

QMATIC

Reference Manual

Orchestra 7.4

211.01F

Copyright notice

The information in this document is subject to change without prior notice and does not represent a commitment on the part of Q-MATIC AB.

All efforts have been made to ensure the accuracy of this manual, but Q-MATIC AB can not assume any responsibility for any errors and their consequences.

This manual is copyrighted and all rights are reserved.

Qmatic and Qmatic Orchestra are registered trademarks or trademarks of Q-MATIC AB.

Reproduction of any part of this manual, in any form, is not allowed, unless written permission is given by Q-MATIC AB.

COPYRIGHT © Q-MATIC AB, 2026.

❗ Send any feedback or questions about this manual to documentation@qmatic.com.

Contents

1. What's New	vii
2. Introduction.....	1
Welcome!	2
Customer Journey Management	2
System description	3
Distributed Operations.....	5
Deployment Scenarios	7
Supported Environments.....	9
Cloud Bridge.....	10
Accessibility.....	10
3. Installation.....	13
Introduction.....	14
Prerequisites - Checklist.....	19
Installation Scenarios	26
Orchestra and Stat on the Same Server	26
Orchestra and Stat on Separate Servers	36
Configuration before using Personal Data Management	47
Orchestra Queue Agent	48
Gateway 1745	53
Installing and Configuring Branch Hub and Hub	53
Redeploying Applications	53
Configuration Variables for Applications	53
4. Upgrade	55
Introduction.....	56
Upgrade Methodology	59
Orchestra Central	63
Upgrading from JBoss to Wildfly	70
Distributed Queue Agent	74
Unit Types	77
Widgets	79
5. System Administration	81
Unit Types	82
Licensing	84
Import and Export.....	90

Widget Administration	93
Queue Agents	97
Parameters.....	112
TrustStore Management	138
Mark Types	138
6. Auditing	141
Introduction	142
What is logged?	142
Where is it stored?	143
Enabling/disabling the functionality	145
7. Stat Aggregation	147
Introduction	148
Stat PDI Jobs/Transformations - Example Configuration.....	148
8. Localisation	155
Introduction	156
Orchestra system GUI.....	156
9. Xtend	161
Introduction	162
Requirements.....	162
10. Network Connectivity	163
Orchestra Bandwidth Requirements	164
11. Secure Communication	165
Introduction	166
Enable HTTPS for Orchestra Central.....	170
Secure communication between Central and distributed Queue Agent.....	178
Secure Communication for a Queue Agent	184
Disabling HTTP and Running HTTPS Only	198
Encryption of JMS connections.....	200
Bandwidth Requirements	202
Enable cookie secure flag	202
Changing the password for keystore.....	203
12. SSO setup	205
Introduction	206
SSO Setup Using Integrated Kerberos	206
SSO Setup Using SAML 2.0 Web SSO	211
13. HA setup	213
Introduction	214
Deployment architecture	215
Load balancer	215
Shared storage.....	217
RDBMS	217

Deployment procedure	217
Notification application	218
HA Proxy Example	219
14. Appendix A - Tuning Database Parameters	223
Tuning of Oracle Parameters	224
Tuning of PostgreSQL Parameters	225
15. Appendix B - Logging.....	231
Enabling Customer Journey Management Logging	232
Enabling Publish Logging	233
16. Appendix C - Security.....	235
Introduction	236
Content-Security-Policy.....	236
X-Frame-Options.....	236
X-Content-Type-Options	237
X-XSS-Protection	237
Strict-Transport-Security	238
Modifying HTTP Response Headers.....	238
17. Appendix D - Open LDAP Setup.....	241
Introduction	242
Install LDAP and Helper Utilities	242
Reconfigure slapd to Select Better Settings.....	242
Install phpLDAPadmin to Manage LDAP with a Web Interface	243
Configure phpLDAPadmin	243
Create an SSL Certificate	244
Create a Password Authentication File	245
Secure Apache.....	245
Modify the phpLDAPadmin Apache Configuration.....	246
Configure the HTTP Virtual Host.....	246
Configure the HTTPS Virtual Host File	247
Log in to the phpLDAPadmin Web Interface.....	248
Adding Nodes.....	249
Adding a User	251
Adding a Group.....	251
Troubleshooting	252
18. Appendix E - Set up SAML.....	253
Introduction	254
Configure trust between Orchestra and the identity provider.....	254
Assign users different access rights.....	261
Configure single logout	266
Troubleshooting	268
19. Index.....	273

1. What's New

Version	Chapter	Change
01.A	Auditing	Added note about when appointment entries in calendar_audit table are removed.
	System Administration	Edited LDAP Settings
	Installation/Upgrade	Updated Wildfly version from 11 to 26. Updated Installation Note - Windows Authentication
01.B	System Administration	Added DD.MM.YYYY as a supported date format. Added information about third-party hardware licensing.
01.C	SSO Setup, Appendix E - Set up SAML	Added instructions for configuring a single logout URL. Changed name from Azure AD to Entra ID
01.D	Xtend	Added a chapter with requirements for using Xtend.
	Installation	Added note about restarting Orchestra service on Windows. Added installation note for installing Orchestra with IPv6.

Version	Chapter	Change
01.E	System Administration	Added setting “Distributed publish delay (milliseconds)” to Central Websocket Server Settings. Added settings for controlling scheduled publishing of branches.
	Localization	Removed section about Connect
	Network Connectivity	Removed sections about Connect and Concierge native apps
01.F	Appendix A – Tuning Database Parameters	Updated the PostgreSQL tuning section to align with Orchestra 7.4, including corrected default values and updated configuration recommendations.

2. Introduction

Topics in this chapter

Welcome!	2
Customer Journey Management	2
System description	3
Distributed Operations	5
Deployment Scenarios	7
Supported Environments	9

Welcome!

Welcome to Qmatic Orchestra, an advanced system for Customer Journey Management.

Qmatic Orchestra is an enterprise-grade platform designed for large, centralized installations, while also supporting distributed deployments. It consists of a collection of components that can be combined to create a wide range of Customer Journey Management solutions.

The platform is highly flexible and can be adapted to meet different customer requirements. This Reference Manual describes a standard, neutral system configuration and does not cover adaptations for specific customers or market segments.

Furthermore, this manual assumes that the reader is familiar with other Qmatic hardware, such as displays and hardware terminals.

Customer Journey Management

To connect people to services and make sure that customers are served in a fair and efficient manner is Qmatic Orchestra's most important task.

Setting up your customer journey solution starts by defining your *services*. Services represent the choices customers can make when they enter the premises. Service are linked to ticket printer buttons which represent the services they see on these devices. A service choice therefore are made on, for example, touch screens or at receptions equipped with a printer. A service choice results in a visit number being generated and usually printed on a ticket.

The queue specifies the visit number sequence. A service always belongs to a queue. Two or more services that belong to the same queue also share the same number sequence.

Visit numbers are selected from queues depending on profiles when a staff member gives the Next command.

System description

Components

The *Home* page of Qmatic Orchestra shows which components of the system are installed and available. Here is an example of what the Home page may look like:



Below, the available components of Qmatic Orchestra are listed:

- Appointment Booking - this application is used by staff at for example a call centre who book and handle Appointments. Note: Appointment Booking is also available as an extension.
- Appointment Reception – application for arriving appointments.
- Business Configuration - branch configuration, Operation Profiles, Services, etc.
- Concierge - gives you the possibility to run a meet-and-greet application for creating visits and arriving appointments. Note: Concierge is also available as an extension.
- Connect - call and serve visits on a mobile device. Note: Connect is also available as an extension.

ⓘ This application is not available as a component on the *Home* page as it is a native app.

- Context Marketing - manage messages and playlists.
- Counter - call and serve visits. Note: Counter is also available as an extension.
- Calendar Admin - configure appointment profiles, resources and similar.
- Calendar Client - manage and get an overview of appointment bookings.
- Operations Panel - branch live information. Note: Operations Panel is also available as an extension.
- Notification Admin - SMS and email administration. Note: Notification Admin is also available as an extension.
- Personal data management – used for handling customer information and retention policies.
- Reception - create visits and print tickets.
- Surface Editor - surface application designer.
- System Administration - parameter settings, LDAP / SAML settings, import / export, license management, unit type templates, widget handling, etc.
- User Management - manage users, roles, and LDAP / SAML integration. For more information, see the Administrator's Guide.
- Xtend - Install and update extensions.

Branch Agnosticism

By default, Qmatic Orchestra is set up so that if a user is assigned a role that is branch agnostic, the user will see *all* branches, even if all the other assigned applications are branch aware.

So, for example, if the user has access to *both* the Counter Workstation (Branch aware) *and* the Central Operations Panel (Branch agnostic), the user will see all available branches in the Counter Workstation.

If you want to change this default behavior, you need to set the *branch_app* variable to *true/false* for the used application(s), in the applications table, accordingly.

To find out which roles are branch agnostic, please see the User Management chapter in the Administrator's Guide.

Distributed Operations

Distributed Operations is a unique Qmatic Orchestra feature, which allows you to configure and manage your Customer Journey platform centrally, while the actual workload is distributed among all your branches.

This also enables your branches to attend to the customers' needs, even if the connection to the central server is lost. Poor network conditions, high latency and low bandwidth will thus no longer be a problem.

When do you need distributed operations?

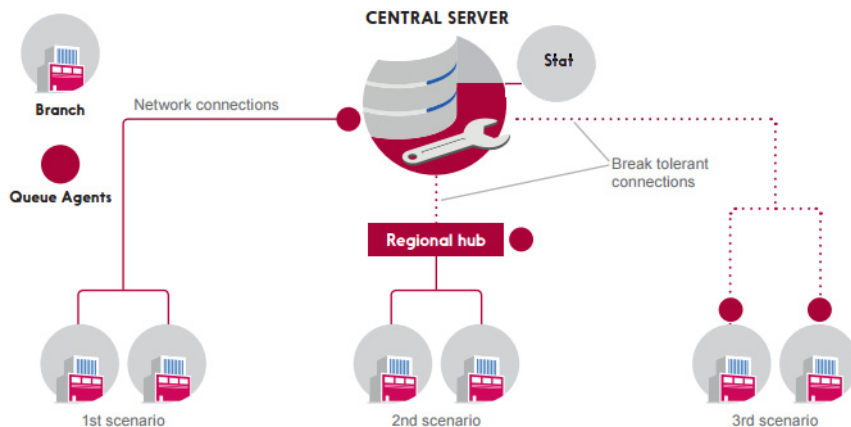
Distributed Operation is needed when at least one of the following applies:

- Available WAN bandwidth is poor or the WAN connectivity is not 100% reliable and suffers from high latency.
- If there are several business critical systems, Qmatic Orchestra being one of them, competing for WAN bandwidth, at peak times.

How does it work?

Distributed Operation is handled in the following way:

- Central stat server handling persistence of all statistical data.
- Branches are configured to operate on distributed queue agent instances.
- Distributed queue agent connects outbound to central Orchestra instance across WAN, communication is bi-directional, once connection is established.
- If the central server, or database, needs planned maintenance, or if there is an unexpected connectivity failure between distributed queue agent and central Orchestra, local users continue to operate as usual, during the outage.
- The distributed queue agent keeps a local copy of all runtime data and configuration.
- The distributed queue agent sends event data for statistical data handling to the central stat server, when a network connection is present.
- The distributed queue agent provides connector interfaces and web terminals for local clients.
- Appointments for the current day are synchronized directly, so long as there is a network connection. Appointments for the next day, however, are synchronized nightly.
- Qmatic Orchestra upgrades can be applied centrally, then applied to all branches, using remote update. For more information, see “Remote Upgrade Overview” on page 74.

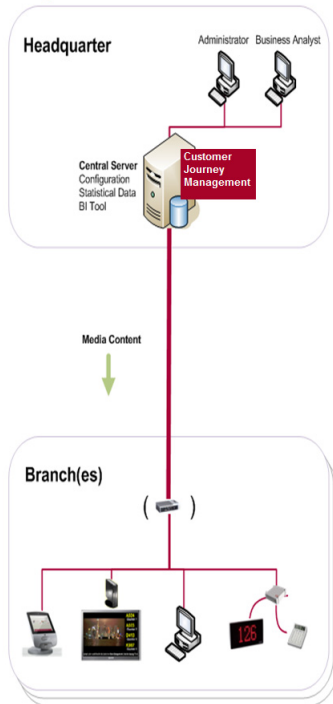


Deployment Scenarios

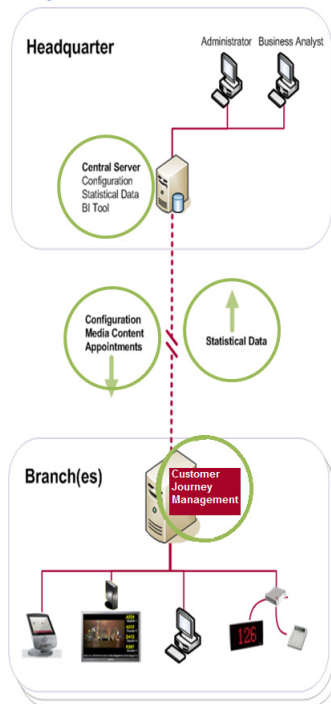
The following pictures show a few examples of deployment scenarios for Qmatic Orchestra.

⚠ There are more deployment options than specified, but these are the most common ones.

Fully Central

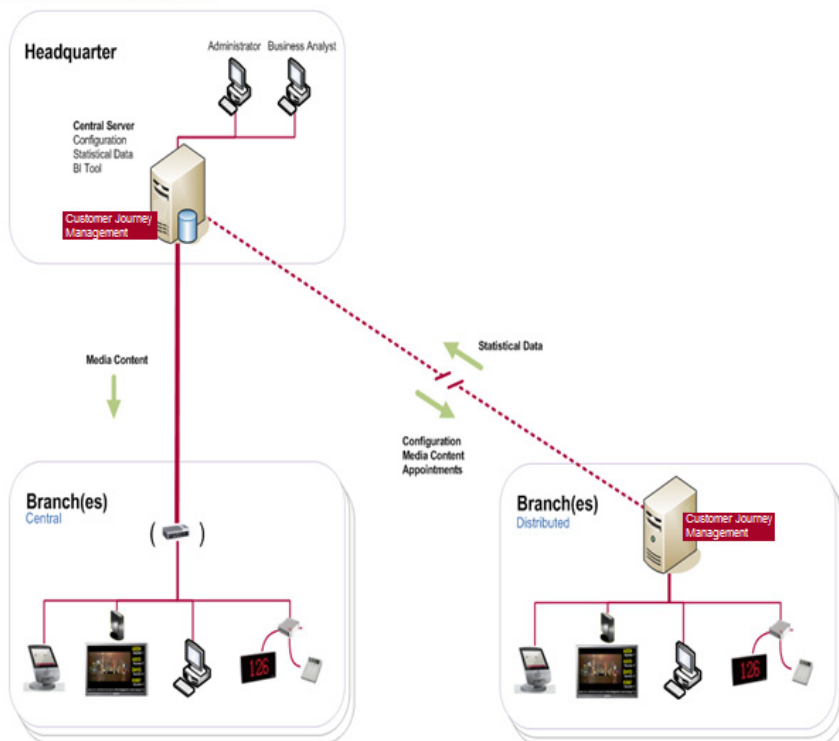


Fully distributed

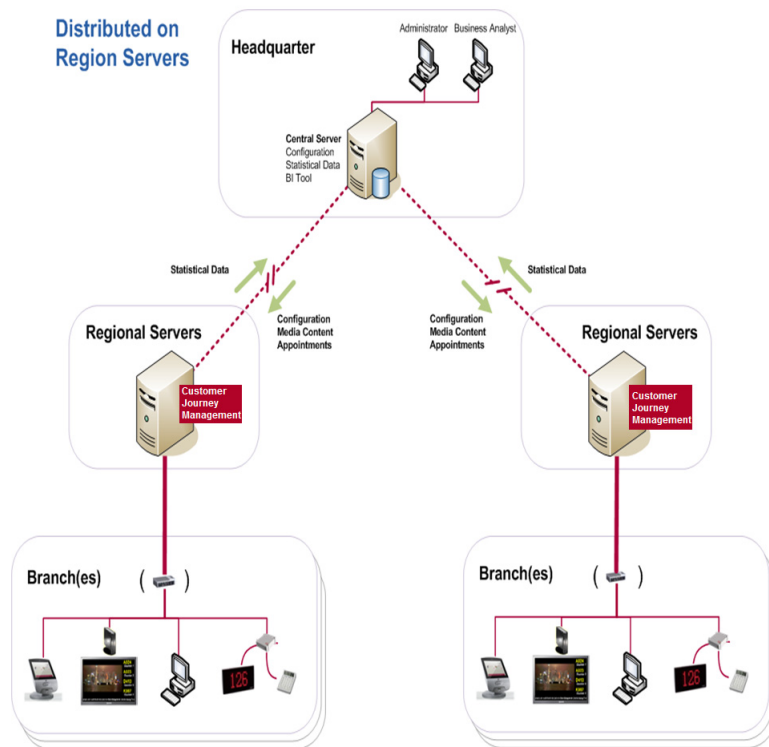


Fully Central vs Fully Distributed

Central & Distributed



Central and Distributed



Distributed on Region Servers

Supported Environments

For more information about supported Operating Systems, Application Servers, Databases and Web Browsers, please refer to the Qmatic Orchestra Data Sheet, which can be found on Qmatic World.

Cloud Bridge

Cloud Bridge is a full duplex communication module that enables communication between Orchestra and cloud services.

Events that are sent to the cloud are the following:

- CFM
- QUEUE
- BRANCH

Orchestra commands that cloud bridge is authorized to call are service point and entry point.

Communication

- Cloud bridge is using MQTT protocol for communication to the cloud. JMS queue is used for events from orchestra and HTTP rest call is used for communication to Orchestra.
- The port 443 needs to be open for secure communication to the cloud via MQTT

Installation and setup

- For installation of Cloud Bridge, check the Cloud Bridge component when running the installation wizard, or set `application.cloud.bridge = true` in `install.properties` if running the installation in silent mode.
- For settings and system parameters, please see “Cloud settings” on page 123.

Accessibility

Applications are considered accessible if they meet certain technical criteria and can be used by people with disabilities. That includes people who are blind, visually impaired and mobility impaired, using for example screen readers, zoom and keyboard only. Accessibility features for our most used staff applications in Orchestra have been implemented according to WCAG 2.1 (Web Content Accessibility Guidelines), including logical tabbing between objects, better contrast and descriptions of buttons and fields.

We strive to be WCAG 2.1 compliant with the following applications:

- Concierge
- Counter
- Mobile Ticket

For best accessibility performance, make sure you have the latest Orchestra update and the latest native app version installed.

ⓘ Note that there are many assistive technology tools and browsers available. Qmatic has primarily optimized accessibility functionality for NVDA on Chrome, Firefox and IE 11 browsers. When applicable, testing is also done for the built-in screen readers for iOS and Android.

3. Installation

Topics in this chapter

Introduction	14
Prerequisites - Checklist	19
Installation Scenarios	26
Orchestra and Stat on the Same Server	26
Orchestra and Stat on Separate Servers	36
Configuration before using Personal Data Management	47
Configuration before using Personal Data Management	47
Orchestra Queue Agent	48
Gateway 1745	53
Installing and Configuring Branch Hub and Hub	53
Redeploying Applications	53
Configuration Variables for Applications	53

Introduction

Preinstalled User

When Qmatic Orchestra is installed, it comes with a preinstalled user:

Username: *superadmin*

Password: *ulan*

This user should only be used when doing the installation and performing setup. It has access to the complete system, and is normally used only by a Qmatic representative.

! The first time you log in with superadmin, you will be prompted to change the password. The password must contain at least 8 characters, 1 digit and both lower case and upper case letters. Note that it may take around a minute for changes to take effect.

Requirements

See the Orchestra Platform Overview document, found on Qmatic World, for requirements regarding versions of: Hardware Operating Systems, Browsers, Databases, Java and the like.

Port Numbers

In a default set up of Orchestra, the ports listed below are used.

Central Orchestra Server

HTTP: 8080

HTTPS: 8443

Queue Agent communication: 8787

Device communication: 8888

Standalone Queue Agent

HTTP: 18080

HTTPS: 18443*

Device communication 18888

* HTTPS listener is not enabled by default.

For more information about HTTPS and port/protocol handling, please see “Secure Communication” on page 165.

Also, see “Ports and Protocols” on page 169.

Connect Agent

If you want to install Connect, the following ports need to be enabled in the firewall:

- 9797
- 4443
- 1434
- 9009
- 5445
- 5432
- 443 (for communication with Azure notification service bus)

HA Setup

- 40010
- 40020

Enabling Ports in the Firewall

This is done in the *Windows Control Panel*:

1. In the *Windows Control Panel*, open *Windows Firewall*.
2. Go to *Advanced settings* and click on *Outbound Rules*.
3. Click *New Rule*, select *Port*, then click *Next*.
4. In the *Specific local ports* field, enter the applicable port number(s), then click *Next*.
5. Select *Allow the connection* and click *Next*, then click *Next* again and give the rule a suitable name.

Changing Port Numbers

Port numbers can be changed in different places. In this section, we will describe where they can be changed.

❗ In the examples below we will change the default http port from *8080* to *15050*.

❗ The instructions below refer to Windows file names and paths. Note that other file names and paths apply if you are using Linux!

Summary

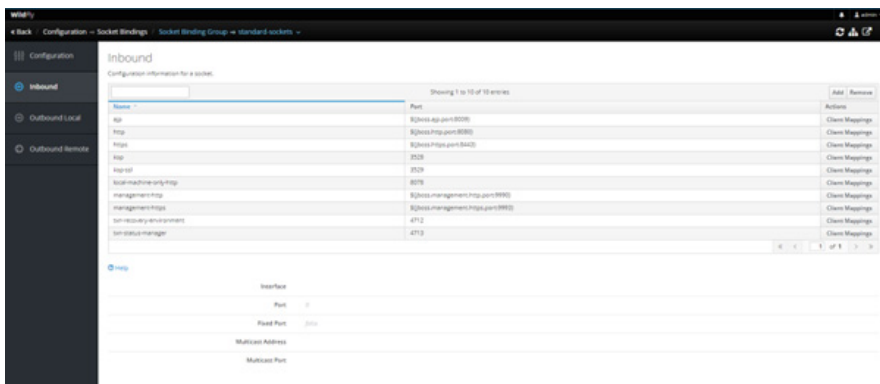
The following steps are included in the procedure and are described in more detail in the sections below:

1. Change the port in the Wildfly Admin console.
2. Change the port in the Configuration Files.
3. Change the port in the Startup Files.
4. Change the port in shiro.ini
5. Restart Orchestra.
6. Change the central http port, in the Orchestra System Parameters.
7. Restart Orchestra.
8. Publish.

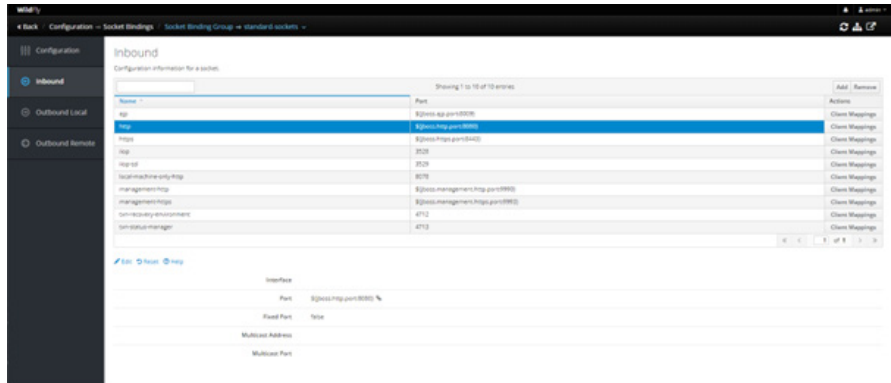
Wildfly Admin console

Follow these steps:

1. In the Wildfly Admin Console (<http://localhost:9990>, User name: admin, Password: ulan), go to Configuration -> Socket Bindings -> standard-sockets -> View -> Inbound.



2. Select http and click Edit.



3. Change the Port to 15050 and click Save.
4. Click Back, go to the Runtime tab, select the server and click Reload. Click Yes to confirm reload the server.

Configuration Files

⚠ These steps are not needed, if you already changed the port from the Wildfly Admin Console.

1. Open the file `<Orchestra install dir>\app\wildfly-x.y.z.Final\standalone\configuration\standalone-full.xml` in your favorite text editor.

2. Locate the line:

```
<socket-binding name="http" port="${jboss.http.port:8080}"/>
```

and change it to:

```
<socket-binding name="http" port="${jboss.http.port:15050}"/>
```

3. Locate the line:

```
<remote-destination host="${agent.ipAddress:localhost}"
port="${jboss.http.port:8080}"/>
```

and change it to:

```
<remote-destination host="${agent.ipAddress:localhost}"
port="${jboss.http.port:15050}"/>
```

4. Save the file.

Startup Files

Follow these steps:

1. Open the file `<Orchestra_install_dir>\app\<wildfly-x.y.z.Final>\bin\standalone.conf.bat`, in your favorite text editor.
2. Locate the line starting like this (but with your corresponding IP address):

```
set "JAVA_OPTS=%JAVA_OPTS% -Dagent.ipAddress=192.168.1.123 -  
DQP_HOME=%JBOSS_HOME%../../../../
```

and change it to (adding the text in bold):

```
set "JAVA_OPTS=%JAVA_OPTS% -Dcentral.http.port=15050 -  
Dagent.ipAddress=192.168.1.123 -DQP_HOME=%JBOSS_HOME%../../../../
```

3. Save the file.

Shiro.ini

In `shiro.ini`, there is a connection to the `oauth2` server which contains the Orchestra port. To change this, follow these steps:

1. Open the file `shiro.ini` and locate the following lines:

```
oauth2Realm.oauth2CheckTokenUrl = http://localhost:8080/oauth2server/  
oauth/check_token
```

```
oauth2Realm.oauth2PublicKeyUrl = http://localhost:8080/oauth2server/oauth/  
token_key
```

2. Change the port number accordingly and save the file.

Restart Orchestra

- Restart the Orchestra service.

Orchestra System Parameters

Follow these steps:

1. Log in to Orchestra.
2. Open the *System Administration* application and go to the *Settings > Parameters*. For more information, see "Parameters" on page 112.
3. In the *Central access Parameters -> HTTP Settings* section, change the parameter *Central HTTP Port* to *15050*.

The settings *Central HTTP Port* and *Central HTTP Protocol* are used for distributed Queue Agents, when connecting to Central, to sync contents, such as widgets.

4. In the *Application Parameters -> Statistics Settings* section, change the parameter *Stat Server Port* to *15050*.
5. Save the parameters.
6. Restart the Orchestra service.

Publish

- Publish all central branches.

Prerequisites - Checklist

The following table can be seen as a checklist for the prerequisites that need to be considered before going ahead with the installation.

We recommend printing it, so that each prerequisite can be carefully checked before going ahead. Some of the prerequisites may not be applicable for you. If that's the case, simply mark them as *N/A* and move on!

Prerequisite	Check	N/A
The hardware and software that is used are of versions supported by the software to be installed. For information about supported versions of Operating Systems, Browsers, Databases, Java and hardware requirements, please refer to the Orchestra Data Sheet, found on Qmatic World.		
All necessary hardware is installed with a supported version of the OS and database (if needed).		
All cabling is in place.		
The software to be installed is available.		
A network of suitable capacity must be in place. For more information, see "Distributed Operations" on page 9.		
Decide which installation scenario suits your organization best. For more information, see "Installation Scenarios" on page 26.		
If you are using CentOS, please read through "Installation Note - CentOS" on page 21 before starting.		
If you will be using Windows Authentication, please read through "Installation Note - Windows Authentication" on page 23, before starting.		
If you, for some reason, after having installed a complete system, need to redirect stat to a separate server, please follow the instructions in "Installation Note - Configuration of Stat on a separate server" on page 24.		

Prerequisite	Check	N/A
If Oracle or Microsoft SQL Server is used as database, database must be up and running. ⚠ For more information about Oracle parameters, that might need tuning, please read “Tuning of Oracle Parameters” on page 282 and for PostgreSQL parameters that might need tuning, please read “Tuning of PostgreSQL Parameters” on page 283.		
If you are to install a distributed setup and will be using Stat, you must open up port 5445 in your firewall.		
If you are using Microsoft SQL Server, the following settings must be applied to the installation: - the <i>TCP/IP protocol</i> (and <i>port number</i>) <i>SQL Server and Windows Authentication mode</i> must be enabled.		
If you will be using named instances, you need to run the installation in silent mode, using the command <i>install.bat -s</i> for Windows, or <i>install.sh -s</i> for Linux.		

Supported Platforms

For information about supported platforms, databases etc, please see the Qmatic Orchestra Datasheet.

Expired password Oracle users

If Oracle user passwords have expired, you will see this error message: “Access to DialectResolutionInfo cannot be null when 'hibernate.dialect' not set”

To find locked accounts in Oracle, run the following query as sys:

SQL

```
select username,account_status from dba_users where
account_status like '%EXPIRED%' or account_status like
'%LOCKED%';
```

Restarting service - Windows

If you have problems with Orchestra failing to start after restarting the service, please try and wait at least two minutes after stopping the service before you start it again. This is to make sure that all lingering port connections are gone.

Installation Note - CentOS

Before running the Orchestra installer on CentOS, there are a few steps you need to go through:

1. Settings for “max user processes” and setting for “max open files”.

We recommend increasing the setting of “max user processes” to 2048 and increasing “max open files” by 30000.

If running other processes on the Orchestra server, these settings might have to be increased further.

Current max user processes limit can be verified by running:

```
ulimit -u
```

Current max open files limit can be verified by running:

```
ulimit -n
```

We suggest increasing this, by adding the lines below to `/etc/security/limits.conf`.

In this example “qmatic” is the username of the user that will run the installer and the user that Orchestra will run as.

```
# Limits for orchestra installer
qmatic hard nofile 30000
qmatic soft nofile 30000
qmatic hard nproc 8192
qmatic soft nproc 8192
```

If you have logged in using graphical user interface, set below value in the files `/etc/systemd/user.conf` and `/etc/systemd/system.conf` under section [Manager]:

```
DefaultLimitNOFILE=30000
```

After modifying this file, the system will require a reboot. Verify the settings again after reboot.

2. Make sure host name is resolvable.

Orchestra installer requires that host name of the machine is resolvable. You can check host name in the file `/etc/sysconfig/network`.

You can verify that it is possible to resolve this by running:

```
ping HOSTNAME
```

If you get a response "*ping: unknown host HOSTNAME*" you will have to add the HOSTNAME to the file */etc/hosts*.

Add the following line to the end of the hosts file:

```
127.0.0.1      HOSTNAME
```

3. If you are installing on a CentOS server without a GUI, you most likely need to install additional fonts that is used by the surface editor, when creating ticket layouts.
4. When running the installation using the embedded PostgreSQL database, a default Linux system sometimes has too low memory settings. The PostgreSQL process requires larger settings for shared memory, compared to the usual default of 32 MB.

Kernel semaphores need to be set according to the following command

```
kernel.sem = SEMMSL SEMMNS SEMOPM SEMMNI
```

```
SEMMMSL = 512
```

```
SEMOPM = 256
```

```
SEMMNI = 256
```

The SEMMNS is calculated from SEMMSL and SMMNI according to the following calculation

```
SEMMNS = SEMMSL * SMMNI = 512 * 256 = 131072
```

5. To make the change permanent, add or change the following line in the file */etc/sysctl.conf*.

```
echo `kernel.sem=512 131072 256 256`>>/etc/sysctl.conf
```

Reboot the server for the changes to take effect.

Installation Note - TLS Security Settings

Mitigation of Denial-of-service attack

During installation of Orchestra, TLS Security settings are automatically enabled, by setting the following parameter to *true*:

```
-Djdk.tls.rejectClientInitiatedRenegotiation
```

If you, for some reason, want to disable this security setting, this is done by setting the above parameter to *false* in the file *<Orchestra install dir>\app\wildfly-x.y.z.Final\bin\standalone.conf*. Then restart the application server.

Mitigation of LOGJAM Man-in-the-Middle attack

During installation of Orchestra, the *Diffie-Hellman* key-size is automatically increased from the default 1024 to 2048.

If you want to enhance your security, you need to un-comment the parameters *ephemeralDHKeySize* and *DHEParameters* in the file `<Orchestra install dir>\app\wildfly-x.y.z.Final\bin\standalone.conf`. Then, restart the application server.

Installation Note - Windows Authentication

If using SQL Server, it is possible to install Orchestra with Windows Authentication.

❗ The user that will be used to authenticate towards the database is the user that runs the Orchestra installation, make sure to run the installation as the correct user.

❗ The database hostname specified must be a host known by the domain, normally the computer name, otherwise windows authentication / integrated security will not work. If you experience issues with the installer failing at an early stage with validating the database connection, please consult the windows event log (on either Orchestra or database server side) to check any error messages related to database access problems.

Follow the steps below:

1. Grant Windows User *Logon as service access*:
 - a) Log on to the computer with administrative privileges.
 - b) Open the desktop app *Local Security Policy*.
 - c) In the left menu, expand *Local Policies* and click on *User Rights Assignment*.
 - d) In the right pane, right-click *Log on as a service* and select *Properties*.
 - e) Click *Add User or Group* button to add a new User.
 - f) In the *Select Users, Computers, Service Accounts, or Groups* window, find the wanted User, then click *OK*.
 - g) Click *OK* in the *Log on as Service Properties* window to save the changes.
2. Create databases:
 - a) Open a connection to the database with the SA account.
 - b) Open and execute the scripts *central-mssql-integrated.sql* and *stat-mssql-integrated.sql*, located in the *db* folder of the installation zip/tgz package.
 - c) Open the script *access-mssql-integrated.sql*, located in the *db* folder of the installation zip/tgz package, and replace the *'DOMAIN\USERNAME'* with the actual domain name and username.
 - d) Save and run the *access-mssql-integrated.sql* script.
3. Install with wizard - for more information, see "Orchestra Installation Wizard - installing both Orchestra and Stat" on page 28 / "Orchestra Installation Wizard - Orchestra and Stat on Separate Servers" on page 39.
4. Install with silent installer:

When running in silent installer mode, there is no need to supply database user or passwords in the `install.properties` file. The password will be prompted for during installation.

- a) Open the `install.properties` file, in the installation package (zip/tgz file).
- b) Locate the property `mssql.integrated.security` and set it to `true`.
`mssql.integrated.security = true`
- c) Make any other wanted updates to the `install.properties` file (for example selecting wanted applications) and save the file.
- d) Run the installer in silent mode, using the command `install.bat -s` for Windows, or `install.sh -s` for Linux.

Installation Note - SQL Server with named instances

If using MS SQL Server with named instances:

1. Open `install.properties` file.
2. For each application db properties.
 - a) Set db host name for the different applications on the format `host-name\instance-name`. Example:
`agent.sqlserver.db.host = sqlserver.acme.com\\DBIN-STANCE`
 - b) Leave port property empty for each application db on SQL Server. Example:
`agent.mssql.db.port =`
3. Make any other wanted updates to the `install.properties` file (for example selecting wanted applications) and save the file.
4. Run the installer in silent mode, using the command `install.bat -s` for Windows, or `install.sh -s` for Linux.

Installation Note - Configuration of Stat on a separate server

Configuration after initial installation standalone stat, no encryption

1. On the central server, in `<ORCHESTRA>/app/wildfly-x.y.z.Final/standalone/configuration/standalone-full.xml`, locate the section

```
<outbound-socket-binding name="stat-remote">
  <remote-destination host="127.0.0.1" port="5445"/>
</outbound-socket-binding>
```

2. Change the IP-address to the stat server.
3. Restart orchestra central

Configuration after initial installation, standalone stat, encryption

Before you begin, please read the section “Encryption of JMS connections” on page 200 and carry out the instructions there.

When that is done, follow these steps:

1. In `<ORCHESTRA>/app/wildfly-x.y.z.Final/standalone/configuration/standalone-full.xml`, locate the section

```
<bridge name="centralAgentEventCoreBridge" queue-  
name="jms.queue.centralAgentEventQueue" forwarding-  
address="jms.queue.statAgentEventQueue" retry-  
interval="60000" reconnect-attempts="-1" use-duplicate-  
detection="true" static-connectors="stat-remote-connector"/  
>
```

2. Change the `static-connectors` parameter to “stat-remote-ssl”.

3. Locate the section

```
<outbound-socket-binding name="stat-remote-ssl">  
  <remote-destination host="127.0.0.1" port="5446"/>  
</outbound-socket-binding>
```

4. Change `remote-destination host` to the IP-address to the stat server.
5. Restart both servers

Installation note - IPv6

Orchestra supports IPv6 alongside IPv4. When installing Orchestra with IPv6, please consider the following bullets:

- When installing Orchestra, use **install_ipv6.bat/sh**
- If using Linux, use the following files to add startup scripts to the operating system
 - For Linux systems using SystemD: **install-services-systemd-ipv6.sh**
 - For Linux systems using SystemV: **install-services-ipv6.sh**

In the system, many default strings contain localhost/127.0.0.1. With ipv6, the short version for localhost is `[::1]`.

