Server Deployment and Configuration

Qlik® Sense
1.1
Copyright © 1993-2015 QlikTech International AB. All rights reserved.
Paths ................................................................. 22
Files ................................................................. 22
Configuration ....................................................... 22
Logs ................................................................. 22
Ports ................................................................. 23
Dependencies ....................................................... 23
Repository database ............................................. 23
File system ........................................................ 23
Directory service ................................................. 23
Start and restart of services ................................... 23
Start-up behavior .................................................. 23
Manual start ....................................................... 23
Selecting the metrics to display ............................... 24
3.4 Web clients ..................................................... 24
Hub ................................................................. 25
Qlik Management Console ...................................... 25
3.5 Apps .............................................................. 25
Default storage ................................................... 25
Portable format .................................................... 25
3.6 Ports overview .................................................. 25
4 Deployment ......................................................... 29
4.1 Deploying single node sites ................................. 29
Services .............................................................. 29
Qlik Sense Repository Service ................................ 29
Qlik Sense Scheduler Service ................................ 30
4.2 Deploying multi-node sites .................................. 30
Synchronization ..................................................... 30
Data to synchronize .............................................. 30
Entity data synchronization .................................... 31
Binary data synchronization ................................... 31
Services .............................................................. 31
Qlik Sense Repository Service ................................ 31
Central node ....................................................... 31
Other nodes ....................................................... 32
Qlik Sense Proxy Service ....................................... 32
Qlik Sense Scheduler Service ................................ 33
Master ............................................................. 33
Slave ............................................................... 33
Master and slave ................................................... 33
Qlik Sense Engine Service ..................................... 33
Guidelines for deploying multi-node sites .................... 33
Data volumes ....................................................... 33
Dedicated roles per node ....................................... 34
Number of nodes .................................................. 34
Network speed and geographic deployment .................. 34
## Contents

Consider using synchronization rules to reduce traffic ........................................ 34
Developing and extending apps ............................................................................. 34
Example deployment scenarios ............................................................................. 35
  Multi-node scenario: Production deployment ................................................... 35
    Node layout ......................................................................................................... 35
    Services on each node ......................................................................................... 36
    Configuration steps ............................................................................................. 36
  Multi-node scenario: Development site ............................................................... 37
    Node layout ......................................................................................................... 38
    Services on each node ......................................................................................... 38
    Configuration steps ............................................................................................. 38

5 **Backing up and restoring** .................................................................................. 40
  5.1 Backing up and restoring a site ...................................................................... 40
    Backing up a site .................................................................................................. 40
      Items to backup .................................................................................................... 40
      Backup procedure ............................................................................................... 41
    Restoring a site ..................................................................................................... 42
      Items to restore ..................................................................................................... 42
      Restore procedure ............................................................................................... 42
  5.2 Backing up and restoring certificates .............................................................. 43
    Backing up certificates ......................................................................................... 44
    Restoring certificates ........................................................................................... 53
  5.3 Moving a node .................................................................................................. 63

6 **Security** ............................................................................................................ 73
  6.1 Protecting the platform .................................................................................... 73
    Network security ................................................................................................... 73
    Server security ....................................................................................................... 75
    Process security ..................................................................................................... 76
    Rugged software .................................................................................................... 76
    Threat analysis ....................................................................................................... 76
    App security ........................................................................................................... 76
  6.2 Authentication .................................................................................................... 77
    Default authentication module .............................................................................. 78
    Certificate trust ..................................................................................................... 78
      Architecture ......................................................................................................... 78
      Requirements ....................................................................................................... 79
      General ................................................................................................................ 79
      Communication ports ......................................................................................... 79
    Unlocking distributed certificates ...................................................................... 80
    Confirming certificates using Microsoft Management Console ......................... 80
    Handling of certificates when a service starts ..................................................... 80
      Client certificate .................................................................................................. 80
      Server certificate ................................................................................................ 81
      Root certificate .................................................................................................... 81
    Definition of invalid certificate ............................................................................ 82
Maximum number of trusted root certificates ...........................................83
Authentication solutions ...........................................................................83
Ticket solution ..........................................................................................83
Session solution ........................................................................................84
Header solution ........................................................................................85
Anonymous users .......................................................................................86
6.3 Authorization ......................................................................................86
Access control ...........................................................................................87
Resource access control ..........................................................................87
Rules ........................................................................................................87
Streams .....................................................................................................88
Administrator access control ..................................................................89
Data reduction ..........................................................................................89
6.4 Security summary ...............................................................................90
Authentication ..........................................................................................90
Authorization ............................................................................................90
Auditing .....................................................................................................90
Confidentiality ..........................................................................................90
Integrity .....................................................................................................91
Availability ................................................................................................91
Security example: Opening an app ............................................................91
7 Logging ..................................................................................................93
7.1 Requirements ......................................................................................93
Securing the file system ...........................................................................93
Synchronizing time ..................................................................................93
Setting time zone .....................................................................................93
7.2 Log file format ....................................................................................93
Storage .......................................................................................................94
Naming .......................................................................................................94
Structure ....................................................................................................95
Rows ...........................................................................................................95
Common fields ..........................................................................................95
Additional fields .......................................................................................98
Application log ........................................................................................98
Audit log .....................................................................................................99
License log ................................................................................................100
Performance log ......................................................................................100
QIX performance log ..............................................................................102
Qlik Management Console log ...............................................................103
Session log ................................................................................................103
System log ................................................................................................104
Task execution log ....................................................................................105
Traffic log ..................................................................................................105
7.3 Configuring the logging ......................................................................105
Appenders .................................................................................................106
QSRollingFileAppender ................................................................. 106
Configuring the appender ............................................................ 106
Converters ................................................................................. 107
Built-in log4net appenders ............................................................. 108
Example: EventLogAppender ......................................................... 108
Example: SmtpAppender ............................................................... 109
Local log configuration file ........................................................... 109
Requirements ............................................................................. 109
XML schema .............................................................................. 109

8 Licensing ................................................................................. 112
8.1 License Enabler File ............................................................... 112
Increase in tokens ...................................................................... 112
Decrease in tokens ...................................................................... 112
8.2 Access passes ........................................................................ 112
Allocation of access passes .......................................................... 113
Login and logout ........................................................................ 114
Login ......................................................................................... 114
Logout ....................................................................................... 114
Removing access passes .............................................................. 115
User access pass ........................................................................ 115
Login access pass ...................................................................... 115
Disconnected node ..................................................................... 115
Multi-deployment sites ................................................................. 116
Development site ........................................................................ 116
Test site ..................................................................................... 116
Anonymous users ...................................................................... 116
8.3 Licensing metrics ................................................................. 116
1 Introduction

This documentation provides details on the Qlik Sense architecture, deployment, security, logging, and licensing.

This document is derived from the online help for Qlik Sense. It is intended for those who want to read parts of the help offline or print pages easily, and does not include any additional information compared with the online help.

1.1 Support services

Contact Qlik if product support, additional training, or consultation concerning application development is needed. Consult the Qlik homepage for current information on how to get in touch with the support services:

www.qlik.com

<table>
<thead>
<tr>
<th>Global headquarters</th>
<th>Qlik</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>150 N. Radnor Chester Road</td>
</tr>
<tr>
<td></td>
<td>Suite E220</td>
</tr>
<tr>
<td></td>
<td>Radnor, PA 19087</td>
</tr>
<tr>
<td></td>
<td>USA</td>
</tr>
<tr>
<td>Phone</td>
<td>+1 (888) 828-9768</td>
</tr>
<tr>
<td>Fax</td>
<td>+1 (610) 975-5987</td>
</tr>
</tbody>
</table>

For other locations, visit the Qlik homepage (see above).

1.2 Conventions

The following conventions are used in the documentation for Qlik Sense.

Style coding

- Menu commands and dialog options are written in **bold**.
- File names and paths are written in *italic*.
- Sample code is written in Lucida Console.

Environment variables

The paths used in the documentation for Qlik Sense may use environment variables. The variables and the equivalent paths in the Microsoft Windows operating system are listed below.
1 Introduction

<table>
<thead>
<tr>
<th>Environment variable</th>
<th>Microsoft Windows</th>
</tr>
</thead>
<tbody>
<tr>
<td>%LocalAppData%</td>
<td>C:\Users&lt;username&gt;\AppData\Local</td>
</tr>
<tr>
<td>%ProgramData%</td>
<td>C:\ProgramData</td>
</tr>
<tr>
<td>%ProgramFiles%</td>
<td>C:\Program Files</td>
</tr>
<tr>
<td>%UserProfile%</td>
<td>C:\Users&lt;username&gt;</td>
</tr>
</tbody>
</table>

### 1.3 Additional documentation

Besides this document, the following related documentation is available for Qlik Sense:

- **Installation Guide**: Describes how to install Qlik Sense.
- **Managing a Qlik Sense Site**: Describes how to manage a Qlik Sense site.
- **Qlik Deployment Console Guide**: Describes how to deploy Qlik Sense sites in cloud computing environments.
2 System requirements

For information on the requirements that must be fulfilled by the target system in order to successfully install and run Qlik Sense, see the Installation Guide.
3 Architecture

Qlik Sense features a distributed architecture that consists of one or more nodes (that is, server machines) that together form a site. One node assumes the role of central node, which is used as the central point of control.

Each node in a site:

- Has a local repository and file set that contains all the data that the node needs to fulfill its role
- Synchronizes its content with the other nodes in the site
- Can perform a different role within the site
- Operates independently, which increases the system resilience, reduces maintenance, and increases the deployment flexibility
- Deploys a set of Qlik Sense services

3.1 Site

A Qlik Sense site is a collection of one or more nodes (that is, server machines) connected to a common logical repository or central node.

In a typical Qlik Sense installation, there is only one production site, which contains a single central node that contains data for the entire site and, optionally, one or more additional nodes that may be used to increase capacity and resilience. All nodes connect with the central node. App data and all necessary meta-data are synchronized between the central node and the other nodes using asynchronous communication.

Single node site

A single node site is the smallest site possible as it consists of a single node (that is, a single server machine), which is also the central node of the site.
Multi-node site

In a multi-node site, the site is spread out across two or more nodes that share the same set of data and license key. Multi-node sites can be used for many purposes:

- Add capacity
- Add resilience
- Move apps or workload onto a specific node
- Fit with customer network deployments
In a multi-node site, each node has a local copy of the data that it needs to fulfill its role, which means that the node can operate independently in the event of a server or network failure. Each node can read and write its local data and a synchronization mechanism in Qlik Sense distributes the changes to other nodes in the site.

One node is configured to be the central node, which is responsible for controlling the multi-node site. The central node is also the point through which the other nodes in the site synchronize their data.

The synchronization in a multi-node site is two-way:

- The central node requests updates from the other nodes every 15 seconds.
- Each of the other nodes initiates a synchronization session with the central node every 15 seconds.

When changes are made on each node, the resulting transactions are recorded in a transaction log. During the synchronization, the latest set of transactions from the log is sent to the other nodes and replayed.

The synchronization is not visible to the users, who can continue to work in their apps while new data is synchronized in the background.

See also:

- Deploying multi-node sites (page 30)

3.2 Node

A Qlik Sense site consists of the following types of nodes:

- A central node: This is the minimum configuration; a site always includes a central node
- Zero or more additional nodes: Used to increase capacity and resilience

As long as the nodes in a site meet the system requirements, the operating system on the nodes may differ.

A node in a Qlik Sense site runs a set of Qlik Sense services. By configuring which services to run on a node, it can be set up to perform a specific role (for example, as a proxy node or a reload node) within a site.
3.3 Services

The Qlik Sense services, which run on the Microsoft Windows operating system, can be deployed in different ways on a node to suit different deployment purposes.

The Qlik Sense services include:

- The Qlik Sense Repository Service (QRS) manages persistence and synchronization of apps and licensing, security, and service configuration data. The QRS is needed by all other Qlik Sense services to run and serve apps. It attaches to a repository database and manages the repository database synchronization in multi-node sites. In addition, the QRS stores the app structures and the paths to the binary files (that is, the app data stored in the local file system).
- In a default Qlik Sense installation, the Qlik Sense Repository Service (QRS) uses the Qlik Sense Repository Database (QRD) service to read and write data in the repository database. A PostgreSQL database is used by default.
- The Qlik Sense Proxy Service (QPS) manages site authentication, session handling, and load balancing.
- The Qlik Sense Scheduler Service (QSS) manages the scheduled reloads of apps as well as other types of reload triggering based on task events.
- The Qlik Sense Engine Service (QES) is the application service, which handles all application calculations and logic.

Qlik Sense Repository Service

The Qlik Sense Repository Service (QRS) manages persistence and synchronization of apps and licensing, security, and service configuration data. The QRS is needed by all other Qlik Sense services to run and serve apps. It attaches to a repository database and manages the repository database synchronization in multi-node sites. In addition, the QRS stores the app structures and the paths to the binary files (that is, the app data stored in the local file system).

Paths

The following table lists the paths used by the Qlik Sense Repository Service (QRS).

<table>
<thead>
<tr>
<th>Executable</th>
<th>%ProgramFiles%\Qlik\Sense\Repository\Repository.exe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>%ProgramData%\Qlik\Sense\Repository</td>
</tr>
</tbody>
</table>

Files

This section lists the files used by the Qlik Sense Repository Service (QRS).
Repository database
All files related to the repository database are stored in the folder defined below.

Path: %ProgramData%\Qlik\Sense\Repository\PostgreSQL

In a default Qlik Sense installation, the repository database is a specific instance of PostgreSQL that runs its own database cluster specifically for the repository.

Logs
Path: %ProgramData%\Qlik\Sense\Log\Repository

See: Logging (page 93)

Ports
The following table lists the ports used by the Qlik Sense Repository Service (QRS).

<table>
<thead>
<tr>
<th>Port no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4241</td>
<td>Communication port within multi-node sites for Qlik Sense Repository Service (QRS)-to-QRS synchronization. This port uses https for communication.</td>
</tr>
<tr>
<td>4242</td>
<td>Qlik Sense Repository Service (QRS) REST API listen port. Also used by the QRS configuration service for the node registration process before the QRS API service is made available. This port uses https for communication.</td>
</tr>
</tbody>
</table>
| 4444     | This port has the following functions:  
  - Security distribution port, only used by Qlik Sense Repository Services (QRSs) on nodes other than the central node to receive a certificate from the master QRS on the central node. The communication is always unencrypted, but the transferred certificate package is password-protected.  
  - Qlik Sense Repository Service (QRS) state port, used to fetch the state of a QRS in a Qlik Sense site. The state is fetched using http://localhost:4444/status/servicestate. The returned state is one of the following:  
    - 0: Initializing. Once the node has been initialized, the node state changes into one of the other states.  
    - 1: Certificates not installed. There are no certificates installed on the node. The node stays in this state until it has received the certificate and the certificate password.  
    - 2: Running. The node is up and running and all APIs have been initiated. |
### 3 Architecture

<table>
<thead>
<tr>
<th>Port no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4570</td>
<td>Certificate password verification port, only used within multi-node sites by Qlik Sense Repository Services (QRSs) on nodes other than the central node to receive the password that unlocks a distributed certificate. The port can only be accessed from localhost and it is closed immediately after the certificate has been unlocked. The communication is always unencrypted.</td>
</tr>
</tbody>
</table>

See also:

- *Ports overview (page 25)*
- *Single node site (page 11)*
- *Multi-node site (page 12)*
- *Certificate trust (page 78)*

### Metrics

This section lists the metrics related to the Qlik Sense Repository Service (QRS).

