Africa/Lagos
Africa/Libreville
Africa/Luanda
Africa/Lubumbashi
Africa/Lusaka
Africa/Malabo
Africa/Maputo
Africa/Maseru
Africa/Mbabane
Africa/Mogadishu
Africa/Nairobi
Africa/Ndjamena
Africa/Niamey
Africa/Porto-Novo
Africa/Tripoli
Africa/Tunis
```

```
Africa/Windhoek
America/Adak
America/Anchorage
America/Anguilla
America/Antigua
America/Araguaina
America/Argentina/Buenos_Aires
America/Argentina/Catamarca
America/Argentina/ComodRivadavia
America/Argentina/Cordoba
America/Argentina/Jujuy
America/Argentina/La_Rioja
America/Argentina/Mendoza
America/Argentina/Rio_Gallegos
America/Argentina/San_Juan
America/Argentina/Tucuman
America/Argentina/Ushuaia
America/Aruba
America/Asuncion
America/Atka
America/Bahia
America/Barbados
America/Belem
America/Belize
America/Boa_Vista
America/Bogota
America/Boise
America/Buenos_Aires
America/Cambridge_Bay
America/Campo_Grande
America/Cancun
America/Caracas
America/Catamarca
America/Cayenne
America/Cayman
America/Chicago
America/Chihuahua
America/Coral_Harbour
America/Cordoba
America/Costa_Rica
America/Cuiaba
America/Curacao
America/Dawson
America/Dawson_Creek
America/Denver
America/Detroit
America/Dominica
America/Edmonton
America/Eirunepe
America/El_Salvador
America/Ensenada
America/Fort_Wayne
America/Fortaleza
America/Glace_Bay
America/Godthab
America/Goose_Bay
America/Grand_Turk
America/Grenada
America/Guadeloupe
America/Guatemala
America/Guayaquil
America/Guyana
America/Halifax
America/Havana
```

```
America/Hermosillo
America/Indiana/Indianapolis
America/Indiana/Knox
America/Indiana/Marengo
America/Indiana/Vevay
America/Indianapolis
America/Inuvik
America/Iqaluit
America/Jamaica
America/Jujuy
America/Juneau
America/Kentucky/Louisville
America/Kentucky/Monticello
America/Knox_IN
America/La_Paz
America/Lima
America/Los_Angeles
America/Louisville
America/Maceio
America/Managua
America/Manaus
America/Martinique
America/Mazatlan
America/Mendoza
America/Menominee
America/Merida
America/Mexico_City
America/Miquelon
America/Monterrey
America/Montevideo
America/Montreal
America/Montserrat
America/Nassau
America/New_York
America/Nipigon
America/Nome
America/Noronha
America/North_Dakota/Center
America/Panama
America/Pangnirtung
America/Paramaribo
America/Phoenix
America/Port-au-Prince
America/Port_of_Spain
America/Porto_Acre
America/Porto_Velho
America/Puerto_Rico
America/Rainy_River
America/Rankin_Inlet
America/Recife
America/Regina
America/Rio_Branco
America/Rosario
America/Santiago
America/Santo_Domingo
America/Sao_Paulo
America/Shiprock
America/St_Johns
America/St_Kitts
America/St_Lucia
America/St_Thomas
America/St_Vincent
America/Swift_Current
America/Tegucigalpa
```