Installation Scenarios

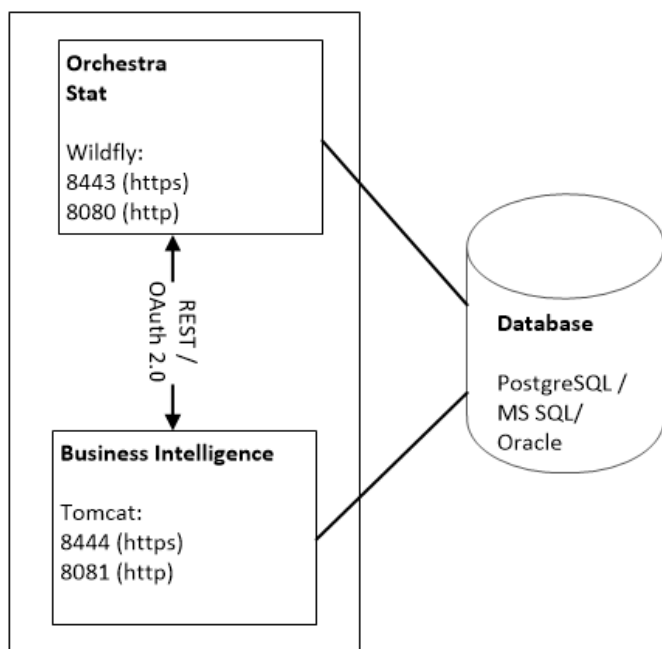
Before you go through the process of installing Orchestra, you should decide which of the following installation scenarios that best corresponds to your wanted system setup and then select it and follow the instructions in the corresponding section(s) of this chapter:

- Orchestra and Stat on the same server. For this installation scenario, please follow the installation steps described in “Orchestra and Stat on the Same Server” on page 26.
- Orchestra and Stat on different servers. For this installation scenario, see “Orchestra and Stat on Separate Servers” on page 36.

Orchestra and Stat on the Same Server

Installation Scenario

The picture below illustrates the installation scenario where Orchestra and Stat are installed on the same server.



❗ For this scenario, you will need to run the Orchestra installation wizard.

In this scenario, the Orchestra installation wizard, see “Orchestra Installation Wizard - installing both Orchestra and Stat” on page 28, installs the following:

- JDK
- Wildfly 26
- Orchestra
- Stat
- PostgreSQL (if used/needed)

Also, it runs the database scripts needed for Orchestra and Stat.

Installation - Windows

Preparation:

1. Copy the *QP_Central__win_<type>-<version_number>.zip* file and unpack the files to a suitable directory, such as *tmp*.

❗ It is recommended that path names containing spaces should *not* be used.

Configuration - Database

Prepare your database by running the scripts corresponding to your database (Oracle Database / Microsoft SQL / PostgreSQL).

❗ If you are going to use PostgreSQL and will be installing it at the same time as installing Orchestra, this configuration can be skipped.

1. Open a database tool that allows you to run a database script (e.g. SQL Developer for Oracle Database, or SQL Server Management Studio for Microsoft SQL). For PostgreSQL see special instructions below.
2. Run the script corresponding to your database in a tool that can execute the script:

Microsoft SQL Server:

```
<tmp_dir>/db/central-mssql.sql  
<tmp_dir>/db/stat-mssql.sql
```

Oracle Database:

```
<tmp_dir>/db/central-oracle.sql  
<tmp_dir>/db/stat-oracle.sql
```

PostgreSQL:

1. Make sure the PostgreSQL bin folder is added to environment variables.
2. Open the command prompt in the *<tmp_dir>/db/* directory.

3. Run the commands:

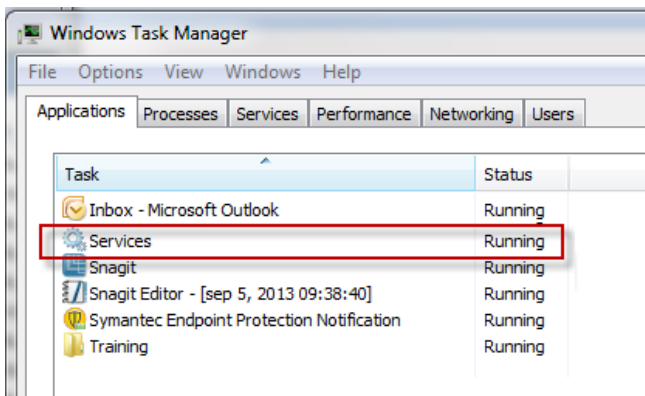
```
psql -U postgres -f central-postgres.sql
```

and:

```
psql -U postgres -f stat-postgres.sql
```

Installation

1. Open the Windows Task Manager and ensure that there is no application called “Services” running, see the picture below. If there is, close the *Services* application. This is done, to not cause any problems with starting and stopping the Orchestra service, during the installation procedure.



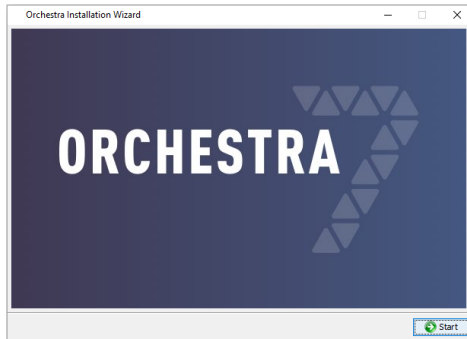
2. Run the Installation Wizard, *install.bat*. See “Orchestra Installation Wizard - installing both Orchestra and Stat” on page 28.

```
/<tmp_dir>/install.bat
```

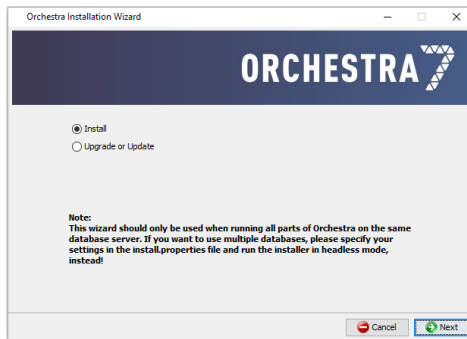
⚠ Using the Installation Wizard to install Orchestra means that *all* modules will automatically be installed. If you know for a fact that you will not be needing certain modules, you should rather use the *install.properties* file to select the wanted modules and then run the installation in silent mode, using the command *install.bat -s* for Windows, or *install.sh -s* for Linux.

Orchestra Installation Wizard - installing both Orchestra and Stat

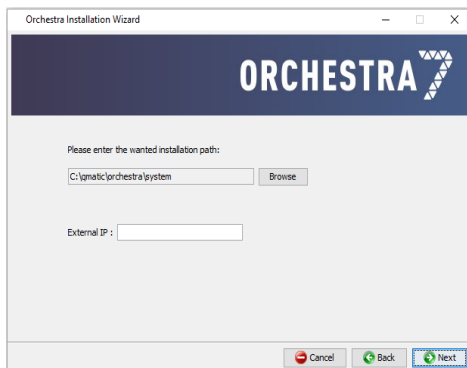
The wizard contains the following steps:



1. Click *Start* to start the installation.



2. Select *Install*.

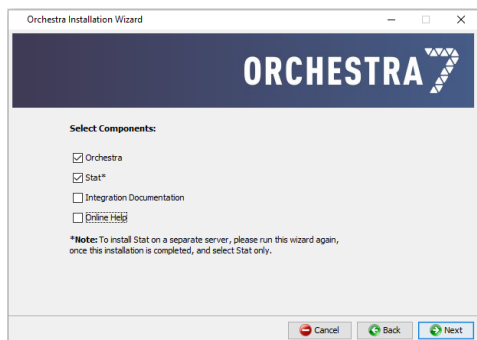


3. Enter the wanted installation path, IP and Hostname.

⚠ It is important that the *External IP* matches what the user will enter in his/her browser.

It is also possible to enter a valid domain name, for example *qmatic.com*, or a host name, such as *GOT-JOHSTA-001*. This can also be updated post installation, in the file *agent.conf*.

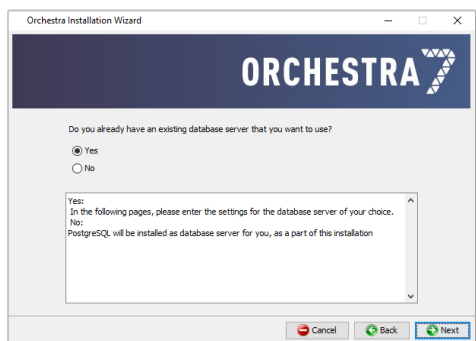
⚠ Host names used by Orchestra should follow the RFC952 specifications. For more information, see <https://tools.ietf.org/html/rfc952>.



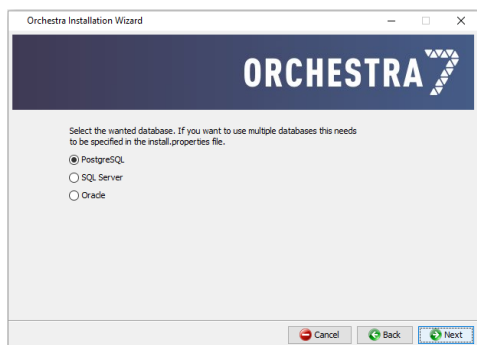
4. Select the wanted components.

If both Orchestra and Stat are selected, they will be installed on the same server.

If however, you want to install Stat on a separate server, see "Orchestra and Stat on Separate Servers" on page 36.



5. Select whether or not you already have an existing database server that you want to use. If you select Yes, you will need to enter the settings for your database server in the following pages. If you select No, PostgreSQL will be installed.



6. If you answered Yes in the previous step, select the wanted database here.

Orchestra Installation Wizard

ORCHESTRA

Database settings

Hostname: localhost

Port: 1433

Windows Authentication: ☐

Test connection

Cancel Back Next

7. Enter your database settings (*Hostname* and *Port* for PostgreSQL and SQL Server, *Hostname*, *Port*, and *SID* (for Oracle). It is also possible (not mandatory) to test the connection here.

! For SQL Server, you can also select *Windows Authentication*. For more information, see “Installation Note - Windows Authentication” on page 23.

Orchestra Installation Wizard

ORCHESTRA

Please provide the Password for this account.

User name : QMATIC\johnyt

Password :

Confirm password :

Cancel Back Next

8. If you selected *SQL Server* and *Windows Authentication* in the previous steps, enter and confirm your password on this page.

Orchestra Installation Wizard

ORCHESTRA

Installation Summary:

C:\qmatic\orchestra\system

With following components :

- Orchestra
- Stat

With following configurations :

- Database : postgres
- Application Server : wildfly

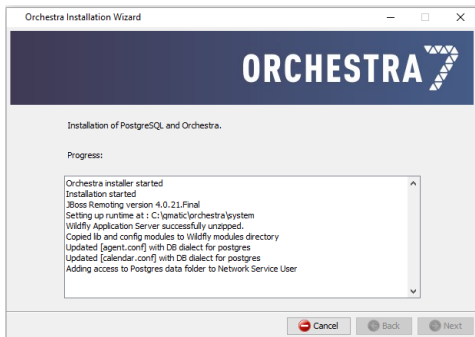
Save Configuration File

Note: If you want to change any of these settings either go back in the installation wizard and change your settings, or press cancel and edit your settings in the configuration file, then restart the installer.

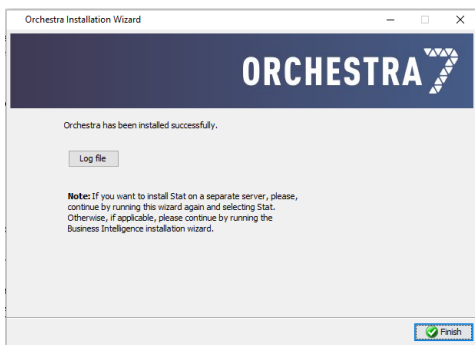
Cancel Back Next

9. This page shows an *Installation Summary*. You can either go back through the wizard and update your settings, or click *Next* to proceed.

Here, it is also possible to save the configuration file.



10. This page shows the progress of the installation.



11. When the installation has been successfully completed, click *Finish*. It is also possible to download the log file.

12. When the installation has finished, Orchestra is automatically started.

13. Open the URL and start configuring the system. Please refer to the Administrator's Guide, that can be found on Qmatic World.

http://<ORCHESTRA_IP>:8080

User: *superadmin*

Password: *ulan*

Once you have successfully managed to log in to Orchestra, you will know that the installation is complete!

⚠ If you want to import a system configuration, we recommend that you do this *before* any other configuration is done.

For more information, see "Import and Export" on page 90.

Installation - Linux

Preparation:

1. Transfer the `QP_Central_linux64-<version>.tgz` file to the server, preferably to a `tmp` directory. Use for example `scp` to transfer the file.

❗ It is recommended that path names containing spaces should *not* be used.

2. Login to the Orchestra Central server with a user with superuser rights.

3. Create a `qmatic` user:

```
#sudo adduser qmatic
```

4. Create Orchestra directory:

```
#sudo mkdir -p /opt/qmatic/orchestra/system
```

5. Grant rights for Orchestra directory to `qmatic` user:

```
#sudo chown -R qmatic:qmatic /opt/qmatic
```

6. Switch to `qmatic` user:

```
#sudo su - qmatic
```

7. Unpack Orchestra `tgz` file, to a `tmp` directory:

```
$ cd /<tmp_dir>
$ tar zxvf QP_Central_linux64-<version>.tgz
```

Configuration - Database

Prepare your database by running the scripts corresponding to your database (Oracle Database/Microsoft SQL/PostgreSQL).

❗ If you are going to use PostgreSQL and will be installing it at the same time as installing Orchestra, this configuration can be skipped.

❗ The scripts are designed for a rudimentary base installation of Orchestra, and should be reviewed and tuned by the database administrator to match the installation size and performance requirements.

1. Open a database tool that allows you to run a database script (e.g. SQL Developer for Oracle Database, or SQL Server Management Studio for Microsoft SQL). For PostgreSQL see special instructions below.
2. Run the script corresponding to your database in a tool that can execute the script:

Microsoft SQL Server:

```
<tmp_dir>/db/central-mssql.sql  
<tmp_dir>/db/stat-mssql.sql
```

Oracle Database:

```
<tmp_dir>/db/central-oracle.sql  
<tmp_dir>/db/stat-oracle.sql
```

ⓘ The Oracle scripts will create required schemas in default user table-space. Please review the scripts and assign appropriate table-spaces for the schemas. Larger installations may require tuning of some Oracle instance parameters.

For more information, see “Tuning of Oracle Parameters” on page 224.

PostgreSQL:

1. Make sure the PostgreSQL bin folder is added to environment variables.
2. Open the command prompt in the `<tmp_dir>/db/` directory.
3. Run the commands:

```
psql -U postgres -f central-postgres.sql
```

and:

```
psql -U postgres -f stat-postgres.sql
```

ⓘ Larger installations may require tuning of some PostgreSQL parameters. For more information, see “Tuning of PostgreSQL Parameters” on page 225.

Installation

1. Switch to qmatic user:

```
#sudo su - qmatic
```

2. Make sure that the Orchestra installation directory is empty and that there is no space in the path name.

3. Run the Installation Wizard, *install.sh*. For more information see “Orchestra Installation Wizard - installing both Orchestra and Stat” on page 28:

```
$ /<install_dir>/install.sh
```

To run the installation in silent mode:

```
$ ./install.sh -s
```

4. When the installation has finished, switch to user with super user rights:

```
$ exit
```

5. Change directory:

```
#cd /opt/qmatic/orchestra/system/bin
```

6. Add startup scripts to operating system

For Linux systems using SystemD:

```
#sudo ./install-services-systemd.sh
```

For Linux systems using SystemV:

```
#sudo ./install-services.sh
```

7. Open the URL and start configuring the system. Please refer to the Administrator’s Guide, that can be found on Qmatic World.

http://<ORCHESTRA_IP>:8080

User: *superadmin*

Password: *ulan*

Once you have successfully managed to log in to Orchestra, you will know that the installation is complete!

❗ If you want to import a system configuration, we recommend that you do this *before* any other configuration is done.

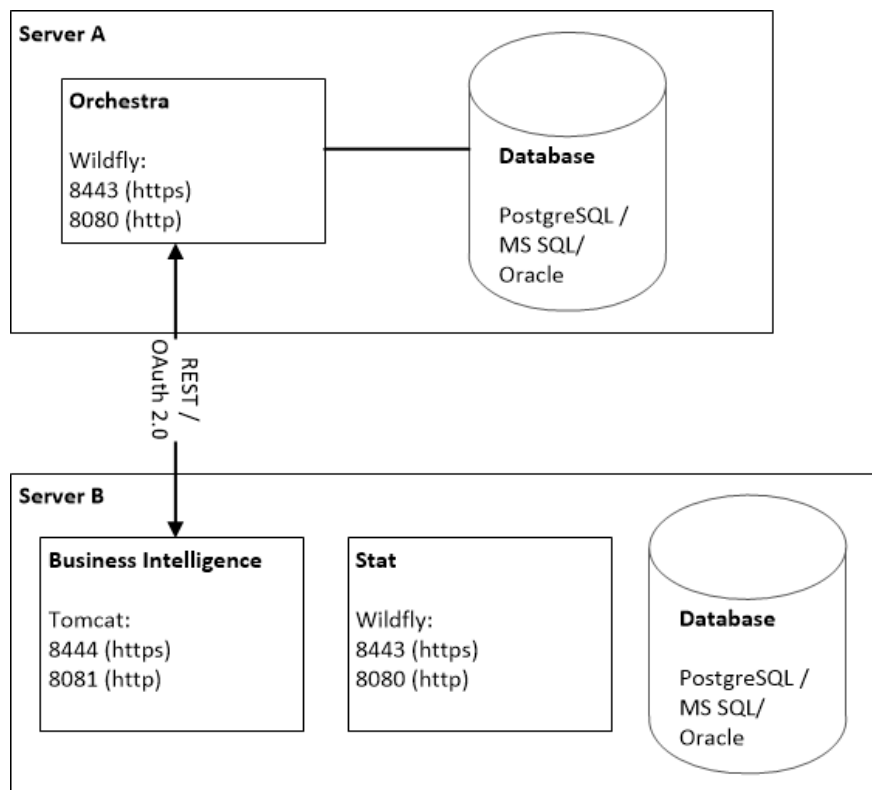
For more information, see “Import and Export” on page 90.

Orchestra and Stat on Separate Servers

❗ If you, for some reason, after having installed a complete system, need to re-direct Stat to a separate server, please follow the instructions in “Installation Note - Configuration of Stat on a separate server” on page 24.

Installation Scenario

The picture below illustrates the installation scenario where Orchestra and Stat are all installed on separate servers:



❗ For this scenario, you will need to run the Orchestra installation wizard *twice*.

The Orchestra installation wizard installs artefacts on two different servers.

First, you run the Orchestra wizard, see “Orchestra and Stat on the Same Server” on page 26, which installs the following, on Server A:

- JDK

- Wildfly 26
- Orchestra
- PostgreSQL (if used/needed)

Also, it runs the database scripts needed for Orchestra.

Then, you run the Orchestra wizard *again* so that it also installs the following, on Server B:

- JDK
- Wildfly 26
- Stat
- PostgreSQL (if used/needed)

Also, it runs the database scripts needed for Stat.

Installation - Windows

Prerequisite:

❗ For distributed systems, port 5445 needs to be opened in the firewall. For an instruction on how to open ports in the firewall, see “Enabling Ports in the Firewall” on page 15.

Data Integration

For more information about Data Integration, please refer to “Hardware Monitoring - Data Integration” on page 176.

Preparation:

1. Copy the *QP_Central__win_<type>-<version_number>.zip* file and unpack the files to a suitable directory, such as *tmp*.

❗ It is recommended that path names containing spaces should *not* be used.

2. Open the file *install.properties*, in the installation package (zip file).
3. Locate the following section:

```
standalone.stat.server.ip = 127.0.0.1
standalone.stat.server.port = 5445
standalone.stat.server.port.secure = 5446
```

4. Edit the file so that you enter the correct ip address (*standalone.stat.server.ip*) and port (*standalone.stat.server.port*) of the server where you will be installing Stat. Update *standalone.stat.server.port.secure* if you are using stat with ssl.

❗ When installing Orchestra, **without** selecting Stat, the installation program will read these properties from the *install.properties* file and automatically add them to the *standalone-full.xml* file.

Configuration - Database

Prepare your database by running the scripts corresponding to your database (Oracle Database / Microsoft SQL / PostgreSQL).

❗ If you are going to use PostgreSQL and will be installing it at the same time as installing Orchestra, this configuration can be skipped.

1. Open a database tool that allows you to run a database script (e.g. SQL Developer for Oracle Database, or SQL Server Management Studio for Microsoft SQL). For PostgreSQL see special instructions below.
2. Run the script corresponding to your database in a tool that can execute the script:

Microsoft SQL Server:

```
<tmp_dir>/db/central-mssql.sql  
<tmp_dir>/db/stat-mssql.sql
```

Oracle Database:

```
<tmp_dir>/db/central-oracle.sql  
<tmp_dir>/db/stat-oracle.sql
```

PostgreSQL:

1. Make sure the PostgreSQL bin folder is added to environment variables.
2. Open the command prompt in the *<tmp_dir>/db/* directory.
3. Run the commands:

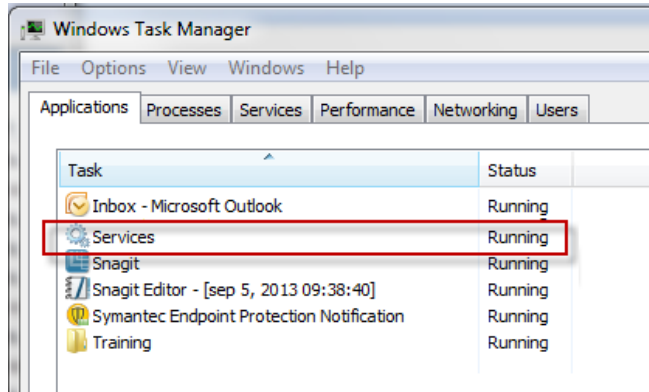
```
psql -U postgres -f central-postgres.sql
```

and:

```
psql -U postgres -f stat-postgres.sql
```

Installation

1. Open the Windows Task Manager and ensure that there is no application called "Services" running, see the picture below. If there is, close the *Services* application. This is done, to not cause any problems with starting and stopping the Orchestra service, during the installation procedure.



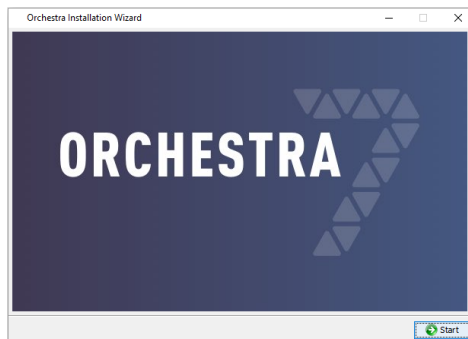
2. Run the Installation Wizard, *install.bat* **twice**. See “Orchestra Installation Wizard - Orchestra and Stat on Separate Servers” on page 39. First, run the wizard and select only *Orchestra*, then run it again and select only *Stat*.

/<tmp_dir>/install.bat

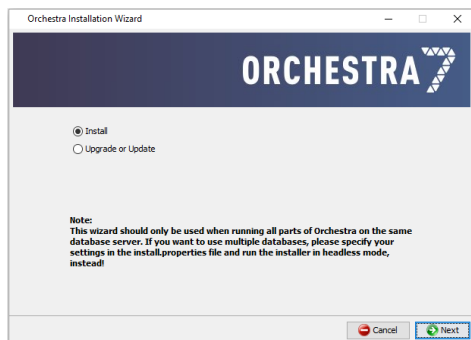
⚠ Using the Installation Wizard to install Orchestra means that *all* modules will automatically be installed. If you know for a fact that you will not be needing certain modules, you should rather use the *install.properties* file to select the wanted modules and then run the installation in silent mode, using the command *install.bat -s* for Windows, or *install.sh -s* for Linux.

Orchestra Installation Wizard - Orchestra and Stat on Separate Servers

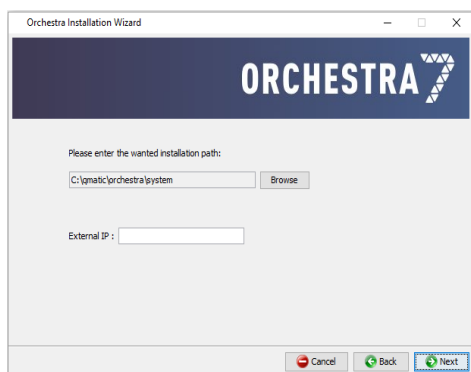
The wizard contains the following steps:



1. Click *Start* to start the installation.



2. Select *Install*.



3. Enter the wanted installation path and *External IP*. Default is *localhost*.

It is also possible to enter a valid domain name, for example *qmatic.com*, or a host name, such as *GOT-JOHSTA-001*. This can also be updated post installation, in the file *agent.conf*.

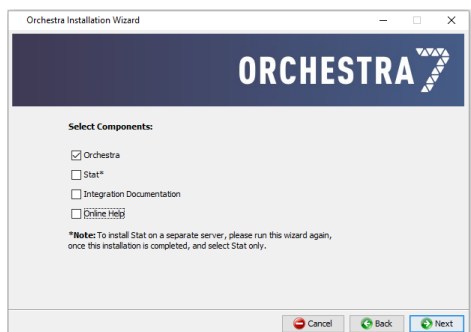


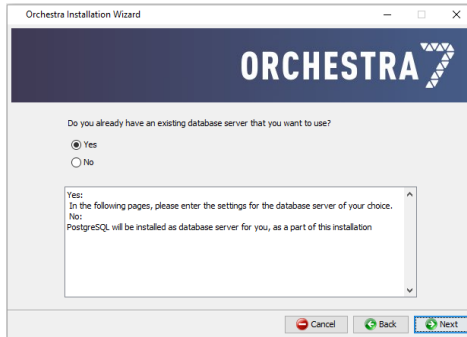
Host names used by Orchestra should follow the RFC952 specifications. For more information, see <https://tools.ietf.org/html/rfc952>.

4. Select the wanted components.

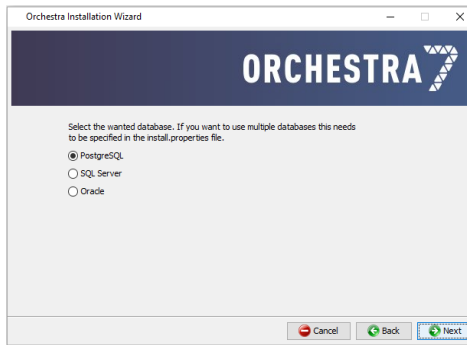


The *first* time you run the installation wizard, select only *Orchestra* (as in the picture) and the *second* time select only *Stat*!

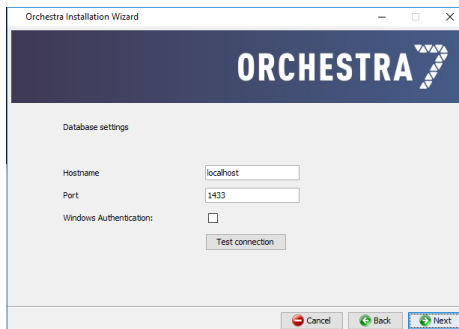




5. Select whether or not you already have an existing database server that you want to use. If you select **Yes**, you will need to enter the settings for your database server in the following pages. If you select **No**, PostgreSQL will be installed.

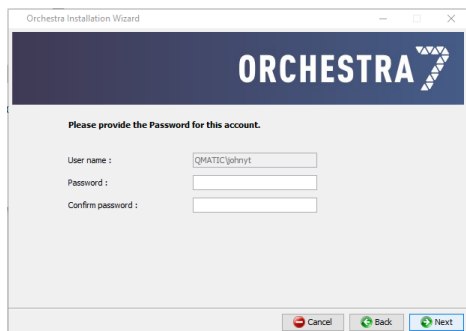


6. If you answered **Yes** in the previous step, select the wanted database here.

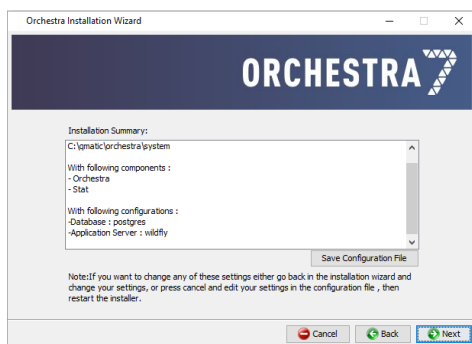


7. Enter your database settings (*Hostname* and *Port* for PostgreSQL and SQL Server, *Hostname*, *Port*, and *SID* for Oracle). It is also possible (not mandatory) to test the connection here.

ⓘ For SQL Server, you can also select *Windows Authentication*. For more information, see “Installation Note - Windows Authentication” on page 23.

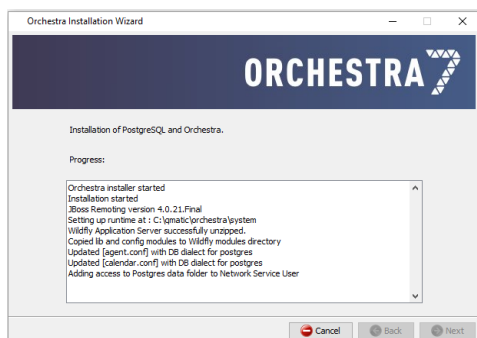


8. If you selected *SQL Server* and *Windows Authentication* in the previous steps, enter and confirm your password on this page.

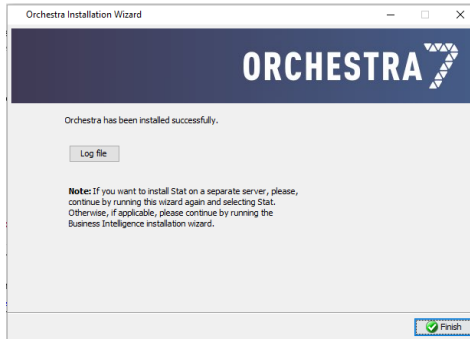


9. This page shows an *Installation Summary*. You can either go back through the wizard and update your settings, or click *Next* to proceed.

Here, it is also possible to save the configuration file.



10. This page shows the progress of the installation.



11. When the installation has been successfully completed, click *Finish*. It is also possible to download the log file.

12. Run the wizard again, this time selecting only *Stat* as component in step 4.

13. When the installation has finished, Orchestra is automatically started.

14. Open the URL and start configuring the system. Please refer to the Administrator's Guide, that can be found on Qmatic World.

http://<ORCHESTRA_IP>:8080

User: *superadmin*

Password: *ulan*

Once you have successfully managed to log in to Orchestra, you will know that the installation is complete!

ⓘ If you want to import a system configuration, we recommend that you do this *before* any other configuration is done.

For more information, see "Import and Export" on page 90.

Installation - Linux

Prerequisite:

ⓘ For distributed systems, port 5445 needs to be opened in the firewall. For an instruction on how to open ports in the firewall, see "Enabling Ports in the Firewall" on page 15.

Data Integration

For more information about Data Integration, please refer to "Hardware Monitoring - Data Integration" on page 176.

Preparation:

1. Copy the *QP_Central__win_<type>-<version_number>.tgz* file and unpack the files to a suitable directory, such as *tmp*.

❗ It is recommended that path names containing spaces should *not* be used.

2. Open the file *install.properties*, in the installation package (tgz file).

3. Locate the following section:

```
standalone.stat.server.ip = 127.0.0.1
standalone.stat.server.port = 5445
standalone.stat.server.port.secure = 5446
```

4. Edit the file so that you enter the correct ip address (*standalone.stat.server.ip*) and port (*standalone.stat.server.port*) of the server where you will be installing Stat. Update *standalone.stat.server.port.secure* if you are using stat with ssl.

❗ When installing Orchestra, **without** selecting Stat, the installation program will read these properties from the *install.properties* file and automatically add them to the *standalone-full.xml* file.

5. Login to the Orchestra Central server with a user with superuser rights.

6. Create a qmatic user:

```
#sudo adduser qmatic
```

7. Create Orchestra directory:

```
#sudo mkdir -p /opt/qmatic/orchestra/system
```

8. Grant rights for Orchestra directory to qmatic user:

```
#sudo chown -R qmatic:qmatic /opt/qmatic
```

9. Switch to qmatic user:

```
#sudo su - qmatic
```

10. Unpack Orchestra *tgz* file, to a *tmp* directory:

```
$ cd /<tmp_dir>
$ tar zxvf QP_Central_linux64-<version>.tgz
```

Configuration - Database

Prepare your database by running the scripts corresponding to your database (Oracle Database/Microsoft SQL/PostgreSQL).

❗ If you are going to use PostgreSQL and will be installing it at the same time as installing Orchestra, this configuration can be skipped.

❗ The scripts are designed for a rudimentary base installation of Orchestra, and should be reviewed and tuned by the database administrator to match the installation size and performance requirements.

1. Open a database tool that allows you to run a database script (e.g. SQL Developer for Oracle Database, or SQL Server Management Studio for Microsoft SQL). For PostgreSQL see special instructions below.
2. Run the script corresponding to your database in a tool that can execute the script:

Microsoft SQL Server:

```
<tmp_dir>/db/central-mssql.sql  
<tmp_dir>/db/stat-mssql.sql
```

Oracle Database:

```
<tmp_dir>/db/central-oracle.sql  
<tmp_dir>/db/stat-oracle.sql
```

❗ The Oracle scripts will create required schemas in default user table-space. Please review the scripts and assign appropriate table-spaces for the schemas. Larger installations may require tuning of some Oracle instance parameters.

For more information, see “Tuning of Oracle Parameters” on page 224.

PostgreSQL:

1. Make sure the PostgreSQL bin folder is added to environment variables.
2. Open the command prompt in the `<tmp_dir>/db/` directory.
3. Run the commands:

```
psql -U postgres -f central-postgres.sql
```

and:

```
psql -U postgres -f stat-postgres.sql
```

❗ Larger installations may require tuning of some PostgreSQL parameters. For more information, see “Tuning of PostgreSQL Parameters” on page 225.

Installation

1. Switch to qmatic user:

```
#sudo su - qmatic
```

2. Make sure that the Orchestra installation directory is empty and that there is no space in the path name.

3. Run the Installation Wizard, *install.sh* **twice**. See “Orchestra Installation Wizard - Orchestra and Stat on Separate Servers” on page 39. First, run the wizard and select only *Orchestra*, then run it again and select only *Stat*. For more information see “Orchestra Installation Wizard - installing both Orchestra and Stat” on page 28:

```
$ /<install_dir>/install.sh
```

To run the installation in silent mode:

```
$ ./install.sh -s
```

4. When the installation has finished, switch to user with super user rights:

```
$ exit
```

5. Change directory:

```
#cd /opt/qmatic/orchestra/system/bin
```

6. Add startup scripts to operating system:

```
#sudo ./install-services.sh
```

7. Open the URL and start configuring the system. Please refer to the Administrator’s Guide, that can be found on Qmatic World.

http://<ORCHESTRA_IP>:8080

User: *superadmin*

Password: *ulan*

Once you have successfully managed to log in to Orchestra, you will know that the installation is complete!

❗ If you want to import a system configuration, we recommend that you do this *before* any other configuration is done.

For more information, see “Import and Export” on page 90.

Configuration before using Personal Data Management

If you are using Orchestra and running stat deployed on a separate server, or if you are using anything else than localhost as your Stat Server Address (see below), there are a few manual steps that you need to follow before using the Personal Data Management application:

1. In Orchestra, go to **System Administration > Settings > Parameters > Statistics Settings**.
2. In the **Stat Server Address** and **Stat Server Port** fields, make sure that you have filled in the correct IP address or Fully Qualified Domain Name (FQDN) and port number for your stat server.
3. Locate the **Customer** parameters. Make sure that the parameters User retention policy for customer object and Delete customers, based on retention policy, at (hh:mm) are set to your liking.
4. Open the `<orchestra_install_dir>\conf\stat.conf` file, **on the Stat server**, and make sure that you have filled in the IP address or FQDN of the Central Orchestra server for the property `stat.personaldata.ip.filter`.

❗ If High Availability (HA) is used, two (2) IP addresses or FQDN for the different Orchestra servers should be added.