#### REST API metrics

The following metrics are available in the Performance Monitor in Microsoft Windows:

- Number of DELETE calls
- Number of GET calls
- Number of POST calls
- Number of PUT calls
- Number of http status 200 (OK)
- Number of http status 201 (Created)
- Number of http status 400 (Bad request)
- Number of http status 401 (Unauthorized)
- Number of http status 403 (Forbidden)
- Number of http status 406 (Not acceptable)
- Number of http status 409 (Conflict)
- Number of http status 415 (Unsupported media type)
- Number of http status 500 (Internal server error)
- Number of http status 503 (Service unavailable)

#### Synchronization metrics

The following metrics are available in the Performance Monitor in Microsoft Windows:

- Number of sessions
- Number of synchronization clients
See also:

- See also: Selecting the metrics to display (page 24)

Qlik Sense Repository Database

In a default Qlik Sense installation, the Qlik Sense Repository Service (QRS) uses the Qlik Sense Repository Database (QRD) service to read and write data in the repository database. A PostgreSQL database is used by default.

Paths

The following table lists the paths used by the Qlik Sense Repository Database (QRD) service.

<table>
<thead>
<tr>
<th>Executable</th>
<th>The QRD spawns the PostgreSQL executable that is located in %ProgramFiles%\Qlik\Sense\Repository\PostgreSQL&lt;database version&gt;\bin.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>%ProgramData%\Qlik\Sense\Repository\PostgreSQL</td>
</tr>
</tbody>
</table>

Files

This section lists the files used by the Qlik Sense Repository Database (QRD) service.

Repository database

All files related to the repository database are stored in the folder defined below.

Path: %ProgramData%\Qlik\Sense\Repository\PostgreSQL\<database version>

Logs

There are no logs for the QRD service.

Ports

The following table lists the ports used by the Qlik Sense Repository Database (QRD) service.

<table>
<thead>
<tr>
<th>Port no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4432</td>
<td>Default listening port for the Qlik Sense Repository Database (QRD). The port is used to listen for connections from the Qlik Sense Repository Service (QRS).</td>
</tr>
</tbody>
</table>
See also:

- Ports overview (page 25)

Qlik Sense Proxy Service

The Qlik Sense Proxy Service (QPS) manages site authentication, session handling, and load balancing.

Paths

The following table lists the paths used by the Qlik Sense Proxy Service (QPS).

<table>
<thead>
<tr>
<th>Executable</th>
<th>%ProgramFiles%\Qlik\Sense\Proxy\Proxy.exe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>%ProgramData%\Qlik\Sense\Proxy</td>
</tr>
</tbody>
</table>

Files

This section lists the files used by the Qlik Sense Proxy Service (QPS).

Configuration

Path: %ProgramData%\Qlik\Sense\Proxy

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app.settings</td>
<td>Proxy settings. The file is created when the service first runs.</td>
</tr>
</tbody>
</table>

Logs

Path: %ProgramData%\Qlik\Sense\Log\Proxy

See: Logging (page 93)

Ports

The following table lists the ports used by the Qlik Sense Proxy Service (QPS).

<table>
<thead>
<tr>
<th>Port no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>Optional Qlik Sense Proxy Service (QPS) listen port.</td>
</tr>
<tr>
<td></td>
<td>This port uses http for communication.</td>
</tr>
</tbody>
</table>

- This port is only available when client-side http communication is enabled. This is managed in the Qlik Management Console (QMC).
### 3 Architecture

<table>
<thead>
<tr>
<th>Port no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>Default Qlik Sense Proxy Service (QPS) listen port. This port uses https for communication.</td>
</tr>
</tbody>
</table>

*Make sure that port 443 is available for the QPS to use, since the port is sometimes used by other software (for example, web servers).*

| 4243     | Qlik Sense Proxy Service (QPS) REST API listen port. |

| 4244     | Default Qlik Sense Proxy Service (QPS) authentication listen port. This port is used by the internal authentication module in the QPS when using NTLM in Microsoft Windows. This port uses https for communication. |

| 4248     | Optional Qlik Sense Proxy Service (QPS) authentication listen port. The port is used by the internal authentication module in the QPS when using NTLM in Microsoft Windows. This port uses http for communication. |

*This port is only available when client-side http communication is enabled. This is managed in the Qlik Management Console (QMC).*

See also:

- *Ports overview (page 25)*

### Metrics

This section lists the metrics related to the Qlik Sense Proxy Service (QPS). The following metrics are available in the Performance Monitor in Microsoft Windows:

- **ActiveConnections:** The number of active connections (in any form or shape) from the client. A connection is a stream (that is, a socket) between a Qlik Sense web client and the Qlik Sense Proxy Service (QPS). This stream is often connected to another stream, which runs from the QPS to the Qlik Sense Repository Service (QRS) or the Qlik Sense Engine Service (QES). The two streams allow the web client to communicate with the QRS or the QES.

- **ActiveStreams:** The number of active data streams (that is, sockets), either from the browser to the QPS or from the QPS to the QRS or the QES.

- **ActiveSessions:** The number of active sessions in the QPS. A Qlik Sense user gets a proxy session when the user has been authenticated. The session is terminated after a certain period of inactivity.
3 Architecture

- LoadBalancingDecisions: The number of users who currently have at least one QES session.
- Tickets: The number of issued login tickets that have not yet been consumed.
- ActiveClientWebsockets: The number of active websockets between the client and the QPS.
- ActiveEngineWebsockets: The number of active websockets between the QPS and the target Qlik Sense service.

The metrics are also available as entries in the Performance log for the QPS.

See also:
- Performance log (page 100)
- Selecting the metrics to display (page 24)

Qlik Sense Scheduler Service

The Qlik Sense Scheduler Service (QSS) manages the scheduled reloads of apps as well as other types of reload triggering based on task events.

Paths

The following table lists the paths used by the Qlik Sense Scheduler Service (QSS).

<table>
<thead>
<tr>
<th>Executable</th>
<th>%ProgramFiles%\Qlik\Sense\Scheduler\Scheduler.exe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>-</td>
</tr>
</tbody>
</table>

Files

This section lists the files used by the Qlik Sense Scheduler Service (QSS).

Logs

Path: %ProgramData%\Qlik\Sense\Log\Scheduler

See: Logging (page 93)

Ports

The following table lists the ports used by the Qlik Sense Scheduler Service (QSS).

<table>
<thead>
<tr>
<th>Port no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5050</td>
<td>Qlik Sense Scheduler Service (QSS) master REST engine. This port uses https for communication.</td>
</tr>
</tbody>
</table>
3 Architecture

<table>
<thead>
<tr>
<th>Port no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5151</td>
<td>Qlik Sense Scheduler Service (QSS) slave REST engine. This port uses https for communication.</td>
</tr>
<tr>
<td>5252</td>
<td>Qlik Sense Scheduler Service (QSS) REST engine monitoring tool. This port is used to get metrics from the QSS. This port uses https for communication.</td>
</tr>
</tbody>
</table>

See also:

- Ports overview (page 25)

Metrics

This section lists the metrics related to the Qlik Sense Scheduler Service (QSS). The following metrics are available in the Performance Monitor in Microsoft Windows:

- Number of connected slaves
- Number of Qlik Sense Engine Service (QES) instances that are running on a slave (this metric is only available on the node where the QES instances run)
- Number of running processes
- Number of running tasks as understood by the master
- Number of running tasks on the slave
- Number of task messages that have been dispatched by the slave
- Number of task messages that have been received by the master
- Number of task retries
- Number of tasks that have completed successfully when executed by the slave
- Number of tasks that have failed when executed by the slave
- Number of tasks that the master has acknowledged as completed
- Number of tasks that the master has acknowledged as failed
- Number of times that the settings have been updated
- Number of tasks that have attempted to start
- Number of tasks that have attempted to stop

See also:

- Selecting the metrics to display (page 24)
Tasks
Tasks are used to perform a wide variety of operations and can be chained together in any arbitrary pattern. The tasks are handled by the Qlik Sense Scheduler Service (QSS) and managed in the Qlik Management Console (QMC).

See: Qlik Management Console (page 25)

Reload
The reload task is used to fully reload the data in an app from the source. Any old data is discarded.

Sync
Within a multi-node site, one instance of the Qlik Sense Repository Service (QRS) runs on each node. The QRS running on the central node is considered to be the master. The master QRS has direct access to the central repository database, whereas the other QRSs only have access to a local repository database on the node where they are running. The master QRS synchronizes the central repository database and the local repository databases.

The sync task is used to schedule the synchronization of the central repository database and the local repository databases within a multi-node site.

Qlik Sense Engine Service
The Qlik Sense Engine Service (QES) is the application service, which handles all application calculations and logic.

Paths
The following table lists the paths used by the Qlik Sense Engine Service (QES).

<table>
<thead>
<tr>
<th>Executable</th>
<th>%ProgramFiles%\Qlik\Sense\Engine\Engine.exe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>%ProgramData%\Qlik\Sense\Engine</td>
</tr>
</tbody>
</table>

Files
This section lists the files used by the Qlik Sense Engine Service (QES).

Configuration
Path: %ProgramData%\Qlik\Sense\Engine

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Settings.ini</td>
<td>Engine settings. The file is created when the service first runs.</td>
</tr>
</tbody>
</table>

Logs
Path: %ProgramData%\Qlik\Sense\Log\Engine

See: Logging (page 93)
Ports
The following table lists the ports used by the Qlik Sense Engine Service (QES).

<table>
<thead>
<tr>
<th>Port no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4747</td>
<td>Qlik Sense Engine Service (QES) listen port. This port is used for communication with the Qlik Sense web clients. This port uses https for communication.</td>
</tr>
</tbody>
</table>

See also:
- Ports overview (page 25)
- Web clients (page 24)

Dependencies
This section describes the dependencies related to the Qlik Sense services (for example, dependencies on the operating system and other software).

Repository database
The Qlik Sense Repository Service (QRS) connects to the repository database to store and retrieve data necessary for the Qlik Sense services on the node on which the QRS is running. In a default Qlik Sense installation, the Qlik Sense Repository Service (QRS) uses the Qlik Sense Repository Database (QRD) service to read and write data in the repository database. A PostgreSQL database is used by default.

File system
The file system stores the binary files for the Qlik Sense apps.

Directory service
The QRS and Qlik Sense Proxy Service (QPS) communicate with a configured directory service (for example, Microsoft Active Directory) using, for example, LDAP or ODBC.

Start and restart of services
Normally, the Qlik Sense services are started automatically on a node.

Start-up behavior
The Qlik Sense Repository Database (QRD) and Qlik Sense Repository Service (QRS) are started first.

When another service is started, it contacts its local QRS to get configuration parameters. If the service is not (yet) configured to run, it periodically checks back with the local QRS.

Manual start
If the services are started manually, make sure to start them in the following order:
The user that installs and runs the Qlik Sense services must be local administrator on the machine.

a. Qlik Sense Repository Database (QRD)
b. Qlik Sense Repository Service (QRS)
c. Qlik Sense Proxy Service (QPS), Qlik Sense Engine Service (QES), and Qlik Sense Scheduler Service (QSS) in no specific order

The order is important because the QRS is dependent on the QRD and the rest of the services are dependent on the QRS.

Selecting the metrics to display

Proceed as follows to select which metrics to display for the Qlik Sense services in the Performance Monitor in Microsoft Windows:

1. Select Start>Run.
2. Enter perfmon and click OK.
3. Expand Monitoring Tools in the left panel.
4. Select Performance Monitor.
   The Performance Monitor is displayed in the right panel.
5. Click the + (plus) icon in the toolbar at the top of the Performance Monitor.
   The Add Counters dialog is displayed.
6. Select the machine to add counters from in the Select counters from computer: drop-down list.
   The Available counters list is populated with counters.
7. Locate the following counter sets in the Available counters list:
   - Qlik Sense Proxy Service
   - Qlik Sense Repository Service - REST API
   - Qlik Sense Repository Service - Synchronization
   - Qlik Sense Scheduler Service
8. Click the + (plus) sign next to a counter set to expand the set.
9. Select the counters to display in the Performance Monitor.
10. Click Add >> to add the counters in the Performance Monitor.
    The added counters are listed in the Added counters list.
11. Click OK.
    The added counters are displayed in the Performance Monitor.

3.4 Web clients

The web clients are used to communicate and interact with Qlik Sense sites.
3. Architecture

Hub

The hub is used to access and publish apps in Qlik Sense. The hub runs in a web browser, so no software installation is required.

Once established, the hub traffic only involves a node other than the central node (unless the site is a single node site) and the hub.

Qlik Management Console

The Qlik Management Console (QMC) is used for configuration and administration of a Qlik Sense site.

The QMC only communicates logically with the central node. This means that:

- The QMC always uses the Qlik Sense Proxy Service (QPS) on the central node.
- For maximum performance within a multi-node site, it is recommended not to allow any user traffic on the central node.

3.5 Apps

A Qlik Sense app is a collection of reusable data items (measures, dimensions, and visualizations), sheets, and stories. It is a self-contained entity that includes the data to analyze in a structured data model.

The apps replace the documents that are used in QlikView.

Default storage

By default, an app is stored as follows:

- Repository database: Contains the app structure, including the paths to the binary files in the local file system.
- Local file system: Stores the app data as binary files. The files are by default stored in %ProgramData%\Qlik\Sense\Apps, but the path can be configured in the Qlik Management Console (QMC).

Portable format

An app can be stored in the local file system in the proprietary .qvf format, which is a portable format.

A single app is stored as <App name>.qvf.

For an app to run in Qlik Sense, it must be stored in the repository database.

3.6 Ports overview

The following figure provides an overview of the ports that are used in a default Qlik Sense installation.
The superscript numbers in the figure refer to the following:

1. The Authentication (ticket) API in the Qlik Sense Proxy Service (QPS).
2. The Session API in the QPS.
3. For example, a Single Sign-On (SSO) portal that uses the Session API.
4. This port is used when the slave Qlik Sense Scheduler Service (QSS) is located on the central node in the site.
5. This port is used when the slave QSS is located on a node other than the central node.

The following table provides details on the ports in the figure.