```
America/Thule
America/Thunder_Bay
America/Tijuana
America/Toronto
America/Tortola
America/Vancouver
America/Virgin
America/Whitehorse
America/Winnipeg
America/Yakutat
America/Yellowknife
Antarctica/Casey
Antarctica/Davis
Antarctica/DumontDUrville
Antarctica/Mawson
Antarctica/McMurdo
Antarctica/Palmer
Antarctica/Rothera
Antarctica/South_Pole
Antarctica/Syowa
Antarctica/Vostok
Arctic/Longyearbyen
Asia/Aden
Asia/Almaty
Asia/Amman
Asia/Anadyr
Asia/Aqtau
Asia/Aqtobe
Asia/Ashgabat
Asia/Ashkhabad
Asia/Baghdad
Asia/Bahrain
Asia/Baku
Asia/Bangkok
Asia/Beirut
Asia/Bishkek
Asia/Brunei
Asia/Calcutta
Asia/Choibalsan
Asia/Chongqing
Asia/Chungking
Asia/Colombo
Asia/Dacca
Asia/Damascus
Asia/Dhaka
Asia/Dili
Asia/Dubai
Asia/Dushanbe
Asia/Gaza
Asia/Harbin
Asia/Hong_Kong
Asia/Hovd
Asia/Irkutsk
Asia/Istanbul
Asia/Jakarta
Asia/Jayapura
Asia/Jerusalem
Asia/Kabul
Asia/Kamchatka
Asia/Karachi
Asia/Kashgar
Asia/Katmandu
Asia/Krasnoyarsk
Asia/Kuala_Lumpur
```

```
Asia/Kuching
Asia/Kuwait
Asia/Macao
Asia/Macau
Asia/Magadan
Asia/Makassar
Asia/Manila
Asia/Muscat
Asia/Nicosia
Asia/Novosibirsk
Asia/Omsk
Asia/Oral
Asia/Phnom_Penh
Asia/Pontianak
Asia/Pyongyang
Asia/Qatar
Asia/Qyzylorda
Asia/Rangoon
Asia/Riyadh
Asia/Riyadh87
Asia/Riyadh88
Asia/Riyadh89
Asia/Saigon
Asia/Sakhalin
Asia/Samarkand
Asia/Seoul
Asia/Shanghai
Asia/Singapore
Asia/Taipei
Asia/Tashkent
Asia/Tbilisi
Asia/Tehran
Asia/Tel_Aviv
Asia/Thimbu
Asia/Thimphu
Asia/Tokyo
Asia/Ujung_Pandang
Asia/Ulaanbaatar
Asia/Ulan_Bator
Asia/Urumqi
Asia/Vientiane
Asia/Vladivostok
Asia/Yakutsk
Asia/Yekaterinburg
Asia/Yerevan
Atlantic/Bermuda
Atlantic/Canary
Atlantic/Cape_Verde
Atlantic/Faeroe
Atlantic/Jan_Mayen
Atlantic/Madeira
Atlantic/South_Georgia
Atlantic/Stanley
Australia/ACT
Australia/Adelaide
Australia/Brisbane
Australia/Broken_Hill
Australia/Canberra
Australia/Currie
Australia/Darwin
Australia/Hobart
Australia/LHI
Australia/Lindeman
Australia/Lord_Howe
```

```
Australia/Melbourne
Australia/NSW
Australia/North
Australia/Perth
Australia/Queensland
Australia/South
Australia/Sydney
Australia/Tasmania
Australia/Victoria
Australia/West
Australia/Yancowinna
Brazil/Acre
Brazil/DeNoronha
Brazil/East
Brazil/West
CET
CST6CDT
Canada/Atlantic
Canada/Central
Canada/East-Saskatchewan
Canada/Eastern
Canada/Mountain
Canada/Newfoundland
Canada/Pacific
Canada/Saskatchewan
Canada/Yukon
Chile/Continental
Chile/EasterIsland
Cuba
EET
EST
EST5EDT
Egypt
Eire
Europe/Amsterdam
Europe/Andorra
Europe/Athens
Europe/Belfast
Europe/Belgrade
Europe/Berlin
Europe/Bratislava
Europe/Brussels
Europe/Bucharest
Europe/Budapest
Europe/Chisinau
Europe/Copenhagen
Europe/Dublin
Europe/Gibraltar
Europe/Helsinki
Europe/Istanbul
Europe/Kaliningrad
Europe/Kiev
Europe/Lisbon
Europe/Ljubljana
Europe/London
Europe/Luxembourg
Europe/Madrid
Europe/Malta
Europe/Mariehamn
Europe/Minsk
Europe/Monaco
Europe/Moscow
Europe/Nicosia
Europe/Oslo
```