Example 1: `stat.personaldata.ip.filter = 127.0.0.1`

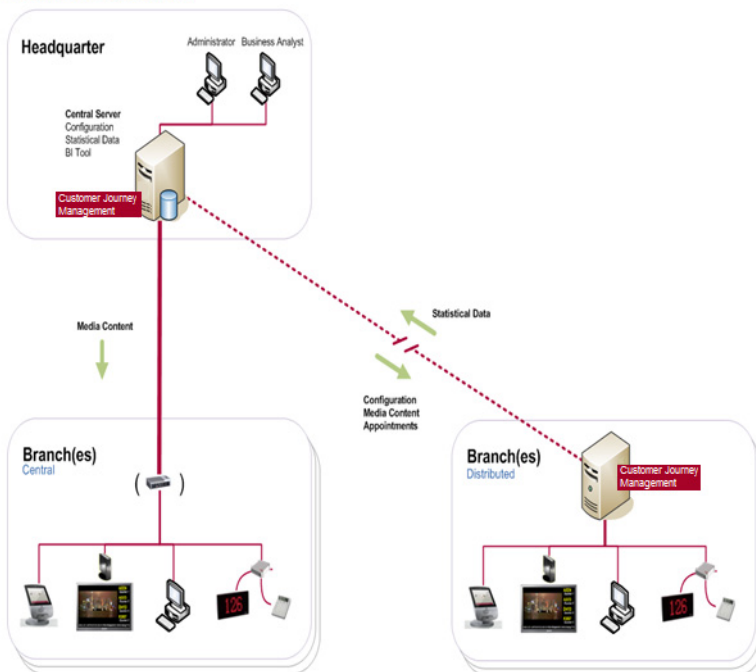
Example 2: `stat.personaldata.ip.filter = mydomainname.com`

Example 3 (using HA): `stat.personaldata.ip.filter = orchestra_1.example.com, orchestra_2.example.com,`

❗ If this is not set up correctly, you will get error messages, such as *“Unable to fetch data from stat. Check the connection to stat.”* or *“Request remote address is unauthorized.”* when for example trying to fetch Custom field options in the Settings page, or trying to fetch customer information when exporting customer information.

Orchestra Queue Agent

Central & Distributed



A schematic picture of the Central and Distributed deployment scenario.

Install Distributed Queue Agent on Windows

Preparation

1. Transfer and unzip the *QP_Qagent_win_<type>-<version_number>.zip* file to a suitable directory.

Installation

1. Install the service by running the following file:

```
<install_dir>/bin/win-install-service.bat
```

ⓘ You need to run the installation as an administrator. Right-click on the file *win-install-service.bat* and select *Run as administrator*.

2. Modify the file *agent.conf*, located in the *conf* folder.

For more information see “Configure Queue Agent” on page 50.

3. Run the database scripts, as described in section “Database Configuration” on page 50.
4. Start the service created in step one.
5. Centrally, in the *System Administration* application in Orchestra, update the *Statistics Settings* section of the *Parameters* tab, so that the *Enabled* check box is checked, *Host Address* is set to *http://<ip-address-to-stat-server>* (normally the Orchestra central server) and *Address Port* is set to 8080.

See “Statistics Settings” on page 126.

Database Configuration

Run the scripts corresponding to your database.

1. Open a database tool that allows you to run a database script (e.g. SQL Developer for Oracle Database, or SQL Server Management Studio for Microsoft SQL) For PostgreSQL see special instructions below.
2. Run the script corresponding to your database in a tool that can execute the script:

Microsoft SQL Server:

```
<Queue Agent_install_dir>/conf/db/qagent-mssql.sql
```

Oracle Database:

```
<Queue Agent_install_dir>/conf/db/qagent-oracle.sql
```

PostgreSQL:

1. Make sure that the PostgreSQL bin folder is added to environment variables.
2. Open the command prompt in `<Queue Agent_install_dir>/conf/db/` directory.
3. Run the command:

```
psql -U postgres -f qagent-postgres.sql
```

Configure Queue Agent

This section is applicable to both Linux and Windows and for all Queue Agents.

1.  Skip this step, if having a central Queue Agent.

In the `<Queue Agent_install_dir>/conf/agent.conf` file, add or remove rows, depending on which database you want to use.

2. Update the values below in `<Queue Agent_install_dir>/conf/agent.conf`:

Parameter	Default value	Description
<i>agent.http.port</i>	18080	Web Server on Queue Agent
<i>central.host</i>	<ip-address-to-central-server>	Location of Central
<i>central.web-socket.port</i>	8787	Fixed, do not change!
<i>device.deviceController.web-socket.port</i>	18888	Port for Gateway to connect
<i>agent.https.enabled</i>	false	Used for changing the start protocol (http/https) of the start url for surfaces.

❗ Optionally, set the agent name to something else than *hostname*, by opening the file *run.bat/run.sh* and uncommenting the row:

```
:: SET AGENT_NAME=auto
```

To set the agent public ip/host address to something else than the default, open the file *run.bat/run.sh* and uncomment the row:

```
:: SET AGENT_HOST_ADDRESS=auto
```

For example, the *ip/host* parameter needs to be updated when using an external router. If the above parameters are not updated, their default values will be used.

3. Optionally, uncomment and configure the heartbeat interval property:

```
#central.heartbeat.intervalMillis = 30000
```

The default value is 30000 ms and the maximum value is 120000 ms.

Install Distributed Queue Agent on Linux

Preparation

1. Transfer the *QP_Qagent_linux<type>-<version>.tgz* file to the server. Preferably to a *tmp* directory. Use for example *scp* to transfer the file.

2. Log in to the Queue Agent server with a user with superuser rights.

3. Create a *qmatic* user:

```
#sudo adduser qmatic
```

4. Create Orchestra directory:

```
#sudo mkdir -p /opt/qmatic/orchestra/qagent
```

5. Grant rights for Orchestra directory to *qmatic* user:

```
#sudo chown -R qmatic:qmatic /opt/qmatic/orchestra/qagent
```

6. Switch to *qmatic* user:

```
#sudo su - qmatic
```

7. Unpack Orchestra *tgz* file:

```
$ tar zxvf QP_Qagent_linux<type>-<version>.tgz -C /opt/qmatic/orchestra/qagent
```

Installation

1. Switch to user with superuser rights:

```
$ exit
```

2. Change directory

```
#cd /opt/qmatic/orchestra/qagent/bin
```

3. Copy startup script to */etc/init.d*

```
#sudo cp qagent /etc/init.d/
```

4. Set appropriate rights:

```
#sudo chmod 755 /etc/init.d/qagent
```

5. Orchestra to be started/stopped at default runlevel:

```
#sudo update-rc.d qagent defaults 92
```

Note: If the command `update-rc.d` does not exist in the linux distribution you are using, you may use the below command instead which will do the same thing:

```
#sudo chkconfig qagent on
```

6. Modify the file *agent.conf*, located in the *conf* folder.

For more information, see “Configure Queue Agent” on page 50.

7. Run the database scripts, as described in section “Database Configuration” on page 50.

8. Start Queue Agent:

```
#sudo /etc/init.d/qagent start
```

9. Centrally, in the *System Administration* application in Orchestra, update the *Statistics* section of the *Parameters* tab, so that the *Enabled* check box is checked, *Host Address* is set to `http://<ip-address-to-stat-server>` (normally the Orchestra central server) and *Address Port* is set to 8080.

See “Statistics Settings” on page 126.

Gateway 1745

Gateway 1745 manages all the Qmatic hardware at the Branch and communicates with the central/distributed Queue Agent.

When there are no Qmatic 1745 devices in Branches, the Gateway 1745 software can be installed on the central server.

For more information about how to install Gateway 1745, please refer to the Gateway 1745 Manual.

Installing and Configuring Branch Hub and Hub

For more information about how to install and configure a Branch Hub, a Branch Hub Basic, or a Hub, please refer to the manual for each Branch Hub/Hub, which can be found on Qmatic World.

Redeploying Applications

If you, for some reason, need to redeploy one of your deployed applications, it is important that you make sure that you are using the wanted version of the application artefact (war-file).

The reason for redeploying an application may for example be if you have received a later version of the Connect application, or if you have translated your Calendar Admin and Calendar View applications.

Follow these manual steps, to make sure that you are really using the wanted application version:

1. In the Wildfly admin console (<http://localhost:9990>, username: *admin*, password: *ulan*), in the *Deployments* tab, undeploy the application that you need to redeploy.
2. Delete the applicable war-file from the `<install_dir>\deploy` folder.
3. Copy the new war-file to the `<install_dir>\deploy` folder.
4. Deploy the new version of the application in the Wildfly admin console, by clicking *Add*, looking up the new file, then clicking *Next* and making sure that the application is correctly enabled.

Configuration Variables for Applications

These variables are linked to their corresponding application.

The following actions are possible:

- For each application, it is possible to create multiple groups of variables.

- Retrieve all groups for an application.
- Create new groups.
- Delete a group.
- Create, update, or delete a variable in a group.

Security

It is also possible to add an extra level of access rights to these variables. In the file *shiro.ini*, it is possible to add a new line to restrict the access.

Example:

If we create an application named *xxx* and an application module named *xxxModule*, add the following line to restrict access to these variables, under the section [main]:

```
/qsystem/rest/config/applications/xxx /variables/** = oauth2, qpBasicAuthc,  
noSessionCreation, modules[xxxModule]
```

4. Upgrade

Topics in this chapter

Introduction	56
Upgrade Methodology	59
Orchestra Central	63
Upgrading from JBoss to Wildfly	70
Distributed Queue Agent	74
Unit Types	77
Widgets	79

Introduction

This chapter describes the prerequisites, dependencies and recommended methodology on how to upgrade both Orchestra Central system and distributed Queue Agents, as well as detailed descriptions of the upgrade scenarios.

❗ We recommend that you read the Upgrade Note sections below, the section “Upgrade Methodology” on page 59, as well as the Release Notes, before the Upgrade process is started.

Upgrade Note - Shiro Environment Class for Web Components

❗ This section should be read if you are using web components, that are not part of the standard Orchestra package, for example customized web applications.

Web Components (i.e. components that provide a web interface) to be deployed on Orchestra 7.1 or later must explicitly set a context parameter in order to work. The name of the parameter is *shiroEnvironmentClass* and the value of the parameter is *com.qmatic.qp.core.aaa.shiro.QPIniWebEnvironment*.

❗ A web component which has been changed according to this instruction will not be deployable to an Orchestra 7.0 update 2 (4.0.0.413) or earlier.

Example Using web.xml

A web component that uses a *web.xml* file can adjust by adding a section to the file, just before the section about *shiroFilter*:

```
<context-param>
  <param-name>shiroEnvironmentClass</param-name>
  <param-value>com.qmatic.qp.core.aaa.shiro.QPIniWebEnvironment</
  param-value>
</context-param>

<filter>
  <filter-name>shiroFilter</filter-name>
  <filter-class>com.qmatic.qp.core.aaa.shiro.QPShiroFilter</
  filter-class>
</filter>
```

Example Using Spring Boot

A web component that uses Spring Boot can adjust by overriding the method *onStartup*:

```
@SpringBootApplication
public class MyApplication extends SpringBootServletInitializer {
    @Override
    public void onStartup(ServletContext context) throws
        ServletException {
        context.setInitParameter("shiroEnvironmentClass",
            "com.qmatic.qp.core.aaa.shiro.QPIniWebEnvironment");
        super.onStartup(context);
    }
}
```

Upgrade Note - Undeploying Jiql.war

If you experience problems that the upgrade gets stuck when undeploying *jiql.war*, please follow these steps to work around these problems:

1. Open the Wildfly Admin console (*http://localhost:9990*, username: *admin*, password: *ulan*).
2. Go to *Deployments* and select *jiql.war*. Click *En/Disable* and then *Confirm*.
3. Run the upgrade again.

Upgrade Note - Stat Aggregation Jobs are Active

If you experience problems that the upgrade gets stuck with error message "*Stat aggregation jobs are active*", you need to wait until the jobs are done and try again.

For more information about controlling Stat aggregation jobs, see Stat REST API in the connector SDK.

Stat Standalone

The upgrade will try to check if aggregation jobs are running on *http://localhost:8080*. If this fails, the upgrade will be aborted.

If this happens, please run the tests manually. Instructions can be found in the Stat REST API. Then change the *application.stat.aggregation.check* flag to *false* in the *install.properties* file, so that the upgrade will skip the tests. Run the upgrade again.

Upgrade Note - Enabling Auditing

For instructions on how to enable auditing when upgrading, please see "Upgrade" on page 145.

Upgrade Note - Blocking Websocket Port

If running a distributed system with one, or more, distributed Queue Agents, there is a possibility that the connected Queue Agents can cause problems as they reconnect during the upgrade procedure.

To work around this, block the port 8787 on the computer running the central system during upgrade.

This can be done in the *Windows Control Panel -> Windows Firewall -> Advanced Settings*:

1. Right-click *Inbound Rules* and select *New Rule*.
2. Select *Port* and click *Next*.
3. Select *TCP* and specify port 8787. Click *Next*.
4. Select *Block the connection*, click *Next*.
5. Select all available networks and click *Next*.
6. Give a name to the new rule, e.g. *"Orchestra upgrade 8787 block"* and click *Finish*.

ⓘ Once the upgrade of the central system is complete, the port block must be removed. This is done in the *Windows Control Panel -> Windows Firewall -> Advanced Settings*.

Select *Inbound Rules*, right click the rule created (e.g. *"Orchestra upgrade 8787 block"*) and click *Delete*.

Upgrade Note - Language Files

When upgrading, there are a couple of things that you need to keep in mind, when it comes to the handling of language files:

- The existing language property files will automatically be backed up to the folder *upgrade_backup*, during upgrade. This way, you can easily restore them, if the encoding somehow fails, during upgrade.
- Since changes in language files are not applied when doing an upgrade, make sure you compare the language files in the installation package with the language files on the server after the upgrade and edit any updated strings.

Example: If the label of an existing system parameter is changed, you must manually make this change

- Make sure that you translate any new strings that have not yet been translated, if applicable.

Upgrade Methodology

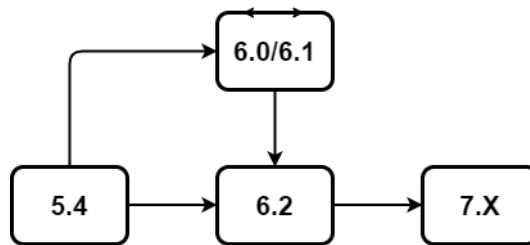
General Information

⚠ Please note that the Upgrade wizard will run all necessary scripts etc for you. There is no need to run any scripts manually, unless specifically stated!

Possible Upgrade Path

Before you start, you need to make sure that you know

- which is your current version of Orchestra. The version number can be found in the footer of Orchestra.
- what possible upgrade path(s) are available to you, depending on your current Orchestra version. The image below illustrates this:



Note that you can upgrade to any minor version within the same major version

Before you begin (Test and Production environment) - Checklist

Step	More information	Check
Check your possible upgrade path.	See "Possible Upgrade Path" on page 59	
Make sure that you have read through the Release Notes document, since it contains important information that needs to be considered before Upgrade commences!		
Read through the applicable Upgrade Note sections.	See: • "Upgrade Note - Shiro Environment Class for Web Components" on page 56 • "Upgrade Note - Undeploying Jiql.war" on page 57 • "Upgrade Note - Stat Aggregation Jobs are Active" on page 57 • "Upgrade Note - Enabling Auditing" on page 57 • "Upgrade Note - Blocking Websocket Port" on page 58 • "Upgrade Note - Language Files" on page 58	
Make sure that you have your license details in place.	ⓘ You can <i>not</i> use the same license key as for your current Orchestra version, if that version is Orchestra 5.x, since the Licensing model has been completely updated for Orchestra 6.x. For more information, see "Licensing" on page 108.	
Obtain / download all software artefacts to be upgraded.		
Make sure that the Orchestra installation folder isn't set to <i>read only</i> (can be configured in "Properties" for that folder)		

Step	More information	Check
Backup installation folder.	⚠ If you use the normal Windows operation of copying and pasting the folder, or dragging and dropping it, you may encounter problems with file names being too long. This means that some files will not be copied and you will be prompted about this problem. If this happens, cancel the copy operation and remove anything that was already copied to the target folder. Then, try using robocopy command line tool, instead. Entering "<i>robocopy system backup /E</i>" on the command line, for example, will copy the folder named "<i>system</i>" and make a copy named "<i>backup</i>". Change these names as desired!	
Backup databases.		
Backup media content files. Optionally snapshot VM if using virtual machine technology for host.	Media content is all media files, such as images and videos, that are running on Media Displays (via Cinematic) and similar surfaces.	
Prepare Regression Tests.	See "Regression Test" on page 120.	
Prepare Performance Tests.	See "Performance Test" on page 120.	

Suggested Upgrade Order

Naturally, depending on how your system is set up and configured, another upgrade order may be applicable for you. However, the following can be seen as a suggested guideline, with references to the applicable sections of this document:

1. Upgrade **central** and **stat**. These two components need to be upgraded at the same time. See “Orchestra Central” on page 63.
2. Upgrade **distributed queue agents**. See “Distributed Queue Agent” on page 74.

ⓘ To be able to verify a successful central setup, you need to upgrade at least one distributed queue agent. When central and stat have been successfully verified, you can upgrade the rest of the distributed queue agents. The reason for this is that central and stat cannot be reverted if queue agents have been upgraded.

However, there is nothing that prevents the upgrade of Central and Stat at the same time as the distributed Queue Agents, given that the Central and Stat upgrade has been verifying before doing the distributed Queue Agent upgrades.

Please note that old queue agents must be upgraded as soon as possible and a combination of different versions must only be used temporary. Central and stat can work together with older queue agents (not more than 2 main versions back), but these older queue agents will not have all the new functionality.

3. When upgrading distributed agents on QEP based products, we advise to upgrade the **QEP firmware** at the same time. New QEP firmware includes security fixes and other improvements.
4. Upgrade **unit types** and **widgets**, if you choose not to upgrade them automatically during the upgrade process. See “Unit Types” on page 77 and “Widgets” on page 79.

If testing fails at any point in the process, restore the environment from backups, address the problem, and repeat the process.

Orchestra Central

General Information

⚠ Upgrade directly from Orchestra 5.4 is not supported. You need to upgrade to Orchestra 6.x before upgrading to Orchestra 7.x. For more information, see “Possible Upgrade Path” on page 59.

⚠ Please note that the Upgrade wizard will run all necessary scripts etc for you. There is no need to run any scripts manually, unless specifically stated!

⚠ When upgrading from 6.0, the Queue Agent database will be removed.

⚠ Note that the ip address for stat is fetched from system parameters (Stat Server Address) while the port is fetch from install.properties (standalone.stat.server.port /...port.secure).

⚠ In the Upgrade Instructions below, the already installed version of Orchestra is named `<old_version>` and the version of Orchestra that you want to upgrade to is named `<new_version>`.

⚠ For an instruction about Daemon Upgrade, please refer to the Release Notes.

The following upgrade scenarios are described in this chapter:

- “Orchestra Central on Windows” on page 65
- “Orchestra Central on Linux, MS SQL Server/Oracle Database” on page 69
- “Distributed Queue Agent” on page 74

Pre-upgrade Checklist

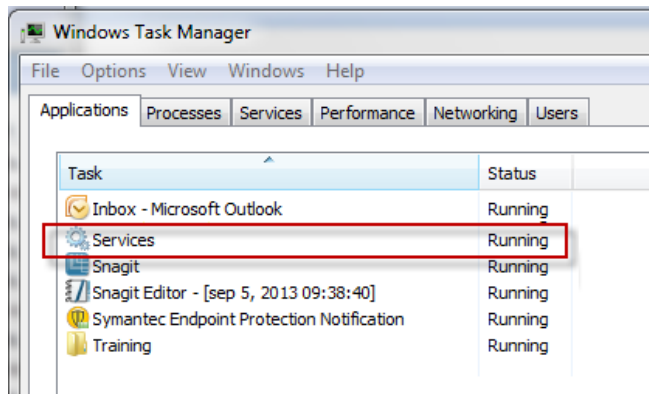
The following table can be seen as a checklist of things that need to be considered before going ahead with the upgrade.

We recommend printing it, so that each prerequisite can be carefully checked before going ahead. Some of the prerequisites may not be applicable for you. If that's the case, mark them as *N/A* and move on.

Prerequisite	Check	N/A
Please read through the Release Notes document before the upgrade procedure is started.		
If you were using JBoss in Orchestra 6.x, please read the section "Upgrading from JBoss to Wildfly" on page 70.		
Backup your system		
If you are upgrading and your current Orchestra system is using MS SQL Server 2008, you first have to upgrade your database server, before upgrading Orchestra. (Orchestra 6.2 and later versions do not support MS SQL Server 2008.)		
Some settings in <i>shiro.ini</i> file are replaced during upgrade. Therefore, you should make a copy of this file before upgrade.		
All distributed Queue Agents must be upgraded to the second latest version, before Central is upgraded.		
When it comes to unit type files and widgets, please read "Unit Types" on page 77 and "Widgets" on page 79, before you begin.		

Orchestra Central on Windows

1. Make sure that Orchestra is running before you proceed.
2. Unzip the `<new_version>` of `QP_Central_win_<type>-<version_number>.zip` files into a `tmp` directory.
3. Open the Windows Task Manager and ensure that there is no application called "Services" running, as in the picture below. If there is, close the *Services* application.

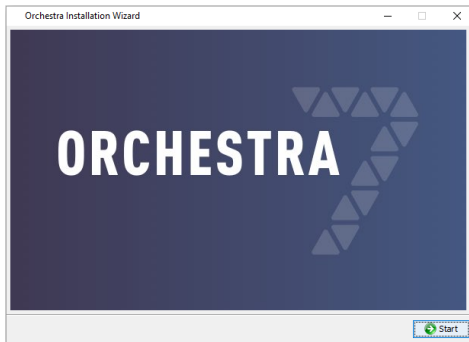


4. Run the Installation Wizard and select *Upgrade*. The steps of the Installation Wizard are described below.

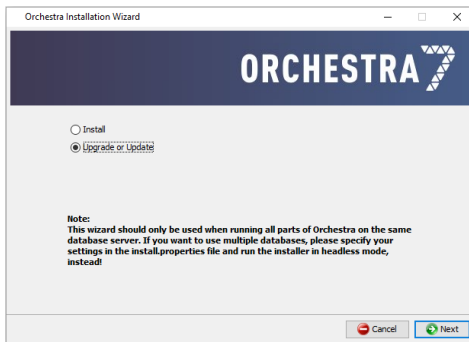
```
/<tmp_dir>/install.bat
```

! You need to run the wizard as an administrator. Right-click on the *install.bat* file and select *Run as administrator*.

Settings in the Installation Wizard

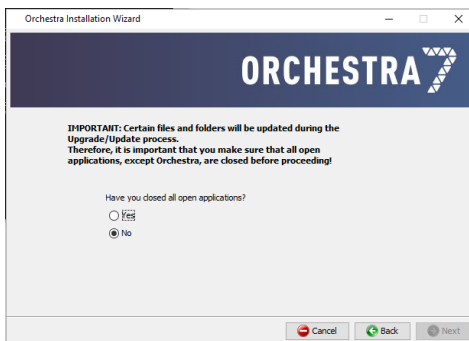


1. Click *Start* to start the upgrade.



2. Select *Upgrade or Update*.

⚠ This wizard should only be used when running all parts of Orchestra on the same database server. If you want to use multiple databases, please specify your settings in the *install.properties* file and run the installer in headless mode instead.



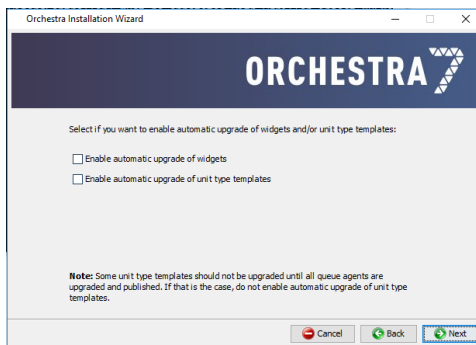
3. Make sure that you have closed all open applications (except Orchestra) before proceeding.

4. Enter the path where your current Orchestra system is installed.



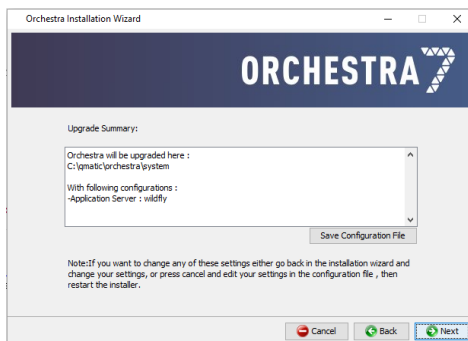
5. If you installed Orchestra, using SQL Server and Windows Authentication, see “Installation Note - Windows Authentication” on page 23, you need to enter and confirm your *Password* in this screen.

⚠ This password needs to be the same as the one entered during installation.

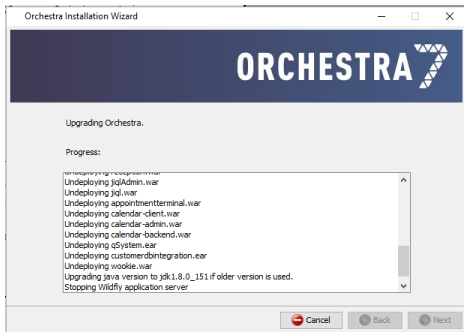


6. Select if you want to automatically upgrade unit type templates and widgets.

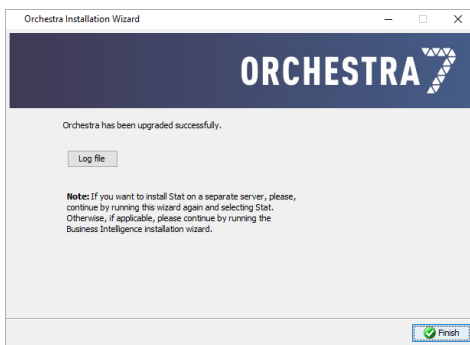
Some unit type templates should not be upgraded until all queue agents are upgraded and published. Please check release notes to see if there are any dependencies. If that is the case, do not enable automatic upgrade of unit type templates.



7. In the *Upgrade Summary* window, you can decide to go back and change your settings before you proceed. Also, here you can save the configuration file, if wanted.



8. Monitor the progress of the upgrade in the next page.



9. Once Orchestra is successfully upgraded, click *Finish*. You can also download the log file from [here](#).

10. Wait for the installation to finish. Orchestra is started.
11. If you choose to not upgrade unit type templates and widgets automatically, please read our recommendations in “Unit Types” on page 77 and “Widgets” on page 79.
12. Publish the branches connected to the central Queue Agent.
13. Start your browser and log in to the Orchestra GUI (if this is not already done).
- ❗ We recommend that you clear the cache of your browser, after the central upgrade procedure has been completed.
14. If you want to import a system configuration, we recommend that you do this *before* any other configuration is done.

For more information, see “Import and Export” on page 90.

- ❗ Note that a new `stat-jobs.xml` file has been installed at `<Orchestra_installation_directory>/conf/stat-jobs/<version>/`. This needs to be merged with the existing `stat-jobs.xml` file, after upgrade, or update.

Orchestra Central on Linux, MS SQL Server/Oracle Database

❗ Please read through the Release Notes document before the upgrade procedure is started.

❗ When it comes to unit type files and widgets, please read “Unit Types” on page 77 and “Widgets” on page 79, before you begin.

1. Make sure that Orchestra is stopped before you proceed.
2. Transfer the `QP_Central_linux<type>-<version>.tgz` file to the server, preferably to a `tmp` directory. Use for example `scp`.
3. Change to a user with superuser rights.
4. Stop Orchestra:

```
/etc/init.d/orchestra stop
```

5. Switch to `qmatic` user:

```
sudo su - qmatic
```

6. Unpack Orchestra `tgz` file in a `tmp` directory:

```
% cd <directory where file is stored>
% tar zxvf QP_Central_linux<type>-<version>.tgz
```

7. When running the installation script, it is recommended to tail the log file, this is done in a separate window:

```
% cd /<new_version_install_dir>/logs
% tail -f qp_server.log
```

8. Run the Installation Wizard, `install.sh` and select the *Upgrade* option.

For information about the steps of the installation wizard, please see “Settings in the Installation Wizard” on page 66.

```
% cd /<install_dir>/install.sh
```

❗ If you are running the Installation Wizard on Linux Server, without a graphical interface, then you must first edit the following values in the *install.properties* file:

- `operation = upgrade`
- `install.path = <your orchestra install dir>`
- `unittemplates.update = false/true`
- `widgets.update = false/true`

Then, execute the following command:

```
% ./install.sh -s
```

9. Wait until the upgrade has finished.

```
$ exit
```

10. If you choose to not upgrade unit type templates and widgets automatically, please read our recommendations in “Unit Types” on page 77 and “Widgets” on page 79.

11. Publish the Branches connected to the central Queue Agent.

12. If you want to import a system configuration, we recommend that you do this *before* any other configuration is done.

For more information, see “Import and Export” on page 90.

❗ Note that a new `stat-jobs.xml` file has been installed at `<Orchestra_installation_directory>/conf/stat-jobs/<version>/. This needs to be merged with the existing stat-jobs.xml file, after upgrade, or update.`

Upgrading from JBoss to Wildfly

If you are using Orchestra 6.x with JBoss, you need to follow the upgrade process below to upgrade to Wildfly, since JBoss is no longer supported in Orchestra 7.0 and later.

❗ The instructions below are for a system with Orchestra and Stat on separate servers and with SSL enabled. For other set ups, please contact the Global Support organization.

Orchestra Server

Follow these steps:

1. Prepare an Orchestra 6.2 system with Oracle/JBoss on a separate server and same domain policy.

❗ Make sure that the Orchestra 6.2 system is update 15 (build 3.2.0.460), or later.

2. Enable HTTPS on the Orchestra 6.2 system. For more information, see the Orchestra 6.2 Reference Manual.

3. Backup the Orchestra central installation folder, by simply renaming it.

4. Install Orchestra 6.2 with Wildfly and PostgreSQL with the same applications as in the JBoss version.

❗ PostgreSQL is only used temporarily, it will be removed later.

5. Copy the *conf/security* folder from your backed up JBoss system to your new Wildfly installation.
6. Start up the new Orchestra installation and open the *System Administration* application. In *System > Settings > Parameters*, enable SSL in the same way as in your previous system. For more information, see the Orchestra 6.2 Reference Manual.

The parameters should be set in the following way:

- *HTTPS server settings*: *HTTPS enabled* parameter enabled.
- *Queue Agent Settings*: Select *https*
- *Central WebSocket Server Settings*: *WebSocket enabled* parameter enabled.

! This is done to get HTTPS configured correctly in the file *standalone-full.xml* for Wildfly.

7. Enable/configure JMS bridges for supporting standalone Stat. Follow the instructions in the *Setting up Stat Standalone* section of the Orchestra 6.2 Reference Manual.
8. Stop the Orchestra server.
9. Copy the *media* folder from the backed up JBoss system to the new Wildfly installation.
10. Copy any customized widgets from the backed up JBoss system to the new Wildfly installation.
11. Copy the *<Orchestra installation directory>\system\conf* folder from the backed up JBoss system to the new Wildfly installation.
12. Reconfigure data sources to point to the old database.

The following files affect database access in some way and should be adapted:

- *<orchestra install dir>\system\pentaho-solutions\system\hibernate\hibernate-settings.xml*
- *<orchestra install dir>\system\pentaho-solutions\system\hibernate\oracle10g.hibernate.cfg.xml*
- *<orchestra install dir>\system\pentaho-solutions\system\jackrabbit\repository.xml*
- *<orchestra install dir>\system\pentaho-solutions\system\quartz\quartz.properties*
- *<orchestra install dir>\system\pentaho-solutions\system\dialects\oracle10g\applicationContext-spring-security-hibernate.properties*
- *<orchestra install dir>\system\pentaho-solutions-jboss-oracle\system\jackrabbit\repository\workspaces\default\workspace.xml*

- `<orchestra install dir>\system\pentaho-solutions-jboss-oracle\system\jack-rabbit\repository\workspaces\security\workspace.xml`
 - `<orchestra install dir>\system\pentaho-solutions\qmaticbi\platform\live.mondrian.xml`
 - `<orchestra install dir>\system\pentaho-solutions\qmaticbi\platform\qmatic.mondrian.xml`
 - `<orchestra install dir>\system\pentaho-solutions\system\audit_sql.xml`
13. Uninstall the PostgreSQL service and the QP service (which has a dependency to PostgreSQL).
 14. Edit the file `install-service.bat` for the QP service and remove the following directive:

```
++DependsOn=QPPostgreSQL
```
 15. Re-install the QP service.
 16. Start the Orchestra server.

Stat Server

Follow these steps:

1. Backup the installation folder by simply renaming it.
 2. Install Orchestra 6.2 with Wildfly and PostgreSQL with the same applications as in the JBoss version (Stat)
- ❗ PostgreSQL is only used temporarily, it will be removed later.
3. Copy the `\conf\security` folder from the backed up JBoss system to the new Wildfly installation.
 4. Copy the `\conf\stat-jobs` folder from the backed up JBoss system to the new Wildfly installation.
 5. Copy the following files from the backed up JBoss system to the new Wildfly installation:
 - `\system\pentaho-solutions\system\hibernate\hibernate-settings.xml`
 - `\system\pentaho-solutions\system\jackrabbit\repository.xml`
 - `\system\pentaho-solutions\system\quartz\quartz.properties`
 - `\system\pentaho-solutions\system\dialects\oracle10g\applicationContext-spring-security-hibernate.properties`
 - `\system\pentaho-solutions\qmaticbi\platform\live.mondrian.xml`
 - `\system\pentaho-solutions\qmaticbi\platform\qmatic.mondrian.xml`
 6. Edit the `<datasources>` section in `standalone-full.xml`, by copying it from your previous JBoss installation.
 7. Stop the Stat server.

8. Uninstall the PostgreSQL service and the QP service (which has a dependency to PostgreSQL).
9. Edit the file *install-service.bat* for the QP service and remove the following directive:

 `++DependsOn=QPPostgreSQL`
10. Re-install the QP service.
11. Start the Stat server.
12. Enable/configure JMS bridges for supporting standalone Stat. Follow the instructions in the *Setting up Stat Standalone* section of the Orchestra 6.2 Reference Manual. Add a jms User.
13. Set up HTTPS manually, by updating the `\app\wildfly-11.0.0.Final\standalone\configuration\standalone-full.xml` file:
 - a) Add an “*orchestraSSL*” security realm to the `<management><security-realms>` section.
 - b) Add the https listener to the `<server name="default-server">` section. Make sure that *socket-binding* uses port 8443.

All Servers

Follow these steps:

1. Upgrade Orchestra and Stat to Orchestra 7. See “Orchestra Central” on page 63.

Distributed Queue Agent

The following information applies to both Linux and Windows.

For a description of the Remote Upgrade procedure, please see “Suggested Working Order, Remote Upgrade” on page 75.

Important information

- Read through the Release Notes document before the upgrade procedure is started, since it may contain special instructions.
- If there are visits in the queues, those will be lost.
- Central must be upgraded before distributed Queue Agent upgrade.
- Remote upgrade will only work from Orchestra R5.3 to later versions. You should always upgrade to the next version, e.g. 5.3.0 to 5.3.1. If the distributed Queue Agent version is lower than Orchestra 5.3, the Queue Agent must be reinstalled.
- When upgrading from 6.0, the Queue Agent database will be removed.
- The file `/conf/security/keystore.jks` will not be synchronized with Agent profile upgrade. This is because you are required to have a secure communication for updating the keystore. If you wish to update the keystore, follow the instructions in “Secure Communication for a Queue Agent” on page 237.
- If you have changed Orchestra Central to run on another port than 8080, it is no longer possible to synchronize a new agent profile to the distributed Queue Agents. Therefore, you must *first* upgrade the distributed Queue Agents, using Remote Upgrade, with the new port, *before* the central port number is changed.

Remote Upgrade Overview

It is possible to upgrade the distributed queue agent software from the central Orchestra installation. This is called remote upgrade.

Before you can upgrade, you need a Queue Agent profile. The profile defines what software files and which parameters that shall be applied for one or more Queue Agents.

An agent profile consists of a zip file containing the web applications (Counter, Reception etc) and required configuration files, such as language files and scripts. An agent profile also consists of configuration parameters (stored in database) that overrides the global system parameters.

Create a new agent profile when a new Queue Agent version is released or when you are about to perform other changes on a Queue Agent, such a replacing Counter with a newer version. By creating a new agent profile, a Queue Agent could easily be reverted in case of any failure during deployment.

It is only recommended to change/update an existing agent profile if e.g. a change of a parameter is needed, or a language property file should be added or changed.

Read more in “Agent Profile” on page 103.

Suggested Working Order, Remote Upgrade

The following is a suggested working order, when it comes to upgrading a distributed queue agent.

1. Create a new agent profile zip file with the new Queue Agent version.

Copy a template profile from `<central install dir>/media/agentProfiles/<version>`. Make any wanted changes, for example replacing Counter with a newer version.

For more information see “Creating an agent profile zip file” on page 106.

2. Create a new agent profile in Orchestra

Go to System Administration > Queue Agents > Agent Profile and click Create Profile. Upload the agent profile zip file and make the wanted configuration. Click Save.

For more information, see “Creating an agent profile in Orchestra” on page 109.

3. Assign the profile to a Queue Agent:

Go to System Administration > Queue Agents > Agents and select the agent you want to upgrade. In the list of available profiles, click Prepare Profile for the profile you just created.

For more information see “Assigning an agent profile” on page 110.