<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>Optional Qlik Sense Proxy Service (QPS) listen port. This port uses http for communication.</td>
</tr>
<tr>
<td>443</td>
<td>Default Qlik Sense Proxy Service (QPS) listen port. This port uses https for communication.</td>
</tr>
<tr>
<td>4241</td>
<td>Communication port within multi-node sites for Qlik Sense Repository Service (QRS)-to-QRS synchronization. This port uses https for communication.</td>
</tr>
<tr>
<td>4242</td>
<td>Qlik Sense Repository Service (QRS) REST API listen port. Also used by the QRS configuration service for the node registration process before the QRS API service is made available. This port uses https for communication.</td>
</tr>
<tr>
<td>4243</td>
<td>Qlik Sense Proxy Service (QPS) REST API listen port.</td>
</tr>
<tr>
<td>4244</td>
<td>Default Qlik Sense Proxy Service (QPS) authentication listen port. This port is used by the internal authentication module in the QPS when using NTLM in Microsoft Windows. This port uses https for communication.</td>
</tr>
<tr>
<td>4248</td>
<td>Optional Qlik Sense Proxy Service (QPS) authentication listen port. The port is used by the internal authentication module in the QPS when using NTLM in Microsoft Windows. This port uses http for communication.</td>
</tr>
</tbody>
</table>

This port is only available when client-side http communication is enabled. This is managed in the Qlik Management Console (QMC).
### 3 Architecture

<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4432</td>
<td>Default listening port for the Qlik Sense Repository Database (QRD). The port is used to listen for connections from the Qlik Sense Repository Service (QRS). In a default Qlik Sense installation, the Qlik Sense Repository Service (QRS) uses the Qlik Sense Repository Database (QRD) service to read and write data in the repository database. A PostgreSQL database is used by default.</td>
</tr>
</tbody>
</table>
| 4444  | This port has two functions:  
- Security distribution port, only used by Qlik Sense Repository Services (QRSs) on nodes other than the central node to receive a certificate from the master QRS on the central node. The communication is always unencrypted, but the transferred certificate package is password-protected.  
- Qlik Sense Repository Service (QRS) state port, used to fetch the state of a QRS in a Qlik Sense site. The state is fetched using `http://localhost:4444/status/servicestate`. The returned state is one of the following:  
  - 0: Initializing. Once the node has been initialized, the node state changes into one of the other states.  
  - 1: Certificates not installed. There are no certificates installed on the node. The node stays in this state until it has received the certificate and the certificate password.  
  - 2: Running. The node is up and running and all APIs have been initiated. |
| 4747  | Qlik Sense Engine Service (QES) listen port. This port is used for communication with the Qlik Sense web clients. This port uses https for communication. |
| 5050  | Qlik Sense Scheduler Service (QSS) master REST engine. This port uses https for communication. |
| 5151  | Qlik Sense Scheduler Service (QSS) slave REST engine. This port uses https for communication. |

**See also:**

- [Services (page 14)]
4 Deployment

The Qlik Sense architecture is based on the concept of sites. A Qlik Sense site is a collection of one or more nodes (that is, server machines) connected to a common logical repository or central node.

Qlik Sense can be deployed in many ways. This section describes different deployment scenarios.

4.1 Deploying single node sites

In this deployment scenario, all Qlik Sense services run on a single node. This kind of deployment works best in a single time zone, where reloads of data can be done during the night.

See also:

- Single node site (page 11)

Services

This section describes how the Qlik Sense services behave when deployed in single node sites.

Qlik Sense Repository Service

Within a single node site, there is only one instance of the Qlik Sense Repository Service (QRS) running and it has direct access to the central repository database.
Qlik Sense Scheduler Service
When deployed in a single node site, the Qlik Sense Scheduler Service (QSS) acts as both master and slave.

See: Master and slave (page 33)

4.2 Deploying multi-node sites
In a multi-node site, the site is spread out across two or more nodes that share the same set of data and license key.

The basic steps for deploying a multi-node site are as follows:

1. Plan the nodes that are needed in the deployment.
2. Install the first node in the site. This node becomes the central node, which contains all apps that are needed.
   See: Installation Guide
3. Install an additional node as a non-central node.
   See: Installation Guide
4. In the Qlik Management Console (QMC) on the central node, add the new node to the site.
   See: Managing a Qlik Sense Site
5. Wait for the first synchronization to complete and then test the new node.
6. Continue to install additional nodes – one at a time – as described in step 3 until the multi-node site is complete.

See also:

- Multi-node site (page 12)

Synchronization
The synchronization in a multi-node site is two-way:

- The central node requests updates from the other nodes every 15 seconds.
- Each of the other nodes initiates a synchronization session with the central node every 15 seconds.

When changes are made on each node, the resulting transactions are recorded in a transaction log. During the synchronization, the latest set of transactions from the log is sent to the other nodes and replayed.

The synchronization is not visible to the users, who can continue to work in their apps while new data is synchronized in the background.

Data to synchronize
There are two types of data that need to be synchronized:
4 Deployment

- Entity data: The repository database contains the system configuration and all meta data about apps. This data is referred to as entities and is usually small in size. The repository database is controlled by the Qlik Sense Repository Service (QRS).
- Binary data: The app data files contain the data models and app definitions. These files are referred to as binary content and the data model element of these files can be large in size. The app data files are controlled by the Qlik Sense Engine Service (QES).

Entity data synchronization
If the transaction log only contains entity data (that is, changes in the repository database), an entity data synchronization is performed. The changes are applied immediately in the repository database on the receiving node. If a conflict occurs, the latest transaction is used.

Example:

A user consumes a license on a node other than the central node. The record is committed to the repository database and a transaction is recorded in the log. During the next synchronization, the central node asks the other node for its latest transactions and applies them to its local database. The rest of the nodes get the same update from the central node during their next synchronization.

Binary data synchronization
If the transaction log contains binary data (that is, changes to app data files), a binary data synchronization, during which the receiving node obtains the updated data, is initiated. The entire app data file does not have to be copied, just the components that have changed. This means that small edits (for example, a new sheet in an app) are synchronized quickly and independently from a large edits (for example, a data model that is synchronized after a reload). During binary data synchronization, the nodes use peer-to-peer replication to speed up the synchronization of large apps and prevent network bottlenecks.

Example:

An app is reloaded on a node other than the central node. During the next synchronization, the central node checks the transaction log and initiates a binary data synchronization. The rest of the nodes get the same update during their next synchronization with the central node. However, the nodes can obtain the binary data not just from the central node, but from any node that already has the updates.

Services
This section describes how the Qlik Sense services behave when deployed in multi-node sites.

Qlik Sense Repository Service
The Qlik Sense Repository Service (QRS) behaves differently depending on if it is deployed on the central node or on another node.

Central node
Within a multi-node site, one instance of the Qlik Sense Repository Service (QRS) runs on each node. The QRS running on the central node is considered to be the master. The master QRS has direct access to the
central repository database, whereas the other QRSs only have access to a local repository database on the node where they are running. The master QRS synchronizes the central repository database and the local repository databases.

When the master QRS starts, it connects to the central repository database. If no database exists, the master QRS builds the database and populates it with initial data. In a default Qlik Sense installation, the repository database is a specific instance of PostgreSQL that runs its own database cluster specifically for the repository.

Other nodes
When the Qlik Sense Repository Service (QRS) on a node other than the central node starts, it connects to the local repository database on the node. If no local repository database exists, the QRS waits until it communicates with the central node.

In a default Qlik Sense installation, the repository database is a specific instance of PostgreSQL that runs its own database cluster specifically for the repository.

Qlik Sense Proxy Service
On the central node in a multi-node site, it is recommended to have a dedicated Qlik Sense Proxy Service (QPS) that is used specifically for the Qlik Management Console (QMC) and not for the hub.

See also:
- Web clients (page 24)
Qlik Sense Scheduler Service

Depending on the type of deployment, the Qlik Sense Scheduler Service (QSS) runs as master, slave, or both on a node.

Master

There is only one master Qlik Sense Scheduler Service (QSS) within a site and it is always located on the central node, where the master Qlik Sense Repository Service (QRS) runs. This means that the central node must have the Qlik Sense Scheduler Service (QSS) installed even if more QSS nodes are added. This is because the QSS on the central node coordinates all QSS activities within the site.

The master QSS handles all task administration (for example, which tasks to execute and when to execute a specific task). When the time comes to execute a task, the master QSS sends the task ID to a slave QSS within the site. Which slave QSS to distribute the task ID to is determined by a load balancing operation performed by the master QSS.

When a slave QSS completes a task, it returns the task state (successful or fail) to the master QSS. The master QSS uses the task state to perform task chaining, which means that it uses the task state to determine if other events are affected by the state of the completed task and need to be executed. This can be configured in the Qlik Management Console (QMC).

If the slave QSS fails to perform the task, the master QSS repeatedly requests the same or another slave QSS to perform the task until it has been completed or until the maximum number of attempts has been reached.

Slave

If a Qlik Sense Scheduler Service (QSS) runs on a node other than the central node, the QSS is considered to be a slave QSS.

When receiving a task ID from the master QSS, the slave QSS reads the task from the local repository database on the node and executes the task.

When a slave QSS completes a task, it returns the task state (successful or fail) to the master QSS.

Master and slave

Within a single node site, the master Qlik Sense Scheduler Service (QSS) also acts as a slave QSS.

Qlik Sense Engine Service

On the central node in a multi-node site, it is recommended to have a dedicated Qlik Sense Engine Service (QES) that is used specifically for the Qlik Management Console (QMC) and not for the hub.

Guidelines for deploying multi-node sites

This section provides guidance on what to consider when planning and designing multi-node sites.

Data volumes

The volume of data in the repository database is fairly small and normally synchronizes quickly. However, the volume of data in a Qlik Sense app can be very large depending on the app.
Each node that is to serve an app needs a local copy of the entire data model in that app before it can allow users access. Each time that the app is reloaded, the changed data needs to be synchronized to the node. Multiple apps can be synchronized simultaneously. This means that the factors that affect the volume of data to synchronize are the number of apps, the size of the apps, and the reload frequency.

It is recommended to change less than 5 GB of data per hour (change of data refers to reloading apps or importing new ones) provided that the network allows for the content to move among the nodes within a reasonable time. If this amount of data is exceeded, there may be performance issues. In the event of an overnight batch of app reloads, such delays may be acceptable.

If the data changes faster than can be handled by the network or the synchronization software (for example, due to a reload before the ongoing synchronization is complete), the synchronization is cancelled and the data is not updated. In some cases a synchronization queue that affects the user experience is built up.

Reloads of QVD files are not included in the figures presented in this section as they are not synchronized.

Dedicated roles per node
It is recommended to give each node a specific role within the multi-node site. For example, make sure that nodes are dedicated to running scheduled reloads or serving user content, not both.

Number of nodes
It is recommended that a multi-node site consists of eight nodes or less. The roles of the nodes may vary.

Network speed and geographic deployment
The ability to synchronize binary content is affected by the network speed. Better throughput means faster synchronization. For best results, the nodes should be on the same network or, if in separate data centers, connected by LAN-like network connections.

Geographically dispersed nodes with slow network connections not only synchronize slowly – they may even slow down the synchronization to other nodes that are on faster connections.

Consider using synchronization rules to reduce traffic
The amount of data to synchronize can be reduced by defining rules in the Qlik Management Console (QMC) that dictate which data is synchronized to which node.

Developing and extending apps
Apps can be created and edited in a Qlik Sense server environment or using Qlik Sense Desktop. Server-based development provides a number of advantages when it comes to app and data governance, security, performance, and collaboration.

The app development process is often iterative and involves steps such as:

- Building a load script
- Running reloads as the data model is built and the user interface is assembled
4 Deployment

- Tuning the app for best performance
- Testing that the analysis produced is correct

The changes made to apps under development are synchronized to the rest of the nodes in the site. This can generate significant load on the network and synchronization mechanism, especially if the apps are large or reloaded often. It is therefore recommended to isolate development activities (specifically reloads) to a dedicated development site or to use Qlik Sense Desktop for app development.

Once the app development is complete, the app can be published, so that other users can access it. The users can add new sheets and stories to the published app. This is considered as extending the app, not as app development.

Example deployment scenarios
This section provides examples of how multi-node sites can be deployed.

The following terms are used in the scenarios:

- Central node: The node that is responsible for management activities and synchronization.
- Reload/scheduler node: A node that reloads apps on a schedule, but serves no content to users.
- Consume node: A node that serves apps to users, but is not used to create, process, or reload data.
- Development node: A node that allows users to create and reload new apps, but does not serve normal consumer traffic.
- Proxy node: A node that provides load balancing of user traffic to other nodes. This node does not contain a Qlik Sense Engine Service (QES).

An alternative to use a proxy node is to have a proxy installed on each consume node and balance the traffic using a hardware load balancer.

Multi-node scenario: Production deployment
This scenario describes how to setup a typical internal production deployment, which provides the ability to scale up both reloads and user load.

Node layout
Each node within the site only contains the services and data that it needs to perform its role.
4 Deployment

Services on each node

The table below lists the Qlik Sense services that are deployed on each node in the site.

<table>
<thead>
<tr>
<th>Node name</th>
<th>Qlik Sense Repository Service (mandatory)</th>
<th>Qlik Sense Engine Service</th>
<th>Qlik Sense Scheduler Service</th>
<th>Qlik Sense Proxy Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Central node</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Reload/scheduler node(s)</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Consume node (s)</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proxy node</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

Configuration steps

**Basic installation**

Proceed as follows to perform the basic installation:

1. **Install the Qlik Sense software and services as described in the table above.**
   
   See: Installation Guide

2. **Add each node via the Qlik Management Console (QMC) on the central node.**
   
   See: Managing a Qlik Sense Site
3. When all nodes have been added, check that they are displayed as being online in the QMC on the central node.

Load balancing
Proceed as follows to configure the load balancing:

1. Select Proxies in the Qlik Management Console (QMC) on the central node.
2. Edit the settings for the proxy node.
   Under Load balancing nodes, specify that the consume nodes should be used.
3. Check that the hub is accessible on the proxy node. In addition, check that the hub lists the apps.

Qlik Sense Scheduler Service
Proceed as follows to configure the Qlik Sense Scheduler Service (QSS):

- Configure the QSS on the central node to run as master only (that is, do not run reloads on the central node). The reload node should be set as slave, which means it will handle all reloads.

License Monitor and Operations Monitor
The data for the License Monitor and Operations Monitor apps in the Qlik Management Console (QMC) only exists on the central node. Proceed as follows to keep the apps reloading even though all reloads occur on other nodes:

1. Create the following file shares on the central node:
   - %ProgramData%\Qlik\Sense\Log
   - %ProgramData%\Qlik\Sense\Repository\Archived Logs
2. Modify the following data connections in the QMC to point to the shares created in step 1:
   - ServerLogFolder
   - ArchivedLogsFolder

Multi-node scenario: Development site
This scenario places the development of apps onto dedicated resources. The number of nodes can be adjusted to support the amount of development activity (for example, a single node can be used).

If more than one development node is used, they can be load balanced using a proxy node. However, when creating a new app there can be a short delay before the app is added on all nodes, which means that the users may be routed to a node that has not yet received the new app.
Services on each node

<table>
<thead>
<tr>
<th>Node name</th>
<th>Qlik Sense Repository Service (mandatory)</th>
<th>Qlik Sense Engine Service</th>
<th>Qlik Sense Scheduler Service</th>
<th>Qlik Sense Proxy Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Central node</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Development node(s)</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

Configuration steps

**Basic installation**

Proceed as follows to perform the basic installation:

1. Install the Qlik Sense software and services as described in the table above.
   
   See: Installation Guide

2. Add each node via the Qlik Management Console (QMC) on the central node.
   
   See: Managing a Qlik Sense Site
3. When all nodes have been added, check that they are displayed as being online in the QMC on the central node.