```
Europe/Paris
Europe/Prague
Europe/Riga
Europe/Rome
Europe/Samara
Europe/San_Marino
Europe/Sarajevo
Europe/Simferopol
Europe/Skopje
Europe/Sofia
Europe/Stockholm
Europe/Tallinn
Europe/Tirane
Europe/Tiraspol
Europe/Uzhgorod
Europe/Vaduz
Europe/Vatican
Europe/Vienna
Europe/Vilnius
Europe/Warsaw
Europe/Zagreb
Europe/Zaporozhye
Europe/Zurich
GB
GB-Eire
GMT
HST
Hongkong
Indian/Antananarivo
Indian/Chagos
Indian/Christmas
Indian/Cocos
Indian/Comoro
Indian/Kerguelen
Indian/Mahe
Indian/Maldives
Indian/Mauritius
Indian/Mayotte
Indian/Reunion
Iran
Israel
Jamaica
Japan
Kwajalein
Libya
MET
MST
MST7MDT
Mexico/BajaNorte
Mexico/BajaSur
Mexico/General
Mideast/Riyadh87
Mideast/Riyadh88
Mideast/Riyadh89
NZ
NZ-CHAT
Navajo
PRC
PST8PDT
Pacific/Apia
Pacific/Auckland
Pacific/Chatham
Pacific/Easter
Pacific/Efate
```