4. Synchronize the profile

Click Synchronize to start the synchronization of the selected profile.

For information, see “Synchronizing the Queue Agent profile” on page 111.

5. Deploy the profile (downtime is required):

When the Queue Agent synchronization is done, click Upgrade to deploy the new profile to the queue agent.

⚠ Do not have files in the Queue Agent installation directory open during upgrade.

6. After the remote upgrade has been completed, you need to manually publish each branch on the queue agent.

Regression and Performance Test

If needed, you should do a regression test when you have updated the distributed queue agent.

What is covered by the regression test naturally largely depends on what your system looks like, how many branches you have, which hardware you are using and so on.

However, we suggest that you perform tests on the branch that you have deployed as the first branch and that you test scenarios that relate to the normal scenarios of your business, such as calling customers, printing tickets, making sure that correct surfaces are displayed on printers, media displays and so on.

For performance tests, we suggest that you run some kind of load tests, using tools and load scripts especially developed for this kind of task.

If testing fails, the failure should be investigated and a re-deployment could be done, if the failure is considered to be related to the deployment. If roll back should be done, it should be decided if all Queue Agents should be rolled back.

Roll back can be done by synchronizing the Queue Agent profile that was previously deployed and then deploying this profile on the Queue Agent.

Unit Types

A new release of Orchestra can include a new version of the Unit Type Files (.utf).

❗ Make sure that you read the release notes, before following the instructions below, since it contains information about what unit types that have new versions.

❗ Once a new unit type has been uploaded, it may include new parameters. It is therefore recommended to go through the Operation Profile(s) and Branch(es) that use the upgraded utf file and make sure that everything is correctly configured.

If you want new unit types to be uploaded automatically to Orchestra after an upgrade, you can either select that option in the wizard or, if you run the silent installer, set the parameter `unittemplates.update = true`.

If you did not set Orchestra to automatically import new unit types before an upgrade, or if you for example want to add third party unit types, you can always copy unit type files to `conf/unittemplates/import` at any point. These will be imported to Orchestra at the next restart.

You can also add unit types manually from the *System Administration* application. In this case, the unit types will be imported directly (no restart needed).

1. Log in as an admin user to Orchestra Central (in order to have access to the new unit types, it is recommended to run a browser on the server where Orchestra is installed, or where you have access to the disk on the Orchestra server).
2. Open the *System Administration* application.
3. Select the *Unit Types* tab.
4. Click the *Add Unit Type* button.
5. Click the *Select File* button and browse for the new unit types, that are stored on the Orchestra server, in the following directory (after a Central upgrade):

`<Orchestra install dir>/conf/unittemplates/R7`

This is where all unit types for the latest Orchestra version can be found. To find out which of the unit types that should be upgraded, please read the Release Notes. Sometimes it is recommended to upgrade all unit types with new versions, after all Queue Agents have been upgraded.

6. Select the file and click *Open*.
7. Click *Upload*.

A message will be displayed that the unit type was successfully uploaded.

❗ If you try to upload a unit type with the same or lower version, the following message will be displayed:



8. Save and publish the Branch(es).

Special considerations regarding Unit Type Files (.utt)

Unit type files (.utt) that are developed for Orchestra version 5.3.x, or earlier, may not work with Orchestra version 7 Queue Agents. Therefore, when upgrading from Orchestra version 5.3.x, or earlier, to Orchestra version 7, or later, we recommend that you follow these steps, during upgrade:

1. Upgrade Central from version 5.3.x to version 6.2. Then, upgrade Central to version 7.

See “Orchestra Central” on page 63.

Leave the unit type files (.utt) as they are, i.e. keep the unit type files and do not upgrade them at this stage.

2. Start to upgrade one Queue Agent after the other.

See “Distributed Queue Agent” on page 74.

The unit type files from version 5.3.x will work in the 7 version Queue Agents.

3. When all the Queue Agents are upgraded, upgrade the unit type files, centrally, to version 7 unit types, as described above.

ⓘ If you want to try the unit type files for Orchestra version 7 *before* all Queue Agents are upgraded, you need to rename them, by changing the unit name in the *unit.xml* file, as in the following example:

```
<unit name="Intro17"
      frameworkVersion="2"
      unitVersion="2.3.0"
      type="ENTRY_POINT"
      defaultNoUnits="1"
```

Then, upload them next to the existing unit type files. Create a new Equipment Profile, using the new unit type files, and link the new Equipment Profile to the Branch(es) where you want to try the new unit types.

Widgets

The upgrade of widgets largely follows the same pattern as the upgrade Unit Type files procedure. See “Unit Types” on page 77.

ⓘ Before going through with an upgrade, please read through the Release Notes document to see if there are any special considerations regarding widgets.

If you want new widgets to be uploaded automatically to Orchestra after an upgrade, you can either select that option in the wizard or, if you run the silent installer, set the parameter `widgets.update = true`.

If you forget to set Orchestra to automatically import widgets before an upgrade, or if you for example want to add third party widgets, you can always copy widget files to `conf/defaultwidgets/import` at any point. These will be imported to Orchestra when Orchestra is restarted.

ⓘ Normally, the widget files in the import folder will be deleted after the files have been imported to Orchestra. However, if the copy fails, the files will not be deleted.

You can also add widgets manually from the *System Administration* application. In this case, the widgets will be imported directly.

Follow these steps:

1. Go to the *System Administration* application with system admin rights and select the *Widgets* tab.
2. Click the *Browse* button and browse for the wanted new widget. After a central upgrade, the new widget files are located in the following directory:
`<Orchestra install dir>/conf/defaultwidgets/R7`.
3. Click *Upload* to add it your list of uploaded widgets.
4. Make sure that all the surfaces that were using the old widget are correctly configured, so that the correct version of the widget is used and that new attributes, if any, are set correctly.
5. Save and publish the branch(es).

5. System Administration

Topics in this chapter

Unit Types	82
Licensing	84
Import and Export	90
Widget Administration	93
Queue Agents	97
Parameters	112
Mark Types	138

Unit Types

Unit Types

Save

Discard changes

Add Unit Type



Info

Name

Intro17

Description

Intro 17 Touch Screen with a Ticket Printer

Template name

Intro17

Template version

2.5.0

Parameters

Tickets on roll

2000



Info

Name

Intro5

Description

Intro5

Template name

Intro5

Template version

2.4.0

Parameters

Tickets on roll

2000



Info

Name

Intro8

Description

Intro8

Template name

Intro8

Template version

2.4.0

Parameters

Tickets on roll

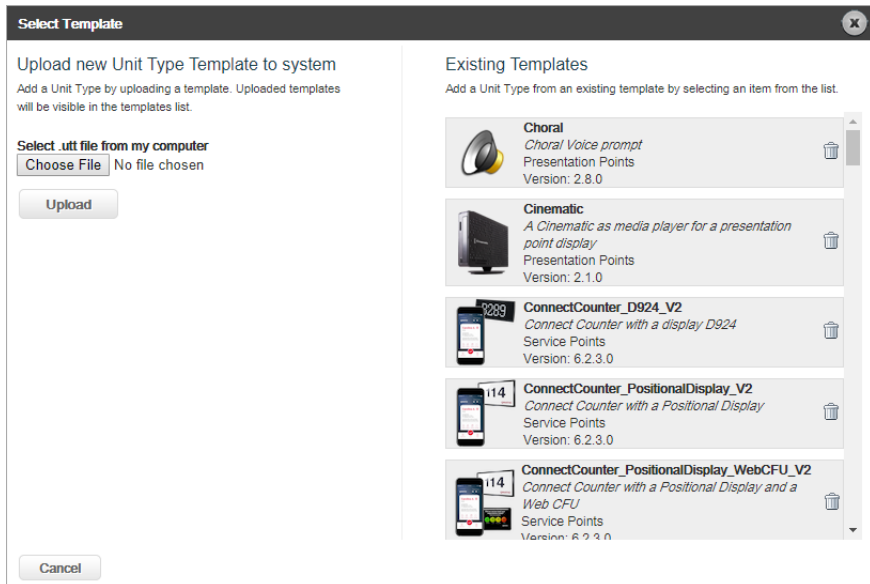
2000

The following Unit Types are available:

- Entry Points.
- Service Points.
- Presentation Points.
- Device Controllers.

For more information about the Standard Unit Types, please see the Standard Unit Types Guide.

To add a new Unit Type, click the *Add Unit Type* button. The *Select Template* window is displayed:



Either select a Template from the *Existing Templates* list, or click the *Choose File* button and then the *Upload and Add* button to upload a Unit Template.

Do not forget to click the *Save* button once you have uploaded all the wanted Unit Types.

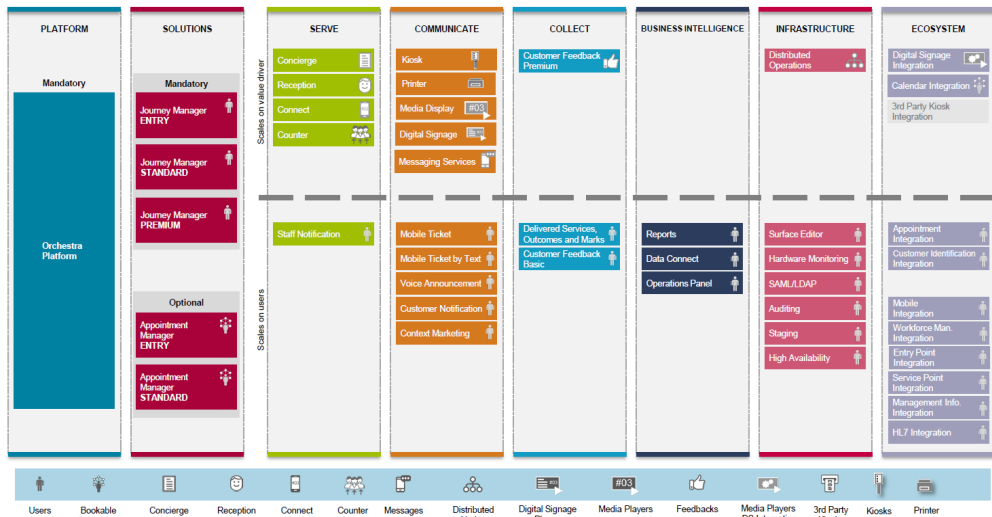
In the Business Configuration application, Units are added to your Equipment Profile. For more information, please see the Administrator's Guide.

Licensing

This section describes the procedures to follow when managing the licence of the system.

Orchestra Modularization Model

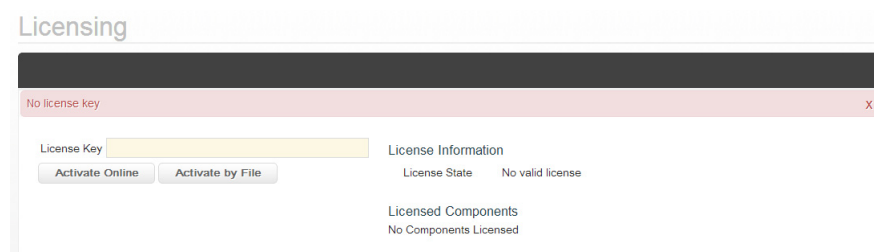
The modularized licensing model for Orchestra 7 looks like this:



For more information, please see the document “*Orchestra 7 Licensing Model*”, found on Qmatic World.

Activating a license

The licence is a file with information about the licence number, number of users, options and applications.



There are two ways to activate a licence:

- On-line. Direct activation in the system. Access to the internet is needed.
- By file, with files that are sent and received.

This is described in more detail below.

Online activation

⚠ It is not possible to activate a licence online when a Proxy server is used to access Internet.

If you have performed an online activation and then want to add more components to your license, you need to update your license online. Please see “Update and Deactivate a licence” on page 87 for more information.

If you want to reinstall Orchestra, or move the installation to another server, you need to deactivate your license first. For more information, please see “Update and Deactivate a licence” on page 87.

From Qmatic, you receive information that your licence is installed on the licence server and ready for activation.

To activate the licence on-line, enter the *License Key* in the field and click the *Activate Online* button.

⚠ If you try to activate an incorrect license, you will get an error message stating “Invalid serial”. The solution is simply to make sure that the correct license file is used.

The license information is read by the system and activated. After a few moments, the window will look something like this:

Licensing

License activation		Licensed Components	
License Key	XXXX-XXXX-XXXX-XXXX-XXXX-XXXX	Component name	License limit Current usage
Update Online Deactivate License		Analysis	60 0
		Concierge	50 0
		Connect	500 0
		Counter	900 1
		Reception	100 2
		Distributed Operations	50 0
		Platform	1000 4
		Appointment Integration	
		Auditing	
		Hardware Monitoring	
		LDAP	
		Operations Panel	1000
		Surface Editor	
		Appointment Manager Standard	100
		Journey Manager Premium	1000

License Information	
License State	Active license
Licensed Product	Orchestra 7 4.0.0
Expiry Date	No expiry date
License Key	QZKP-CVAB-ZMXH-VEOI-BLLX-GIRT-FXON
Licensed Customer	value

Licensed packages	
No packages in license	

The following information is displayed:

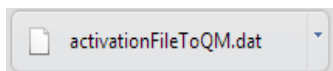
- License Information
 - License State - whether or not the license is active.
 - Licensed Product - name and version of the licensed product.
 - Expiry Date
 - License Key
 - Licensed Customer
- Licensed packages - article numbers of packages deployed via the Orchestra Provisioning Tool.
- Licensed Components - in this table, all licensed components are listed. You can also see each component's *License limit* and *Current usage*. It is possible to click on the name of the module, to navigate to a filtered view of the *User* list, to see which Users currently have that module.

Offline activation

⚠ The process of activating a license offline must be completed within 14 days. Otherwise, the activation will fail.

To activate a license off-line, enter your licence key in the field and click the *Activate by File* button.

A file will be available for download, in your browser:



Save the file in a suitable directory.

Go to the *Licensing* tab in Qmatic World to upload the generated file and then download the licence file.

With the licence file in a suitable directory, click the *File Activation* link.

Click the *Add* button and browse to the directory where the licence is located and open it. Click the *Upload* button to make the system read the file, then click the *Activate* button.

The licence information is read by the system and activated.

Update and Deactivate a licence

With an on-line licence, it is possible to directly Update and Deactivate the licence on-line: just use either the *Deactivate License*, or the *Update Online* button. This is useful if you, for example, want to add more components to your license. Another scenario where this needs to be done is when you want to reinstall Orchestra, or when you want to move your installation to another server.

With a File-activation licence, you can either update by file (*Update by File* button) or online (*Update Online* button).

Reaching and Passing the License Limit

When you reach and pass the limit of how many users you are allowed to have, according to your license, creation of new users is blocked. Also, it is not possible to add a role that contains a module that has passed the license limit, to a user. If a module has passed the license limit, this will be indicated in the GUI.

In the *Licensing* tab, click on the name of the module, e.g *Reception*, to navigate to a filtered view of the *User* list, to see which users currently have that module.

Third-party hardware licensing

Software licenses are required for third-party hardware devices that connect with Qmatic Orchestra. The following license types exist:

- Kiosk
- Display
- Printer

You can see the license type in the settings for the unit type template.

On the *Edit branch* page, there is a section with hardware license information where you can see the number of licenses used by the branch, the number of remaining licenses, and the total number of licenses.

License count	This branch	Total remaining
Media Display	101	-1 of 100
Printer	0	5 of 5
Kiosk	2	3 of 5

If you try to add too many units on branch level, you won't be able to save the branch.

If you add too many units on equipment profile level, those units will be disabled on the branch until you enable them.



Please note the following:

- Hardware licenses for Qmatic-manufactured hardware is free of charge

- Disabled branches and disabled unit types are not included in the license count
- If a license is breached due to a license being deactivated, or due to an upgrade, license information is shown on the *Branches* page and the *Publishing branches* page.

Branches

Create New Branch

License count	Total	Used	Remaining
Media Display	0	100	-100
Kiosk	0	2	-2
Printer	0	0	0

Name	Address	City	Operation Profile	Equipment Profile	Dr
------	---------	------	-------------------	-------------------	----

Import and Export

An Orchestra system configuration can be saved to a *zip* file and then imported into another system. The importing system must be a newly installed system, that is no configuration has been done.

Also, an exported system configuration can be sent to Qmatic for problem solving in a service assignment.

Several similar system configurations can be set up, by first creating one configuration containing most settings and then exporting it, importing it into the new systems, and then making adjustments.

⚠ The Import and Export functionality can *not* be used to clone a system, since that will result in severe problems with id's in the database.

The Import/Export functionality is database agnostic. For example, it is possible to export a configuration on a system using Oracle and import it on a system using PostgreSQL.

⚠ When importing a configuration, the import will reset the state of any previously registered distributed Queue Agents, including the current profile of the Queue Agent.

For more information, see “Queue Agents” on page 97.

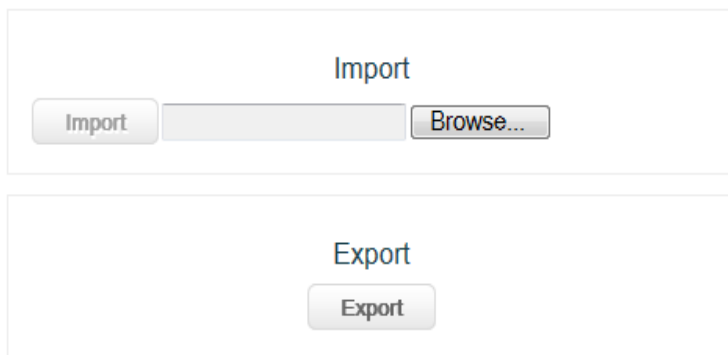
⚠ Note that only import/export from and to the same version of Orchestra is supported.

⚠ Note that only Appointments in Central are included in the export.

To export a system configuration, do the following:

⚠ Only the Central configuration is exported, i.e. no data on the Queue Agents is exported.

In the *System Administration* application, open the *Import/Export* tab:



Click *Export*.

Once the file is downloaded, you will be asked if you want to open or save the file.

The following are included in an export:

- Customer Journey Management Global configuration settings
 - Branches
 - System settings
 - General Parameters
 - Branch Hierarchy
 - AAA - i.e the three security layers in an application: *Authentication* (the act of confirming the truth of an entity, e.g a User); *Authorisation* (confirming the access rights, privileges of an entity e.g a User); *Audit* (evaluation of a security token, e.g a User or process).
 - Context Marketing Messages (including content)
 - Touch screen applications
 - Playlists (including content)
 - Appointments
- ⚠ Note that **only** Appointments in Central are included in the export.
- Unit Types and Unit Templates

- Units (fonts, voice-files and unit images)
- Wookie widgets and Orchestra Wookie Widget configuration
- Surface Applications
- Queue Agents
- Queue Agent profiles (and their connections to parameters)
- Stat job transformations, i.e. the *stat-jobs.xml* file.

ⓘ The statistics tables and data are not included in the exported zip file. However, the user Branch Group mappings and hierarchy are included.

The exported data is saved in a nested JSON structure. Binary data is stored in binary files, referenced from the JSON structure. The exported data is finally stored in a *zip* file. A link to the *zip* file is displayed on the page. Click the file link and save the file in a suitable directory.

To import a system configuration, do the following:

ⓘ When importing Users to Orchestra, using the Import/Export functionality, System administrator Users are sent to the database directly after import. However, Users with Branch privileges are not sent until the Branch is published.

By importing a system configuration, a complete Orchestra set-up can be installed in one simple operation.

Import must be performed directly after an Orchestra installation, before any configuration is done.

If any Queue Agents are connected to the system that you are performing import on, then the databases of those Queue Agents must be reset. This is done in the *Queue Agent* tab of the *System Administration* application.

For more information, see “Queue Agents” on page 97. If the databases are not reset, publishing to these Queue Agents will fail after import.

1. Click *Browse* and select the file to import.
2. Click *Import*.

Widget Administration

A Web Widget is a small application that can be inserted on web pages. Qmatic bundles a small set of Widgets with Orchestra. These are normally used on web pages for Touch Screens, Media Displays and Positional Displays.

For a description of the widgets that come with Orchestra, please see the Widgets Guide, found on Qmatic World.

It is possible for customers to build their own special purpose Widgets. For more information, see Widget Developers Guide on the Development Community in the Wiki section of Qmatic World.

Internal Widget Communication

Whenever a Surface application requests that Widgets shall be loaded onto a Surface, the Unit makes a request to Orchestra (or a distributed Queue Agent, depending on your configuration). Orchestra central (or a distributed Queue Agent) will then perform certain internal HTTP requests to itself, in order to get hold of the required data. This communication by default goes through *http://127.0.0.1:8080* (port varies, depending on your Orchestra configuration).

If this needs to be changed, due to environment constraints, the following configuration parameter can be added to the *conf/agent.conf* file:

wookie.base.url=http://192.168.1.2:8181



Change protocol, host and port, to match your configuration.

The Web Widgets are found on the *Widgets* tab, in the *System Administration* application:

Widgets

Upload Widget

Upload Choose File No file chosen Manage whitelist

Installed Widgets

Icon	Name	Description	Version	Dimensions	Mappings
	Workstation name	Shows the name and number(s) of the workstation(s) associated with the display the widget is running in. Mainly useful for Positional displays.	1.1	300x100	
	Ding	This invisible widget plays a "ding" sound when a customerCalled event is received.	2.0	48x48	
	Ticket call	Shows the ticket call popup	3.4	300x400	Generic Positional Display, Generic Media
	eventsim	Event simulator for Surface Editor	1.0	440x310	
	Positional Display Ticket call	Shows the ticket call, workstation status	3.3	300x400	
	Customer feedback for tablet	Customer feedback for tablet	1.2	400x400	
	Ticket call history	Keeps track of latest called tickets to allow customers to see what tickets have been called where	3.4	300x400	Generic Positional Display
	News Ticker	Ticker text widget, takes strings separated by ., font settings and scroll speed/direction)	2	600x160	
	Scrolling text	Scrolling/Sliding text widget, takes a string list (separated by .), font settings and scroll speed/direction)	1.0	600x160	Generic Positional Display

1-9 of 9

Icon

An icon for the widget is displayed here.

Name

Name of the widget.

Description

Description of the widget.

Version

Version number for the widget.

Dimensions

Dimensions for the widget (in pixels).

Mappings

Here you can see which Surface Types that have been mapped to the widget. To edit this, simply click on the row for the applicable widget to open the *Edit Surface type to Widget mapping* window.

For more information, see Surface Type - Widget Mapping below.

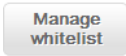
Upload Widgets

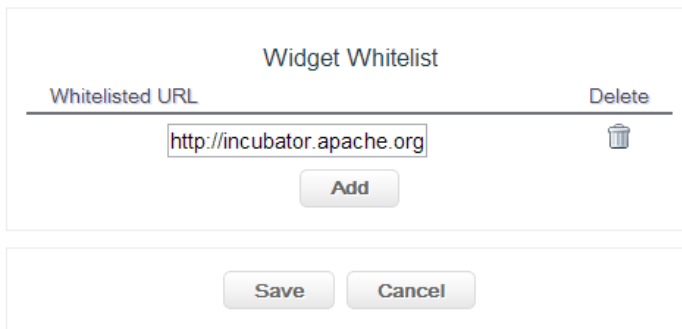
Click *Choose File* and browse for the widget (.wgt) file on your computer. Then, click *Upload* to upload the widget to the list of installed widgets. Do not forget to map it to the applicable Surface Types, so that it can be used in the Surface Editor application.

For more information, see Surface Type - Widget Mapping below.

Manage Widget Whitelist

The Whitelist is a list of URL:s that the Widgets are allowed to access.

By clicking on the *Manage whitelist* button,  , in the *Widget Administration* window, you will open the *Widget Whitelist* window where you can add or delete URL:s to/from the Whitelist. Make sure that you *Save* the list when done!



The screenshot shows a window titled "Widget Whitelist". It contains a table with two columns: "Whitelisted URL" and "Delete". The "Whitelisted URL" column has a text input field containing "http://incubator.apache.org". The "Delete" column has a trash icon. Below the table is an "Add" button. At the bottom of the window are "Save" and "Cancel" buttons.

Whitelisted URL	Delete
http://incubator.apache.org	

Surface Type - Widget Mapping

To make it possible to insert a Widget on a web page for a device, it must be mapped to the Surface Type used by the device.

This is done by clicking on the row of the applicable widget in the *Widget administration* window. The mapping form is shown:

Mapped	Surface Type
☐	TP-3115 Touch Printer/Intro 8
☐	Vision 15" Touch Printer
☐	Vision 17" Touch Printer
☐	Media Display HD 720
☐	Media Display Full HD 1080
☐	Positional Display HD 720
☐	Media Display HD 720 Portrait
☐	Media Display Full HD 1080 Portrait
☐	Positional Display HD 720 Portrait
☐	Intro 17" Touch Printer
☐	Positional Display 600 x 1024
☐	Positional Display 1024 x 600
☐	Positional Display 768 x 1024
☐	Positional Display 1024 x 768
☐	Positional Display 1536 x 2048



 1-15 of 19
 


Check the check box(es) in the *Mapped* column, to enable the Widget for that Surface Type. Click *Save* when you are done.

ⓘ There may be several pages of available Surface Types to go through. To view further pages, click on the arrow buttons at the bottom of the page,



 1-15 of 19
 


Be careful to only map Widgets that work on the intended Surface Type.

For more information, see the Widgets Guide, found on Qmatic World.

Queue Agents

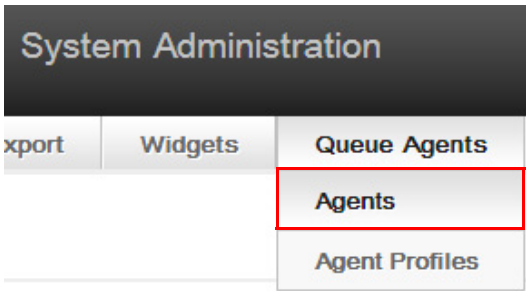
Overview

A distributed Queue Agent registers with a name and a unique identifier. The name is typically a hostname and must conform to the hostname standard. The identifier can be any string, but is by default set to the MAC-address of the network interface of the Queue Agent.

When a Queue Agent registers for the first time, the agent profile currently deployed is *unknown*. It is only when an agent profile is deployed from central Orchestra, that the profile is known.

There are, however, some cases, described below, where the agent profile can change directly on the Queue Agent and where the deployed profile should be set to *unknown* again.

In the *Queue Agents* tab there are two sub tabs: *Agents* and *Agent Profiles*:



In the *Agents* sub tab, you can see the following information about available Queue Agents.

Queue Agents

Start	Stop	Reset Database	Reset CFM Engine	Reset CFM Media	Prepare Profile	Synchronize	Upgrade
Name	Id	Host	Version	Profile	Status	Connection	Latest activity
Central CFM Engine	central	127.0.0.1	2.3.0.287		Started		CFM Engine connected
☐ ubsvr64	00:0C:29:D2:E7:35		2.3.0.287		Unknown	Disconnected	CFM Engine disconnected
☐ vm-w2k8r2-174	00:0C:29:82:93:92	192.168.8.70	2.3.0.287	2.3.0.287_SQL	Started	Connected	Deployment completed

Name

Name of the Queue Agent.

Id

Id of the Queue Agent.

Host

The IP address of the Queue Agent.

Version

Version number.

Profile

Name of Queue Agent Profile.

Status

Status of the Queue Agent. Possible values are: *Started*, *Starting*, *Stopping*, *Unknown*, *Stopped*, *Resetting Queue Agent*, *Resetting Queue Agent Media* and *Resetting Database*.

Connection

Status of Connection. Possible values are: *Disconnected* and *Connected*.

Latest activity

Shows the latest activity of the Queue Agent.

It is possible to sort each column by clicking on the column header. If you need to find a certain Queue Agent, you can use *Ctrl + F*.



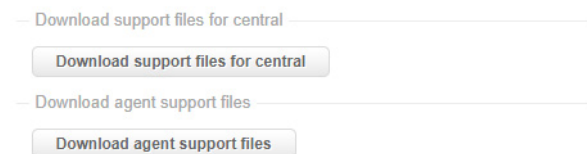
If you decide to sort the list on *Connection* status and have many Queue Agents that change their connection status often, it may take up to 10 seconds before the sorting is correct.

Import a configuration

See “Import and Export” on page 90.

Retrieve log files for all queue agents

To retrieve and download support files for all queue agents, go to Queue Agents > Agents > Central Queue Agent.



- Click *Download support files for central* or *Download agent support files*. For agent support files, they will first be retrieved from the agents before they can be downloaded. After around 30 seconds, refresh the page to see the available zip files. The click on the file names to download them.

- Note that log files can also be downloaded from each queue agent, see “Handling a Queue Agent” on page 100.

Replacing a Branch Hub or Hub

When replacing or switching out a Branch Hub, or Hub, the MAC-address of the Branch Hub/Hub is generally also changed, which means that the Queue Agent on the Branch Hub/Hub will re-register with central Orchestra. In most cases, one does however keep the name of the Queue Agent, thus two Queue Agents with the same name will appear in the list of Queue Agents, in the *System Administration* application.

It is safe to remove a Queue Agent from the list using the *Delete* icon, since any active Queue Agents will re-register, upon connect. However, by de-registering a Queue Agent, one will also clear out the information about the deployed Agent Profile.

Upgrading the Branch Hub/Hub firmware

When upgrading the entire Branch Hub/Hub firmware, the Agent Profile, e.g. the binaries, libraries, configuration and resources of the Queue Agent, is also replaced. In this case, an undetectable error may occur. The Queue Agent will register with the same id and name, but central Orchestra will not be able to detect the changes on the Queue Agent, and it will falsely continue to state the currently applied Agent Profile.

To avoid this, it is important to remove the upgraded Branch Hub/Hub Queue Agent from the list of Queue Agents, thus clearing out the stated deployed Agent Profile.

Handling a Queue Agent

To see more detailed information about a Queue Agent, click on it in the list and the following information window will be displayed:

Queue Agent

The screenshot shows the 'Queue Agent' information window. At the top, there is a toolbar with buttons: Start, Stop, Reset Database, Reset Queue Agent, and Reset Queue Agent Media. The main content is divided into several sections:

- Information:** A table showing agent details.

Id	44:39:C4:55:6F:A5
Name	GOT-JOHSTA-003
Host	127.0.0.1
Version	4.0.0.126
Status	Started
- Connections:** A table showing connection status.

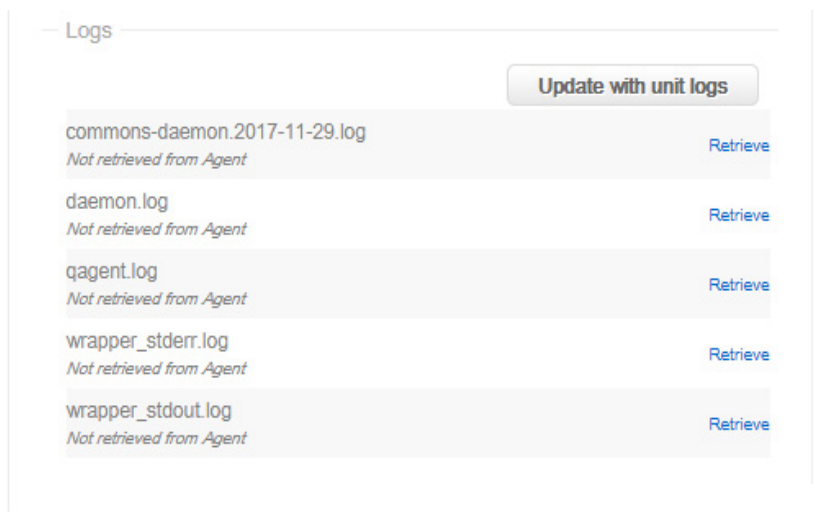
Daemon	Connected	2017-11-29 15:28:57
Queue Agent	Connected	2017-11-29 15:28:51
- Activity History:** A list of recent events.
 - 2017-11-29 15:28:57 Daemon connected
 - 2017-11-29 15:28:51 Queue Agent connected
- Logs:** A list of log files with a 'Retrieve' button for each.

commons-daemon.2017-11-29.log	Retrieve
daemon.log	Retrieve
qagent.log	Retrieve
wrapper_stderr.log	Retrieve
wrapper_stdout.log	Retrieve
- Security:** A section for security settings.
- Upgrade:** A section for upgrade handling.
- Current profile:** No profile set.
- Prepared profile:** No profile prepared.
- Available profiles:** A list of available profiles with a 'Prepare Profile' button for each.

4.0.0.126	Prepare Profile
4.0.0.126-branchhub	Prepare Profile
4.0.0.126-hub	Prepare Profile

In this page you can find general *Information* about the Queue Agent, information about *Connections*, an *Activity History*, *Logs*, *Security*, as well as *Upgrade* handling.

You can retrieve and download (as a zip file) a number of log files in the *Logs* area, by clicking on the *Retrieve/Download* links:



Here, you can also update the Queue Agent with unit logs.

You can also perform the following actions:

- Start the Queue Agent.
- Stop the Queue Agent.
- Reset the Database.
- Reset the Queue Agent.
- Reset the Queue Agent Media.
- Prepare profile. Click the button next to the wanted Profile in the *Available Profiles* area.
- Force synchronization.
- Upgrade

When performing an action, you can follow the activity in the Activity History section.

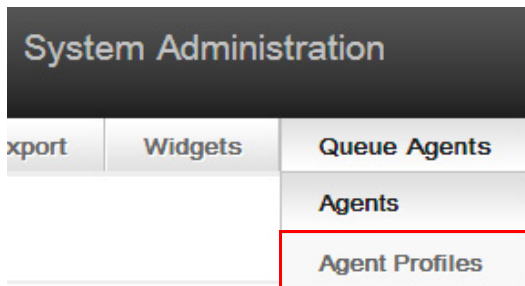
Handling Multiple Queue Agents

You can also perform an action on multiple Queue Agents at once. In the list of queue agents, select the wanted Queue Agents by marking the check boxes. Click one of the highlighted actions (Start, Stop etc) at the top of the window, depending on which action you want to perform.

Agent Profiles

Overview

In the *Queue Agents* tab, you can also open the *Agent Profiles* sub tab:



Agent Profiles

Create Profile		
Name	▲ In use	
4.0.0.126	Yes	
4.0.0.126-branchhub	No	
4.0.0.126-hub	No	

Here, you can find a list of all agent profiles that are present in the system. The list can be sorted, by clicking on the column headers.

You can see whether or not they are used and you can also delete profiles that are not used. Also, you can create new agent profiles.

⚠ After deleting an agent profile, please make sure that you refresh the browser.

At install and upgrade, there are three pre-installed agent profiles in the system, in the `<installation_directory>/media/agentProfiles` folder:

- A default agent profile for the current Orchestra version. The name of this agent profile is the same as the version name in the footer.
- A default Hub agent profile for the current Orchestra version. The name of this agent profile is the same as the version name in the footer followed by `-hub`.
- A default Hub first generation agent profile for the current Orchestra version. The name of this agent profile is the same as the version name in the footer followed by `-hub_first_gen`.

These default agent profiles fetch their settings for *host ip*, *protocol*, *http port* and *websocket port* from the central configuration.

Agent Profile

An agent profile consists of configuration parameters (stored in database) and a zipped file. The agent profile zip file typically contains Counter, Reception, Concierge, and other applications

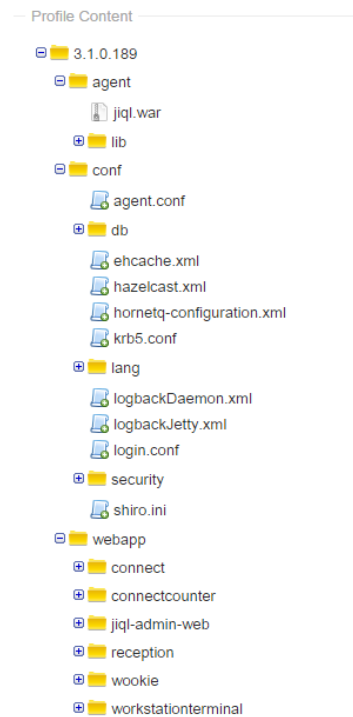
For more information, see “Creating an agent profile zip file” on page 106.

When uploading, the expected library files should be included in the *lib* directory of the agent profile that was created to be run with the latest version of the Queue Agent. The contents of the directory is validated and If something is missing in the file, this is presented in the *Upload* window.

It is possible to upload a new file for an agent profile. In that case, the existing uploaded file will be replaced by the new file.

⚠ We recommend that you think twice before uploading a new file for an already existing agent profile, since it may be hard to keep track of the contents of the different file versions. In many cases it may be a better idea to create a completely new agent profile instead of uploading a new file for an existing agent profile.

The file structure of the uploaded file is displayed, once the agent profile has been successfully created:



Time Zone Considerations - Configuring your Branch Hub / Hub

In order for the time stamps to be correct in your log files (*qagent.log* and *daemon.log*), you need to make sure that you have configured the time zone correctly in your Branch Hub / Hub in the agent profile that is then sent out to the Branch Hub / Hub.

This is done as described below.