Load balancing

Proceed as follows to configure the load balancing:

1. Select Proxies in the Qlik Management Console (QMC) on the central node.
2. Edit the settings for the proxy node.
   Under Load balancing nodes, specify that the local Qlik Sense Engine Service (QES) should be used for each proxy.
3. Check that the hub is accessible on the development nodes. In addition, check that the hub lists the apps on the nodes and that new apps can be created.
5 Backing up and restoring

This section describes how to back up and restore Qlik Sense sites and certificates and how to move a node to a new machine.

5.1 Backing up and restoring a site

This section describes how to back up and restore a Qlik Sense site.

These instructions define the minimum steps required. The use of specific backup software may further extend the options for backup and restore.

In a single node site, the single node is referred to as the central node.

In a multi-node site, the central node is the master record that contains all data about the site. All other nodes in the multi-node site contain either a full copy or a limited subset of the data, which is maintained by the synchronization mechanism. This means that the central node is the only node that needs to be backed up in order to keep the data and configuration safe. The rest of the nodes can be restored by simply re-adding them as new nodes, since they will have their data restored by the synchronization mechanism.

Nodes other than central nodes maintain local log files that may be worth backing up in order to identify and investigate issues. It may also be worth backing up any general operating system data that may be required.

See also:

- Single node site (page 11)
- Multi-node site (page 12)

Backing up a site

This section describes how to backup a Qlik Sense site in a default installation where a PostgreSQL database is used as the repository database.

Items to backup

The following items need to be considered when backing up a site:

- Repository database: The database contains all configuration data for the site.
- Certificates for the Qlik Sense services: The certificates are used to encrypt the traffic between the services and the users. Make sure to backup the certificates in order not to lose any encrypted data (for example, passwords for data connections).
- Log data
- Application data: The data models in the Qlik Sense apps.
- Any content that supports the apps (for example, QVD files)
Backup procedure

Proceed as follows to backup a Qlik Sense site:

1. Stop the Qlik Sense services.
2. Make a backup of the repository database:
   a. Open the Command Prompt in Microsoft Windows.
   b. Produce a dumpfile for the repository database (that is, a single file for the entire database):
      i. cd "C:\Program Files\Qlik\Sense\Repository\PostgreSQL\<database version>\bin"
      ii. pg_dump.exe -h localhost -p 4432 -U postgres QSR > c:\dumpfile
          If you are prompted for the PostgreSQL super user password, enter the password that was given during the installation of Qlik Sense.
      iii. Make a backup of the dumpfile for the repository database.
3. Make a backup of the certificates used to secure the Qlik Sense services. This only needs to be done once.
   See: Backing up certificates (page 44)
4. Make a backup of the log files, which are stored in %ProgramData%\Qlik\Sense\Log.
5. Make a backup of the %ProgramData%\Qlik\Sense\Apps folder, which contains the binary files for the apps.
6. Make a backup of any locations where content that supports the Qlik Sense environment may be kept (for example, QVD files created by load scripts).
7. Start the Qlik Sense services. If the services are started manually, start them in the following order:

   The user that installs and runs the Qlik Sense services must be local administrator on the machine.

   a. Qlik Sense Repository Database (QRD)
   b. Qlik Sense Repository Service (QRS)
   c. Qlik Sense Proxy Service (QPS), Qlik Sense Engine Service (QES), and Qlik Sense Scheduler Service (QSS) in no specific order.
   The order is important because the QRS is dependent on the QRD and the rest of the services are dependent on the QRS.

See also:

- Restoring a site (page 42)
- Backing up and restoring certificates (page 43)
Restoring a site

This section describes how to restore a Qlik Sense site in a default installation where a PostgreSQL database is used as the repository database.

Items to restore

The following items need to be considered when restoring a site:

- Qlik Sense software
- Repository database: The database contains all configuration data for the site.
- Certificates for the Qlik Sense services: The certificates are used to encrypt the traffic between the services and the users. Make sure to backup the certificates in order not to lose any encrypted data (for example, passwords for data connections).
- Log data
- Application data: The data models in the Qlik Sense apps.
- Any content that supports the apps (for example, QVD files)

Restore procedure

1. Install the Qlik Sense software on the machine targeted for the restore.  
   See: Installation Guide

   The machine targeted for the restore can have a different machine name than the machine from which the node was backed up.

   Make sure to deselect Start the Qlik Sense services when the installation has completed during the installation setup. If the services are started, new certificates and a new repository database are created and they must be removed before proceeding with the restore procedure.

2. Restore the certificates used to secure the Qlik Sense services.  
   See: Restoring certificates (page 53)
3. Restore the repository database:
   a. Place the backed up repository database on the machine targeted for the restore.
   b. Open the Command Prompt in Microsoft Windows.
   c. Run the following commands to restore the repository database (adjust the paths as needed):
      i. `cd "C:\Program Files\Qlik\Sense\Repository\PostgreSQL\<database version>\bin"
      ii. `createdb -h localhost -p 4432 -U postgres -T template0 QSR`
          If the command fails because a database already exists, run the following command and then repeat the createdb command:
          `dropdb -h localhost -p 4432 -U postgres QSR`
      iii. `psql -h localhost -p 4432 -U postgres QSR < c:\dumpfile`

4. Restore the binary files for the Qlik Sense apps to the `%ProgramData%\Qlik\Sense\Apps` folder.
5. Restore the Qlik Sense log files to the `%ProgramData%\Qlik\Sense\Log` folder.
6. Restore any supporting content to its original location as required.
7. Start the Qlik Sense services. If the services are started manually, start them in the following order:

   The user that installs and runs the Qlik Sense services must be local administrator on the machine.

   a. Qlik Sense Repository Database (QRD)
   b. Qlik Sense Repository Service (QRS)
   c. Qlik Sense Proxy Service (QPS), Qlik Sense Engine Service (QES), and Qlik Sense Scheduler Service (QSS) in no specific order

   The order is important because the QRS is dependent on the QRD and the rest of the services are dependent on the QRS.

See also:

- Backing up a site (page 40)
- Backing up and restoring certificates (page 43)

5.2 Backing up and restoring certificates

It is recommended that you backup the certificates on the central node in a Qlik Sense site so that they can be restored, if needed.

The backed up certificates can be used for different purposes:
5 Backing up and restoring

- Restore the certificates on the **same** node as they were exported from.
- Move a node to **another** node in the site. This means that the repository database and its associated crypto key are reused on another node, but with new certificates for communication.

**Backing up certificates**

Proceed as follows to make a backup of the certificates on the central node in a Qlik Sense site:

1. Select **Start > Run**.
2. Enter *mmc* and click **OK**.
3. Select **File > Add/Remove Snap-in**.
4. Double-click **Certificates**.

![Image of the Run dialog box](Image)

![Image of the Add or Remove Snap-ins dialog box](Image)
5. Select **Computer account** and click **Next**.

6. Select **Local computer** and click **Finish**.

7. Double-click **Certificates**.
8. Select **My user account** and click **Finish**.

9. Click **OK**.
10. Expand **Certificates (Local Computer)** in the left panel.

11. Expand the **Trusted Root Certification Authorities** folder and select the **Certificates** folder.

12. Right-click the certificate that is Certificate Authority (CA) for all nodes in the site and select **All Tasks > Export**. The CA is named `<machine_that_issued_the_certificate>-CA` by default.
13. Click **Next**.

14. Select **Yes, export the private key** and click **Next**.
15. Select **Personal Information Exchange**.
16. Select **Include all certificates in the certification path if possible** and **Export all extended properties**. Then click **Next**.
17. Enter and confirm a password. Then click **Next**.
   The password is needed when importing the certificate.
18. Enter a file name for the .pfx file and click **Next**.

*It is recommended to include the server name in the file name to avoid confusion with other certificate files.*
19. Click **Finish**.

The .pfx file that contains the CA for all nodes in the site is stored in the selected location.
20. Starting at step 11, repeat the procedure and export the server certificate, which is located under **Certificates (Local Computer) > Personal > Certificates**. The server certificate a) has the same name as the Domain Name System (DNS) name of the machine, and b) is signed by the CA for all nodes in the site.

21. Starting at step 11, repeat the procedure and export the client certificate (that is, the ID of the client), which is located under **Certificates - Current User > Personal > Certificates**. The client certificate is named *QlikClient* and is signed by the CA for all nodes in the site.

22. Close the MMC console. 
   No changes have to be saved.

---

**See also:**

- *Restoring certificates (page 53)*

**Restoring certificates**

In case of a system crash, the certificates may have to be restored on the central node in the Qlik Sense site.

Proceed as follows to restore the certificates on the central node in a site:
5 Backing up and restoring

1. Open the Task Manager in Microsoft Windows and stop all Qlik Sense services except the Qlik Sense Repository Database (QRD) service.
2. Select **Start>Run**.
3. Enter `mmc` and click **OK**.

![Run](image)

4. Select **File>Add/Remove Snap-in**.
5. Double-click **Certificates**.

![Add or Remove Snap-ins](image)

6. Select **Computer account** and click **Next**.
5 Backing up and restoring

7. Select **Local computer** and click **Finish**.

8. Double-click **Certificates**.
9. Select **My user account** and click **Finish**.

10. Click **OK**.
11. Expand **Certificates (Local Computer)** in the left panel.

12. Right-click the **Trusted Root Certification Authorities** folder and select **All Tasks > Import**.
13. Click Next.

14. Browse to the file that contains the backed up Certificate Authority (CA) for all nodes in the site and then click Next. The CA is named <machine_that_issued_the_certificate>-CA by default.
15. Enter the password for the .pfx file (that is, the password that was given when the file was exported).

16. Select **Mark this key as exportable** and **Include all extended properties**. Then click **Next**.
17. Select **Place all certificates in the following store** and click **Next**.
18. Click **Finish**.
19. Click the **Refresh** button and check that the imported CA is available in the **Trusted Root Certification Authorities** folder.

20. Starting at step 11, repeat the procedure and import the server certificate to **Certificates (Local Computer) > Personal > Certificates**. The server certificate a) has the same name as the Domain Name System (DNS) name of the machine, and b) is signed by the CA for all nodes in the site.

21. Starting at step 11, repeat the procedure and import the client certificate (that is, the ID of the client) to **Certificates - Current User > Personal > Certificates**. The client certificate is named *QlikClient* and is signed by the CA for all nodes in the site.

22. Close the MMC console.

   No changes have to be saved.

23. Start the Qlik Sense services. If the services are started manually, start them in the following order:

   *The user that installs and runs the Qlik Sense services must be local administrator on the machine.*

   a. Qlik Sense Repository Database (QRD)
   b. Qlik Sense Repository Service (QRS)
c. Qlik Sense Proxy Service (QPS), Qlik Sense Engine Service (QES), and Qlik Sense Scheduler Service (QSS) in no specific order
The order is important because the QRS is dependent on the QRD and the rest of the services are dependent on the QRS.

See also:

- Backing up certificates (page 44)

5.3 Moving a node

A backed up server certificate can be used to move a node in a Qlik Sense site to another node in the same site by moving the existing repository database and its associated crypto key to the new node.

Proceed as follows to install the crypto key for the repository database on a new node in the site:

1. Open the Task Manager in Microsoft Windows and stop all Qlik Sense services except the Qlik Sense Repository Database (QRD) service.
2. Select Start>Run.
3. Enter `mmc` and click OK.

![Run Dialog]

4. Select File>Add/Remove Snap-in.
5. Double-click Certificates.
6. Select **Computer account** and click **Next**.

7. Select **Local computer** and click **Finish**.
8. Click **OK**.

9. Expand **Certificates (Local Computer)** in the left panel.
10. Right-click the **Personal** folder and select **All Tasks>Import**.

11. Click **Next**.
12. Browse to the file that contains the backed up server certificate. The server certificate a) has the same name as the Domain Name System (DNS) name of the machine, and b) is signed by the CA for all nodes in the site. Then click Next.
13. Enter the password for the .pfx file (that is, the password that was given when the file was exported).

14. Select **Mark this key as exportable** and **Include all extended properties**. Then click **Next**.
15. Select **Place all certificates in the following store** and click **Next**.
16. Click Finish.
17. Click the **Refresh** button ( ) and check that the imported server certificate is available in the **Personal** folder.

18. Close the MMC console.
   - No changes have to be saved.

19. Start the Qlik Sense services. If the services are started manually, start them in the following order:

   - **The user that installs and runs the Qlik Sense services must be local administrator on the machine.**

   a. Qlik Sense Repository Database (QRD)
   b. Qlik Sense Repository Service (QRS)
   c. Qlik Sense Proxy Service (QPS), Qlik Sense Engine Service (QES), and Qlik Sense Scheduler Service (QSS) in no specific order

   The order is important because the QRS is dependent on the QRD and the rest of the services are dependent on the QRS.
5 Backing up and restoring

See also:

- Backing up certificates (page 44)
- Restoring certificates (page 53)
6 Security

The security in Qlik Sense consists of the following parts:

- Protection of the platform: How the Qlik Sense platform itself is protected and how it communicates and operates.
- Authentication: Who is the user and how can the user prove it? Qlik Sense uses standard authentication protocols (for example, Integrated Windows Authentication (IWA), http headers, and ticketing) to authenticate every user requesting access to data.
- Authorization: What does the user have access to? Authorization is the procedure of granting or denying users access to resources.

6.1 Protecting the platform

The security in Qlik Sense does not depend only on the Qlik Sense software. It also relies on the security of the environment that Qlik Sense operates in. The figure below shows the components that have to be considered in order to maximize the security.

**Network security**

**Server security**

**Process security**

**App security**

Result: Qlik Sense security

Network security

For all Qlik Sense components to communicate with each other in a secure way, they need to build trust.

In Qlik Sense, all communication between the Qlik Sense services and web clients is based on web protocols. The web protocols use Transport Layer Security (TLS) for:
- Encryption and exchange of information and keys
- Certificates for authentication of the communicating parties

TLS provides a way to build encrypted tunnels between identified servers or services. The parties that communicate are identified using certificates. Each tunnel needs two certificates; one to prove to the client that it is communicating with the right server and one to prove to the server that the client is allowed to communicate with the server.

So, how to make sure that the certificates are from the same trust zone? All certificates that belong to a trust zone are signed with the same signature. If the signature exists in the certificate, it is accepted as proof that the certificate belongs to the trust zone.

When the protected tunnels and the correct certificates are in place, the Qlik Sense services have a trust zone to operate within. Within the trust zone, only services that belong to the specific Qlik Sense site can communicate with each other.

The Qlik Sense web clients are considered to be outside of the trust zone because they often run on less trusted end-user devices. The Qlik Sense Proxy Service (QPS) can bridge the two zones and allow communication with servers within the trust zone, if the user can authenticate (that is, show who the user is) to the system.

TLS-protected tunnels are used to secure the communication between the Qlik Sense web clients and the QPS. As the web clients are outside of the trust zone, the communication between the web clients and the QPS uses a certificate with a different signature than the one used within the trust zone.