```
Pacific/Enderbury
Pacific/Fakaofo
Pacific/Fiji
Pacific/Funafuti
Pacific/Galapagos
Pacific/Gambier
Pacific/Guadalcanal
Pacific/Guam
Pacific/Honolulu
Pacific/Johnston
Pacific/Kiritimati
Pacific/Kosrae
Pacific/Kwajalein
Pacific/Majuro
Pacific/Marquesas
Pacific/Midway
Pacific/Nauru
Pacific/Niue
Pacific/Norfolk
Pacific/Noumea
Pacific/Pago_Pago
Pacific/Palau
Pacific/Pitcairn
Pacific/Ponape
Pacific/Port_Moresby
Pacific/Rarotonga
Pacific/Saipan
Pacific/Samoa
Pacific/Tahiti
Pacific/Tarawa
Pacific/Tongatapu
Pacific/Truk
Pacific/Wake
Pacific/Wallis
Pacific/Yap
Poland
Portugal
ROK
Singapore
Turkey
US/Alaska
US/Aleutian
US/Arizona
US/Central
US/East-Indiana
US/Eastern
US/Hawaii
US/Indiana-Starke
US/Michigan
US/Mountain
US/Pacific
US/Samoa
UTC
W-SU
WET
```

D

Files Managed by Deployment Manager

This appendix lists SenSage AP files that are managed by Deployment Manager. Any edits to these files are overwritten, usually when Deployment Manager starts the service. One exception is `/etc/logrotate.d/syslog-ng`, which is deleted by Deployment Manager rather than overwritten.

Service	File Managed by Deployment Manager
OAE	<postgres_data_dir>/pg_hba.conf <postgres_data_dir>/postgresql.conf <postgres_data_dir>/server.key <postgres_data_dir>/server.crt /etc/init.d/sensage_atpgsql /opt/hexis/hawkeye-ap/etc/atpgsql/postgresql.conf /opt/hexis/hawkeye-ap/etc/atpgsql/pg_hba.conf /opt/hexis/hawkeye-ap/etc/atpgsql/sensage_atpgsql.conf /opt/hexis/hawkeye-ap/etc/atpgsql/server.key /opt/hexis/hawkeye-ap/etc/atpgsql/server.crt /usr/lib/python2.6/site-packages/ambari_agent/hexis-atpgsql-pidpathmap.json /etc/ganglia/hdp/HAPPostgres/conf.d/modpythonHAPPostgres.conf /etc/ganglia/hdp/HAPPostgres/conf.d/postgresql.pyconf /usr/lib64/ganglia/python_modules/postgresql.py /opt/hexis/hawkeye-ap/etc/atpgsql/sensage_sls_<sls_instance>/shared_secret.asc
Analyzer	<postgres_data_dir>/pg_hba.conf <postgres_data_dir>/postgresql.conf <postgres_data_dir>/server.key <postgres_data_dir>/server.crt /etc/init.d/sensage_atpgsql /opt/hexis/hawkeye-ap/etc/atpgsql/postgresql.conf /opt/hexis/hawkeye-ap/etc/atpgsql/pg_hba.conf

Service	File Managed by Deployment Manager
	/opt/hexis/hawkeye-ap/etc/atpgsql/sensage_atpgsql.conf /opt/hexis/hawkeye-ap/etc/atpgsql/server.key /opt/hexis/hawkeye-ap/etc/atpgsql/server.crt /usr/lib/python2.6/site-packages/ambari_agent/hexis-atpgsqlpidpathmap.json /etc/ganglia/hdp/HAPPostgres/conf.d/modpythonHAPPostgres.conf /etc/ganglia/hdp/HAPPostgres/conf.d/postgresql.pyconf /usr/lib64/ganglia/python_modules/postgresql.py /opt/hexis/hawkeye-ap/etc/atpgsql/sensage_sls_<sls_instance>/shared_secret.asc
atslapd	/etc/init.d/sensage_atslapd /opt/hexis/hawkeye-ap/etc/atslapd/omnisight.schema /opt/hexis/hawkeye-ap/etc/atslapd/atslapd.key /opt/hexis/hawkeye-ap/etc/atslapd/atslapd.crt /opt/hexis/hawkeye-ap/etc/atslapd/ldapadd.conf /opt/hexis/hawkeye-ap/etc/atslapd/slapd.conf /opt/hexis/hawkeye-ap/etc/atslapd/sensage_atslapd.conf /usr/lib/python2.6/site-packages/ambari_agent/hexis-slapdpidpathmap.json /opt/hexis/hawkeye-ap/etc/atslapd/ldapsearch.sh
Collector	/etc/init.d/sensage_collector /opt/hexis/hawkeye-ap/etc/collector/sensage_collector.conf /usr/lib/python2.6/site-packages/ambari_agent/hexis-collectorpidpathmap.json
NSS	/etc/init.d/sensage_nss /opt/hexis/hawkeye-ap/etc/nss/nss.conf /opt/hexis/hawkeye-ap/etc/nss/sensage_nss.conf
OpenSSL	/opt/hexis/hawkeye-ap/etc/keys/openssl.cnf /opt/hexis/hawkeye-ap/etc/keys/cert.pem /opt/hexis/hawkeye-ap/etc/keys/private.pem
EDW	/opt/hexis/hawkeye-ap/etc/sls/instance/<instance>/cluster.xml /opt/hexis/hawkeye-ap/etc/sls/instance/<instance>/rootdc.ldif /opt/hexis/hawkeye-ap/etc/sls/instance/<instance>/onisight.ldif /opt/hexis/hawkeye-ap/etc/init.d/sensage_sls_<instance> /opt/hexis/hawkeye-ap/etc/sls/instance/<instance>/athttpd.conf /opt/hexis/hawkeye-ap/etc/sls/instance/<instance>/ldap.conf /opt/hexis/hawkeye-ap/etc/sls/instance/<instance>/ldapadd.conf /opt/hexis/hawkeye-ap/etc/sls/instance/<instance>/atslapd.crt /etc/security/limits.d/99-hexis.conf /opt/hexis/hawkeye-ap/etc/sls/instance/<instance>/LICENSE /usr/lib/python2.6/site-packages/ambari_agent/hexis-edwpidmap.json /usr/lib/python2.6/site-packages/ambari_agent/hexis-edwpidpathmap.json /op/etc/sls/instance/<instance>/ldapsearch.sh /etc/init.d/sensage_sls_<instance> /opt/hexis/hawkeye-ap/etc/init.d/sensage_edw /etc/init.d/sensage_edw

Service	File Managed by Deployment Manager
syslog-ng	/opt/hexis/hawkeye-ap/etc/roll_logs/roll_logs.conf