Offset

The offset is controlled by the logback configuration files. For the Queue Agent, the file is `<installation_directory>/media/agentProfiles/<agent_profile_name>/conf/logbackJetty.xml`.

By default, the log file configuration is set to UTC time.

The section for the file appender can be changed so the correct time zone is used for the printouts (this should match what has been configured for the Branch Hub / Hub).

To set the time zone *Europe/Stockholm*, for example, in the log file for the Branch Hub, open the file *logbackJetty.xml* in the correct agent profile and locate the *startupDailyRolloverAppender*. Then change the line:

```
<pattern>%d
{yyyy-MM-dd HH:mm:ss.SSS}
[%thread] %-5level %logger
{36} - %msg%n </pattern>
```

to:

```
<pattern>%d{yyyy-MM-dd HH:mm:ss.SSS,Europe/Stockholm} [%thread] %-
5level %logger{36}
- %msg%n</pattern>
```

Example of the section, after the change:

```
<!-- Daily rollover appender that also appends timestamp and rolls
over on startup -->
<appender name="startupDailyRolloverAppender"
class="com.qmatic.qp.logging.QPRollingFileAppender">
<file>${
AGENT_HOME}/logs/qagent.log</file>
<rollingPolicy
class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
<fileNamePattern>${AGENT_HOME}
/logs/qagent-%d
{yyyy-MM-dd}
_%d
```

```
{HHmmss}
.%i.log</fileNamePattern>
<timeBasedFileNamingAndTriggeringPolicy
  class="com.qmatic.qp.logging.StartupTimeAndSizeBasedTriggeringPolicy">
<maxFileSize>10MB</maxFileSize>
<maxBackupIndex>10</maxBackupIndex>
</timeBasedFileNamingAndTriggeringPolicy>
</rollingPolicy>
<encoder>
<pattern>%d
{yyyy-MM-dd HH:mm:ss.SSS,Europe/Stockholm}
[%thread] %-5level %logger
{36}
- %msg%n</pattern>
</encoder>
<filter class="ch.qos.logback.classic.filter.ThresholdFilter">
<!-- In order to DEBUG change this filter level, along with required
  components below -->
<level>INFO</level>
</filter>
</appender>
```

After this change, update the Queue Agent profile on the Branch Hub / Hub.

Without Offset

To get time stamps *without* offset to UTC time in *daemon.log*, modify the file `<installation_directory>/media/agentProfiles/<agent_profile_name>/conf/logbackDaemon.xml` in the same way as described for *logbackJetty.xml*, above.

Agent Profile zip file structure

Below is the agent profile zip file structure, where you can see mandatory and optional files. Those that are not marked are mandatory:

```
/agent/<jiql_war>
/agent/preDeploy.groovy [Optional]
/agent/postDeploy.groovy [Optional]

/agent/lib/<required_libs> [Not validated during upload but required
  to run jiql.war]

/app/<application>/<application_package> [Optional]

/webapp/<web_application>/<war_file> [Optional]
/webapp/<web_application>/preDeploy.groovy [Optional]
/webapp/<web_application>/postDeploy.groovy [Optional]

/conf/<required_conf_content>
/conf/db [Optional]
```

```
/conf/css/ [Optional]
/conf/images/ [Optional]
/conf/lang/<required_lang_content>
/conf/lib [E.g. java segmentation rules] [Optional]
/conf/security/<required_security_content>
/conf/scripts/ [Optional]
```

Folder content lists

```
required_conf_content = agent.conf, ehcache.xml, hazelcast.xml (only
    for 5.4 Queue Agents and below), hornetq-configuration.xml (new in
    6.1), krb5.conf, logbackDaemon.xml, logbackJetty.xml, login.conf,
    shiro.ini
required_security_content = keystore.jks
required_lang_content = commonMessages.properties,
    graphicalDisplayMessages.properties
required_libs = This folder needs to contain the libs (jar files etc)
    required by the version of jiql.war included in the profile.
```

❗ Files placed in the folder */app/<application>* and any subfolders are automatically synchronized with the corresponding folder on a distributed Queue Agent. It can be used to automatically distribute files, e.g. third party applications and binaries to a distributed Queue Agent.

Creating an agent profile zip file

1. Copy the template profile from the Central server *<central install dir>/media/agentProfiles/<version>*.

❗ Each new Orchestra release comes with a new default profile that contains correct versions of all mandatory files.

2. Mandatory: */agent/jiql.war*

❗ If the default profile is used as template, the *jiql.war* SHOULD NOT be changed.

❗ If updating the zip file for an existing profile (in an upgraded system), the *jiql.war* MUST be changed.

3. Optional : */agent/preDeploy.groovy/*

If required, add a *preDeploy.groovy* script.

4. Optional : */agent/postDeploy.groovy/*

If required, add a *postDeploy.groovy* script.

5. Mandatory: */agent/lib*

❗ If the default profile is used as template, the content in the *lib* directory SHOULD NOT be changed.

❗ If updating the zip file for an existing profile (in an upgraded system), the content in the *lib* directory MUST be changed.

6. Optional: `/webapp/<web application>/<war file>`, `/webapp/<web application>/preDeploy.groovy`, `/webapp/<web application>/postDeploy.groovy`

❗ The default profile contains the following default web applications: *Appointment Reception*, *Cloud Bridge*, *Concierge*, *Jiqi Admin Web*, *Reception*, *Wookie*, *Counter*.

7. Mandatory: `/conf/<required_conf_content>`

For information about what content is required, please refer to the list above that defines `<required_conf_content>`

Make necessary changes to the *agent.conf* file, e.g. set correct ports, ip addresses and ensure that settings for correct database type are made.

8. Optional: `/conf/db`

9. Optional: `/conf/css`

10. Optional: `/conf/images`

11. Mandatory: `/conf/lang/<required_lang_content>`

If the default profile is used as template, the content in the `/conf/lang` directory contains the latest language property files.

If additional languages are used on branches on a Queue Agent, those should be added here.

If updating the zip file for an existing profile (in an upgraded system), it is highly recommended to go through the existing language property files, since new parameters might have been added.

12. Optional: `/conf/lib` [E.g. java segmentation rules]

13. Mandatory: `/conf/security/<required_security_content>`

❗ If the default profile is used as template the `<required_security_content>` SHOULD NOT be changed.

❗ If updating the zip file for an existing profile (in an upgraded system), the `<required_security_content>` MUST be changed.

14. Optional: `/conf/scripts`

16. When all updates have been made, the zip file can be created. The zip is created by zipping the following directories:

- *agent*
- *conf*
- *webapp*

It is recommended to give the zip file a name that describes the content.

Queue Agent Pre Deployment Script

A typical use case for the pre deployment script is to reset the database.

Queue Agent Post Deployment Script

A typical use case for the post deployment script is to include some tests to ensure that the Queue Agent is working properly. In case of failure, a roll back of the agent profile will be done.

Creating an agent profile in Orchestra

To create a Queue Agent profile:

1. Click the *Create Profile* button.

Agent Profile

Save

Discard changes

General

Name

Software Upgrade

File

Choose File No file chosen

Folder Name

Profile Content

Override Global Parameter values

Appointment Management Settings

Delete appointments where endtime passed by (days)

1

Cron trigger for synchronizing appointments

0 0 3 * * *

Cron trigger for deleting old appointments

0 0 2 * * *

Block early appointments (minutes)

Recycle Settings

Recycle Max no Recycles

3

Recycle Insert Delay

60

Fault Manager Settings

Upload Level

WARN

Resend Interval (minutes)

60

Upload Schedule

[[{"day": "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun", "time": "0 0:00-23:59"}]]

Fault Manager enabled

false

Queue Agent Settings

Central HTTP Port

8080

Central HTTP Protocol

http

User Settings

Min Login Code

1000

Max Login Code

9999

UserName Validation Pattern

[a-z][a-z,0-9]\$

Password Validation Pattern

^(?=.*[0-9])(?=.*[a-z])(?=.*[A-Z]).*\$

Event Manager Settings

Upload Events to Central

true

Sorting Policy Settings

Multi service visit sort policy

SORTED

Queue Agent Media Settings

Cron trigger for deleting old media

0 0 23 * * *

Allowed Download Interval

00:00-23:59

Download media in advance (days)

5

Cron trigger for downloading media

0 0 3 * * *

Statistics Settings

2. Enter a unique name for the agent profile.

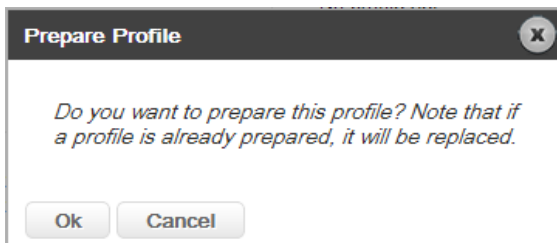
3. Browse for and upload the required zip file and enter a unique folder name,
4. For each agent profile, it is possible to override the *Global Parameter values* that have been configured in the *Parameters* tab. This is why all the parameters that can be overridden are also listed in the *Queue Agent Profile* window, as can be seen above.
5. Save the agent profile.

Assigning an agent profile

1. In the list of Queue Agents on the *Agents* sub tab, select the Queue Agent that you want to assign the agent profile to.
2. In the Available profiles section, click the *Prepare Profile* button next to the name of the wanted agent profile:



3. Click *Ok* in the confirmation window:



4. Force start the synchronization of the selected Queue Agent, see "Synchronizing the Queue Agent profile" on page 111.

Synchronizing the Queue Agent profile

The performance bottleneck when it comes to remote upgrade of distributed Queue Agents can be the synchronization of the agent profiles. It is highly depending on the network bandwidth between the Central Orchestra installation and the Queue Agents. Once the synchronization is done, the upgrade is taken care of by each Hub or the distributed Queue Agent PC. This is triggered by a command in Central.

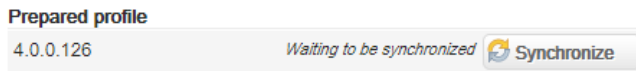
Synchronization can take very long if the bandwidth is low. The upgrade can however not be started until that step is done and the upgrade itself will not take long but should be done outside business hours, so that it will not disturb your daily routines too much.

Synchronization can be started by three different events:

- At reconnect to the central system
- At a scheduled time, triggered by a spring cron job
- When you force start synchronization in the GUI.

The latter is described below:

1. In the list of Queue Agents on the *Agents* sub tab, select the Queue Agent that you want to synchronize, by marking the check box to the left in the list. Alternatively, open up the detailed information page for the Queue Agent.
2. After having prepared the profile, as described above, click the *Synchronize* button, either in the Queue Agent list view, or in the detailed information page, as in the picture below:



During synchronization, you will be able to see the status (*Synchronizing*, *Synchronized* and *Synchronization interrupted*). The latter simply means that an ongoing synchronization for some reason was interrupted, for example by stopping the Queue Agent).

⚠ The file `/conf/security/keystore.jks` will not be synchronized. This is because you are required to have a secure communication for updating the keystore. If you wish to update the keystore, follow the instructions in “Secure Communication for a Queue Agent” on page 237.

Parameters

In the Settings> *Parameters* tab of the *System Administration* application, you can change the following types of parameters. For each parameter, you see the parameter name and its value and it is also possible to restore the default value, by clicking the Restore Defaults, , button.

 Please note that all of the parameters are not available and used for the Queue Agent profiles.

- General Parameters
 - System
 - Connectors
 - Weekend Settings
 - Scheduled publishing
- Central Access Parameters
 - HTTP Settings
- HTTPS Parameters
 - Certificate and key store Settings
 - HTTPS server settings
 - Central WebSocket Server Settings
- Application Parameters
 - Appointment Management Settings
 - Recycle Settings
 - Browser Settings
 - Mobile API (Central)
 - Cloud settings
 - Customer
 - User Settings
 - Event Manager Settings
 - Sorting Policy Settings
 - Queue Agent Media Settings
 - Statistics Settings
 - SSO Settings
 - LDAP Settings
 - SAML v2 Web Single Sign On Settings
 - Orchestrated Logout Settings

General Parameters

Parameter	Description	Default value
System	General parameter, regarding the whole system.	
System Locale	Language code for language used in the system. This setting only applies to users created manually in Orchestra (DB users) that have been assigned the default language. For more information, see "Localisation" on page 155.	en
Time convention	From the drop-down list, select <i>24 hour</i> or <i>AM/PM</i> , depending on which time format you want to use throughout the system.	24 hour
Date convention	Select a date format depending on which date format you want to use throughout the system. The following formats are allowed: • YY-MM-DD • YY/MM/DD • DD.MM.YYY You can change the position of YY, MM and DD.	YY-MM-DD

Parameter	Description	Default value
Connectors	General parameters regarding connectors.	
Date Time Format	Date and time format.	HH:mm
Customer External JNDI Name	Customer External JNDI Name	java:global/customCustomerDb/customerdintegration/CustomCentralManagerBean
Weekend Settings	Parameters regarding scheduling of Context Marketing Messages.	
Weekend days	Select which days of the week that should be regarded as weekend days. This is used when scheduling Context Marketing Messages. For more information, see the Administrator's Guide.	Saturday Sunday
Scheduled publishing	Parameters to control scheduled publishing	
Retain publish items (hours)	Controls how many hours a schedule should be kept after it is completed. A schedule is marked complete only if all branches in the schedule have completed the publish.	24
Allow late publish (minutes)	If the system is slow or even shut down during the scheduled time, this setting controls how late to allow the schedule to fire when the scheduling mechanism becomes active again.	15
Cron trigger to delete old schedules	When to run the clean-up job to delete old, completed, schedules.	0 0 */2 * * * (every 2nd hour)

Central Access Parameters

Parameter	Description	Default value
HTTP Settings		
Central HTTP Port	Enter the port number that should be used to access the Central Orchestra Server. This is usually the port number that Wildfly uses. In case of a load balancer in front of Orchestra, enter the port number of the load balancer.	8080 for http and 8443 for https.
Central HTTP Protocol	Select the HTTP protocol (http or https) that should be used to access the Central Orchestra Server. This is usually the protocol that Wildfly uses. In case of a load balancer in front of Orchestra, enter the protocol of the load balancer.	http or https

HTTPS Parameters

Parameter	Description	Default value
Certificate and key store Settings		
(Re)generate certificate	Enabling this check box and saving the parameter list will cause the certificate to be generated in the key store. ⓘ If a certificate with the same alias already exists, it will be overwritten, so be careful!	
KeyStore alias	The alias of the certificate key entry, in key store.	orchestra
Distinguished name	The distinguished name of the certificate. The first (CN) section is the host name of the server and the subsequent sections describe the organization.	CN=local-host,OU=orgUnit,O=org,L=city,S=state,C=country-Code

Parameter	Description	Default value
Subject alternate name	This field needs to be set to both IP address and host name of the server if both are going to be used for HTTPS communication. Separate each entry with a comma. Example: <i>myhost.com, 10.0.10.0</i> .	localhost

Parameter	Description	Default value
HTTPS server settings	For more information, see “Secure Communication” on page 165.	
HTTPS enabled	Controls whether HTTPS should be enabled or not in Wildfly.  An Orchestra restart is recommended after enabling HTTPS.	Disabled
KeyStore alias	Determines which key entry in the key store to use as a server certificate.  HTTPS must be disabled before selecting a new KeyStore alias.	
HTTPS port	Decides which port to use for HTTPS communication.	8443
Central WebSocket Server Settings	Settings connected to the Central WebSocket server, heartbeat, etc.	
WebSocket enabled	If this check box is checked, Web socket communication over unencrypted channels is allowed. WARNING! Disabling this will cause any distributed Queue Agent connected over unencrypted WebSocket to stop functioning!	Enabled
WebSocket port	The port to use for unencrypted WebSocket communication. WARNING! Changing this will cause any distributed Queue Agent connected over unencrypted WebSockets to stop functioning, until they are re-configured!	8787

Parameter	Description	Default value
Secure WebSocket enabled	Enabling this parameter will cause the WebSocket server on Central to support WebSocket secure. Make sure that the certificate setting is properly configured. WARNING! Disabling this setting will cause any Queue Agents that are currently connected using secure WebSocket to stop functioning!	Disabled
Secure WebSocket port	The port to use for secure WebSocket communication. WARNING! Changing this parameter will cause any Queue Agents that are currently connected using secure WebSocket to stop functioning, until they are re-configured!	9150
Netty worker thread pool size	The number of worker threads available to handle web socket traffic. Minimum 5, maximum 1000.	100
Init commands thread pool size	The number of threads available to handle init commands from the Queue Agents. Minimum 1, maximum 500.	20
Non-init commands thread pool size	The number of threads available to handle all non-init commands from the Queue Agents. Minimum 1, maximum 500.	20
Event thread pool size	The number of threads available to handle events from the Queue Agents. Minimum 1, maximum 500.	20
Command pool size	The number of threads tasked with notification of results to commands. Minimum 5, maximum 1000.	20
Command timeout (milliseconds)	The time in milliseconds to wait for a response from a command sent to a Queue Agent. Minimum 500, maximum 180000.	60000

Parameter	Description	Default value
Client connection timeout (milliseconds)	The time, in milliseconds, to wait before a connection to a Queue Agent is considered lost. Minimum 1000, maximum 600000.	120000
Heartbeat interval (milliseconds)	The maximum time, in milliseconds, before a heartbeat message is sent to a Queue Agent if nothing else is sent. Minimum 5000, maximum 120000.	30000
Enable IP-address filtering	This check box determines whether the web socket server should only allow connections from certain IP-addresses. If enabled, only addresses specified in the definition clause, below, will be allowed to connect.	Disabled
Allowed IP-addresses	A comma-separated list of allowed IP-addresses. Wildcards are allowed. Localhost (127.0.0.1) is always allowed. Example: <i>192.168.2.2*, 192.168.1.100</i> will allow 192.168.2.2, a range from 192.168.2.20 to 192.168.2.29, a range from 192.168.2.200 to 192.168.2.255, 192.168.1.100 as well as 127.0.0.1.	
Send extended heartbeat message	If this check box is checked, time-stamps are included in the heartbeat message. This is combined with trace logging both centrally and on selected Queue Agent(s) that have the <i>agent.conf</i> property <i>central.websocket.heartbeat.extended</i> set to <i>true</i> .	Disabled

Parameter	Description	Default value
Delay start of web socket server (seconds)	Increasing this value will delay the start of the web socket server and prevent any Queue Agent connections, until the web socket server is started. This can, for example, be used for a central system with many distributed Queue Agents.	0
Distributed publish delay (milliseconds)	This parameter controls how publishing branches to distributed agents behave. The system will introduce a delay between each publish event sent to a distributed agent. This is done in order to not overwhelm the central server when publishing many branches on distributed agents, or when many distributed agents reconnect. This does not affect branches on the central agent. Setting this value to 0 will effectively disable this throttling, but is not recommended.	250

Application Parameters

Parameter	Description	Default value
Appointment Management Settings	Parameters regarding Appointment management.	
Delete appointments where endtime passed by (days)	Applicable to Central. Number of days that should pass, since the end time of an appointment, before that appointment is deleted.	1
Delete appointments at (hh:mm)	Applicable to Central. Time when appointments are deleted.	02:00
Cron trigger for synchronizing appointments	Cron job trigger indicating when appointments should be synchronized.	0 0 0 * * *

Parameter	Description	Default value
Cron trigger for deleting old appointments	Cron job trigger indicating when appointments should be deleted.	0 0 2 * * *
Appointment Status callbacks enabled	This parameter is enabled in order to get updates for appointments. Example: CREATED: 20 RESCHEDULED: 21 CALLED: 40 ARRIVED: 30 CANCELLED: 53 COMPLETE: 50 NO_SHOW: 51 ENDED_BY_RESET: 52	Enabled
Appointment Status update callback URLs (comma-separated).		/calendar-backend/public/api/v1/appointments/callback
Appointment life cycle events enabled	Enable or disable sending of events when appointments created, updated, or deleted.	Disabled
Block early appointments (minutes)	Specify the number of minutes before the appointment start time, that it is possible to call an appointment visit. Note that it is possible to have a different number for different agent profiles, here.	
Recycle Settings	Parameters regarding recycling of tickets.	
Recycle Max no Recycles	The maximum number of times a visit can be recycled.	3
Recycle Delay	Number of seconds after which a recycled visit can be called again.	60

Parameter	Description	Default value
Browser Settings		
Allow Browser Chrome Frame	When this check box is checked, Chrome Frame is enabled. Chrome Frame is designed to expand Internet Explorer's functionality, by adding support for open web technologies and Google Chrome's fast rendering engine.	Disabled
HttpOnly cookie flag enabled	Cookies with HTTP only flag set to true indicates that the cookie shall only be accessed from server side and not applications running in the browser. ⚠ Note that the <i>Use the HttpOnly cookie flag</i> setting should also be enabled in the <i>Calendar Admin</i> application, under <i>System Settings</i> if applicable. For more information, see the Administrator's Guide. ⚠ An Orchestra restart is needed after enabling the <i>HttpOnly cookie flag</i>.	Disabled
Secure cookie flag enabled	This enables the secure attribute for the cookie used by surface applications. Only set this to true if all surface applications connects using https. For more information, see "Enable cookie secure flag" on page 202	Disabled
Mobile API (Central)		
Parameters regarding username and password for the Mobile API.		
Username	Username used to access the Mobile API.	mobile
Password	Password used to access the Mobile API.	

Parameter	Description	Default value
Mobile Ticket Base URL	Base URL used when generating URLs in e.g. barcodes, SMS messages and other mobile ticket implementations.	http://MobileTicket/MyVisit/CurrentStatus
Cloud settings	Parameters for connecting to the cloud	
System Id	System id, can be retrieved from the cloud portal	
System Secret	Secret for fetching cloud bridge certificate, can be retrieved from cloud portal	
Certificate Url	The url to fetch cloud bridge certificate from, as shown in cloud portal	
Customer	Parameters regarding handling of Customer data, mainly due to GDPR regulations.	
Include customers in export	If this check box is checked, customers will be included in export/import.	Disabled
Use retention policy for customer object	If this check box is checked, Customer objects will automatically be updated with a new interaction timestamp and deletion timestamp when Appointments or Visits are created or updated for that Customer.	Enabled
Delete customers, based on retention policy, at (hh:mm)	Customers will automatically be deleted, based on their retention policy, at this time (hh:mm).  Delete job will only run if <i>Use retention policy for customer object</i> is enabled.	01:00
User Settings	Parameters regarding User settings.	

Parameter	Description	Default value
Number of Login attempts before deactivating user	Number of failed Login attempts before deactivating user. Counter will be reset if successful login occurs before limit is reached.  Counter will also reset when user is updated.	
Min Login Code	Lowest valid login code number.	1000
Max Login Code	Highest valid login code number. Default is set to 9999.	9999
UserName validation Pattern	Regular expression used for user name validation.	^[a-z]+[a-z,0-9]{3,}\$
Password Validation Pattern	Regular expression used for password validation.	^(?=.*{8,})(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).*\$
Enable automatic deactivation of users	Check this check box to enable automatic deactivation of users that have not logged in for a specified period of time.	Enabled
Deactivate inactive users at (hh:mm)	Users will automatically be deactivated, based on when they were last logged in, at this time (hh:mm).	23:30
Deactivate users not logged in for given number of days	Number of days since last log in, after which users will automatically be deactivated.	180
Password expiration	Enable/disable password expiration	Disabled
Password expiration days	Set the number of days before a password expires.	90
Require new password at first login	Require new users to change their password at first login or after password has been changed by an administrator.	Disabled
Event Manager Settings	Parameter regarding events.	

Parameter	Description	Default value
Upload Standard Events to Central	Whether or not standard events should be uploaded to Central (yes/no).	Events are uploaded.
Publish Custom Events on Central Topic	Whether or not custom events should be published on Central topic.	Events are published.
Number of days to keep saved generic events for offline agents	When an integration attempts to send generic events to a distributed agent through the central REST connectors (/qsystem/rest/...) the call will fail if the agent is offline. In this case the event will be queued up for later execution. When the agent reconnects (or the branch is published) the system will attempt to send the event. This setting limits how many days a saved event is kept for execution.	30

Parameter	Description	Default value
Sorting Policy Settings	Settings regarding sorting policy for visits.	
Multi service visit sort policy	This setting affects how the visit is transferred to the Queue of the next Service in a multi-service Visit. From the drop-down list, select the wanted sorting policy; SORTED, FIRST, or LAST.	SORTED
Queue Agent Media Settings	Parameters regarding handling of media on Queue Agents.	
Cron trigger for deleting old media	Cron job trigger indicating when old media will be deleted.	0 0 23 * * *
Allowed Download Interval	Time period when download is allowed.	00:00-23:59
Download media in advance (days)	Number of days in advance that media is downloaded.	5
Cron trigger for downloading media.	Cron job trigger indicating when media will be downloaded.	0 0 3 * * *
Statistics Settings	Parameters regarding the handling of statistics.	
Enable Stat Messages	Whether or not sending of Stat messages should be enabled.  Changing this value requires a restart of both central and all Queue Agents.	Enabled.

Parameter	Description	Default value
Stat Server Address	IP v4 address to the stat resource server. When upgrading, the address entered here is used. Should it fail, the value from stand-alone.stat.server.ip from install.properties is used. ⓘ Note that when creating a Queue Agent profile, this parameter needs to be changed to the IP address where Stat.war is located.	http://127.0.0.1
Statistics Settings for Agents Version 6.0 or Earlier	If you have agent 6.0 or earlier, the parameters below can be configured in addition to those above.	
Stat Server Port	Port number where the stat resource is configured. ⓘ This is only used for old agents. For newer agents, this parameter is set in install.properties.	8080
Stat Server Resource	Application name for the stat resource.	/stat/message/
Queue Agent Resend Interval (minutes)	Resend interval (minutes).	10
Queue Agent Upload schedule.	JSON definition of allowed time slots for stat sending, leave empty to always allow stat upload. Example for upload during the night: <pre>[{"days":["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"], "time": "00:00-20:00-04:00"}]</pre>	<pre>[{"days":["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"], "time": "00:00-23:59"}]</pre>
SSO Settings	For more information about the SSO Settings, see “SSO setup” on page 205.	
Enabled	Mark this check box to enable SSO. By default it is not enabled.	Not enabled

Parameter	Description	Default value
Allow basic authentication		Not allowed
Allow localhost		Allowed
Allow unsecure basic		Not allowed
Client module		spnego-client
Pre-authentication user name	User name of pre-authentication user.	preauthuser
Pre-authentication password	Password of pre-authentication user.	
Login server module		spnego-server
Prompt ntlm		Not enabled.
Allow delegation		Allowed
Logger level		1

Parameter	Description	Default value
LDAP Settings	LDAP configuration in Orchestra consists of two tasks: • Server configuration • System parameters - found in this table. • Certificate handling - see “LDAP Certificate Handling” on page 135. • LDAP/AD Group Mappings, e.g how LDAP objects, groups are mapped to Orchestra entities (Roles, Branches/Branch Groups). For more information, see the Administrator’s Guide. Each Orchestra attribute has a corresponding LDAP field attribute. For more LDAP information, see “LDAP Hosts/Urls” on page 134. ⓘ The user <i>superadmin</i> will always log in locally. It is also possible to create users that are not authenticated towards the Active Directory. For an instruction about setup of Open LDAP, please refer to “Appendix D - Open LDAP Setup” on page 241.	
Enabled	When the check box is checked, all users are authenticated towards the configured LDAP server, that is the Active Directory. ⓘ This parameter needs to be set to <i>true</i>, in order to be able to perform LDAP mappings in the <i>User Management</i> application.	Not enabled.
Validate settings	Validate settings against LDAP server, when enabling LDAP. Disabling allows settings to be saved, even if LDAP server is not available.	Enabled

Parameter	Description	Default value
Server URL(s)	Space separated list of full LDAP URL:s, e.g. <i>ldap://somehost:some-Port</i> See "LDAP Hosts/Urls" on page 134.	ldap://local-host:389
Bind user Dn	Bind user name, either <i>account-Name@domain.foo</i> or full DN.	addEntry-User@domain Name.se
Bind user password	Bind user password.	
Base search context DN	Defines the root context, from which searches will origin.	CN=Users,DC=your_domain,DC=com
Account search filter	Defines how a user account DN should be searched for.	(&(object-Class=user)(sAMAccountName={0}))
Search timeout (milis)	Defines the timeout for an LDAP query in milliseconds.	1000
User groups attribute name	User attribute that defines the groups of the user.	memberOf
Account name mapped attribute	Defines the user attribute mapped to the account name. Leave empty if you do not wish to map this attribute.	sAMAccountName
First name mapped attribute	Defines the user attribute mapped to the first name of the user. Leave empty if you do not wish to map this attribute.	givenName
Last name mapped attribute	Defines the user attribute mapped to the last name of the user. Leave empty if you do not wish to map this attribute.	sn

Parameter	Description	Default value
Locale mapped attribute	Defines the user attribute mapped to the locale of the user. Leave empty if you do not wish to map this attribute.	msExchUser-Culture
RTL mapped attribute	Defines the user attribute mapped to the right-to-left setting of the user. Should be evaluated to true/false. Leave empty if you do not wish to map this attribute.	rtl
Login code mapped attribute	Defines the user attribute mapped to the login code of the user. Leave empty if you do not wish to map this attribute.	loginCode
Email mapped attribute	Defines the user attribute mapped to the email address of the user. Leave empty if you do not wish to map this attribute.	email
Phone mapped attribute	Defines the user attribute mapped to the phone number of the user. Leave empty if you do not wish to map this attribute.	phone
Orchestrated Logout Settings	Settings for single logout	
RedirectURL	Redirect URL for Orchestra logout. System will redirect to this URL on logout.	/login.jsp
Logout URLs	URLs to log out from connected services when logging out from Orchestra. If several URLs, use comma as delimiter.	

Parameter	Description	Default value
SAML v2 Web Single Sign On Settings	For more information, see “SSO Setup Using SAML 2.0 Web SSO” on page 211.	
SAML v2 web SSO enabled	Whether SAML v2 web SSO is enabled, or not. Changing this value requires a restart to take effect.	false
Service provider entity ID	The ID of the service provider. This shall match the value configured in the Identity Provider.	Orchestra
Max authentication token lifetime	The maximum allowed age for an authentication token, in seconds. Tokens older than this will not be allowed and the user will need to log in to the IdP again.	3600
Role attribute identifier	The SAML attribute used to identify which role(s) the user has	http://schemas.microsoft.com/ws/2008/06/identity/claims/role
Username attribute identifier	The SAML attribute used to identify the user name of the user. ⓘ This parameter will be used to identify the user name of the user. For some Identity Providers, this is the user's email address. Orchestra will truncate such addresses, so that the domain name will not be included when the user name is stored. Example: "myuser@orchestra.com" will be treated as "myuser".	http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name
First name attribute identifier	The SAML attribute used to identify the first name of the user.	http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname

Parameter	Description	Default value
Last name attribute identifier	The SAML attribute used to identify the last name of the user.	http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname
Locale attribute identifier	The SAML attribute used to find the user's locale	http://schemas.xmlsoap.org/ws/2005/05/identity/claims/preferredLanguage
RTL attribute identifier	The SAML attribute used to identify if user is configured to use right-to-left layout	http://schemas.xmlsoap.org/ws/2005/05/identity/claims/rtl
Email attribute identifier	The SAML attribute used to identify the user's email address	
Phone attribute identifier	The SAML attribute used to identify the user's phone number	
Identity Provider Metadata URL		
SAML Keystore alias	If a custom keystore alias is used, it needs to be added here.	
Callback URL	Callback URL for the Orchestra client, e.g. https://<orchestra_host>:<port>/callback?client_name=SAML2Client.	
Callback URL pattern	If a callback URL is not provided the system will construct one based on the pattern.	https://{HOST-NAME}{PORT_DEFINITION}/callback

LDAP Hosts/Urls

❗ For an instruction about setup of Open LDAP, please refer to “Appendix D - Open LDAP Setup” on page 241.

The LDAP Hosts/Urls define what Active Directory hosts that should be used for authentication.

Orchestra will try to connect to the first one in the list, if Orchestra establishes a connection, the authentication will be performed towards that server. If the connection fails because the server is unavailable/down, Orchestra will automatically try the next server in the list.

❗ Authentication will only be performed once, using the first available connection.

It is possible to specify the LDAP server port, if the server does not use the default port, 389.

If the Active Directory server uses encrypted communication, that is LDAP over SSL (Secure Sockets Layer), one must import the appropriate server certificates in order to secure the connection.

For more information, see “Example:” on page 136.

❗ Note that the server url is different in this case, *ldaps://server:636*. The default secure port is 636.

LDAP Certificate Handling

❗ You must export the CA certificate from the Active Directory server to enable Secure Sockets Layer (SSL) security.

Different Corporate organizations have different methods and processes to create a CA root certificate. The procedure below provides information on creating a personal CA for Active Directory 2003.

Procedure

1. Log on as a domain administrator on the Active Directory domain server.
2. Install the certificate authority (CA) on the Microsoft Windows Server, which installs the server certificate on the Active Directory server. To do so, complete the following steps:
 - a) Click *Start > Control Panel > Administrative Tools > Certificate Authority* to open the CA Microsoft Management Console (MMC) GUI.
 - b) Highlight the CA computer, and right-click to select CA Properties.
 - c) From *General* menu, click *View Certificate*.
 - d) Select the *Details* view, and click *Copy to File* on the lower-right corner of the window.
 - e) Use the Certificate Export wizard to save the CA certificate in a file.

❗ You can save the CA certificate in either *DER Encoded Binary X-509* format or *Based-64 Encoded X-509* format.

Adding AD/LDAP server certificate to Orchestra Central

The Active Directory/LDAP server certificate must be added to the trusted store of certificates on Orchestra.

Preparation:

1. Copy the Active Directory Server certificate to a temporary location, for example:

```
C:\TEMP\myADcert.cer
```

SSL Certificate import:

1. In Orchestra *conf/security* folder, execute the following:

```
keytool -import -keystore truststore -file <cert-file> -storepass  
changeit
```

2. Edit *app/<wildfly-x.y.z.Final>/bin/standalone.conf.bat*.

❗ Make sure that the password entered above matches the following parameter value:

```
javax.net.ssl.trustStorePassword=changeit
```

Example:

```
Owner: CN=elkcertificate, DC=elk, DC=se
Issuer: CN=elkcertificate, DC=elk, DC=se
Serial number: 423e55f0dff557b24ffe5e41a7df65c8
Valid from: Thu Nov 13 11:02:53 CET 2008 until: Wed Nov 13 11:10:07
          CET 2013
Certificate fingerprints:
          MD5: 36:C0:35:A0:FA:4B:81:B4:F2:35:F2:F3:13:CF:73:92
          SHA1:
C3:F6:56:A5:F0:49:65:DE:DA:D5:64:94:FD:88:8D:32:8E:95:8F:87
          SHA256:
76:6A:02:FF:6C:A3:3C:74:BC:CF:C8:A6:23:F1:13:3F:69:6E:F0:BE:53:6E:
AB:AF:78:2A:6F:22:F1:73:A9:9A
          Signature algorithm name: SHA1withRSA
          Version: 3
```

Extensions:

```
#1: ObjectId: 1.3.6.1.4.1.311.21.1 Criticality=false
0000: 02 01 00                                     ...

#2: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#3: ObjectId: 2.5.29.31 Criticality=false
CRLDistributionPoints [
  [DistributionPoint:
    [URIName: ldap:///
    CN=elkcertificate,CN=testtpc2,CN=CDP,CN=Public%20Key%20Services,CN=
    Services,CN=Configuration,DC=elk,DC=se?certificateRevocationList?b
    ase?objectClass=cRLDistributionPoint, URIName: http://
    testtpc2.elk.se/CertEnroll/elkcertificate.crl]
  ]]

#4: ObjectId: 2.5.29.15 Criticality=false
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#5: ObjectId: 2.5.29.14 Criticality=false
```

```
SubjectKeyIdentifier [  
KeyIdentifier [  
0000: 39 C9 64 7C 58 79 86 A2    62 01 80 FF 09 1E F5 43  
    9.d.Xy..b.....C  
0010: 5C 23 14 DD                                \#..  
]  
]
```

Trust this certificate? [no]: yes

Certificate was added to keystore

3. Orchestra can now use SSL for communicating with the Active Directory.

❗ If either of the SSL parameters in the configuration file was changed, for example trust store file name or trust store password, the Orchestra must be restarted in order to commit the changes.

Adding AD/LDAP server certificate to Distributed Queue Agent

1. Copy the Active Directory server certificate to a temporary location, e.g. *C:\TEMP\fooAD-DER.cer, /tmp/fooAD-DER.cer*

2. Go to the profile template for the distributed Queue Agent, e.g.

*nix:

```
<install_dir>/media/agentProfiles/2.3-master-custom/conf/security
```

Windows:

```
<install_dir>\media\agentProfiles\2.3-master-custom\conf\security
```

3. Import the certificate using the standard Java command key tool:

```
keytool -import -keystore truststore.jks -file C:\TEMP\fooAD-DER.cer  
-storepass changeit
```

TrustStore Management

All truststore certificates are listed in System Administration > Settings > TrustStore Management. For each certificate in the list, you can see the alias and their validity period. To see more details, click on the certificate in the list.