See also:

- Certificate trust (page 78)
Server security

Qlik Sense uses the server operating system to gain access to resources. The operating system provides a security system that controls the use of the server resources (for example, storage, memory, and CPU). Qlik Sense uses the security system controls to protect its resources (for example, files, memory, processes, and certificates) on the server.

Through the use of access control, the security system grants access to Qlik Sense files (for example, log files, database files, certificates, and apps) only to certain users on the server.

The security system also protects the server memory, so that only authorized processes are allowed to write to the Qlik Sense part of the memory.

In addition, the security system is responsible for assigning users to processes. This is used to restrict who is allowed to interact with the Qlik Sense processes on the server. The processes are also restricted in terms of which parts of the operating system they are allowed to access.
So, by using the controls in the security system, a secure and protected environment can be configured for the Qlik Sense processes and files.

Process security

As described in Server security (page 75), each process executes in an environment that poses different threats to the process. In this layer of the security model, the focus is on ensuring that the software is robust and thoroughly analyzed from a security perspective.

Rugged software

For software to be considered as rugged, it must cope with all potential threats to the confidentiality, integrity, and availability of the information, and be robust when used in ways not anticipated.

Several mitigating actions have been implemented in the Qlik Sense software in order to make it rugged:

- Authorization of communication using certificates
- Validation of all external data that is sent to the system
- Encoding of content to avoid injection of malicious code
- Use of protected memory
- Encryption of data
- Audit logging
- Use of checksums
- Isolated execution of external components
- Escaping of SQL data

Threat analysis

To ensure that the Qlik Sense software is secure and rugged, threat analyses of the design have been performed as part of the development process. The following threat areas, often abbreviated as STRIDE, have been covered:

- Spoofing
- Tampering
- Repudiation
- Information disclosure
- Denial of service
- Elevation of privilege

In addition to the threat analyses, exploratory security testing has also been performed on the Qlik Sense software.

App security

The major components of the Qlik Sense app security are:
6 Security

- Access control system: The access control system grants users access to the resources in Qlik Sense.
- Data reduction: The data reduction functionality is based on the concept of section access, which is a way to dynamically change which data a user can view. This makes it possible to build apps that can be consumed by many users, but with different data sets that are dynamically created based on user information. The reduction of data is performed by the Qlik Sense Engine Service (QES).

Using these components, the resources and data (that is, the content) consumed by the Qlik Sense users can be secured.

6.2 Authentication

All authentication in Qlik Sense is managed by the Qlik Sense Proxy Service (QPS). The QPS authenticates all users regardless of Qlik Sense client type. This means that the QPS also authenticates users of the Qlik Management Console (QMC).

In Qlik Sense, authentication and authorization are two distinct, unconnected actions. In addition, the sources of information used for authentication do not have to be the same as for authorization, and the other way around.

Qlik Sense always asks an external system to verify who the user is and if the user can prove it. The interaction between Qlik Sense and the external identity provider is handled by authentication modules.

For a module to communicate with Qlik Sense, it has to be trusted. Transport Layer Security (TLS) and certificate authentication are used to authorize external components for communication with Qlik Sense.

In Qlik Sense, the authentication of a user consists of three distinct steps:

1. Authentication module: Get the user identity and credentials.
2. Authentication module: Request an external system to verify the user identity using the credentials.
3. Transfer the user to Qlik Sense using the Ticket API, the Session API, or headers.

The first two steps are always handled by the authentication module. It is up to the authentication module to verify the user in an appropriate way.

The third step can be performed in the following ways:

- Using the ticket API, which transfers the user and the user's attributes using a one-time ticket.
- Using the session API, whereby an external module can transfer web sessions that identify the user and the user's attributes to Qlik Sense.
- Using headers, with which a trusted system can transfer the user using http headers. This is a common solution for integrating with Single Sign-On (SSO) systems.
- Qlik Sense can be configured to allow anonymous users.

See also:

- Network security (page 73)
Default authentication module

After a default installation of Qlik Sense, the Qlik Sense Proxy Service (QPS) includes a module that handles authentication of Microsoft Windows users. The module supports the use of Kerberos, NTLM, and basic authentication.

Certificate trust

Qlik Sense uses certificates for authentication. A certificate provides trust between nodes within a site.

This section describes how to deploy certificates for use in Qlik Sense.

Architecture

Certificates are used within a Qlik Sense site to authenticate communication between services that reside on different nodes. In addition, certificates can be used to build a trust domain between services that are located in different domains or areas (for example, internal networks, extranets, and Internet) without having to share a Microsoft Active Directory (AD) or other user directories.

The architecture is based on the master Qlik Sense Repository Service (QRS) on the central node acting as the certificate manager or Certificate Authority (CA). The master QRS creates and distributes certificates to all nodes within a site. The master QRS is therefore an important part of the security solution and has to be managed from a secure location to keep the certificate solution secure.

The root certificate for the installation is stored on the central node in the site, where the master QRS runs. All nodes with Qlik Sense services that are to be used within the site receive certificates signed with the root certificate when added to the master QRS. The master QRS (that is, the CA) issues digital certificates that contain keys and the identity of the owner. The private key is not made publicly available – it is kept secret by the nodes. The certificate enables the master QRS to validate the authenticity of the node. This means that the master QRS is responsible for making sure that a service that is deployed on a node is a service within the site.
After the nodes have received certificates, the communication between the Qlik Sense services is encrypted using Transport Layer Security (TLS) encryption.

**Requirements**

The requirements described in this section must be fulfilled for the certificate trust to function properly.

**General**

In Microsoft Windows environments, permission to access the certificate private key is a prerequisite for using Transport Layer Security (TLS). The access is needed both on the server-side and for certificate authentication between services.

**Communication ports**

To set up certificate trust, the Qlik Sense Repository Services (QRSs) require that the ports listed in the following table can be opened and used for communication. If any communication passes through a network firewall, the ports in the firewall must be opened and configured for the services.

<table>
<thead>
<tr>
<th>Port no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4570/TCP</td>
<td>Certificate password verification port, only used within multi-node sites by Qlik Sense Repository Services (QRSs) on nodes other than the central node to receive the password that unlocks a distributed certificate. The port can only be accessed from localhost and it is closed immediately after the certificate has been unlocked. The communication is always unencrypted.</td>
</tr>
<tr>
<td>4444/TCP</td>
<td>Security distribution port, only used by Qlik Sense Repository Services (QRSs) on nodes other than the central node to receive a certificate from the master QRS on the central node. The communication is always unencrypted, but the transferred certificate package is password-protected.</td>
</tr>
</tbody>
</table>

The Qlik Sense services use the following protocols:
The Qlik Sense Engine Service (QES) uses the Qlik Engine API over Transport Layer Security (TLS).

All other services use REST/JSON as the protocol over TLS.

See: Ports overview (page 25)

Unlocking distributed certificates
When adding a new node to a site, the distributed certificate needs to be unlocked.

See: Managing a Qlik Sense Site

Confirming certificates using Microsoft Management Console
Certificates can be visually confirmed in the Microsoft Management Console (MMC) with the certificate snap-in added.

If the certificates have been properly deployed, they are available in the locations listed in the table.

<table>
<thead>
<tr>
<th>Certificate</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>QlikClient</td>
<td>Certificates - Current User&gt;Personal&gt;Certificates</td>
</tr>
<tr>
<td>&lt;full computer name&gt;-CA</td>
<td>Certificates - Current User&gt;Trusted Root Certification Authorities&gt;Certificates</td>
</tr>
<tr>
<td>&lt;full computer name&gt;-CA</td>
<td>Certificates (Local Computer)&gt;Trusted Root Certification Authorities&gt;Certificates</td>
</tr>
<tr>
<td>&lt;computer name&gt;</td>
<td>Certificates (Local Computer)&gt;Personal&gt;Certificates</td>
</tr>
</tbody>
</table>

Handling of certificates when a service starts
This section describes how the certificates are handled when a Qlik Sense service starts.

Client certificate
This section describes how the master Qlik Sense Repository Service (QRS) on the central node in a site handles the client certificate when a Qlik Sense service starts.

The client certificate is located in the following place in the Microsoft Windows certificate store:

Current User>Personal>Certificates

When a Qlik Sense service starts, the QRS searches the certificate store to see if there are any Qlik Sense certificates. Depending on the results of the search, the QRS does the following:
If no client certificate is found, the QRS creates a new certificate.

If only one client certificate is found, the QRS checks if it is valid. If the certificate is not valid, the QRS deletes the certificate and creates a new one. In addition, the QRS logs that an invalid certificate was found and deleted.

If more than one client certificate is found, the QRS deletes all certificates and creates a new one. Duplicates are not allowed. In addition, the QRS logs the number of valid and invalid certificates that were found and deleted.

See also:

- Definition of invalid certificate (page 82)
- Services (page 14)

Server certificate

This section describes how the master Qlik Sense Repository Service (QRS) on the central node in a site handles the server certificate when a Qlik Sense service starts.

The server certificate is located in the following place in the Microsoft Windows certificate store:

Local Computer>Personal>Certificates

When a Qlik Sense service starts, the QRS searches the certificate store to see if there are any Qlik Sense certificates. Depending on the results of the search, the QRS does the following:

- If no server certificate is found, the QRS creates a new certificate.
- If only one server certificate is found, the QRS checks if it is valid. If the certificate is not valid, the QRS deletes the certificate and creates a new one. In addition, the QRS logs that an invalid certificate was found and deleted.
- If more than one server certificate is found, the QRS deletes all certificates and creates a new one. Duplicates are not allowed. In addition, the QRS logs the number of valid and invalid certificates that were found and deleted.

See also:

- Definition of invalid certificate (page 82)
- Services (page 14)

Root certificate

This section describes how the master Qlik Sense Repository Service (QRS) on the central node in a site handles the root certificate when a Qlik Sense service starts.

The root certificate is located in the following places in the Microsoft Windows certificate store:

Current User>Trusted Root Certification Authorities>Certificates
Local Computer>Trusted Root Certification Authorities>Certificates

When a Qlik Sense service starts, the QRS searches the certificate store to see if there are any Qlik Sense certificates. Depending on the results of the search, the QRS does the following:

- If no root certificate is found, the QRS creates a new certificate.
- If only one root certificate is found, the QRS checks if it is valid. If it is not valid, the QRS logs a fatal error that an invalid root certificate was found, which means that the service is shut down and that the administrator manually has to delete any unwanted certificates. In addition, the QRS logs information on the certificates that are affected by this.
- If more than one root certificate is found, the QRS logs a fatal error that an invalid root certificate was found, which means that the service is shut down and that the administrator manually has to delete any unwanted certificates. In addition, the QRS logs information on the certificates that are affected by this.

\[\text{In order not to break any certificate trust between machines, the QRS does not remove any root certificates. It is up to the administrator to decide on what to do with invalid root certificates.}\]

See also:

- Definition of invalid certificate (page 82)
- Services (page 14)

Definition of invalid certificate

The definition of an invalid certificate is as follows:

- The operating system considers the certificate to be too old or the certificate chain is incorrect or incomplete.
- The Qlik Sense certificate extension (OID “1.3.6.1.5.5.7.13.3”) is missing or does not reflect the location of the certificate:
  - Current User/Personal certificate location: Client
  - Local Machine/Personal certificate location: Server
  - Local Machine/Trusted Root certificate location: Root
  - Current User/Trusted Root certificate location: Root
- The server, client, and root certificates on the central node do not have a private key that the operating system allows them to access.
- The server and client certificates are not signed by the root certificate on the machine.
Maximum number of trusted root certificates
When a Qlik Sense service starts, it checks the number of trusted root certificates on the machine where it is running. If there are more than 300 certificates on the machine, warning messages containing the following information are logged:

- There are too many root certificates for the service to trust.
- The Microsoft Windows operating system will truncate the list of certificates during the Transport Layer Security (TLS) handshake.

If the Qlik Sense root certificate (\<host-machine>-CA) that the Qlik Sense client certificate belongs to is deleted from the list of certificates because of the truncation, the service cannot be authenticated.

To manually view the root certificates on a machine, open the Microsoft Management Console (MMC) and go to Certificates (Local Computer) > Trusted Root Certification Authorities.

Authentication solutions
This section describes various authentication solutions for Qlik Sense.

Ticket solution
The ticket solution is similar to a normal ticket. The user receives a ticket after having been verified. The user then brings the ticket to Qlik Sense and, if the ticket is valid, is authenticated. In order to keep the tickets secure, the following restrictions apply:

- A ticket is only valid for a short period of time.
- A ticket is only valid once.
- A ticket is random and therefore hard to guess.

All communication between the authentication module and the Qlik Sense Proxy Service (QPS) uses Transport Layer Security (TLS) and must be authorized using certificates.

The figure below shows a typical flow for authenticating a user with tickets.
6 Security

1. The user accesses Qlik Sense.
2. Qlik Sense redirects the user to the authentication module. The authentication module verifies the user identity and credentials with an identity provider.
3. Once the credentials have been verified, a ticket is requested from the QPS. Additional attributes may be supplied in the request.
4. The authentication module receives a ticket.
5. The user is redirected back to the QPS with the ticket. The QPS checks that the ticket is valid and has not timed out.
6. A session is created for the user.
7. The user is now authenticated.

Session solution

The session solution allows the Qlik Sense Proxy Service (QPS) to use a session from an external system to validate who the user is.

All communication between the authentication module and the QPS uses Transport Layer Security (TLS) and must be authorized using certificates.

The figure below shows a typical flow for authenticating a user using a session from an external system.
1. The user accesses the identity provider, which, for example, can be integrated into a portal. The identity provider gets the user identity and credentials and then verifies them. After that, the identity provider creates a new session.

2. The identity provider registers the session token with the Qlik Sense session module.

3. The identity provider sets the session token as a session cookie.

4. The user accesses the QPS to get content (for example, through an iframe in the portal).

5. The QPS validates the session to the session module.

6. If the session is valid and has not yet timed out, the user is authenticated.

The name of the session cookie used by the authentication module can be configured in the Qlik Management Console (QMC).

Header solution

Header authentication is often used in conjunction with a Single Sign-On (SSO) system that supplies a reverse proxy or filter for authenticating the user.

The figure below shows a typical flow for authenticating a user using header authentication.
1. The user accesses the system and authenticates to the reverse proxy.
2. The reverse proxy injects the username into a defined http header. The header must be included in every request to the Qlik Sense Proxy Service (QPS).
3. The user is authenticated.

For this solution to be secure, the end-user must not be able to communicate directly with the QPS but instead be forced to go through the reverse proxy/filter.

The name of the http header used for the user can be configured in the Qlik Management Console (QMC).

Anonymous users
If anonymous use of Qlik Sense is allowed, users who are not authenticated are not automatically redirected to an authentication module. Instead, the user first gets anonymous access and is then, if the user chooses to sign in, redirected to the authentication module to supply user identity and credentials.

6.3 Authorization
Authorization is the procedure of granting or denying users access to resources.
In Qlik Sense, authentication and authorization are two distinct, unconnected actions. In addition, the sources of information used for authentication do not have to be the same as for authorization, and the other way around.

In Qlik Sense, there are two authorization systems:

- **Access control**: The access control system grants users access to the resources in Qlik Sense. The access control system is implemented in the Qlik Sense Repository Service (QRS) and is, hence, independent of the operating system.
- **Data reduction**: The data reduction functionality is based on the concept of section access, which is a way to dynamically change which data a user can view. This makes it possible to build apps that can be consumed by many users, but with different data sets that are dynamically created based on user information. The reduction of data is performed by the Qlik Sense Engine Service (QES).

The two authorization systems are unconnected and configured separately.

**Access control**

This section describes the different types of access control:

- **Resource access control**: Is the user allowed to access the app? Which functions in the app is the user allowed to use (for example, printing, exporting, and snapshots)?
- **Administrator access control**: Which access rights are needed for the different roles and responsibilities of the administrators?

**Resource access control**

The resource access control system in Qlik Sense is based on attributes. This means that the access is based on rules that refer to attributes connected to resources and users in Qlik Sense.

All authorization to resources is enforced by the Qlik Sense Repository Service (QRS). The QRS only gives other Qlik Sense services access to resources that the current user is allowed to access.

The resource access control system determines the access based on the following parameters:

- **User name and user properties**: The user name and user properties are supplied by the Qlik Sense Proxy Service (QPS) that authenticated the user.
- **Action**: The method that the user is trying to perform on a resource (for example, create, read, or print).
- **Resource**: The entity that the user is trying to perform an action on (for example, app, sheet, or object).
- **Environment**: The environment is supplied by the QPS and describes, for example, time, location, protection, and the type of Qlik Sense web client used.

**Rules**

The system administrator can set up rules for the resources access control. The rules are divided into three parts:
• Resource filter: The resources that the rule applies to.
• Condition: A logical condition that, if evaluated as true, grants access.
• Action: The action that the user is allowed to perform, if the condition is true.

Attributes connected to resources or users may be used in the rules. Examples of attributes include the name of user or resource, type of resource, and Active Directory groups for users or custom-defined attributes.

Streams
To make the management of the Qlik Sense authorization systems efficient, apps can be grouped into streams. From an authorization perspective, a stream is a grouping of apps that a group of users has read (often referred to as “subscription”) or publish access to.

By default, Qlik Sense includes the following streams:

• Everyone: All users have read and publish rights to this stream.
• Administration: Contains a number of apps for monitoring of Qlik Sense.

Streams are created and managed in the Qlik Management Console (QMC).
Administrator access control

In addition to setting up the access control for the users, it is important to configure the access control for the administrators so that they get access rights in the Qlik Management Console (QMC) that correspond to their roles and responsibilities.

Common administrator roles include:

- **RootAdmin**: Full access to all Qlik Sense resources.
- **AuditAdmin**: Read access to all resources.
- **ContentAdmin**: Full access to all resources except nodes, engines, repositories, schedulers, and syncs.
- **DeploymentAdmin**: Full access to apps, tasks, licenses, nodes, repositories, schedulers, proxies, virtual proxies, and engines.
- **SecurityAdmin**: Same as ContentAdmin, but with full access to proxies and virtual proxies and no access to tasks.

Data reduction

Data reduction is used to determine which data a user is allowed to see: all of it or just parts of it?

The data reduction functionality is based on the concept of section access, which is a way to dynamically change which data a user can view. This makes it possible to build apps that can be consumed by many users, but with different data sets that are dynamically created based on user information. The reduction of data is performed by the Qlik Sense Engine Service (QES).

The definition of access rights for section access is maintained in the apps and configured through the load script.
6.4 Security summary

This section provides a summary of the Qlik Sense security system.

Authentication

Qlik Sense supports authentication in the following ways:

- The users are authenticated by the Qlik Sense Proxy Service (QPS).
- The QPS supports the use of multiple proxies and each proxy can use multiple authentication methods over a network protected by Transport Layer Security (TLS).

Authorization

Qlik Sense supports authorization in the following ways:

- Inter-server communication is authorized through Transport Layer Security (TLS) using certificates for authentication.
- The Qlik Sense Repository Service (QRS) provides attribute-based access control of user content.
- Authorization to data is managed using section access.

Auditing

Qlik Sense supports auditing in the following ways:

- The repository database stores information about when the database was last changed and who made the change.
- The logging framework provides audit and security logs.
- The logs are centrally stored.
- The log format is resistant to injection from the Qlik Sense web clients.
- The license logs are signed with a signature to protect them from tampering.

Confidentiality

Qlik Sense supports confidentiality in the following ways:

- The network uses Transport Layer Security (TLS) for encryption and certificates for authentication.
- The locally stored information on a node, including Qlik Sense content, is protected by the operating system using server access control and file system controls.
- The process memory and loaded data for Qlik Sense are protected by the physical server and the operating system controls.
- The apps are secured using access control on the resource level.
- Sensitive information (for example, passwords and connection strings) that is used to access external data sources is stored with encryption.
- The app data is protected using data reduction.
Integrity

Qlik Sense supports integrity in the following ways:

- Stored data is protected using the operating system controls (for example, the file system).
- Sensitive information (for example, passwords and connection strings) that is used to access external data sources is stored with encryption.
- Qlik Sense does not support write back to the source system (that is, the Qlik Sense web clients cannot edit the data sources).

Availability

Qlik Sense supports availability in the following ways:

- The nodes in a multi-node site are resilient by design. Each node has a local copy of the data that it needs to fulfill its role, which means that the node can operate independently in the event of a server or network failure.
- The Qlik Sense protocols are designed to be fault tolerant.
- Qlik Sense supports load sharing and failover between servers.

Security example: Opening an app

The figure below shows the flow in the Qlik Sense security system when a user logs in and opens a app.
1. Authentication: The authentication module in the Qlik Sense Proxy Service (QPS) handles the authentication. The credentials provided by the user are verified against information from the identity provider (for example, a directory service such as Microsoft Active Directory).

2. Session creation: When the user credentials have been successfully verified by the authentication module, a session is created for the user by the session module in the QPS.

3. Access control system: When the user tries to open an app, the Qlik Sense Engine Service (QES) requests the Qlik Sense Repository Service (QRS) to check if the user is authorized to perform the action. The QRS then checks the repository database, where, among other things, all users and access rights are stored.

   The users are imported into the repository database from one or more User Directories (UDs) (for example, Microsoft Active Directory) using Qlik Sense User Directory Connectors (UDCs). The import is triggered by the Qlik Sense Scheduler Service (QSS) and the intervals in-between imports can be scheduled.

4. Dynamic data reduction: When the user has been successfully authorized by the QRS, the app is opened. Before the data is displayed to the user, the QES performs a dynamic data reduction, where the data that the user is allowed to see is prepared.

See also:

- App security (page 76)
7 Logging

The log messages produced by Qlik Sense provide important information that can be used to detect security incidents, operational problems, and policy violations.

The logging is based on the log4net component in Apache Logging Services. This means that Qlik Sense uses a standardized logging framework and conforms to standard logging procedures.

The log files can be read and analyzed using Qlik Sense.

See also:

- Apache Logging Services

7.1 Requirements

The requirements described in this section must be fulfilled for the Qlik Sense logging to function properly.

Securing the file system

The system administrator must secure the file system so that the log files cannot be tampered with.

> By default, the account used for the Qlik Sense installation gets full permissions for the log folder, %ProgramData%\Qlik\Sense\Log, whereas the Users group only gets read permission. No other accounts or users get any permissions for the log folder.

Synchronizing time

The nodes within a Qlik Sense site must have synchronized time.

For the date and time stamps to be correct, all nodes within a site must be configured to synchronize their system clocks with either an internal or an external Network Time Protocol (NTP) service to ensure that all log entries are time-stamped accurately. NTP is a networking protocol for synchronizing the clocks of computer systems over packet-switched, variable-latency data networks.

Setting time zone

It is recommended that every node within a Qlik Sense site is set to the correct time zone so that the time zone corresponds to the geographical location of the node.

7.2 Log file format

This section describes the format of the Qlik Sense log files.
Storage

The default log files are stored in `%ProgramData%\Qlik\Sense\Log\<Service>.

The local log configuration file can be used to set up the logging so that the log files are not only stored in the default location, but also in another location.

See also:

- Local log configuration file (page 109)

Naming

The Qlik Sense log files are named in accordance to the following file rollover procedure:

1. The log is stored in a file named `<Machine>_<Facility>_<Service>.txt.
2. When the file is full or a pre-defined amount of time has passed, the file extension is automatically changed to .log and a time stamp is appended to the file name for uniqueness and archiving. This means that the new file name becomes `<Machine>_<Facility>_<Service>_<YYYY-MM-DDTHH.mm.ss>Z.log`. The file is then moved to the repository database on the central node by the Qlik Sense Repository Service (QRS).
3. A new log file, named `<Machine>_<Facility>_<Service>.txt`, is created.

If the .log file is deleted before being copied to the repository database on the central node, the file is lost and cannot be recreated.

The format of the file name is as follows:

- `<Machine>` = Name of the server where the log was created.
- `<Facility>` = The type of events that are covered by the log.
  See: Logger (page 97)
- `<Service>` = The service that the log originates from (for example, Proxy or Repository).
- `<YYYY-MM-DDTHH.mm.ss>Z` = Time stamp for when the log file was closed for new entries, where:
  - `YYYY`: Year
  - `MM`: Month
  - `DD`: Day in month
  - `T`: Delimiter, time designator
  - `HH`: Hour
  - `mm`: Minutes
7 Logging

- **ss**: Seconds
- **Z**: UTC designator, indicates that the time stamp is in UTC format

See also:

- Logger (page 97)

Structure

This section describes the rows and fields used in the Qlik Sense log files.

Rows

The first row of each log file contains a header that, in turn, contains the names of all fields, separated by tabs.

Each log entry is one row and the characters listed in the following table are replaced with Unicode characters.

<table>
<thead>
<tr>
<th>Character</th>
<th>Unicode replacement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\t</td>
<td>\u21d4</td>
<td>Symbol for horizontal tabulation, HT.</td>
</tr>
<tr>
<td>\n</td>
<td>\u2193</td>
<td>Symbol for line feed, LF.</td>
</tr>
<tr>
<td>\f</td>
<td>\u2192</td>
<td>Symbol for form feed, FF.</td>
</tr>
<tr>
<td>\r</td>
<td>\u21b5</td>
<td>Symbol for carriage return, CR.</td>
</tr>
</tbody>
</table>

Common fields

The following table lists the fields (in order of appearance) included in all log entries.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence#</td>
<td>1 - 2147483647 by default, but can be configured using custom logging as described in Appenders (page 106). Each row in the log starts with a sequence number that is used to ensure that the log is not tampered with (that is, that no rows are inserted or deleted). The sequence number wraps either when the last sequence number is reached or when the logging, for some reason, is restarted without the last sequence number being reached.</td>
</tr>
</tbody>
</table>
## 7 Logging

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestamp</td>
<td>Timestamp in ISO 8601 format, <code>YYYYMMDDTHHMMSS.fffK</code>, where:</td>
</tr>
<tr>
<td></td>
<td>- <code>YYYY</code>: Year</td>
</tr>
<tr>
<td></td>
<td>- <code>MM</code>: Month</td>
</tr>
<tr>
<td></td>
<td>- <code>DD</code>: Day in month</td>
</tr>
<tr>
<td></td>
<td>- <code>T</code>: Delimiter</td>
</tr>
<tr>
<td></td>
<td>- <code>HH</code>: Hour</td>
</tr>
<tr>
<td></td>
<td>- <code>MM</code>: Minutes</td>
</tr>
<tr>
<td></td>
<td>- <code>SS</code>: Seconds</td>
</tr>
<tr>
<td></td>
<td>- <code>fff</code>: Milliseconds</td>
</tr>
<tr>
<td></td>
<td>- <code>K</code>: Time zone offset</td>
</tr>
<tr>
<td></td>
<td>For example, <code>20110805T145657.000+02:00</code> means year 2011, month 8, day 5 at</td>
</tr>
<tr>
<td></td>
<td>14:56:57 GMT+2.</td>
</tr>
</tbody>
</table>
| Level       | Row log level, can be configured using custom logging as described in [Appenders (page 106)](page_106):
<p>|             | - Debug: Information useful to developers for debugging purposes. This level is |
|             |   not useful during normal operation since it generates vast amounts of logging |
|             |   information.                                                              |
|             | - Info: Normal operational messages that may be harvested for reporting,    |
|             |   measuring throughput, and so on. No action required.                      |
|             | - Warn: Not an error message, but an indication that an error may occur, if  |
|             |   no action is taken (for example, the file system is 85% full). Each item must |
|             |   be resolved within a given time.                                          |
|             | - Error: Non-urgent failures that are relayed to developers or administrators.|
|             |   Each item must be resolved within a given time.                          |
|             | - Fatal: Indicates a failure in a primary system (for example, loss of      |
|             |   primary ISP connection) and must be corrected immediately.               |
|             | - Off: No logs, except for license logs, are produced.                      |
| Hostname    | Server name.                                                                |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logger</td>
<td>Logger in <code>&lt;facility&gt;..&lt;service&gt;..&lt;fully qualified name of class&gt;</code> format, where:</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;facility&gt;</code>:</td>
</tr>
<tr>
<td></td>
<td>• Application: Log events that are related to the app running in Qlik Sense.</td>
</tr>
<tr>
<td></td>
<td>• Audit: Log events that provide an audit trail of a user’s activities and administration of the Qlik Sense platform.</td>
</tr>
<tr>
<td></td>
<td>• Exit: Log events that are related to the shutdown of the Qlik Sense Engine Service (QES).</td>
</tr>
<tr>
<td></td>
<td>• License: Log events that are related to the Qlik Sense license.</td>
</tr>
<tr>
<td></td>
<td>• ManagementConsole: Log events that are related to the Qlik Management Console (QMC).</td>
</tr>
<tr>
<td></td>
<td>• Performance: Log events that are related to the performance of the Qlik Sense platform or app.</td>
</tr>
<tr>
<td></td>
<td>• QixPerformance: Log events that are related to the performance of the QIX protocol in the QES.</td>
</tr>
<tr>
<td></td>
<td>• Security: Log events that are related to security issues.</td>
</tr>
<tr>
<td></td>
<td>• Session: Log events that are related to the termination of a client session.</td>
</tr>
<tr>
<td></td>
<td>• Synchronization: Log events that are related to the synchronization of the Qlik Sense Repository Service (QRS) instances in a multi-node site.</td>
</tr>
<tr>
<td></td>
<td>• System: Log events that are related to the Qlik Sense platform and not to the app running on the platform (for example, log messages related to the QMC, QRS, Qlik Sense Proxy Service (QPS), and so on).</td>
</tr>
<tr>
<td></td>
<td>• TaskExecution: Log events that are related to the execution of tasks by the Qlik Sense Scheduler Service (QSS).</td>
</tr>
<tr>
<td></td>
<td>• Traffic: Log events that are related to debugging.</td>
</tr>
<tr>
<td></td>
<td>• UserManagement: Log events that are related to the management of the users.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;service&gt;</code>: The Qlik Sense service that the log originates from (for example, QRS or QPS).</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;fully qualified name of class&gt;</code>: Indicates the part of the service that generated the log message.</td>
</tr>
<tr>
<td>Thread</td>
<td>Thread name or Managed Thread ID (if available).</td>
</tr>
<tr>
<td>Id</td>
<td>Globally Unique Identifier (GUID) for the log message.</td>
</tr>
<tr>
<td>ServiceUser</td>
<td>Name of the user or account used by the Qlik Sense service.</td>
</tr>
<tr>
<td>Message</td>
<td>Log message.</td>
</tr>
<tr>
<td>Exception</td>
<td>Exception message.</td>
</tr>
</tbody>
</table>

*This field is only present when there is an exception message.*
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StackTrace</td>
<td>A trace to the place in Qlik Sense where the exception occurred.</td>
</tr>
<tr>
<td></td>
<td><em>This field is only present when the Exception field is present.</em></td>
</tr>
<tr>
<td>ProxySessionId</td>
<td>The ID of the Qlik Sense Proxy Service (QPS) session for the user.</td>
</tr>
<tr>
<td></td>
<td><em>This field is not present in all log files.</em></td>
</tr>
<tr>
<td>Id2 or Checksum</td>
<td>The last field in a log entry either contains an Id2 or a Checksum:</td>
</tr>
<tr>
<td></td>
<td>• Id2: Log message GUID (same as Id described earlier). This is the normal value.</td>
</tr>
<tr>
<td></td>
<td>• Checksum: To protect logs that contain sensitive information (for example, audit, security,</td>
</tr>
<tr>
<td></td>
<td>and license logs) from tampering, the last field in such log entries contains a cryptographic</td>
</tr>
<tr>
<td></td>
<td>hash of the entire row up to the hash itself.</td>
</tr>
</tbody>
</table>

See also:
- Additional fields (page 98)
- Synchronization (page 30)

**Additional fields**
The logs produced by Qlik Sense have common fields that are present in all log files. Some logs contain additional fields, which are listed in this section. In addition, optional fields can be defined.

See also:
- Common fields (page 95)

**Application log**

**Qlik Sense Repository Service (QRS)**
The following fields are specific to the Application log for the QRS:

- Application: The name of the application (if there is a name to associate with the log entry).

**Qlik Sense Scheduler Service (QSS)**
The following fields are specific to the Application log for the QSS:

- TaskName: The name of the task that was executed.
- TaskId: The ID of the task that was executed.
7 Logging

- User: The name of the user who executed the task. When the QSS starts a scheduled execution of a task, the QSS is listed as user.
- ExecutionId: A unique ID that identifies the execution of the task. A task gets a new ExecutionId every time it is executed.
- AppName: The name of the app that executed the task (if any).
- AppId: The ID of the app that executed the task (if any).

See also:

- Common fields (page 95)

Audit log

**Qlik Sense Repository Service (QRS)**
The following fields are specific to the Audit log for the QRS:

- Action: The action that the user performed (add, update, delete, export).
- ActiveUserDirectory: The user directory for the user.
- ActiveUserId: The ID of the user.
- ResourceId: The ID of the resource on which the user performed the action.

**Qlik Sense Proxy Service (QPS)**
The following fields are specific to the Audit log for the QPS:

- ConnectionId: The ID of the connection.
  See: ActiveConnections field in Performance log (page 100)
- ActiveUserDirectory: The user directory for the user.
- ActiveUserId: The ID of the user.
- TicketId: The ID of the login ticket that was issued for the user. The ticket ID exists until it is consumed by the QPS.
- IpAddress: The IP address of the client.
- AppId: The ID of the app (empty if no app is loaded).
- TargetHost: The call from the client is forwarded to a Qlik Sense Engine Service (QES) or QRS. This field contains the name of the machine on which the service is running.
- VirtualProxy: The virtual proxy prefix in \{prefix\} format.

**Qlik Sense Engine Service (QES)**
The following fields are specific to the Audit log for the QES:

- ActiveUserDirectory: The user directory for the user.
- ActiveUserId: The ID of the user.
- EngineTimestamp: The time when the QES wrote the log message to file.
- EngineThread: The ID of the thread that was used when the QES wrote the log message to file.
• ProcessId: The ID of the QES process from which the log message originates.
• ServerStatus: The time when the QES started.
• AppId: The ID of the app.
• Type: The type of operation that the user performed to generate the audit message.
• Qlik Sense User: The user who generated the audit message.

See also:

▷ Common fields (page 95)

License log

**Qlik Sense Repository Service (QRS)**
The following fields are specific to the License log for the QRS:

• AccessTypeld: The ID of the access type entity.
• AccessType: The name of the access type (LoginAccess or UserAccess).
• Operation: The operation that was performed (Add, Update, Delete, Usage Granted, Usage Denied, Available, Timedout, or Unquarantined).
• UserName: The name of the user (who, for example, uses an access pass).
• UserId: The ID of the user in the repository database.

See also:

▷ Common fields (page 95)

Performance log

**Qlik Sense Repository Service (QRS)**
The following fields are specific to the Performance log for the QRS:

• Tracenumber: A unique ID for the call to the QRS REST API.
• Httpmethod: The http method that was used (Get, Put, Post, or Delete).
• Url: The URL that was used.
• Resourcetype: The type of resource.
• Method: The backend code that was called.
• Elapsedmilliseconds: The time (in milliseconds) to complete the call to the QRS REST API.

Example: Get http://mytest/cars/4

• Httpmethod: Get
• Url: http://mytest/cars/4
• Resource type: cars
• Method: get/cars/{0}

Qlik Sense Proxy Service (QPS)

The following fields are specific to the Performance log for the QPS:

• ActiveConnections: The number of active connections (in any form or shape) from the client.
  A connection is a stream (that is, a socket) between a Qlik Sense web client and the Qlik Sense Proxy Service (QPS). This stream is often connected to another stream, which runs from the QPS to the Qlik Sense Repository Service (QRS) or the Qlik Sense Engine Service (QES). The two streams allow the web client to communicate with the QRS or the QES.

• ActiveStreams: The number of active data streams (that is, sockets), either from the browser to the QPS or from the QPS to the QRS or the QES.

• ActiveSessions: The number of active sessions in the QPS.
  A Qlik Sense user gets a proxy session when the user has been authenticated. The session is terminated after a certain period of inactivity.

• LoadBalancingDecisions: The number of users who currently have at least one QES session.

• Tickets: The number of issued login tickets that have not yet been consumed.

• ActiveClientWebsockets: The number of active websockets between the client and the QPS.

• ActiveEngineWebsockets: The number of active websockets between the QPS and the target Qlik Sense service.

The logging entries are also available as metrics; see Metrics (page 19).

Qlik Sense Engine Service (QES)

Each entry (that is, row) in the Performance log corresponds to a snapshot (that is, a number of measurements) of the performance of the QES at the given point in time.

The following fields are specific to the Performance log for the QES:

• ActiveUserDirectory: The user directory for the user.
• ActiveUserId: The ID of the user.
• EngineTimestamp: The time when the QES wrote the log message to file.
• EngineThread: The ID of the thread that was used when the QES wrote the log message to file.
• ProcessId: The ID of the QES process from which the log message originates.
• Exe Type: The configuration type (release or debug version) of the QES process.
• Exe Version: The version number of the QES process.
• Server Started: The time when the QES started.
• Entry Type: The reason (Server Starting, Normal, or Server Shutting Down) for the log entry in the Performance log.
• ActiveDocSessions: The number of active app sessions in the QES at the given point in time.
• DocSessions: The number of app sessions in the QES at the given point in time.
7 Logging

- **ActiveAnonymousDocSessions**: The number of active anonymous app sessions in the QES at the given point in time.
- **AnonymousDocSessions**: The number of anonymous app sessions in the QES at the given point in time.
- **ActiveTunneledDocSessions**: The number of active tunneled app sessions in the QES at the given point in time.
- **TunneledDocSessions**: The number of tunneled app sessions in the QES at the given point in time.
- **DocSessionStarts**: The number of started app sessions in the QES since the previous snapshot.
- **ActiveDocs**: The number of active apps in the QES at the given point in time.
- **RefDocs**: The number of apps in the QES at the given point in time.
- **LoadedDocs**: The number of loaded apps in the QES at the given point in time.
- **DocLoadFails**: The number of failed app loads in the QES since the previous snapshot.
- **Calls**: The number of calls to the QES since the previous snapshot.
- **Selections**: The number of selections in the QES since the previous snapshot.
- **ActiveIpAddrs**: The number of IP addresses of active connected clients in the QES at the given point in time.
- **IpAddrs**: The number of IP addresses of all connected clients in the QES at the given point in time.
- **ActiveUsers**: The number of active users in the QES at the given point in time.
- **Users**: The total number of users in the QES at the given point in time.
- **CPULoad**: A measurement of the load on the CPU on which the QES runs at the given point in time.
- **VMCommitted(MB)**: The committed Virtual Memory (in megabytes) at the given point in time.
- **VMAvailable(MB)**: The allocated Virtual Memory (in megabytes) at the given point in time.
- **VMFree(MB)**: The freed Virtual Memory (in megabytes) at the given point in time.
- **VMLargestFreeBlock(MB)**: The largest freed Virtual Memory block (in megabytes) at the given point in time.

**See also:**

- Common fields (page 95)

**Qlik performance log**

**Qlik Sense Engine Service (QES)**

The following fields are specific to the QIIX performance log for the QES:

- **ActiveUserDirectory**: The user directory for the user.
- **ActiveUserId**: The ID of the user.
- **EngineTimestamp**: The time when the QES wrote the log message to file.
- **EngineThread**: The ID of the thread that was used when the QES wrote the log message to file.
- **ProcessId**: The ID of the QES process from which the log message originates.
- **CServerId**: The ID of the server instance that handled the request.
- SessionId: The ID of the session for which the QIX method call was made.
- Server Started: The time when the QES started.
- Method: The name of the QIX method that was called.
- RequestId: The ID of the request in which the QIX method call was handled.
- Target: The memory address of the target for the QIX method call.
- RequestException: The ID of an exception (if any) that occurred as a result of the QIX method call.
- ProcessTime: The amount of time that was needed to process the request.
- WorkTime: The amount of time that the request did actual work.
- LockTime: The amount of time that the request had to wait for an internal lock.
- ValidateTime: The amount of time that the request used for validation.
- Handle: The ID of the interface that handled the request.

See also:

- Common fields (page 95)

Qlik Management Console log

*The Qlik Management Console log is not created until there is an event (for example, an error message) for the Qlik Management Console (QMC) to write in the log.*

Qlik Sense Repository Service (QRS)
The following fields are specific to the Qlik Management Console log for the QRS:

- Browser: The web browser that is used to run the QMC.

See also:

- Common fields (page 95)

Session log

Qlik Sense Engine Service (QES)
The following fields are specific to the Session log for the QES:

- ActiveUserDirectory: The user directory for the user.
- ActiveUserId: The ID of the user.
- EngineTimestamp: The time when the QES wrote the log message to file.
- EngineThread: The ID of the thread that was used when the QES wrote the log message to file.
- ProcessId: The ID of the QES process from which the log message originates.
- Exe Type: The configuration type (release or debug version) of the QES process.
- Exe Version: The version number of the QES process.
- Server Started: The time when the QES started.
- AppId: The ID of the app that was loaded by the finished session.
- App Title: The title of the loaded app that was used during the finished session.
- Doc Timestamp: The last modified timestamp of the app that was loaded by the finished session.
- Qlik Sense User: The user that started the finished session.
- Exit Reason: The reason for the session to finish.
- Session Start: The time when the session started.
- Session Duration: The duration (in milliseconds) of the finished session.
- CPU Spent (s): The CPU time (in seconds) that was spent handling requests during the finished session.
- Bytes Received: The number of bytes of data that were received during the session.
- Bytes Sent: The number of bytes of data that were sent during the session.
- Calls: The number of calls that were made during the session.
- Selections: The number of selections that were made during the session.
- Authenticated User: The authenticated user that used the session.
- Client Machine Identification: The ID of the client machine that opened the session.
- Serial Number: The serial number that was used during the session.
- Client Type: The type of client that was used for the session.
- Client Build Version: The build version of the client.
- Secure Protocol: An on/off flag that indicates whether the protocol was run over a secure connection.

See also:

- Common fields (page 95)

System log

Qlik Sense Engine Service (QES)
The following fields are specific to the System log for the QES:

- ActiveUserDirectory: The user directory for the active user who was logged in when the log message was generated in the QES.
- ActiveUserId: The user ID for the active user who was logged in when the log message was generated in the QES.
- EngineTimestamp: The time when the QES wrote the log message to file.
- EngineThread: The ID of the thread that was used when the QES wrote the log message to file.
- ProcessId: The ID of the QES process from which the log message originates.
- Server Started: The time when the QES started.

See also:

- Common fields (page 95)
Task execution log

Qlik Sense Scheduler Service (QSS)
The following fields are specific to the Task execution log for the QSS:

- **TaskId**: A unique ID of the task that was executed.
- **TaskName**: The name of the task that was executed.
- **AppId**: A unique ID of the app that executed the task (if any).
- **AppName**: The name of the app that executed the task (if any).
- **ExecutionId**: A unique ID that identifies the execution of a task. A task gets a new ExecutionId every time it is executed.
- **ExecutionNodeId**: A unique ID that identifies the node in the site on which the specific execution of the task was performed.
- **Status**: The result of the execution of the task (successful, failed, aborted, skipped, or retry).
- **StartTime**: The time when the execution of the task started.
- **StopTime**: The time when the execution of the task stopped.
- **Duration**: The time (in milliseconds) for the execution of the task to be completed.
- **FailureReason**: Empty, unless an error occurred during the execution of the task.

See also:

- Common fields (page 95)

Traffic log

Qlik Sense Engine Service (QES)
The following fields are specific to the traffic log for the QES:

- **ActiveUserDirectory**: The user directory for the user.
- **ActiveUserId**: The ID of the user.
- **EngineTimestamp**: The time when the QES wrote the log message to file.
- **EngineThread**: The ID of the thread that was used when the QES wrote the log message to file.
- **ProcessId**: The ID of the QES process from which the log message originates.

See also:

- Common fields (page 95)

7.3 Configuring the logging

The standard logging in Qlik Sense is configured using the Qlik Management Console (QMC).

Customized logging is setup using appenders and the local log configuration file, LocalLogConfig.xml.
Appenders

The logging in Qlik Sense implements a custom appender, QSRollingFileAppender, which is based on the log4net component. The custom appender is used internally by the Qlik Sense logging system.

QSRollingFileAppender and some of the built-in, predefined appenders in the log4net framework can be used to configure customized logging, which is specified in the local log configuration file, LocalLogConfig.xml.

QSRollingFileAppender can also log events in the local log file (for example, the Microsoft Windows event log) or send log information to a remote log server.

QSRollingFileAppender

QSRollingFileAppender inherits from log4net.Appenders.FileAppender and all parameters, except for AppendToFile, are also available to QSRollingFileAppender. QSRollingFileAppender stores the log files in accordance to the MaxFileSize and MaxFileTime parameters.

Configuring the appender

The QSRollingFileAppender configuration is as follows:

```xml
<appender name="MyQSRollingFileAppender" type="Qlik.Sense.Logging.log4net.Appender.QSRollingFileAppender">
  <param name="threshold" value="info" />
  <param name="encoding" value="utf-8" />
  <param name="file" value="C:/ProgramData/Qlik/Sense/Log/output.log" />
  <param name="maximumfiletime" value="720" />
  <param name="maximumfilesize" value="512KB" />
  <layout type="log4net.Layout.PatternLayout">
    <converter>
      <param name="name" value="rownum" />
      <param name="type" value="Qlik.Sense.Logging.log4net.Layout.Pattern.CounterPatternConverter" />
    </converter>
    <converter>
      <param name="name" value="longIso8601date" />
      <param name="type" value="Qlik.Sense.Logging.log4net.Layout.Pattern.Iso8601TimeOffsetPatternConverter" />
    </converter>
    <converter>
      <param name="name" value="hostname" />
      <param name="type" value="Qlik.Sense.Logging.log4net.Layout.Pattern.HostNamePatternConverter" />
    </converter>
    <converter>
      <param name="name" value="guid" />
      <param name="type" value="Qlik.Sense.Logging.log4net.Layout.Pattern.GuidPatternConverter" />
    </converter>
    <converter>
      <param name="name" value="user" />
      <param name="type" value="Qlik.Sense.Logging.log4net.Layout.Pattern.ServiceUserNameCachedPatternConverter" />
    </converter>
  </layout>
</appender>
```
<param name="name" value="encodedmessage" />
</converter>

<converter>
<param name="name" value="encodedexception" />
<param name="type" value="Qlik.Sense.Logging.log4net.Layout.Pattern.EncodedExceptionPatternConverter" />
</converter>

<param name="ignoresexception" value="false" />

<param name="header" value="Sequence#&nbsp;Timestamp&amp;nbsp;Level&amp;nbsp;Hostname&amp;nbsp;Logger&amp;nbsp;Thread&amp;nbsp;Id&amp;nbsp;User&amp;nbsp;Message&amp;nbsp;Exception&amp;nbsp;Id2
" />

<param name="conversionpattern" value="%rownum(9999) &amp;nbsp;%longIso8601date&amp;nbsp;%level&amp;nbsp;%hostname&amp;nbsp;%logger&amp;nbsp;%thread&amp;nbsp;%guid&amp;nbsp;%user&amp;nbsp;%encodedmessage&amp;nbsp;%encodedexception[innermostmessage]&amp;nbsp;%guid%newline" />
</layout>
</appender>

Converters

log4net.Layout.PatternLayout and a couple of custom converters are used to format rows in logs based on QSRollingFileAppender:

- Qlik.Sense.Logging.log4net.Layout.Pattern.CounterPatternConverter: Add a sequence number to the log row. Parameter:
  - Integer: The last number of the sequence before it is reset.


- Qlik.Sense.Logging.log4net.Layout.Pattern.HostNamePatternConverter: Add the host name to the log row.


- Qlik.Sense.Logging.log4net.Layout.Pattern.EncodedExceptionPatternConverter: Add information on the logged exception to the log row. Parameter (one of the following):
  - MESSAGE: The message in the logged exception.
  - INNERMOSTMESSAGE: The message in the innermost exception of the logged exception.
  - SOURCE: The source of the exception (that is, the name of the app or the object that caused the error).
  - STACKTRACE: The stack trace for the exception.
  - TARGETSITE: The target site for the exception (that is, the method that threw the current exception).
  - HELPLINK: Link to the help file associated with the exception.
Built-in log4net appenders

In addition to the Qlik Sense custom appender, QSRollingFileAppender, the log4net framework comes with a set of built-in predefined appenders that also can be used in the local log configuration file, LocalLogConfig.xml:

- AdoNetAppender
- AnsiColorTerminalAppender
- AspNetTraceAppender
- ColoredConsoleAppender
- ConsoleAppender
- EventLogAppender
- FileAppender
- NetSendAppender
- RemoteSyslogAppender
- RemotingAppender
- RollingFileAppender
- SmtpAppender
- SmtpPickupDirAppender
- TelnetAppender
- UdpAppender

Each appender has its own set of parameters to control the output.

See also:

- Apache Logging Services

Example: EventLogAppender

The following example shows how to use the EventLogAppender in the local log configuration file, LocalLogConfig.xml, for the Qlik Sense Proxy Service (QPS). In the example, all QPS audit log entries at warning level are sent to the Microsoft Windows event log.

```xml
<appender name="EventLogAppender" type="log4net.Appender.EventLogAppender">
   <param name="threshold" value="warn" />
   <param name="applicationName" value="Qlik Sense Proxy Service" />
   <layout type="log4net.Layout.PatternLayout">
      <param name="conversionPattern" value="%message" />
   </layout>
</appender>

<logger name="Audit.Proxy">
   <appender-ref ref="EventLogAppender" />
</logger>
```
Example: SmtpAppender

The following example shows how to use the SmtpAppender in the local log configuration file, `LocalLogConfig.xml`, for the Qlik Sense Proxy Service (QPS). In the example, all QPS audit log entries at warning level are sent to an email address (to@domain.com).

```xml
<appender name="MyMailAppender" type="log4net.Appender.SmtpAppender">
  <param name="threshold" value="warn" />
  <param name="to" value="to@domain.com" />
  <param name="from" value="from@domain.com" />
  <param name="subject" value="test logging message" />
  <param name="smtpHost" value="SMTPServer.domain.com" />
  <param name="port" value="25" />
  <param name="buffersize" value="512" />
  <param name="lossy" value="true" />
  <layout type="log4net.Layout.PatternLayout">
    <param name="conversionPattern" value="%newline%date %-5level %message%newline%newline%newline" />
  </layout>
</appender>

<logger name="Audit.Proxy">
  <appender-ref ref="MyMailAppender" />
</logger>
```

Local log configuration file

The logging in Qlik Sense can be set up to produce customized logging as an addition to the default logging.

To set up customized logging, create a local log configuration file named `LocalLogConfig.xml` in the `%ProgramData%\Qlik\Sense\<Service>` folder.

> The logging defined by the local log configuration file does not affect the default logging.

Requirements

The requirements described in this section must be fulfilled for the customized logging to function properly.

Conforming to the XML schema

The local log configuration file must conform to the XML schema because the Qlik Sense Repository Service (QRS), Qlik Sense Proxy Service (QPS), and Qlik Sense Scheduler Service (QSS) have built-in schema validation.

If the local log configuration file is not accepted by the services, an error is logged in the System log.

Maximum file size

The size of the local log configuration file must not exceed 1 MB.

XML schema

The XML schema for the local log configuration file, `LocalLogConfig.xml`, is as follows:
In this example, the local log configuration file is configured to write the system logs at debug level in %ProgramData%\Qlik\Sense\Log\Proxy\Debug_System_Proxy.txt.

```xml
<?xml version="1.0"?>
<configuration>
  <appender name="LocalApp_AppenderSystem" type="Qlik.Sense.Logging.log4net.Appender.QSRollingFileAppender">
    <param name="threshold" value="debug" />
    <param name="encoding" value="utf-8" />
    <param name="file" value="C:\ProgramData\Qlik\Sense\Log\Proxy\Debug_System_Proxy.txt" />
    <param name="maximumfiletime" value="720" />
    <param name="maximumfilesize" value="512KB" />
    <layout type="log4net.Layout.PatternLayout">
      <converter value="Qlik.Sense.Logging.log4net.Layout.Pattern.CounterPatternConverter" />
      <converter value="Qlik.Sense.Logging.log4net.Layout.Pattern.HostNamePatternConverter" />
      <param name="ignoresexception" value="false" />
      <param name="header" value="Sequence#	Timestamp	Level	Hostname	Logger	Thread	Id	ServiceUser	Message	Exception
ActiveUserDirectory	ActiveUserId	ProxyTimestamp	ProxyThread" />
```

<param name="conversionpattern" value="%rownum\[9999]\%longIso8601date\%null;\%level\%hostname\%logger\%thread\%guid\%serviceuser\%encodedmessage\[1000000]\%encodedexception\[innermostmessage:1000000]\;\%property\[ActiveUserDirectory]\%property\[ActiveUserId]\%property\[ProxyTimestamp]\%property\[ProxyThread]\%guid\\newline" />
</layout>

<logger name="System.Proxy">
  <appender-ref ref="LocalAppenderSystem" />
</logger>
</configuration>

See also:

- Converters (page 107)
8 Licensing

The licensing in Qlik Sense is based on tokens, which are used to allocate access passes that allow users to access Qlik Sense. There are different types of access passes to choose from and each type corresponds to a specific consumption model for accessing Qlik Sense.

The tokens used in Qlik Sense are not compatible with the Client Access Licenses (CALs) used in QlikView. In addition, QlikView licenses cannot be used in Qlik Sense.

8.1 License Enabler File

The Qlik Sense licensing is administered using a License Enabler File (LEF), which holds the number of tokens available for the central node in a site. This means that a Qlik Sense site needs at least one (1) LEF.

The LEF can be downloaded when the serial number and the control number have been entered in the Qlik Management Console (QMC). The LEF can also be pasted directly into the QMC, if, for example, no network connection is available.

Increase in tokens

When the number of tokens in the LEF increases (for example, when buying additional tokens), the new tokens are added to the pool of unallocated tokens that can be used to allocate access passes that allow users to access Qlik Sense.

Decrease in tokens

When the number of tokens in the LEF decreases, the procedure is as follows:

1. Unallocated tokens are removed.
2. If step 1 is not enough to meet the decreased number of tokens in the LEF, the QMC informs the Qlik Sense administrator that allocated tokens have to be freed up within a certain period of time.
3. If the number of allocated tokens is still too large at the end of the period specified in step 2, access passes are removed until the number of allocated tokens equals the number of tokens specified in the LEF.

See also:

- Removing access passes (page 115)

8.2 Access passes

The licensing in Qlik Sense is based on tokens, which are used to allocate access passes that allow users to access Qlik Sense. There are different types of access passes to choose from and each type corresponds to a specific consumption model for accessing Qlik Sense.
A user connection is the combination of device and browser that is used by a single user to connect to Qlik Sense. If a user who already has a user connection connects to Qlik Sense from another browser or device, an additional user connection is established.

The following table lists the types of access passes that are available in Qlik Sense.

<table>
<thead>
<tr>
<th>Access type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User access pass</td>
<td>This type of access pass allows a unique and identified user to access the hub. The access pass is valid within an entire Qlik Sense site. For example, if a user first connects to a node in the USA and then, at a later stage, connects to a node in the UK, the user consumes the same access pass, if the two nodes are connected to the same central node. See: Site (page 11) The maximum number of parallel user connections for a single user of this type of access pass is five (5). One (1) token corresponds to one (1) user access pass. User access passes are allocated using the Qlik Management Console (QMC).</td>
</tr>
<tr>
<td>Login access pass</td>
<td>This type of access pass allows an identified or anonymous user to access the hub for a maximum of 60 continuous minutes per 28-day period. If the user exceeds the 60 minutes time limitation, the user connection does not time out. Instead, another login access pass is used. If no more login access passes are available, the user connection is discontinued. If an identified user is disconnected, the user can re-connect and continue to use the same access pass, if re-connecting within the 60 minutes. If an anonymous user is disconnected, the user gets a new access pass when re-connecting. The login access pass tracks the number of logins and runs over 28 days. For example, if 1000 logins are assigned to Group A, the users in Group A can use 1000 logins over 28 days. If 100 logins are consumed on Day 1, the 100 logins are available again on Day 29. The maximum number of parallel user connections for a single user of this type of access pass is five (5). Note that this only applies to identified users. An anonymous user can only have one (1) user connection. One (1) token corresponds to ten (10) login access passes. Login access passes are allocated using login access groups in the QMC.</td>
</tr>
</tbody>
</table>

**Allocation of access passes**

The following figure shows how the Qlik Management Console (QMC) is used to manage the allocation of access passes.
Login and logout

Login
When a user logs in to Qlik Sense, an access pass of the applicable type is used to provide the user with access to Qlik Sense.

Logout
When a user logs out of Qlik Sense, the following happens depending on the type of access pass used:
User access pass: The access pass is not affected when the user logs out.

Login access pass: The access pass that was used to access Qlik Sense is considered to be used and will not be available for a new login until the period specified in Login access pass (page 113) has passed.

Removing access passes
This section describes how to free up tokens for new allocations of access passes by removing existing access passes in the Qlik Management Console (QMC).

User access pass
When a user access pass is removed, it enters a quarantine for seven (7) days, counting from the last time that the access pass was used. For example, if the access pass is used on January 10, the tokens used to allocate the access pass are not available for new allocations until January 18.

During the quarantine period, the original allocation of the access pass can be reinstated, which means that the quarantine period ends and the user can start using the access pass again.

Login access pass
When a login access group is removed, the tokens used to allocate the access pass become available in accordance to the following procedure:

1. For every ten (10) unused login access passes, one (1) token is freed up.

2. For every ten (10) login access passes that leave the used state after the period specified in Login access pass (page 113) has passed, one (1) token is freed up.

Disconnected node
If a node becomes disconnected from the central node in a Qlik Sense site (that is, if the node fails to synchronize with the central node), the disconnected node serves up to its current allocation of licenses.

If the disconnected node cannot reach the central node within two (2) hours, the node does not allow any new user connections. When the two hour period has passed, it is still possible to connect to the disconnected node from localhost (that is, from the local computer on which the disconnected node is running).

When logging in to a disconnected node from localhost, the following applies:

- An authentication mechanism that does not require network access (for example, the authentication that is integrated in the Microsoft Windows operating system) is needed.
- Operations that require acknowledgment by the master Qlik Sense Repository Service (QRS) are disabled.
- The QRS handles the synchronization when the disconnected node is reconnected to the Qlik Sense site.
- The Qlik Sense licensing for any other nodes is not affected by the node being disconnected from the central node of the Qlik Sense site.
The disconnected node must connect to the central node of the Qlik Sense site at least every five (5) days. If the disconnected node fails to connect to the central node within the given time, it is no longer possible to log in to the disconnected node.

Multi-deployment sites
This section describes how the Qlik Sense licensing is handled within multi-deployment sites, where apps are promoted from a development site to a test site and finally to a production site.

Development site
In a Qlik Sense deployment that includes a development site and a production site, two (2) License Enabler Files (LEFs) are needed (that is, one per site).

Each node within the development site is licensed with one (1) access pass type (for example, user access passes), if only disconnected users are expected.

Test site
The License Enabler File (LEF) for a test site mirrors the LEF for a development site.

See also:
- Disconnected node (page 115)
- Deploying multi-node sites (page 30)

Anonymous users
Anonymous users only use login access passes.

See also:
- Login access pass (page 113)

8.3 Licensing metrics
The most current Qlik Sense license metrics are available at www.qlik.com/license-terms and incorporated by reference.