To upload a new certificate:

1. Enter an alias.
2. Click **Choose file**. Valid file extensions are .cer, .crt,.pem, .der.
3. Click **Upload**.

To delete a certificate, click on the trash can icon.

To read more about certificates and truststore, see “Secure Communication” on page 165.

ⓘ You can only handle the truststore for Central Orchestra here.

Mark Types

Mark Types are typically used for when a Customer Feedback Unit (CFU) of some kind (either an Expressia, or a CFU tablet) is part of your system. Before those units can be set up, Mark Types must be created, in this tab and Marks must be created in the *Business Configuration* application.

Mark Types and Marks are also used to define the Outcomes that can be connected to Services. For this, there is a pre-installed Mark Type available in Orchestra.

For more information, see the Administrator’s Guide, found on Qmatic World.

Another possible way to use Mark Types and Marks is for NPS[®] scoring. NPS¹ stands for Net Promoter Score. For more information, see http://en.wikipedia.org/wiki/Net_Promoter.

There is a pre-installed, default *NPS* Mark Type available in Orchestra. For this Mark Type, the following marks are available: *0-10* and *skip*. By default, *Codes enabled* is disabled and *Visible in configuration* is enabled for this Mark Type.

When the *Mark Types* tab is opened, all available Mark Types are listed:

1. Net Promoter, NPS, and the NPS-related emoticons are registered service marks, and Net Promoter Score and Net Promoter System are service marks, of Bain & Company, Inc., Satmetrix Systems, Inc. and Fred Reichheld.

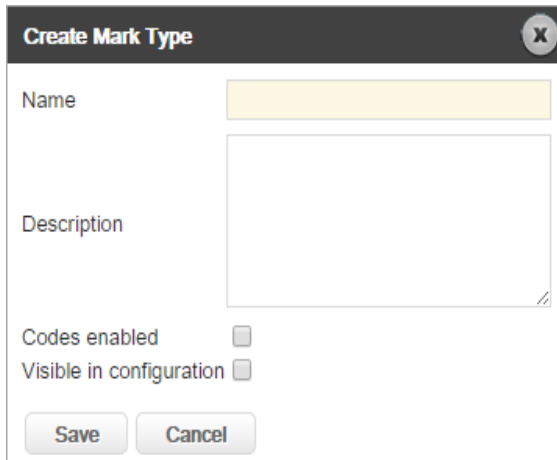
Mark Types

Create Mark Type					
Id	Name	Description	Codes enabled	Visible in configuration	Number of Marks
nps	NPS	System default mark type for Net Promoter Score (NPS)	No	Yes	12
outcome	Outcome	System default mark type for service outcomes	No	Yes	0

In this tab, it is also possible to delete existing Mark Types.

ⓘ The pre-installed *Outcome* Mark Type can, however, not be deleted. Also, it is not possible to update its *Id*.

To create a new Mark Type, click the *Create Mark Type* button. The following window is opened:



The 'Create Mark Type' dialog box contains the following fields and controls:

- Name:** A text input field with a yellow background.
- Description:** A larger text area for entering a description.
- Codes enabled:** A checkbox.
- Visible in configuration:** A checkbox.
- Buttons:** 'Save' and 'Cancel' buttons at the bottom.

Here, you enter a *Name* of the Mark Type. Optionally, enter a *Description*. Also, decide if Codes should be enabled for the Mark Type and if it should be Visible in the configuration, by checking the applicable check boxes.

When a new Mark Type has been created, it is possible to add Marks to it. This is done in the *Business Configuration* application.

7. Auditing

Topics in this chapter

Introduction	142
What is logged?	142
Where is it stored?	143
Enabling/disabling the functionality	145

Introduction

As businesses grow, we see a larger and larger need to track and store information about who does what in the system.

Below is a description of how auditing events are captured and stored in Orchestra as well as how to enable/disable the auditing functionality.

What is logged?

❗ The Calendar application has its own auditing functionality.

Below is a description of the different areas where audit information is logged.

Login/logout events

The following is logged:

- Successful login
- Unsuccessful login
- Logout

❗ Only login/logout attempts towards Central are logged.

Configuration changes

Database changes (CUD) in *qp_central* are logged.

For example:

- Branch publish.

❗ System activities, triggered by the Queue Agent, are not logged.

- Branch configuration.
- Service configuration.
- Work Profile configuration.
- Segmentation Rule configuration and Queue handling.
- Context Marketing configuration.
- Unit Type configuration.
- Surface Application configuration.
- Mark Types and Marks.
- User management.

❗ No auditing events are stored for handling of widgets (for example uploading or deleting widgets).

❗ Some tables are filtered, i.e not logged to auditing.

Customer object

For Customer objects, auditing events are stored for Create, Update and Delete (CUD).

❗ Due to GDPR regulations, actual Customer information is *not* stored. However, the username of the person who performed the change and when the change was done will be stored, if auditing is enabled.

Queue Agent

The following is logged:

- Configuration changes, such as Agent Profile synchronization and Upgrade.

❗ Updates triggered by the Queue Agent are not logged.

- Starting and stopping a Queue Agent

Calendar

The following is logged:

- Who (which User) booked/deleted/confirmed/updated the Appointment and at what time?

Where is it stored?

Central

Auditing is stored in the *qp_auditing* database, in the *events* table. This table contains the following columns:

- *id* (integer)
- *event_date_time* (timestamp without time zone)
- *category* (text) - example: *login*
- *old_value* (text)
- *new_value* (text)
- *principal* (text) - Describes the User in the local custom fashion.
- *remote_host* (text)
- *server_host* (text)

- *origin_code* (text) - The source code location for the place where the action of the event audited is taking place.
- *entity_name* (text)
- *entity_id* (text) - For example the Branch Id.
- *description* (text) - May only be *null* if either of the columns *old_value* or *new_value* have content. Example "A failed login attempt occurred."

For *CREATE* events, *old_value* is stored as *null* and *new_value* is stored as a shallow JSON string of the created object.

For *UPDATE* events, *old_value* contains the detected updated parameters with the value before the change and *new_value* is stored as a shallow JSON string of the updated object.

For *DELETE* events, the old value is a shallow JSON string of the deleted object and the *new_value* is stored as *null*.

A shallow JSON string means that referenced objects will be presented as a list of id:s.

Calendar

Auditing is stored in the database *qp_calendar*, in the *calendar_audit* table. This table contains the following columns:

- *id* (bigint)
- *branch_id* (bigint)
- *change* (text)
- *entity_id* (bigint)
- *entity_type* (character varying, 255)
- *audit_operation* (character varying, 255)
- *time_stamp* (character varying, 255)
- *user_name* (character varying, 255) - the name of the logged in User. *QP_* - *PUBLIC* means that no User was logged in when the event took place.

To reduce the size of this table, entries with *entity_type* *APPOINTMENT* older than a threshold value which by default is 365 days, are removed nightly at 03:30.

The threshold value can be changed in the database table *qp_calendar.settings*, jsonproperty *auditSettingsGroup.auditDeleteOldAppointmentThresholdDays*. The time when old entries are removed can be configured in *calendar.conf* by changing the property *delete.audit.cron.expression*.

Enabling/disabling the functionality

Installation - Orchestra Central

If you run the Installation Wizard, Auditing will be installed automatically, just like all other modules.

However, if auditing is enabled or not depends on if the Auditing module is included in your license, or not.

The auditing database, *qp_auditing* is created by default during a fresh installation of Orchestra.

❗ If the license is removed or disabled, existing auditing data is kept.

In the *install.properties* file, which should be used when running the installation in silent mode, you can find the application called *application.auditing*, which by default is set to *true*. The properties for the *qp_auditing* database can also be found in the *install.properties* file.

Upgrade

If the Auditing module is not included in your license, auditing is disabled.

❗ If auditing is not installed on the system, auditing will be disabled, regardless of if the Auditing module is licensed, or not.

The auditing database, *qp_auditing*, is **not** automatically created during upgrade. Enabling auditing in an upgrade requires setting up the required user and table, as well as modifying the *install.properties* file.

To enable auditing, follow the steps described in Configuration - Database, below.

Configuration - Database

Prepare your database by running the scripts corresponding to your database (Oracle Database / Microsoft SQL / PostgreSQL).

1. Open a database tool that allows you to run a database script (e.g. SQL Developer for Oracle Database, or SQL Server Management Studio for Microsoft SQL). For PostgreSQL, see special instructions below.

2. Run the script corresponding to your database in a tool that can execute the script:

Microsoft SQL Server:

```
<tmp_dir>/db/auditing-mssql.sql
```

Oracle Database:

```
<tmp_dir>/db/auditing-oracle.sql
```

PostgreSQL:

1. Make sure that the PostgreSQL bin folder is added to environment variables.
2. Open the command prompt in the `<tmp_dir>/db/` directory.
3. Run the command:

```
psql -U postgres -f auditing-postgres.sql
```

Configuration - *install.properties* file

Set *application.auditing* = *true*.

Modify the *audit db* settings for the database picked in the previous step to match those of your db.

Example for PostgreSQL:

```
auditing.db = postgres
```

```
auditing.postgres.db.host = localhost
auditing.postgres.db.user = qp_auditing_insert
auditing.postgres.db.password = qp_auditing_insert
auditing.postgres.db.port = 5432
auditing.postgres.db.name = qp_auditing
```

Run the upgrade in silent mode, using the command *install.bat -s* for Windows, or *install.sh -s* for Linux.

8. Stat Aggregation

Topics in this chapter

Introduction	148
Stat PDI Jobs/Transformations - Example Configuration	148

Introduction

Pentaho Data Integration (PDI)

Pentaho Data Integration (PDI), also referred to as Kettle/Spoon, is a tool that can be used for creating jobs/transformations that are used for data aggregation.

To be able to create your own, customized, data aggregations, you need to install PDI. The installation file can be found on Qmatic World. Unzip the installation file to a suitable location and run the *Spoon.bat* file.

For more information about PDI, please refer to <http://www.pentaho.com/product/data-integration>.

❗ Do *not* install PDI on production servers, only for development of transformations!

Disclaimer

Pentaho Data Integration (PDI) is a third party tool. Qmatic is in no way responsible for support, maintenance, or translation of this tool.

Existing Jobs/Transformations in Orchestra

Orchestra Stat supports running PDI jobs/transformations by triggering them for configured events. Any event sent to Stat can be configured to trigger jobs.

There is a file called *stat-jobs.xml*, located in *<orchestra installation directory>/conf/stat-jobs*, where events and jobs/transformations are configured. The following section explains how to configure them.

Stat PDI Jobs/Transformations - Example Configuration

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<kettle-jobs>
  <job filename="qmatic/visit-transaction.ktr" max-events-
before-processing="10000" batch-limit="10000" max-time-before-
processing="10" >
    <event name="VISIT_END" />
    <event name="VISIT_END_TRANSACTION" />
    <event name="VISIT_REMOVE" />
    <event name="END_SERVICE" />
    <event name="VISIT_TRANSFER_FROM_QUEUE" />
    <event name="VISIT_NOSHOW" />
  </job>
  <job filename="qmatic/service-point-session.ktr" max-events-
before-processing="10000" batch-limit="10000" max-time-before-
processing="10" >
    <event name="SERVICE_POINT_CLOSE" />
  </job>
```

```

        <job filename="qmatic/staff.ktr" max-events-before-
processing="10000" batch-limit="10000" max-time-before-
processing="10" >
            <event name="USER_SESSION_END" />
        </job>
</kettle-jobs>

```

The root element *<kettle-jobs>* contains all configurations. This element contains a list of *<job>* elements that define which type of jobs and transformations are configured in the system. Job definitions can be either kettle jobs (*.kjb* files) or kettle transformations (*.ktr* files). Each job definition also contains settings for how often the job is supposed to run, and optionally a list of *<event>* sub-elements which define which events that trigger the particular job.

The transformation file to process when the event arrives is configured with attribute *filename* that expects the file to exist relative to Orchestra's *conf/stat-jobs* folder. Here *filename* is *qmatic/visit-transaction.ktr* which means the file must be in folder *conf/stat-jobs/qmatic*. If any configured file is missing at Orchestra start up, stat will fail with an error message.

Attribute *max-events-before-processing* controls how many events should pass through the system before the job is executed. If this is set to *1*, or left out entirely, the job will execute for every event (this is not advisable for events with a high throughput). All events defined for the job are counted towards this *max-events-before-processing* setting.

The attribute *max-time-before-processing* controls the maximum amount of time the system should wait before triggering the job execution.

The job will be executed when the configured (*max-events-before-processing* setting) number of events have passed through the system, or when the configured time (*max-time-before-processing*) has passed. Each time the job is triggered, the counter for *max-events-before-processing* is reset.

The setting *batch-size* is a setting that will be sent to the job as a parameter, which will indicate to the job how many events it should process in one execution.

Qmatic and Custom jobs/transformations

There are two separate folders in the *conf/stat-jobs* directory. These are *qmatic* and *custom*. The *qmatic* folder contains all internal Orchestra jobs/transformations that are required by Stat, to get correct system functionality.

❗ These (or their configuration) must not be altered in any way.

Any custom requirements are met by creating custom jobs/transformations that shall be added to the *custom* directory. Custom jobs/transformations are referenced in *stat-jobs.xml* by prepending the job filename with the "*custom/*" folder.

Example:

```
<job filename="custom/test.ktr"/>
```

Processing events

The following example illustrates how event and job/transformation processing is handled:

```
<kettle-jobs>
  <job filename="qmatic/visit-transaction.ktr" max-events-
    before-processing="10" batch-limit="1+" max-time-before-
    processing="10" >
    <event name="VISIT_END" />
    <event name="VISIT_END_TRANSACTION" />
  </job>
  <job filename="custom/myJob.ktr" max-events-before-
    processing="1" max-time-before-processing="10" >
    <event name="VISIT_NOSHOW" />
  </job>
  <job filename="custom/myOtherJob.ktr" max-time-before-
    processing="10" />
</kettle-jobs>
```

In this example, every time a visit is completed (*VISIT_END*) or a multi-service-visit has a service completed (*VISIT_END_TRANSACTION*), the counter for the *visit-transaction.ktr* is incremented. When this counter reaches 10, (configured by *max-events-before-processing*) the job will be executed.

If no (or fewer than 10) events are processed in a 10 minute interval (configured by *max-time-before-processing*), the job will be executed anyway. Each time the job executes the counter and timer is reset.

Here is a timing example:

00:00 – system initializes – timer is started for job to run at 00:10.

00:08 – 10 events have been processed, job is started. Counter is reset to 0 and timer is reset to start at 00:18.

00:18 – 5 events have been processed. Job is started by timer, counter is reset to 0 and timer is reset to start at 00:28.

The job *custom/myJob.ktr* in this example has *max-events-before-processing* set to 1, meaning it will trigger on all events with name *VISIT_NOSHOW*. It is not recommended to use this setup with an event that is very common, as it could severely impact performance.

The job *custom/myOtherJob.ktr* does not have any events configured for it, meaning the job will be run on schedule only. In this configuration, the job will run

once every 10 minutes. This type of configuration is suitable when no reasonable event exists that could trigger the job, and certain calculations / actions need to be done periodically.

Jobs that fail due to errors in the job/transformation itself are logged but not processed further. They will be stored in the *failed_jobs* table in the database. For more information, see *Stat REST API* in the connector SDK for the rest endpoint to replay failed aggregations. Also, see “Failed jobs” on page 153.

Filtering on event type

Certain events are common across multiple domains, an example of this is the *CONNECT* event, which is sent both when Queue Agents connect and when devices (printers) and device controllers (gw1745) connect.

In order to be able to separate these events, one can use the *event-type* attribute to separate between these different types of event domains.

The defined event-types are:

- *AGENT_DAEMON*
- *AGENT_JIQL*
- *DEVICE_CONTROLLER*
- *DEVICE*
- *STAFF*
- *SERVICE_POINT*
- *VISIT*
- *APPOINTMENT*

An example configuration could be:

```
<event name="CONNECT">
  <job filename="custom/device-connect.ktr" event-type="DEVICE" />
  <job filename="custom/agent-connect.ktr" event-type="AGENT_JIQL" />
</event>
```

The different event types are used in the following scenarios:

- *DEVICE*: Printer connect and disconnect events. Printer status events (e.g. *OUT_OF_PAPER*)
- *DEVICE_CONTROLLER*: Device controller (gw1745) connect and disconnect events
- *AGENT_JIQL*: Publish events. End-of-day events sent from Queue Agents. Branch events sent from central (at e.g. Branch update or delete). Queue Agent connect / disconnect events.

- **AGENT_DAEMON**: End-of-day events sent from central. Retire events sent from central (when Queue Agent is deleted). Daemon connect / disconnect events.
- **STAFF**: Staff events (session start / end etc).
- **SERVICE_POINT**: Service point events (service point open / close).
- **VISIT**: Visit events (create, call, end etc).
- **APPOINTMENT**: Appointment events sent from central (create, update, cancel).

Transaction validation

The transaction count validation performed before running a job/transformation whenever the attribute *validate-transaction-count-before-execution* is set to *true* does the following:

1. Count the number of entries in *fact_visit_transaction* for the Staff/Service Point transaction in question.
2. Count the number of entries in *failed_events* for the same transaction.
3. If the sum of these two totals are the same as the number of transactions that the Staff/Service Point has served during the session, then the validation will succeed.

If the validation fails (e.g. an event belonging to the transaction has not yet arrived to Stat), the following steps will occur:

1. The job will be rescheduled for execution 10 seconds later (default value, can be configured).
2. The job will be rescheduled as long as the validation fails, up to 5 times before giving up (default value, can be configured).
3. If the validation has failed the maximum number of times, an entry will be added to the *failed_jobs* table.

The following *stat-jobs.xml* entry gives an example on how to reconfigure the rescheduling values mentioned above:

```
<event name="USER_SESSION_END">
  <job filename="qmatic/test_user_end_session.ktr" validate-
transaction-count-before-execution="true" maximum-validation-
failure-retry-count="1" validation-failure-retry-interval-
seconds="20"/>
</event>
```

Failed events

To list events that have failed for some reason:

GET <host:port>/stat/rest/events/listFailedEvents

To process failed events (using the result of the above REST call as input):

POST <host:port>/stat/rest/events/processFailedEvents

Failed jobs

To return a list of failed jobs:

GET <host:port>/stat/rest/kettleJobs/failed

To take a list of failed jobs as input to replay - this command is preferably combined with the above REST call:

POST <host:port>/stat/rest/kettleJobs/replay

Example of how this can be used:

GET localhost:8080/stat/rest/kettleJobs/failed

Answer: [1,3,4]

POST localhost:8080/stat/rest/kettleJobs/replay

Body: [1,3,4]

Outcome: Will try to replay jobs 1,3 and 4.

Failover Behaviour

If a message cannot be processed, it will get saved to the *failed_events* table, with some information that enables the message to be re-played in the future.

In the case of a serious system error, e.g. the database cannot be reached, the message will not be able to be saved to the *failed_events* table.

In this scenario, the following will happen:

- The message listener will stop processing messages for a while (default 5 minutes).
- The failed message will be put back on the queue and will be processed later.
- In the case that the event(s) in the message has been successfully saved to the database, and only the aggregation is missing, the message will only be re-sent 10 times.
- After 10 times, the message will end up in a "dead letter" queue, e.g. *stat/VisitEventQueueDeadLetterQueue* for visit events.
- Messages that have not had their event(s) saved to the relevant event table(s) will not be put on the dead letter queue, however if there are multiple failures

where a new instance of the message cannot be re-sent to the queue the message might end up in the dead letter queue eventually.

Settings to control the behaviour are defined in *stat.conf*, except for the limit of 10 consecutive retries, which is in *standalone-full.xml* (as it is a hornetq setting).

9. Localisation

Topics in this chapter

Introduction	156
Orchestra system GUI	156

Introduction

This chapter describes the procedures to follow to make localized versions of Qmatic Orchestra.

Orchestra system GUI

❗ This section is applicable the central parts of Orchestra.

Language files must use the character encoding utf-8.

You need to know the 2-character language code for the new language, it is the code that will show up in Orchestra; it is *sv* for Swedish, for example. See the standard *ISO 639-1*.

The language property handling procedures can be seen as three different parts:

- Central and Central Queue Agent
- Distributed Queue Agent
- Unit Type Templates and Widgets handling

Central and Central Queue Agent

To make a language specific version of Orchestra, do the following:

1. Make copies of all the original files in the *<Orchestra central installation directory>\conf\lang* folder (they are in English):

```
businessConfigMessages.properties
commonMessages.properties
and so on...
```

and rename them like this (for example): *businessConfigMessages_temp.properties*

❗ The corresponding original file will be used when a specific language file can not be found for some reason.

2. Translate all strings in all the files
3. The characters in the file must be Unicode (utf-8), If necessary for your language, run the program *native2ascii* on all the files, at the same time renaming them by changing *temp* to the language code (the code for Arabic in this example):

```
native2ascii -encoding utf-8 businessConfigMessages_temp.properties
businessConfigMessages_ar.properties
```

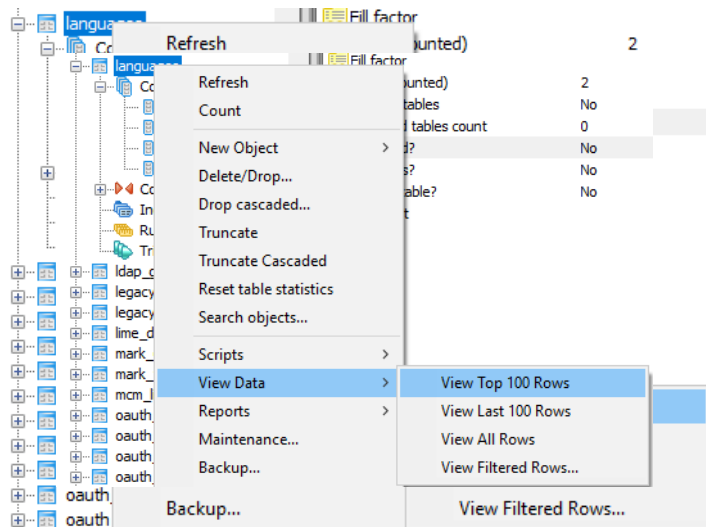
To find out the encoding used in a file, open it in a text editor or MS-Word, most programs will show the encoding. Word displays a list of suggested encodings, use trial and error to find the correct one.

4. All available languages *must* be entered, manually into the database table *LANGUAGES*. This is also where the left-to-right (ltr) and right-to-left (rtl) setting is set for each language.

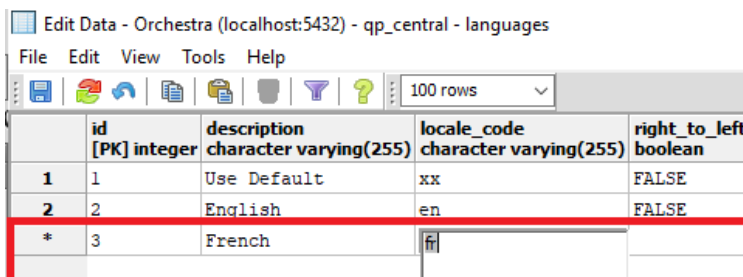
⚠ You must, yourself, make sure that you add your language correctly to this table!

Example. PostgreSQL:

- a) Open a tool such as *PGAdmin*.
- b) Locate the *qp_central* database, expand *Schemas* and *qp_central*.
- c) Expand *Tables* and locate the *languages* table. Right click and select *View Data* -> *View Top 100 Rows*, as in the following example:



- d) In the bottom row, enter your wanted language, as in the following example:



	id [PK] integer	description character varying(255)	locale_code character varying(255)	right_to_left boolean
1	1	Use Default	xx	FALSE
2	2	English	en	FALSE
*	3	French	fr	FALSE

! Right-to-left is set by checking the check box in the *right_to_left_boolean* column.

e) When done, click *Save*, in the upper left corner.

- If not already done so in the previous step, rename all the files by changing *temp* to the language code, like this: `businessConfigMessages_fr.properties`.
- Place the new files in the language folder along with the original files.

! A restart is needed for the changes to take place. If *lang* files have changed on the queue agent, then the queue agent needs to be restarted. If, however, *lang* files have changed on central, then central needs to be restarted.

As each new user is added in Orchestra, you select which should be the preferred language for that user, from a drop-down list.

Distributed Queue Agent

Properties files for distributed Queue Agents are placed in the Queue Agent profile. If you want to edit the files, you either need to create a new Agent profile, or update an old one.

For more information see “Suggested Working Order, Remote Upgrade” on page 75

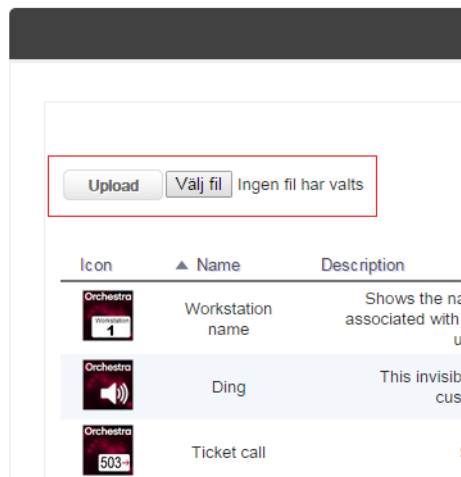
During deploy, the Queue Agent is restarted.

Browser Specific Settings

Please note that a few items in the GUI depend on the specific language settings of your browser.

For example, the buttons used to upload files, as illustrated in the following picture, depend on the language used in your browser:

Widgets



Unit Type Templates

When it comes to unit type templates (utt files), there are a couple of things to consider:

- Each unit type template (utt) that is to be used in your system needs to be edited and translated one by one.
- The *Name* and *Description* of the unit type can be translated in the actual fields of the GUI, marked in the following picture:



You can change the Name and Description to whatever you want. For more information, please see the Standard Unit Types Guide, found on Qmatic World.

Widgets

When it comes to widgets, the *Name* and *Description* (displayed in the GUI) are translated in the file *config.xml*, as in the following example for widget *callhistory*, where you can see both the Swedish (sv) and the English (en) version of the name:

```
<name>callhistory</name>
<name xml:lang="sv">Biljetthistorik</name>
```

```
<name xml:lang="en">Ticket call history</name>
<description>Keeps track of latest called tickets to allow customers
to see what tickets have been called where</description>
```

-

9. Xtend

Topics in this chapter

Introduction	162
Requirements	162

Introduction

Xtend is an application within Qmatic Orchestra where you can find extensions which will add value to your solution. In most cases, extensions are applications such as Counter, Concierge, Notification Admin etc and contain a war file, properties files, release notes and documentation.

From Xtend, you can install and deploy extensions directly, but you can also download the extension artifacts and install them manually.

Xtend is installed with Qmatic Orchestra by default.

Requirements

To be able to use Xtend, you need to make sure that the requirements below are fulfilled.

Open firewall

Xtend needs to be able to connect to <https://xtend.qmatic.cloud>. If you have a firewall or similar that doesn't allow access to the internet, it should be enough to open https traffic for this domain. If you don't do this, you cannot use Xtend.

Configure proxy settings

If you use a proxy, make sure to add the proxy settings to:

- standalone.conf.bat (Windows)
- standalone.conf (Linux)

There are proxy settings both for http and https and Xtend needs the https settings, i.e. -Dhttps.proxyHost and -Dhttps.proxyPort. In a system using http, you need to set proxy settings for both http and https.

10. Network Connectivity

Topics in this chapter

Orchestra Bandwidth Requirements 164

Orchestra Bandwidth Requirements

The bandwidth need is very different, depending on how you configure and run Orchestra.

For an Orchestra not using any special integrations, or for example Calendar, global variables, or media sync, the only data sent between the central and distributed Queue Agent are stat messages, as well as a keep alive message sent every 30 seconds.

Stat messages for a Visit are less than 3 kb in size and Service Point sessions are less than 5 kb in size. The needed bandwidth is thus depending on how many Visits and Service Point Sessions you have in your system.

However, when synchronizing a new Agent Profile or media, the bandwidth requirement is much larger.

Internal tests have been performed, with bandwidth as poor as 250 kbit/s as well as 4% packet loss. Profile and media synchronization still works under those conditions, but it is recommended to have higher bandwidth available.

11. Secure Communication

Topics in this chapter

Introduction	166
Enable HTTPS for Orchestra Central	170
Secure communication between Central and distributed Queue Agent	178
Secure Communication for a Queue Agent	184
Secure Communication for a Queue Agent	184
Disabling HTTP and Running HTTPS Only	198
Encryption of JMS connections	200
Bandwidth Requirements	202
Enable cookie secure flag	202
Changing the password for keystore	203

Introduction

Why security?

As our dependency on the internet has grown, the risks have also grown. Unencrypted HTTP requests reveal information about a user's behavior, and the interception and tracking of unencrypted browsing has become commonplace.

When properly configured, HTTPS can provide a fast, secure connection that offers the level of privacy and reliability that users should expect.

Therefore, we *highly* recommend that you configure Orchestra to use HTTPS / SSL / WSS for all communication.

First, let's go through the basics.

What is HTTPS?

Hyper Text Transfer Protocol Secure (HTTPS) is the secure version of HTTP, the protocol over which data is sent between your browser and the website that you are connected to. The 'S' at the end of HTTPS stands for 'Secure'. It means all communications between your browser and the website are encrypted.

What is SSL?

Secure Sockets Layer (SSL) is a standard security protocol for establishing encrypted links between a web server and a browser in an online communication. The usage of SSL technology ensures that all data transmitted between the web server and browser remains encrypted.

What is WSS?

WebSocket Secure (WSS) is a computer communications protocol, providing full-duplex communication channels over a single TCP connection. WebSocket is designed to be implemented in web browsers and web servers, but it can be used by any client or server application.

Why do you need a certificate?

When you request a HTTPS connection to a web page, the website will initially send its SSL certificate to your browser. This certificate contains the public key needed to begin the secure session. Based on this initial exchange, your browser and the website then initiate the 'SSL handshake'. The SSL handshake involves the generation of shared secrets to establish a uniquely secure connection between yourself and the website.

Notes on certificate host names and IP addresses

When creating a certificate, the certificate is only valid for the actual host name that it is created for (the "CN" attribute).

This should be the only host name which is used when communicating securely with an Orchestra system.

For example, if the host *orchestra-central.company.com* is used when setting up SSL for the Central system, all distributed Queue Agents should be configured (in *agent.conf*) to communicate towards that hostname and not the IP-address of the host.

If a mix of hostnames and IP-address needs to be used for self-signed certificates, the "*-ext san=...*" flag should be used when creating the certificate.

Different kinds of certificates

In this manual, we distinguish between the following kinds of certificates:

1. A certificate generated by Orchestra, this can be:
 - a) A certificate that can be used as it is - this should only be used for demo and test purposes, not for production environments!
 - b) A certificate that needs to be sent off to a well-known CA authority, such as Verisign or Thawte, to be signed
2. An existing certificate, created *outside* Orchestra, this can be:
 - a) A certificate signed by a Private CA Body
 - b) A certificate already signed by a well-known CA authority, such as Verisign or Thawte.

For both a)-versions above, the truststore needs to be updated. For both b)-versions above, the Queue Agent already trusts the certificate, so no changes are needed on the Queue Agent.

The instructions below describe what to do, depending on what kind of certificate you are using.

ⓘ You need to only follow the instructions for your certificate. Make sure that you follow the right set of instructions!

The difference between a Truststore and a Keystore

The purpose of the truststore is to *verify* credentials and the purpose of the keystore is to *provide* credentials.

In other words, the main difference between a truststore and a keystore is that truststore is used to store certificates from trusted Certificate Authorities (CA) which are used to verify certificates presented by the Server in SSL Connection, while keystore is used to store private key and own identity certificates which a program should present to other parties (Server or client) to verify its identity.

Orchestra is installed with a default keystore and a default truststore. The truststore is a copy of the Java Runtime Environment (JRE) truststore and the keystore is empty.

The truststore can be managed in System Administration > Settings > TrustStore Management.

Both files are located in the folder *<installation directory>/conf/security*.

Keystore files for standalone Queue Agents also follow the same structure and the files are included in the default Queue Agent profiles.

In order to use another keystore or truststore, the settings are located in the following files:

- Central Orchestra: *<installation directory>/app/<wildfly>/bin/standalone.conf.bat* (*standalone.conf* for Linux).
- Standalone Queue Agent: *<Queue Agent installation directory>/conf/agent.conf*

The values that affect keystore and truststore settings:

-Djavax.net.ssl.trustStore

-Djavax.net.ssl.trustStorePassword

-Djavax.net.ssl.keyStore

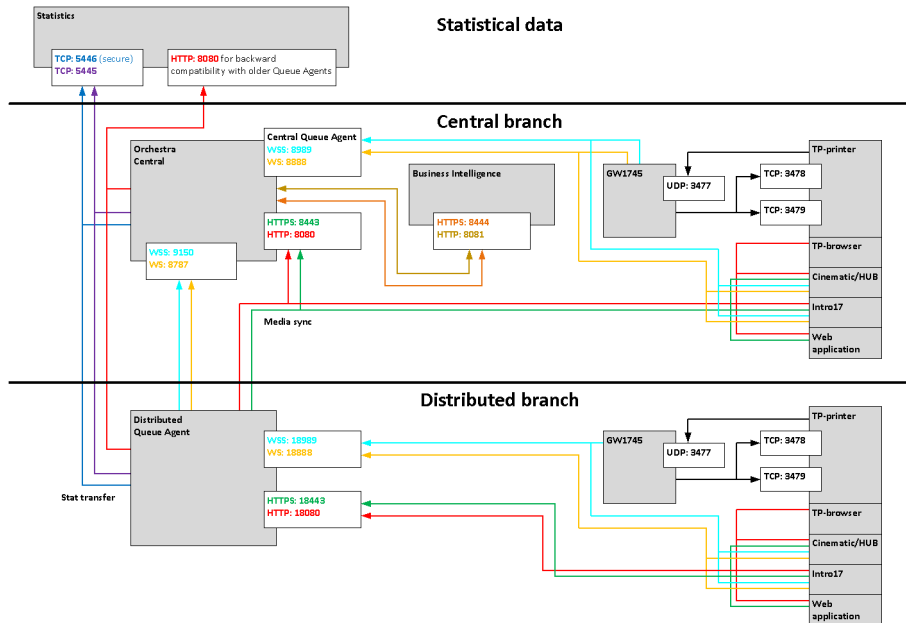
-Djavax.net.ssl.keyStorePassword

❗ If upgrading from Orchestra 5.4, it will be required to import any certificates previously used when running Glassfish, as they will not be included in the upgrade to Wildfly.

❗ TP3115, TP Touch and Intro 8 do not support encrypted communication at all. Those units need to go through HTTP.

Ports and Protocols

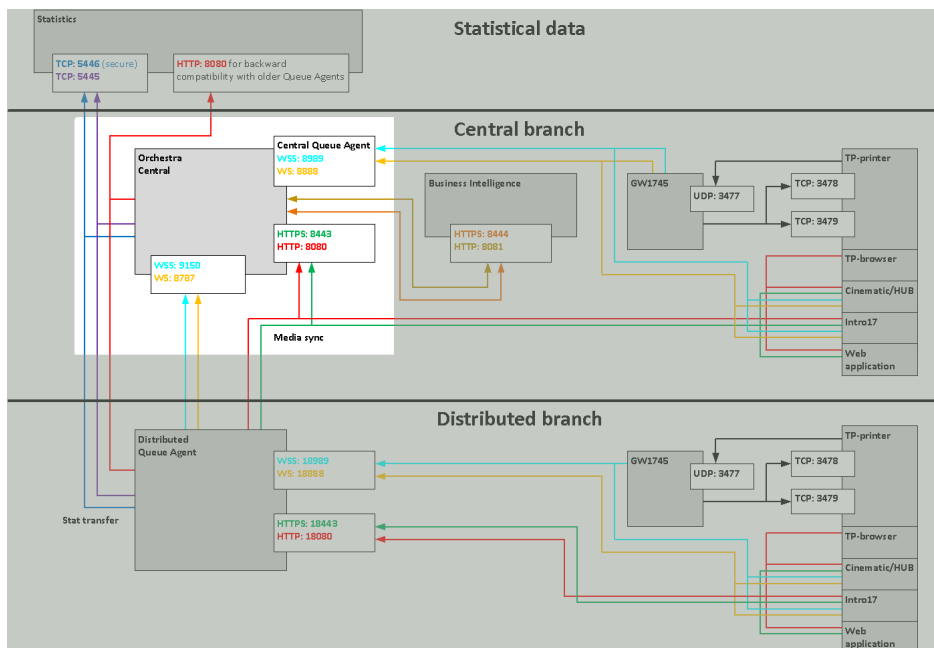
The following image illustrates the ports and protocols used in Orchestra:



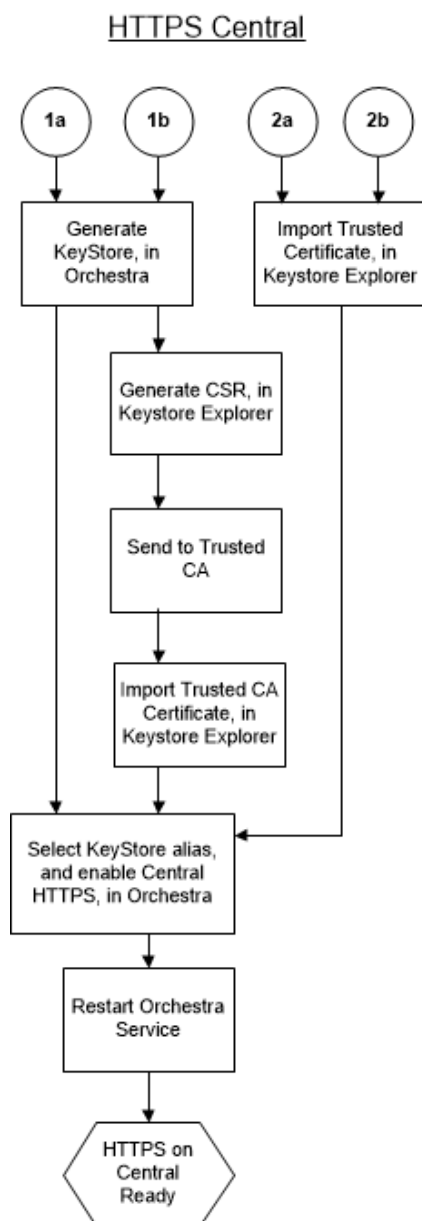
In the instructions below, we will go through the image above and how to set up secure communication for each part.

Enable HTTPS for Orchestra Central

In this first scenario, we will enable HTTPS for Orchestra Central, highlighted in the following picture:



The following flow chart illustrates the steps that need to be taken to set up HTTPS for Central. Each number (1a, etc) refers to a certificate type (see “Different kinds of certificates” on page 167.), described by a corresponding section below.



1a: Certificate Generated by Orchestra, used as is

! We recommend using a tool such as Keystore Explorer, <http://keystore-explorer.org/>, for managing your keystores and truststore.

1. First, a private key pair must be generated.
 - a) Navigate to the *Parameters* section of the *System Administration* application and locate the *HTTPS Parameters* section.
 - b) In the Certificate and key store Settings section, adjust the values below:

HTTPS Parameters

▼ Certificate and KeyStore Settings

(Re)generate certificate	☐	
KeyStore alias	orchestra	
Distinguished name	CN=localhost,OU=orgUnit,O=org.	
Subject alternate name	localhost	

- *KeyStore alias* - The name of the certificate entry in the key store. For OAuth2 integration to work, the default value *orchestra* must be used.
- *Distinguished name* - The distinguished name of the certificate. The "CN" part should be the host name of the Orchestra server, the other parts, i.e which organization, country etc need to be filled in, so they match the customer's organization.
- *Subject alternate name* - This needs to be filled in, if the certificate should be valid for more than just one host name. For example, both host name and IP-address. Separate each entry with a comma.
Example: *testrd.q-matic.net, 192.168.8.206*

Once all settings have been set properly, Save the configuration.

2. Set the value for *(Re)generate certificate* to *true* and Save the configuration again.

This will generate a key entry in the keystore.

! Do **not** enable this setting once the HTTPS setup is complete, as it **will** overwrite any certificate using the same alias in the key store, potentially destroying the HTTPS setup.

3. Navigate to the *Parameters* section of the *System Administration* application and locate the *HTTPS Parameters* section.

4. In the *HTTPS server settings* section, adjust the values below:

▼ HTTPS server settings

HTTPS enabled ☒

KeyStore alias orchestra ▼

HTTPS port 8443

- *HTTPS port* - This is the port you want to use when running HTTPS, default is *8443*.
- *KeyStore alias* - This should match the alias in the key store, used when creating / importing the alias, default is *orchestra*.

When both these settings are set correctly, set the value for *HTTPS enabled* to *true* and click *Save*.

This will enable HTTPS in the underlying server (Wildfly).

5. Next, restart the Orchestra service.
6. Test your https connection by entering the url *https://<orchestra_ip>:<https port>*, in your browser and making sure that everything works as expected.

1b: Certificate Generated by Orchestra, to be sent to a CA Authority

The certificate needs to be sent to a certificate authority to be properly signed.

! We recommend using a tool such as Keystore Explorer, <http://keystore-explorer.org/>, for managing your keystores and truststore.

Follow these steps:

1. In Keystore Explorer open the keystore. This should be the default keystore for Orchestra. This file is located in *<installation directory>\conf\security* and is called *keystore.jks*.
2. Right-click and select *Generate CSR*. The default password is *changeit*.
3. In the *Generate CSR* window, click *Ok*.
4. Send the certificate request (*server-2048.csr*) to your certificate authority (for example Verisign or Thawte).
5. Once the certificate response has been received, it must be imported into the keystore.

The certificate response normally consists of 3 parts:

- The certificate authority's signer certificate, example: *symantec-ca.cer*
- The certificate authority's root certificate, example: *symantec-root.cer*

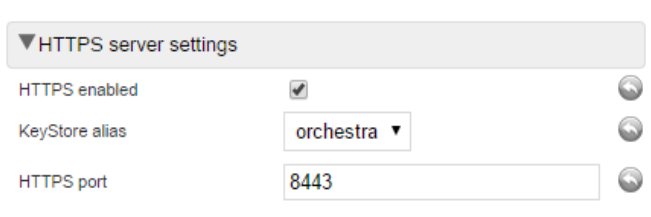
- Your signed certificate, example: *server-2048.cer*

All these certificates must be imported into the keystore in the correct order (the one defined above).

Open Keystore Explorer and select *Import Trusted Certificate*, select the wanted files and import them.

Enter *yes* when prompted if the certificate(s) should be trusted.

6. Navigate to the *Parameters* section of the System Administration application and locate the *HTTPS Parameters* section.
7. In the *HTTPS server settings* section, adjust the values below:



▼ HTTPS server settings	
HTTPS enabled	☒
KeyStore alias	orchestra ▼
HTTPS port	8443

- *HTTPS port* - This is the port you want to use when running HTTPS, default is *8443*.
- *KeyStore alias* - This should match the alias in the key store, used when creating / importing the alias, default is *orchestra*.

When both these settings are set correctly, set the value for *HTTPS enabled* to *true* and click *Save*.

This will enable HTTPS in the underlying server (Wildfly).

8. Next, restart the Orchestra service.
9. Test your https connection by entering the url *https://<orchestra_ip>:<https port>*, in your browser and making sure that everything works as expected.

2a: Certificate Generated outside Orchestra, Signed / Created by Private CA Body

If your enterprise/company uses a certificate already created and signed by a Private Certificate Authority (CA) body, this certificate needs to be imported to your truststore, before you can use it.

The keystore must contain only one alias, and that alias must use the name *orchestra*. So the default alias *orchestra* must be removed before importing the already signed certificate into the keystore.

1. Navigate to the *Parameters* section of the System Administration application and locate the *HTTPS Parameters* section.
2. In the *HTTPS server settings* section, adjust the values below:

▼ HTTPS server settings

HTTPS enabled ☒

KeyStore alias orchestra ▼

HTTPS port 8443

- *HTTPS port* - This is the port you want to use when running HTTPS, default is *8443*.
- *KeyStore alias* - This should match the alias in the key store, used when creating / importing the alias, default is *orchestra*.

ⓘ Your private CA certificate should be available in this drop-down.

When both these settings are set correctly, set the value for *HTTPS enabled* to *true* and click *Save*.

This will enable HTTPS in the underlying server (Wildfly).

3. Next, restart the Orchestra service.
4. Test your https connection by entering the url *https://<orchestra_ip>:<https port>*, in your browser and making sure that everything works as expected.

2b: Certificate Generated outside Orchestra, already signed by CA authority

If your enterprise/company uses an already signed Certificate Authority (CA) certificate, this certificate needs to be imported to your truststore, before you can use it.

The keystore must contain only one alias, and that alias must use the name *orchestra*. So the default alias *orchestra* must be removed before importing the already signed certificate into the keystore.

❗ We recommend using a tool such as Keystore Explorer, <http://keystore-explorer.org/>, for managing your keystores and truststore.

Follow these steps:

1. In Keystore Explorer, select *Open an existing KeyStore*, browse for and open the keystore. which is located in `<installation directory>\conf\security` and is called `keystore.jks`. Enter the password for the keystore (the default password is *changeit*).
2. In the *Tools* menu, select *Import Key Pair*.
3. Select the wanted type of key pair, for example *PKCS #12*.

- ❗ The format supported by Orchestra is pkcs12 files (i.e *.p12 and *.pfx files).
4. Specify the password for the certificate, then browse for and select the certificate file, and click *Import*.
 5. Enter the alias *orchestra*. (OAuth2 integration will not work if another alias is used.) This will be displayed in the Orchestra GUI.
 6. Enter the Key Pair password. This must be the same as the password you entered in 1. Default is *changeit*.
 7. Make sure that the key pair is imported correctly and save the updated keystore.
 8. Navigate to the *Parameters section of the System Administration* application and locate the *HTTPS Parameters* section.
 9. In the *HTTPS server settings* section, adjust the values below:

▼ HTTPS server settings

HTTPS enabled	☒	↻
KeyStore alias	orchestra ▼	↻
HTTPS port	8443	↻

- *HTTPS port* - This is the port you want to use when running HTTPS, default is *8443*.
- *KeyStore alias* - This should match the alias specified in 5. Default is *orchestra*.

❗ Your private CA certificate should be available in this drop-down.

When both these settings are set correctly, set the value for *HTTPS enabled* to *true* and click *Save*.

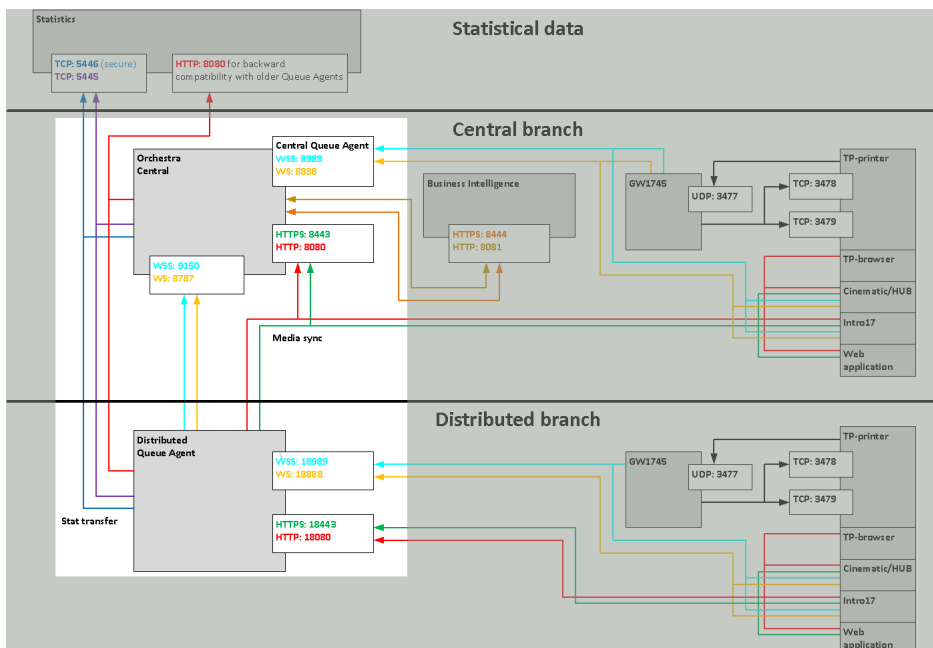
This will enable HTTPS in the underlying server (Wildfly).

10. Next, restart the Orchestra service.
11. Test your https connection by entering the url *https://<orchestra_ip>:<https port>*, in your browser and making sure that everything works as expected.

Secure communication between Central and distributed Queue Agent

! This section is only applicable if using a distributed Queue Agent!

In this scenario, we will secure the communication between a distributed Queue Agent and Orchestra Central, as highlighted in the following picture:



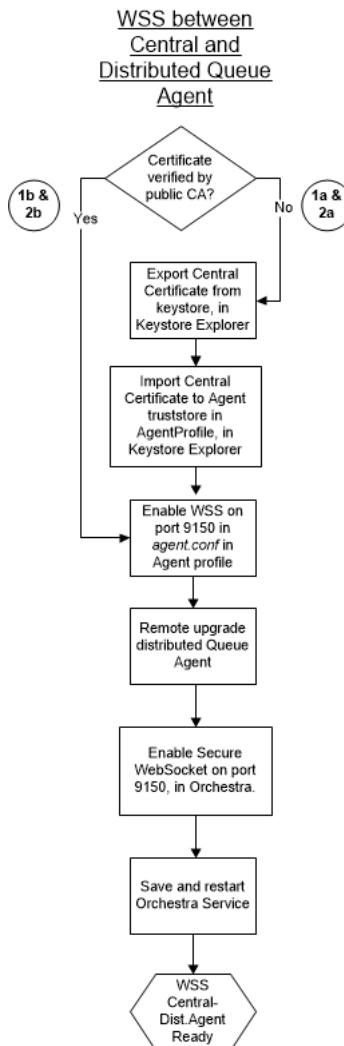
! Note that you need to secure the communication between the Daemon and Central. Make sure that the Daemon is version 6.2.xx or later. Since the Daemon component in a distributed Queue Agent cannot be upgraded as part of a remote upgrade, special considerations are required:

- If running on Hub or BranchHub, a firmware patch is required to make the Daemon able to communicate securely with Central.
The Daemon will use the settings configured in *agent.conf* for this purpose, more precisely the following settings: *"central.websocket.secure"*, *"central.websocket.secure.port"* and *"jetty.jvm.args"*.
- If running the distributed Queue Agent on a PC or similar, the easiest way is to replace the entire Queue Agent distribution with a newer version.

Apply the necessary upgrades and then continue with the configuration steps below for the distributed Queue Agents, which will enable secure communications for both Daemon and jiq.

Enable WebSocket Secure between Central and distributed Queue Agent

The flow chart below illustrates the steps needed to enable Websocket Secure between Central and distributed Queue Agent. These steps are described in more detail below.



1b & 2b: Certificate Verified by a Public CA

Preconditions:

- The steps outlined in the section “Enable HTTPS for Orchestra Central” on page 170 have been performed.
 - Section “Disabling Unencrypted WebSocket Communications” on page 181 should have been read through and considered.
1. Copy the existing Agent Profile to a separate folder, such as *tmp*.
 2. Open the <installation directory>\media\agentProfiles\your_agent_profile>\conf\agent.conf file and edit the following:
 - Change the setting for “central.websocket.secure” to “true”.
 - Change the port setting “central.websocket.secure.port” to the wanted port. Default is 9150.
 3. In the *System Administration* application, open the *Parameters* tab and locate the *Central WebSocket Server Settings* section.

Change these parameters:

Secure WebSocket enabled



Secure WebSocket port

9150

- *Secure WebSocket enabled* - true
- *Secure WebSocket port* - the wanted port (default 9150).



The *Secure WebSocket port* can **not** be the same as the *Unencrypted WebSocket port* (default: 8787).

4. Prepare, synchronize and upgrade the new Agent Profile.
5. Click *Save* to save the parameters.
6. Restart Orchestra.
7. Verify that the Queue Agent connects.

1a & 2a: Certificate NOT Verified by a Public CA

Preconditions:

- The steps outlined in the section “Enable HTTPS for Orchestra Central” on page 170 have been performed.
- Section “Disabling Unencrypted WebSocket Communications” on page 181 should have been read through and considered.



We recommend using a tool such as Keystore Explorer, <http://keystore-explorer.org/>, for managing your keystores and truststore.

1. In Keystore Explorer, open the file *keystore.jks*, located in *<installation directory>\conf\security*. The default password is *changeit*.
2. Right-click and select *Export -> Export Certificate Chain*. Click *Export*.
3. In Keystore Explorer, open the file *truststore.jks*, located in *<installation directory>\media\agentProfiles\<your_agent_profile>\conf\security*.
The default password is *changeit*.
4. Select *Tools -> Import Trusted Certificate* and import the *.cer file that you exported above.

⚠ Do not forget to click *Save*

5. Copy the existing Agent Profile to a separate folder, such as *tmp*.
6. Open the *<installation directory>\media\agentProfiles\<your_agent_profile>\conf\agent.conf* file and edit the following:
 - Change the setting for *"central.websocket.secure"* to *"true"*.
 - Change the port setting *"central.websocket.secure.port"* to the wanted port. *9150* is default.
7. In the *System Administration* application, open the *Parameters* tab and locate the *Central WebSocket Server Settings* section.

Change these parameters:

Secure WebSocket enabled	☒	
Secure WebSocket port	9150	

- *Secure WebSocket enabled* - checked
- *Secure WebSocket port* - the wanted port (default 9150).

⚠ The *Secure WebSocket port* can **not** be the same as the *Unencrypted WebSocket port* (default: 8787).

8. Prepare, synchronize and upgrade the new Agent Profile.
9. Click *Save* to save the parameters.
10. Restart Orchestra.

Disabling Unencrypted WebSocket Communications

⚠ Make sure that your Queue Agent is successfully connected before you disable unencrypted WebSocket communications.

1. In the *System Administration* application, open the *Parameters* tab and locate the *Central WebSocket Server Settings* section.

2. Uncheck the check box for the parameter *WebSocket enabled*.
3. Click *Save* to save the changes.
4. Restart Orchestra.

ⓘ If there are distributed Queue Agents in the setup that have not yet been configured to use secure WebSocket communication, they will stop working properly and it will not be able to e.g. remote upgrade them to support secure communications.

This is especially important for distributed Queue Agents that run on Hub or BranchHub, as they require a firmware patch to add support for secure WebSocket communication to the Daemon component. See [Disabling Unencrypted WebSocket Communications](#) above.

ⓘ Disabling unencrypted WebSocket will make it impossible for software running on other machines to communicate with the Central system using unencrypted communication.

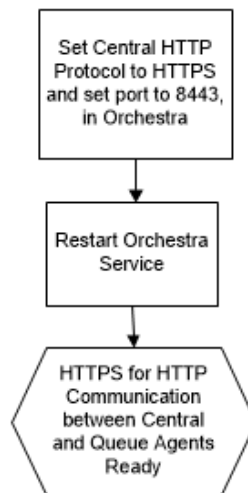
However, the unencrypted channel will not be 100% disabled, it will simply start to listen on the *localhost* / *127.0.0.1* network interface only.

The reason for this is that the Central Queue Agent still uses unencrypted communication to Central.

Enable HTTPS between Queue Agents and Central

The following flow chart illustrates the steps needed to enable HTTPS for HTTP communication between Queue Agents and Central. The steps are described in more detail below:

Enabling HTTPS for HTTP Communication Between Queue Agents and Central



The section “Secure communication between Central and distributed Queue Agent” on page 178 must be completed before this step.

1. In the *System Administration* application, open the *Parameters* tab and locate the *Central Access Parameters* section:

The screenshot shows the "Central Access Parameters" section. Under the "HTTP Settings" sub-section, there are two fields: "Central HTTP Port" with the value "8443" and "Central HTTP Protocol" with the value "https". Both fields have a small circular icon to their right.

2. Select *https* as *Central HTTP Protocol* and *Central HTTP Port* to 8443.
3. Save the Parameters and restart the Orchestra service.

Secure Communication for a Queue Agent

Configuration of secure WebSocket and HTTPS for Queue Agents is handled centrally and the necessary runtime configuration and certificates are published to each Queue Agent.

The central Queue Agent will always use the same certificate as that configured for the central WebSocket server (see “Enable WebSocket Secure between Central and distributed Queue Agent” on page 179).

ⓘ This requires secure WebSocket to be configured on the central server **before** configuring it on the central Queue Agent.

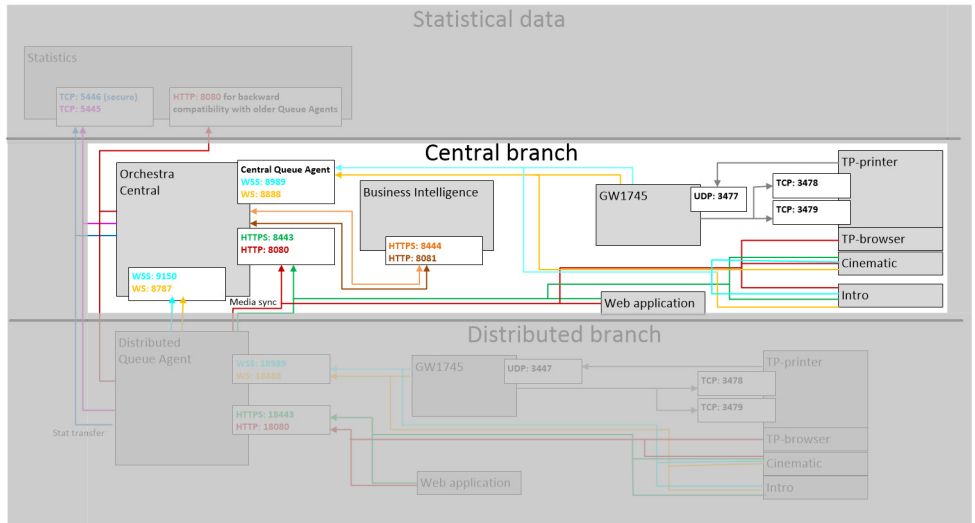
Distributed Queue Agents each have their own set of certificates, which are stored and provisioned from the central server.

ⓘ The same certificate will be used for both secure WebSocket and HTTPS, on distributed Queue Agents.

Enabling WebSocket Secure on a Central Queue Agent

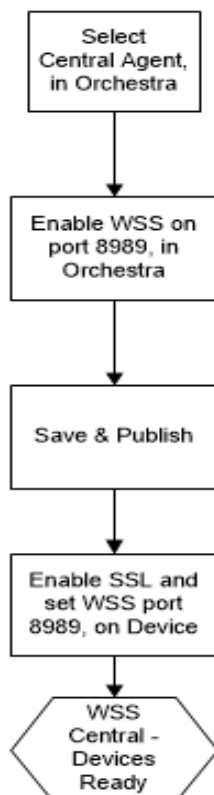
⚠ This step is only necessary if you have hardware devices, such as a printer or a GW1745 connected to your Central Queue Agent.

In this scenario, we will secure the communication for a central Queue Agent, as highlighted in the following picture:



The following flow chart illustrates the steps needed to enable Web Socket Secure between a Central Queue Agent and all connected devices, such as GW1745 / Hub:

WSS between
Central Queue
Agent and Devices



1. First, set up HTTPS and secure WebSocket for Central (see “Enable WebSocket Secure between Central and distributed Queue Agent” on page 179).
2. Navigate to *System Administration -> Queue Agents -> Agents*. Select the Central Queue Agent.

3. In the *Security* section, under *Prepared security configuration*, enable the check box for *WebSocket Secure enabled* and select a port for *WebSocket Secure port* (default: 8989):

WebSocket Secure enabled ☒

WebSocket Secure port

4. Click *Save* to stage the configuration.
5. Click *Publish* to enable secure WebSocket.

ⓘ This action will cause the device controller ports (8888 and 8989) to be restarted on the central Queue Agent, causing all connected devices (e.g. gw1745, Cinematic, HubMedia, Intro17 etc) to reconnect.

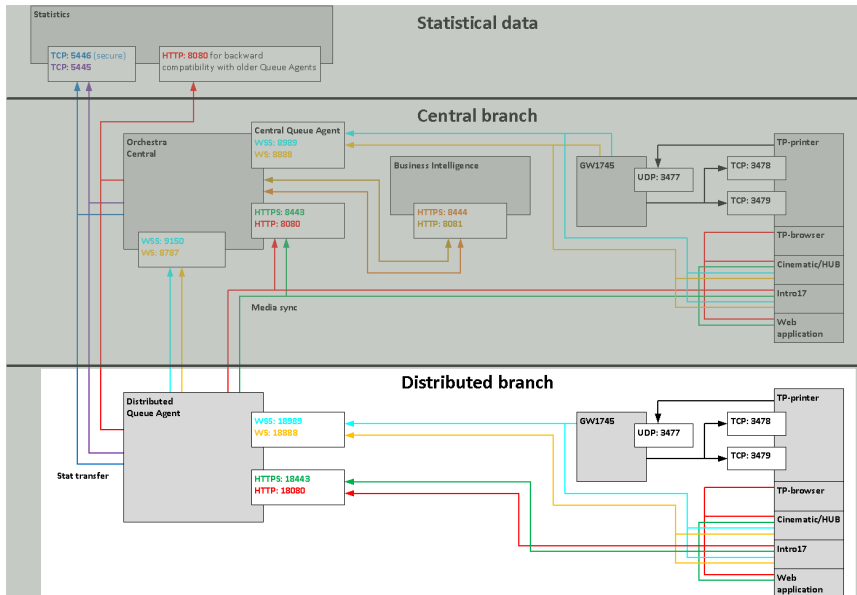
This might cause an interruption for ongoing business and should therefore be done outside business hours.

6. Log in to the applicable device, for example Hub, in the *Device Controller* tab, enable *SSL* and set the WSS port to 8989. For more information, please read the applicable hardware manual, found on Qmatic World.

Enabling WebSocket Secure on a distributed Queue Agent

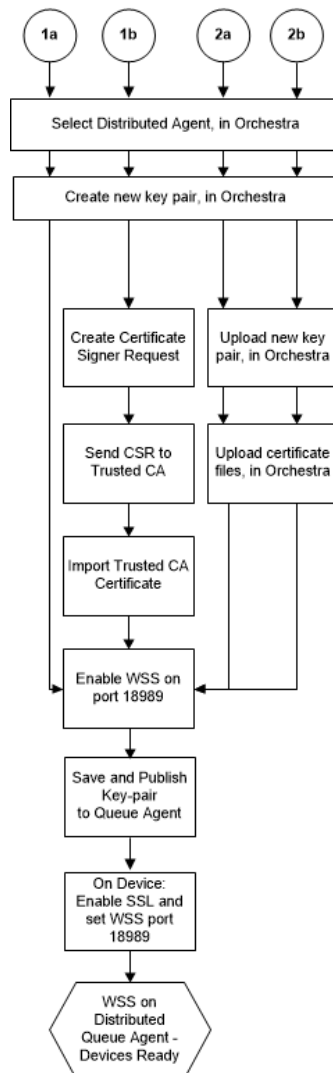
⚠ This step is only necessary if you have hardware devices, such as a printer, Hub or a GW1745 connected to a distributed Queue Agent.

In this scenario, we will secure the communication for a distributed Queue Agent, as highlighted in the following picture:



The following flow chart illustrates the steps needed to enable Web Socket Secure between a distributed Queue Agent and all connected devices, such as Hub / GW1745. Each number (1a, etc) refers to a certificate type, described by a corresponding section below.

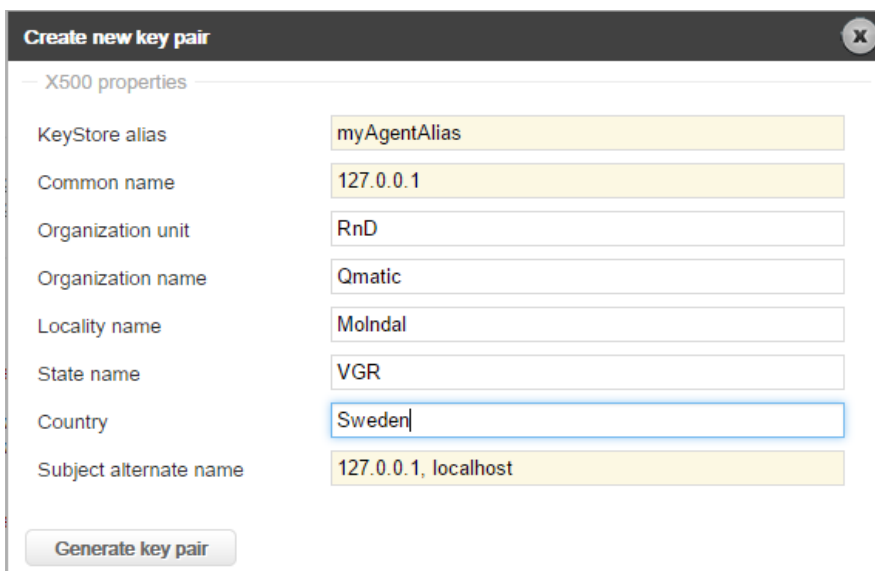
WSS between Distributed
Queue Agent and for
example GW1745



ⓘ The Queue Agent will keep the previously configured keystore during the Agent Profile synchronization part of the remote upgrade. The file *keystore.jks* will not match the one in the Agent Profile, centrally. Thus, the keystore must be managed through the *Security* section, as described in this section.

1a: Certificate Generated by Orchestra, used as is

1. Navigate to *System Administration* -> *Queue Agents* -> *Agents*. Select the wanted Queue Agent.
2. In the *Security* -> *Key pairs* section, click the *Create new* button.
3. A new window, *Create new key pair*, will be shown. Below is an example of what it may look like when filled in:



X500 properties	
KeyStore alias	myAgentAlias
Common name	127.0.0.1
Organization unit	RnD
Organization name	Qmatic
Locality name	Molndal
State name	VGR
Country	Sweden
Subject alternate name	127.0.0.1, localhost

Generate key pair

Enter the necessary fields:

- *Keystore alias*: The name the alias/certificate will be referenced in the configuration. (Mandatory)
- *Common name*: This is the host name or IP address of the Queue Agent. (Mandatory)
- *Organization unit*: The organizational unit (e.g. a department name). (Optional)
- *Organization name*: The organizational name (e.g. the company name). (Optional)
- *Locality name*: The locality (e.g. city). (Optional)
- *State name*: The state/region. (Optional).

- *Country*: The country. (Optional)
- *Subject alternate name*: additional host names / IP addresses the certificate should be valid for, separated by a comma. (Mandatory).

Click *Generate key pair* when the fields are filled in.

These fields will be combined to form the fully distinguished name of the certificate.

Example, using these values :

Key store alias: myAgentAlias

Common name: 127.0.0.1

Organization unit: RnD

Organization name: Qmatic

Locality name: Molndal

State name: VGR

Country: SE

Subject alternate name: 127.0.0.1, localhost

This will generate a certificate entry with the distinguished name "*cn=127.0.0.1, ou=RnD, o=Qmatic, l=Molndal, s=VGR, c=Sweden*". This certificate will be valid for host names 127.0.0.1, localhost.

❗ When the certificate is stored, the alias name will be suffixed with a timestamp (e.g. *myAgentAlias-2017-04-10T13:37:00*). This is to avoid conflicts and overwriting previously used certificates.

4. In the *Security -> Prepared security configuration* section, enable the check box for *WebSocket secure enabled* and select a port for *WebSocket Secure port* (default: 18989 for Distributed Queue Agents:

WebSocket Secure enabled	☒
WebSocket Secure port	18989
Agent hostname	

❗ It is also possible to enter *Agent hostname* here. This is the hostname of the Queue Agent and it must be resolvable by DNS. A Branch publish is needed for a hostname change to take effect.

5. Ensure that the selected alias in the *WebSocket Secure key pair* setting is correct. If not, select a different alias in the drop-down list in the *Key pairs* section. Click *Select* to choose it as the active alias/certificate.
6. Click *Save* to stage the configuration.

7. Click *Publish* to enable secure WebSocket.

ⓘ This action will cause the device controller ports (18888 and 18989) to be restarted on the Queue Agent, causing all connected devices (e.g. gw1745, Cinematic, HubMedia, Intro17 etc) to reconnect.

This might cause an interruption for ongoing business and should therefore be done outside business hours.

8. Log in to the applicable device, for example Hub, in the *Device Controller* tab, enable *SSL* and set the WSS port to 18989. For more information, please read the applicable hardware manual, found on Qmatic World.

1b: Certificate Generated by Orchestra, to be sent to a CA Authority for signing

1. Navigate to *System Administration* -> *Queue Agents* -> *Agents*. Select the wanted Queue Agent.
2. In the *Security* -> *Key pairs* section, click the *Create new* button.
3. A new window, *Create new key pair*, will be shown. Below is an example of what it may look like when filled in:

Create new key pair

— X500 properties

KeyStore alias	myAgentAlias
Common name	127.0.0.1
Organization unit	RnD
Organization name	Qmatic
Locality name	Molndal
State name	VGR
Country	Sweden
Subject alternate name	127.0.0.1, localhost

Generate key pair

Enter the necessary fields:

- *Keystore alias*: The name the alias/certificate will be referenced in the configuration. (Mandatory)
- *Common name*: This is the host name or IP address of the Queue Agent. (Mandatory)

- *Organization unit*: The organizational unit (e.g. a department name). (Optional)
- *Organization name*: The organizational name (e.g. the company name). (Optional)
- *Locality name*: The locality (e.g. city). (Optional)
- *State name*: The state/region. (Optional).
- *Country*: The country. (Optional)
- *Subject alternate name*: additional host names / IP addresses the certificate should be valid for, separated by a comma. (Mandatory).

Click *Generate key pair* when the fields are filled in.

These fields will be combined to form the fully distinguished name of the certificate.

Example, using these values :

Key store alias: myAgentAlias

Common name: 127.0.0.1

Organization unit: RnD

Organization name: Qmatic

Locality name: Molndal

State name: VGR

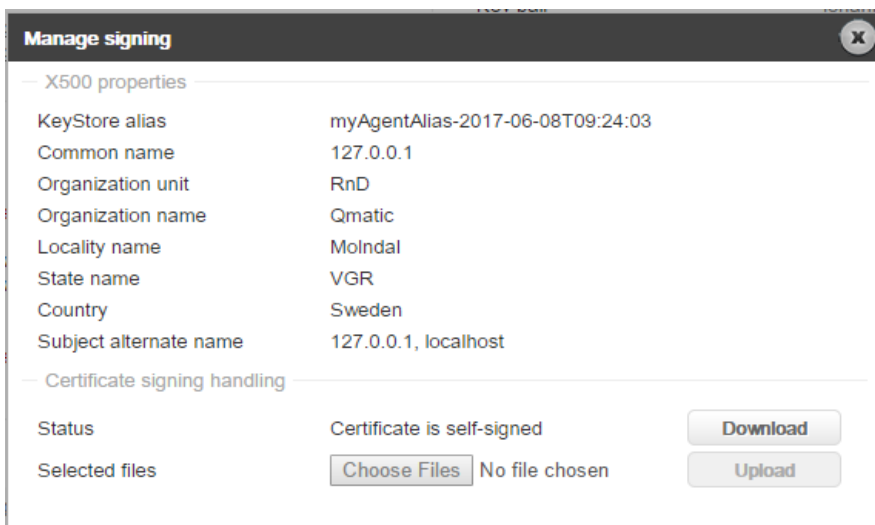
Country: SE

Subject alternate name: 127.0.0.1, localhost

This will generate a certificate entry with the distinguished name "*cn=127.0.0.1, ou=RnD, o=Qmatic, l=Molndal, s=VGR, c=Sweden*". This certificate will be valid for host names 127.0.0.1, localhost.

ⓘ When the certificate is stored, the alias name will be suffixed with a timestamp (e.g. *myAgentAlias-2017-04-10T13:37:00*). This is to avoid conflicts and overwriting previously used certificates.

4. In the *Key pairs* section, in the drop-down list, select the certificate and click *Manage signing*, the following window will be opened:



Click the *Download* button. This will generate a Certificate Signing Request that will be downloaded in your browser.

5. Send the Certificate Signing Request file to the desired Certificate Authority to get it signed.

The response should be one file for the signer response as well as a number of files describing the trust chain of the Certificate Authority.

Supported file formats are *.cer*, *.der* and *.crt*.

6. Navigate to *System Administration -> Queue Agents -> Agents*. Select the wanted Queue Agent.
7. In the *Security -> Key pairs* section, select the correct alias in the drop-down box and click the *Manage signing* button. The following window will be opened:

Manage signing

— X500 properties —

KeyStore alias	myAgentAlias-2017-06-08T09:24:03
Common name	127.0.0.1
Organization unit	RnD
Organization name	Qmatic
Locality name	Molndal
State name	VGR
Country	Sweden
Subject alternate name	127.0.0.1, localhost

— Certificate signing handling —

Status: Certificate signed by certificate authority

Selected files: No file chosen

- Click the *Choose Files* button. Select the certificate signer response, as well as the files containing the trust chain of the Certificate Authority.
- Click the *Upload* button.
- The certificate is now fully signed and can be properly used.
- Navigate to *System Administration* -> *Queue Agents* -> *Agents*. Select the wanted Queue Agent.
- In the *Security* -> *Prepared security configuration* section, enable the check box for *WebSocket secure enabled* and select a port for *WebSocket Secure port* (default: 18989 for Distributed Queue Agents):

WebSocket Secure enabled ☒

WebSocket Secure port

Agent hostname

⚠ It is also possible to enter *Agent hostname* here. This is the hostname of the Queue Agent and it must be resolvable by DNS. A Branch publish is needed for a hostname change to take effect.

- Ensure that the selected alias in the *WebSocket Secure key pair* setting is correct. If not, select a different alias in the drop-down list in the *Key pairs* section and click *Select* to choose it as the active alias/certificate.
- Click *Save* to stage the configuration.
- Click *Publish* to enable secure WebSocket.

ⓘ This action will cause the device controller ports (18888 and 18989) to be restarted on the Queue Agent, causing all connected devices (e.g. gw1745, Cinematic, HubMedia, Intro17 etc) to reconnect.

This might cause an interruption for ongoing business and should therefore be done outside business hours.

16. Log in to the applicable device, for example Hub, in the *Device Controller* tab, enable *SSL* and set the WSS port to 18989. For more information, please read the applicable hardware manual, found on Qmatic World.

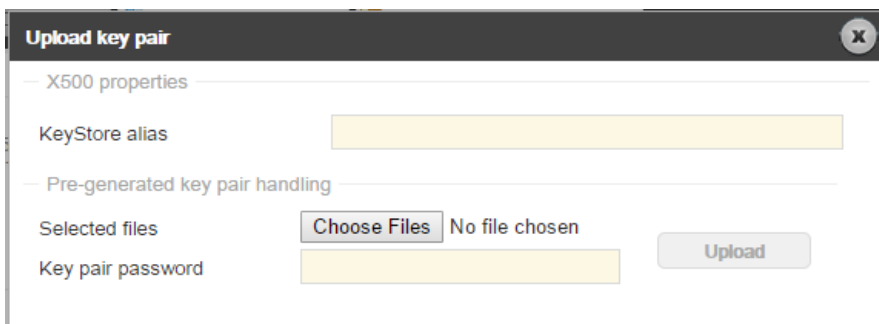
2a & 2b: Already existing Certificate, signed by public CA or Private Body

Similar to a response from a Certificate Authority, a pre-signed certificate will contain one file with the key-pair and more files making up the trust chain of the Certificate Authority.

ⓘ Supported file formats for the key-pair are *.pfx* and *.p12*.

ⓘ Supported file formats for the trust chain are *.cer*, *.der* and *.crt*.

1. Navigate to *System Administration* -> *Queue Agents* -> *Agents*. Select the wanted Queue Agent.
2. In the *Security* -> *Key pairs* section click the *Upload new* button. The following window will be opened:



3. Enter a new *KeyStore alias name*.
4. Click the *Choose Files* button and select the key-pair file as well as the files containing the trust chain of the Certificate Authority.
5. Enter the *Key pair password* for the key-pair file.
6. Click *Upload*.
7. The certificate and trust chain should now be imported and can be properly used.

8. Navigate to *System Administration -> Queue Agents -> Agents*. Select the wanted Queue Agent.
9. In the *Security -> Prepared security configuration* section, enable the check box for *WebSocket secure enabled* and select a port for *WebSocket Secure port* (default: 18989 for Distributed Queue Agents):

WebSocket Secure enabled	☒
WebSocket Secure port	18989
Agent hostname	

- ⓘ It is also possible to enter *Agent hostname* here. This is the hostname of the Queue Agent and it must be resolvable by DNS. A Branch publish is needed for a hostname change to take effect.
10. Ensure that the selected alias in the *WebSocket Secure key pair* setting is correct. If not, select a different alias in the drop-down list in the *Key pairs* section and click *Select* to choose it as the active alias/certificate.
 11. Click *Save* to stage the configuration.
 12. Click *Publish* to enable secure WebSocket.

ⓘ This action will cause the device controller ports (18888 and 18989) to be restarted on the Queue Agent, causing all connected devices (e.g. gw1745, Cinematic, HubMedia, Intro17 etc) to reconnect.

This might cause an interruption for ongoing business and should therefore be done outside business hours.

13. Log in to the applicable device, for example Hub, in the *Device Controller* tab, enable *SSL* and set the *WSS* port to 18989. For more information, please read the applicable hardware manual, found on Qmatic World.

Enable HTTPS communication for a distributed Queue Agent.

To enable HTTPS for a distributed Queue Agent, first enable WebSocket Secure communication for the distributed Queue Agent.

Once that is done, the only thing that is needed is to restart the Queue Agent.

The HTTPS connector in Jetty will use the same certificate that was used when configuring WebSocket Secure.

Enabling HTTP to HTTPS redirect

To enable HTTP to HTTPS redirect:

1. Navigate to *System Administration -> Queue Agents -> Agents*. Select the wanted Queue Agent.

2. In the *Security* section check the *HTTP to HTTPS redirect enabled* check box. Save the settings.
3. Restart the Queue Agent.

Selecting HTTPS protocol strength

By default, only the TLSv1.2 HTTPS protocol is enabled.

Depending on the client environment, connected hardware, browsers etc, it might be necessary to enable weaker HTTPS protocols.

The protocols that can be enabled are:

- SSLv3
- TLSv1
- TLSv1.1
- TLSv1.2

SSLv3 is necessary if a Vision unit is going to be connected to the system. If no such units are going to be connected to the Central Queue Agent, this protocol should not be enabled.

TLS versions prior to 1.2 might be needed for older types of browsers, this would depend on the client environment.

Wildfly

Edit the `app\wildfly-x.y.z.Final\standalone\configuration\standalone-full.xml` file.

Locate the `https-listener` configuration tag, change the `enabled-protocols` attribute, so that it matches the requirements, example:

```
enabled-protocols="TLSv1,TLSv1.1,TLSv1.2"
```

Save the file and restart Orchestra.

Disabling HTTP and Running HTTPS Only

⚠ This section should not be followed until you have made sure that your HTTPS communication is working properly.

You can configure Orchestra so that it exposes only HTTPS to connecting systems.

In order to do this, a few preconditions must be met:

⚠ Some hardware cannot use HTTPS at all, so if you have these types of hardware connected to the Central Queue Agent disabling HTTP is *not* an option:

- TP3115

- Intro8
- Cinematic

HTTP cannot strictly be disabled, since some components use it for internal communication.

However, it can be set to listen only to *localhost* (*127.0.0.1*) so no one outside of the Orchestra machine can use it.

To disable HTTP:

1. First configure HTTPS properly and ensure that it works.
2. Open the *System Administration* application and the *Parameters* page and change the system parameters *Central HTTP Port* and *Central HTTP Protocol*.

The port setting should be the port of the Central HTTPS port, default is *8443*.

The protocol setting should be *https*.

3. Stop Orchestra.
4. Edit the server configuration file to set address *127.0.0.1* for HTTP traffic.

Wildfly:

File: `app\wildfly-x.y.z.Final\standalone\configuration\standalone-full.xml`

Change this line:

```
<socket-binding name="http" port="${jboss.http.port:8080}"/>
```

To this:

```
<socket-binding name="http" interface="unsecure" port="${jboss.http.port:8080}"/>
```

Removing HTTPS Warnings in Browsers

If you use a self-signed certificate, you will get a HTTPS warning in the browser. To remove HTTPS warnings in your browser:

1. Go to your browser's *Settings* page and locate the section where certificates are managed. Import the *.cer file that you exported in the flows above.
2. Place the certificate on *Trusted Root Certification Authorities* and make sure that it is located there.
3. Restart your browser for the settings to take place.

Encryption of JMS connections

JMS is used to transfer statistical events from Central and Distributed Queue Agents to the Stat server (where the application *stat.war* is deployed).

The Stat server can be deployed together with Orchestra Central or on a standalone server.

To be able to use SSL for encryption of stat messages, it is necessary to configure matching certificates in a keystore on the stat server and a truststore on the Queue Agent. The process of setting up certificates and using remote upgrade to transfer them to remote Queue Agents is described earlier in this chapter.

In the example below, tcp port 5446 will be used for encrypted JMS traffic. This port needs to be open in the firewall.

To configure Orchestra with a standalone stat server to use encryption of all JMS traffic between central and distributed agents and stat and between central server and stat, please follow these steps:

1. Add certificates to keystore and truststore

You need to create a certificate that references both the central and stat server that will need to be added to the keystore on the stat server and to the truststore of both central server and distributed agents. If a self signed certificate is used the procedure for this is:

- a) In Orchestra central navigate to System Administration -> Parameters

- b) Select (Re)generate certificate
 - c) Add central server and stat server ip addresses to Subject alternate name
 - d) Click Save
 - e) Export the generated certificate from conf/security/keystore.jks
 - f) Import the certificate in the following locations
 - Central server: conf/security/truststore.jks
 - Central server: media/agentProfiles/<agent profile>/conf/security/truststore.jks for all agent profiles in use. Use remote upgrade to update distributed agents
 - Stat server: conf/security/keystore.jks
2. Open the file app/wildfly-x.y.z.Final/standalone/configuration/standalone-full.xml on the stat server and update key-store-password value.
- ```
<acceptor name="netty-ssl" factory-
class="org.apache.activemq.artemis.core.remoting.impl.netty.NettyAcceptorFact
ory">

<param name="host" value="0.0.0.0"/>

<param name="port" value="5446"/>

<param name="ssl-enabled" value="true"/>

<param name="key-store-path" value="${jboss.home.dir}/../../system/conf/
security/keystore.jks" />

<param name="key-store-password" value="changeit" />

</acceptor>
```
3. Open the file conf/broker.xml
- a) Locate the section stat-remote-connector and add parameters to enable ssl

```
<connector name="stat-remote-connector">

tcp://localhost:5446?sslEnabled=true;

trustStorePath=conf/security/truststore.jks;

trustStorePassword=changeit

</connector>
```
  - b) Locate the section audit-remote-connector and add parameters to enable ssl

```
<connector name="audit-remote-connector">

tcp://localhost:5446?sslEnabled=true;

trustStorePath=conf/security/truststore.jks;
```

```
trustStorePassword=changeit
```

```
</connector>
```

4. Change the connectors in the agent profiles

- a) In the file(s) `media/agentProfiles/<agent profile>/conf/broker.xml` make the same change as in 3) above
- b) Update the agents using remote upgrade

5. Open the file `app/wildfly-x.y.z.Final/standalone/configuration/standalone-full.xml` on the central server and update `key-store-password` value.

```
<remote-connector name="stat-remote-ssl-connector" socket-binding="stat-remote-ssl">
```

```
<param name="ssl-enabled" value="true"/>
```

```
<param name="trust-store-path" value="{jboss.home.dir}/../../../../system/conf/security/truststore.jks"/>
```

```
<param name="trust-store-password" value="changeit"/>
```

```
</remote-connector>
```

6. Restart all servers.

## Bandwidth Requirements

When HTTPS is used, the browser will disable some of its caching, which will cause higher bandwidth requirements as follows:

- The link between the web browser and the Orchestra web server requires 0.03 Mbit/s per concurrent user.
- This is about 150% higher bandwidth requirements compared to accessing the system with HTTP.

## Enable cookie secure flag

### SSO Cookie

This is the login/auth cookie used in Orchestra.

This can be set to use the Secure attribute in the following way:

- Edit `conf/shiro.ini` and add the following to the `[main]` section `"cookie.secure = true"`, e.g:

```
[main]
cookie.secure = true
```

Once `conf/shiro.ini` has been edited, save and exit the file, then restart Orchestra.

ⓘ Do not set this value if Orchestra is accessed by both authenticated HTTP and HTTPS connections, as HTTP will not work.

### **BAYEUX\_BROWSER cookie**

This is the cookie used by the CometD servlet, the mechanism used for e.g. counters and surfaces to receive events.

To set this cookie to have the secure attribute do the following:

1. Log in to Orchestra central as an administrator and navigate to System Administration -> Parameters.
2. Locate the "Browser Settings" section in the right pane.
3. Check the "Secure cookie flag enabled" checkbox.
4. Save.
5. Restart Orchestra.

ⓘ This might require that all surfaces (including media displays) are connected via HTTPS.

ⓘ For the Calendar application, note that the *Use the Secure cookie flag* setting should also be enabled in the *Calendar Admin* application, under *System Settings*, if applicable. For more information, see the Administrator's Guide.

## **Changing the password for keystore**

To change the keystore password, you need to run one script for changing the password for the keystore, and one script for changing the password within the file.

ⓘ `keytool.exe` can be found in `app/zulu8.58.0.13-ca-jdk8.0.312-linux_x64/bin` in the Orchestra installation.

ⓘ `keystore.jks` is located under `conf/security` in the Orchestra installation.

1. To change password for a key store, run the following command in a command-line window

```
keytool -keystore keystore.jks -storepass changeit -storepasswd
```

This command will prompt for a new password twice.

2. To set the password for the key inside the keystore to the same password, run the command below. Adjust alias as needed - the default is "orchestra". The "new storepass" argument should be the new password you set for the key store.

```
keytool -keystore keystore.jks -storepass <new storepass> -keypasswd -alias orchestra
```

This command will prompt once for the old key password ("changeit") and twice for the new password.

When you have changed the keystore password, you also need to update the password in the following files:

- standalone-full.xml
- standalone.conf (Linux) or standalone.conf.bat (Windows)

If you do a remote upgrade for a distributed agent, you also need to align the keystore password in the zip-files.

# 12. SSO setup

## Topics in this chapter

Introduction .....	206
SSO Setup Using Integrated Kerberos .....	206
SSO Setup Using SAML 2.0 Web SSO .....	211

## Introduction

Qmatic Orchestra supports Single Sign-On (SSO). SSO means that the users only need to log in once for several applications. They also only need to keep track of one username and password.

You can setup SSO using either Integrated Kerberos or SAML 2.0 Web SSO.

## SSO Setup Using Integrated Kerberos

Useful information can be found in the official SPNEGO documentation. The documentation includes complete descriptions of parameters and a thorough example of how to set up SSO using Kerberos. At the time of writing, the documentation can be found at the SPNEGO SourceForge webpage.

The setup can then be divided in two areas, the Orchestra server area and the Orchestra client (or Orchestra terminal user) area.

ⓘ Note that SSO using Kerberos is not working if the central or distributed agent resides in a different domain than the Active Directory.

### Client Area

Only one procedure is required to configure the Orchestra client(s). Set the client browser to identify the Orchestra server as an intranet site.

**For Microsoft Internet Explorer, follow these steps:**

1. Login to the Windows domain using the Windows account of the Orchestra terminal user.
2. Start Internet Explorer.
3. Open the tab: *Tools -> Internet Options -> Security*.
4. Select *Local Intranet*.
5. Set the *Security Level* to *Low*.
6. Click *Sites*.
7. Click *Advanced*.
8. Fill in the IP address or the host name of your Orchestra server and click *Add*.
9. Close this window.
10. Click *OK*.
11. Click *Advanced*.
12. In the *Security* section, make sure that *Enable Integrated Windows Authentication* is checked. If not, check the check box, and click *Apply*.

13. Close the Internet Options window by clicking OK.
14. Close the browser.

The browser is now be configured to support SPNEGO.

**For Mozilla Firefox, follow these steps:**

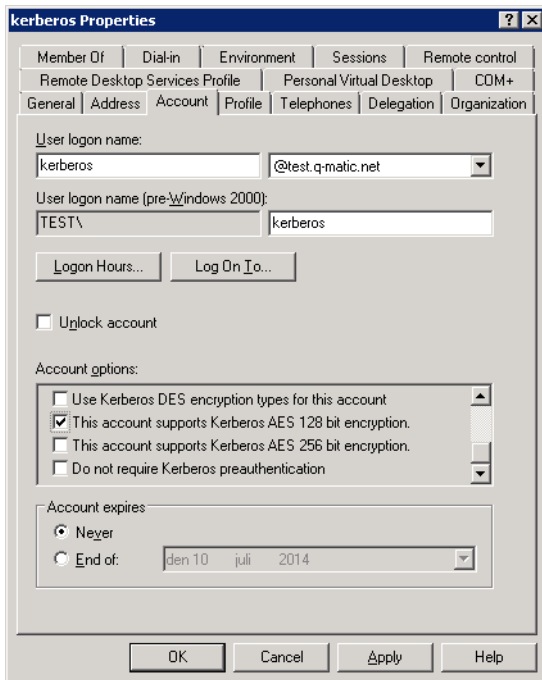
To authenticate Firefox automatically through a proxy (avoiding NTLM prompt), you have to modify 4 parameters.

1. Start Mozilla Firefox with Administrator privileges.
2. Open the page about:config (in the address bar)
3. Add your uris like hostname of FQDN (without port like 8080 / 8443) (separate with ,) in the following 3 parameters:
  - network.automatic-ntlm-auth.trusted-uris
  - network.negotiate-auth.delegation-uris
  - network.negotiate-auth.trusted-uris
4. Modify signon.autologin.proxy to be true
5. Restart Mozilla Firefox and it should now be working

**Server Area**

On the Active Directory domain controller, perform the following steps:

1. Register the host name of the Orchestra service in DNS.
  - This hostname should be in the same domain as the AD domain and **MUST BE** a DNS A-record, i.e. **NOT** a CNAME record.
  - E.g. if your AD domain is *test.q-matic.net*, then the host name could be *testrd.test.q-matic.net*.
2. Create a user that will hold the kerberos / SSO configuration. Alternatively re-use another user.
  - The user should have a pre-set password that should not be possible to change, and the password should never expire.
  - Example is a user named "*kerberos*" in the AD domain "*test.q-matic.net*".
  - To edit the properties of the user, right-click on its name in the *Active Directory Users and Computers* window.
  - The following window is opened:



In the *Account options* area of the *Account* tab, make sure that you check the check box next to *This account supports Kerberos AES 128 bit encryption*, as shown in the picture above.

3. Register the SPN (Service Principal Name, i.e. the servername) with the user. This is done via the *"setspn"* command.

The command should be executed like this:

```
setspn -A HTTP/<hostname> <username>
```

⚠ It is important that the *-A HTTP* part of the command is in upper case, for this to work!

Using the examples above the command would look like this:

```
setspn -A HTTP/testrd.test.q-matic.net kerberos
```

⚠ Note that *setspn* is not available in all versions of Windows 2003 server. You can, however, download the Windows server 2003 support tools from the Microsoft Web site.

The Orchestra server requires a few things in order to enable SSO.

Start by editing the `<install_dir>/conf/krb5.conf` file or `<agent_profile_dir>/conf/krb5.conf` file, if the configuration is part of a configuration change in a remote upgrade. Change the following entries in the file to match what is correct for your domain:

- `default_realm`
- `realms`
- `kdc`
- `default_domain`
- `domain_realm`

The simplest way to configure is to do the following when starting with the default `krb5.conf` file:

1. Replace (case-sensitively) all instances of "*YOUR.DOMAIN*" (in UPPERCASE) with the domain that the Orchestra server resides in, e.g. "*MYSUBNET.MYCOMPANY.COM*".

❗ This should be in 4 places and in UPPERCASE.

2. Replace the value for `kdc` "*nameofyourkdc.your.domain*" with the domain controller (Active directory) in your environment, e.g. "*adserver01.mysubnet.mycompany.com*".
3. Replace the domain-mapping "*your.domain*" (in lowercase) with the domain that your Orchestra server resides in, prefixed by a dot (.),

Example: "*.mysubnet.mycompany.com*"

In our example, the file looks like this:

```
[libdefaults]
 default_realm = MYSUBNET.MYCOMPANY.COM
 default_tkt_enctypes = aes128-cts rc4-hmac des3-cbc-sha1 des-cbc-md5 des-cbc-crc
 default_tgs_enctypes = aes128-cts rc4-hmac des3-cbc-sha1 des-cbc-md5 des-cbc-crc
 permitted_enctypes = aes128-cts rc4-hmac des3-cbc-sha1 des-cbc-md5 des-cbc-crc

[realms]
 MYSUBNET.MYCOMPANY.COM = {
 kdc = adserver01.mysubnet.mycompany.com
 default_domain = MYSUBNET.MYCOMPANY.COM
 }

[domain_realm]
 .mysubnet.mycompany.com = MYSUBNET.MYCOMPANY.COM
```

Next, in Orchestra, navigate to the *System Administration* application and the *Parameters* tab. In the SSO section, mark the enabled check box as active.

Enter the user created in AD as *Pre-authentication username*, e.g. "kerberos" and enter the password for the user as *Pre-authentication password*.

Save the changes and restart Orchestra.

❗ For any distributed QueueAgent instances the configuration made in the UI will be enabled upon restart. However, the *krb5.conf* file must be modified for each QueueAgent, or modified in an agent profile and sent to the Queue Agent via remote upgrade. It should be identical to the one defined for the central installation, unless the agent profile configuration has been overridden via remote upgrade.

Once configuration is complete, restart the Orchestra server.

If a misconfiguration is made, SSO can be disabled by editing the database.

Change the table *qp\_central.system\_parameter*. The field value should be set to "false" for the row with ID "aaa.sso.enabled". This can also be done in the *Parameters* tab of the *System Administration* application.

E.g. UPDATE qp\_central.application\_parameters SET value = 'false' WHERE id = 'aaa.sso.enabled';

SSO can also be disabled for an Agent Profile by remote upgrade. Changing the setting Enabled to false in System Administration -> Queue Agents -> Agent Profiles -> Override Global Parameters values -> SSO and performing a remote upgrade disables SSO for every Queue Agent that is currently assigned that profile.

Once the setup is complete, SSO should work.

❗ Once SSO is enabled, there can be issues logging in as *superadmin*, if using a client that is a member of the domain.

The reason is that the client will attempt to log in automatically, not giving the user the option to log in as superadmin.

A workaround exists:

1. Restart the browser to clear any login information.
2. Manually enter the URL into the browser address field: *http://<orchestra host>:<orchestra port>/login.jsp?failMessage=error.login.noApplications*

Example:

*http://testrd.test.q-matic.net:8080/  
login.jsp?failMessage=error.login.noApplications*

An error message is shown "User has no applications.", this can be disregarded and log on as *superadmin* should be possible.

## SSO Setup Using SAML 2.0 Web SSO

### Introduction

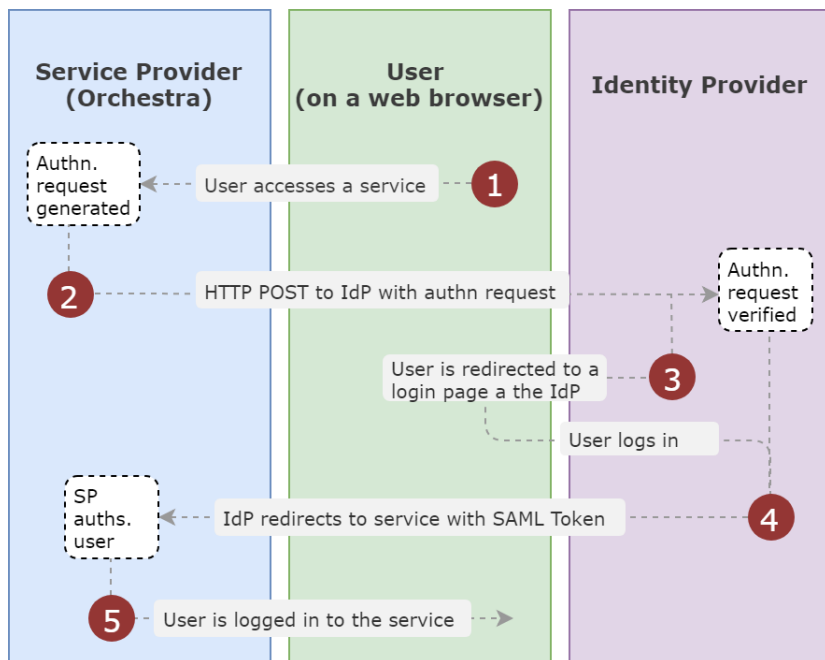
SAML 2.0 Web SSO is a standard that enables users defined in an external directory to log in to Orchestra without configuring the users beforehand. This means that no users need to be added into Orchestra.

It also allows the user a single-sign-on experience, as long as they have previously logged in to the external directory.

There are three core actors involved in SAML 2.0 Web SSO:

- The Identity Provider (IdP): The external directory that stores and provides the authentication information (user accounts, passwords, access rights etc.) and authenticates users. This could be for example Entra ID, Microsoft Active Directory Federated Services (ADFS), Keycloak or Okta.
- The Service Provider (SP): The service that needs the authentication information from the Identity Provider. In this case it's Orchestra.
- User: The end user (eg. the staff member) that tries to log in using a web browser.

In the picture below you can see how the SAML flow works.



- 1 A user tries to access an Orchestra page.
- 2 If the user is not authenticated in Orchestra, the user is redirected to the configured IdP.
  - a) If the user is *not* authenticated at the IdP, -> step 3
  - b) If the user *is* authenticated at the IdP -> step 4
- 3 The IdP redirects the user to the login page and the user logs in.
- 4 The user is redirected back to Orchestra with a SAML Token containing information about the user and its access rights.
- 5 Orchestra checks the SAML Token, authenticates the user, and gives the user access to the page he/she was trying to access in step 1.

### Restrictions and prerequisites

- It is not possible to set up SAML Web SSO in Orchestra, when either LDAP, SSO (Kerberos) or PreAuth is enabled.
- To use SAML, you need an identity provider, such as Microsoft ADFS or Entra ID. If you don't already have an identity provider, a third-party identity provider can be used.
- To use SAML 2.0 Web SSO, HTTPS must be enabled in Orchestra and for all distributed agents. Otherwise there is a risk that the authorization tokens are compromised.

### Configure SAML

To configure Orchestra to use SAML 2.0 Web SSO, the following main steps need to be completed:

1. Configure Orchestra and the identity provider to trust each other with the proper redirect links.
2. Assign different access rights to the users in the Identity Provider.
3. Configure single logout.

For a more detailed instruction about how to configure SAML 2.0 Web SSO, see "Appendix E - Set up SAML" on page 253.

# 13. HA setup

## Topics in this chapter

Introduction .....	214
Deployment architecture .....	215
Load balancer .....	215
Shared storage .....	217
RDBMS .....	217
Deployment procedure .....	217
Notification application .....	218
HA Proxy Example .....	219

## Introduction

This chapter details architecture and deployment for implementing Orchestra in an HA (High Availability) configuration.

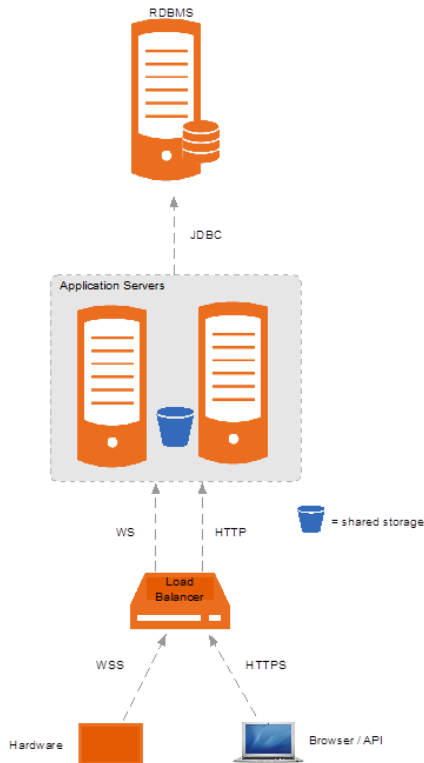
### **Supported environments – Orchestra and stat**

The following set-ups are supported:

- Hot/Cold – both servers are running but Orchestra service is only started on one of them. Supported for all scenarios.
- Hot standby– Orchestra service is started on both servers, but all traffic is directed to one of them. This is only supported when all Queue Agents are distributed, i.e when there are no central branches.

ⓘ The descriptions below only cover Hot/Cold configuration.

## Deployment architecture



- Multiple application servers
- Shared storage and RDBMS between application servers
- Load Balancer that supports websocket communication
- Hot/Cold configuration with all traffic directed to primary, failover to secondary

## Load balancer

A Load Balancer / reverse proxy is installed in front of the application servers and serves, as the main connection point for all communication to Orchestra.

The Load Balancer must be configured with the Orchestra application servers in a hot / cold setup. Whereby all traffic is directed to the primary instance and the secondary is used as a backup in case the primary fails.

It is recommended to use the same ports (e.g. 8080 for http traffic) in the Load Balancer frontend as in the Orchestra backend.

To support Qmatic hardware devices that communicate with the Queue Agent directly (e.g. GW1745, Intro 17), the Load Balancer will need to support the websocket protocol and be capable of directing websocket traffic to the backend instances, either at layer 7 or at least layer 4 TCP.

To prevent a SPOF (single point of failure) in the architecture, the Load Balancer / reverse proxy should be deployed in an HA configuration also. The setup of this configuration is specific for the selected HA solution and is not in the scope of this manual.

If HTTPS is required, it is advised that SSL be terminated at the Load Balancer and proxy HTTP to the application servers to offload the SSL processing.

The following HTTP headers should be configured at the Load Balancer to send to Orchestra application servers:

Header	Value	Comments
X-Forwarded-Proto	https	Use only when offloading SSL at Load Balancer.
X-Forwarded-For	<default>	Identifies the source client IP address.
Proxy-IP	Same as X-Forwarded-For	Used by Glassfish instead of 'X-Forwarded-For' in some places.

See “HA Proxy Example” on page 219, for an example configuration with a popular open source Load Balancer. In the example configuration for HAProxy is included a frontend for terminating SSL traffic at the Load Balancer.

## Shared storage

It is recommended that the following directories are synchronised between the application servers:

```
/conf
/media
/custdeploy
/deploy/wookie.war/wservices
```

Synchronisation can be either a manual process, or automatic, by way of shared storage mechanisms. This will be a design choice, based on the requirements and system usage. For example, the contents of the */conf* folder may never (or rarely) be touched on a production system, once deployed, and can be copied manually. If widgets or custom applications are not utilised then the */deploy/wookie.war/wservices* and */custdeploy* directories do not need to be synchronised.

To utilise shared storage on the application server instances, NAS (Network Attached Storage) should be mounted to the same location on both servers and symbolically link the required folders to this mount point (using *mklink* to create symbolic links on Windows file systems).

❗ Symbolic links on Windows for the directory */media* are only supported on Orchestra 6, Hotfix1, and later.

Alternatively, on Windows, DFS replication can be used to synchronise the directories.

## RDBMS

Both application servers use the same database instance.

To prevent a SPOF in the architecture, the RDBMS should be deployed in an HA configuration also as per the vendors specifications (e.g. Oracle RAC, MSSQL Clustering).

## Deployment procedure

### Prerequisites

The following ports need to be opened up in the firewall:

- 40010
- 40020

For more information on how to open up ports, see “Enabling Ports in the Firewall” on page 15.

## Procedure

1. Setup databases using scripts provided with Orchestra installation.
2. Install Orchestra on one application server.
3. Open the following files, located under `<orchestra install dir>/conf/`:
  - `ehcache-central.xml`
  - `ehcache-calendar.xml`
4. Follow the instructions in each file and enable the applicable sections ("*Enable this section when using HA mode*"). Also, replace the `<ip-host-A>` and `<ip-host-B>` parameters with the ip addresses of your nodes.
5. Test installation.
6. Adjust JMS bridge configuration. For this, follow the instructions in "Installation Note - Configuration of Stat on a separate server" on page 24.
7. Either clone existing application server, or install Orchestra again on second server, pointing to the same databases. The second installation must be made to the exact same path as the first.
8. Test second installation.
9. Install and configure Load Balancer / reverse proxy.
10. Configure both Orchestra systems with the address to the Load Balancer / reverse proxy, where necessary:
  - The JVM parameter **agent.ipAddress** in the startscript (`standalone.conf.bat` on Windows and `standalone.conf` on Linux) for the Wildfly application server is used to configure certain hardware, e.g. Cinematic, and should be the address to the Load Balancer /reverse proxy. This file is located in `<Orchestra install>\app\<application server>\bin\`.
  - All connected hardware should connect to the Load Balancer. If hardware was connected to the system before the Load Balancer was set up, it is necessary to change that configuration both in the hardware and in Orchestra
  - Test access via Load Balancer from browser.
11. Test access via Load Balancer for hardware.
12. Test failover, stop Orchestra service on primary, confirm users re-directed to secondary. (Users will have to re-login, as session data is not replicated. Hardware should loose connection and re-connect automatically).

## Notification application

❗ If Orchestra is deployed in a clustered environment, the `"org.quartz.jobStore.isClustered = false"` flag needs to be set to `true`, in the `notificationQuartz.properties` file, found in the Orchestra `conf` folder.

## HA Proxy Example

```

global
 daemon
 nbproc 1
 user haproxy
 group haproxy
 log 127.0.0.1:514 local0
 pidfile /var/run/haproxy.pid
 stats socket /var/run/haproxy.stat mode 777

 tune.comp.maxlevel 5
 maxcompcpuusage 98
 maxconn 30000
 spread-checks 5

defaults
 mode http
 log global

 option httplog
 option http-server-close
 option dontlognull
 option redispatch
 option splice-auto
 option clitcpka
 option srvtcpka
 option tcp-smart-accept
 option tcp-smart-connect
 option contstats

 retries 3

 timeout http-request 5s
 timeout http-keep-alive 5s
 timeout connect 5s
 timeout client 25s
 timeout client-fin 30s
 timeout tunnel 1h
 timeout server 25s
 timeout queue 10s
 timeout tarpit 15s

 backlog 10000

 compression algo gzip
 compression type text/html text/html;charset=utf-8 text/
html;charset=ISO-8859-1 text/plain text/css text/javascript
application/x-javascript application/javascript application/
ecmascript application/rss+xml application/atomsvc+xml
application/atom+xml application/atom+xml;type=entry application/

```

```
atom+xml;type=feed application/cmismquery+xml application/
cmisallowableactions+xml application/cmismatom+xml application/
cmistree+xml application/cmismacl+xml application/msword
application/vnd.ms-excel application/vnd.ms-powerpoint

errorfile 408 /dev/null # workaround Chrome pre-connect bug

frontend http
 bind *:80

 maxconn 1000

 http-request set-header X-Forwarded-For %[src]
 option forwardfor header Proxy-ip

 rspidel ^Server:.*
 rspidel ^X-Powered-By:.*

 acl is_ws path_beg /wsdc hdr(Upgrade) -i websocket
 acl is_comet path_beg /cometd/

 use_backend bk_comet if is_comet
 use_backend bk_ws if is_ws
 default_backend bk_web

#Optional example frontend for terminating SSL at the load balancer
frontend https
 mode http
 option forwardfor
 bind *:8443 ssl crt <path_to_pem_file>
 reqadd X-Forwarded-Proto:\ https
 rspiprep ^Location:\ http:(.*) Location:\ https:\1

 acl is_ws path_beg /wsdc hdr(Upgrade) -i websocket
 acl is_comet path_beg /cometd/

 use_backend bk_comet if is_comet
 use_backend bk_ws if is_ws
 default_backend bk_web

backend bk_web
 option httpchk GET /ping.html HTTP/1.1\r\nHost:\ www

 acl is_static capture.req.uri -m end .css .png .jpg .gif .js
 .favicon.ico

 rspadd Cache-Control:\ public if is_static

 server oas1 10.0.41.172:8080 check
 server oas2 10.0.47.106:8080 check backup

backend bk_comet
```

```
timeout server 120s

option httpchk GET /ping.html HTTP/1.1\r\nHost:\ www

cookie comet insert

server oas1 10.0.41.172:8080 check
server oas2 10.0.47.106:8080 check backup

backend bk_ws

 server oas1 10.0.41.172:8888 check on-marked-up shutdown-backup-
sessions
 server oas2 10.0.47.106:8888 check backup

listen stats :9090
mode http
no log

stats enable
stats refresh 15s
stats uri /stats
```



# Appendix A - Tuning Database Parameters

Topics in this Appendix

Tuning of Oracle Parameters .....	224
Tuning of PostgreSQL Parameters .....	225

## Tuning of Oracle Parameters

Depending on the size of the Orchestra installation and the number of database connections set in the different database connection pools in the Orchestra application server, Oracle parameters may have to be tuned.

The database must be able to cope with the sum of all database connection pools in Orchestra.

The three most commonly tuned parameters are:

- PROCESSES
- SESSIONS

Should match number of configured connections in the Orchestra database connection pools.

Oracle recommend setting it to  $\text{PROCESSES} \times 1.1 + 5$

- TRANSACTIONS

Oracle recommends setting it to  $\text{SESSIONS} \times 1.1$

### Example:

```
ALTER SYSTEM SET PROCESSES=1200 SCOPE=SPFILE;
ALTER SYSTEM SET TRANSACTIONS=1458 SCOPE=SPFILE;
ALTER SYSTEM SET SESSIONS=1325 SCOPE=SPFILE;
```

## Additional parameters to review

### Oracle memory

- MEMORY\_TARGET
- MEMORY\_MAX\_TARGET

### Example:

```
ALTER SYSTEM SET MEMORY_TARGET=8000M SCOPE=SPFILE;
ALTER SYSTEM SET MEMORY_MAX_TARGET=10000M SCOPE=SPFILE;
```

## Adaptive log file sync

Oracle 11 introduced an new mechanism to communicate with the log writer, a part of the commit-process. Several Oracle database administrators have experienced a considerable performance gain when they have disabled this new "adaptive log file synchronization".

```
ALTER SYSTEM SET "_use_adaptive_log_file_sync"= false;
```

## Cursor sharing

Qmatic Orchestra gains additional performance if cursor sharing is set to

```
ALTER SYSTEM SET CURSOR_SHARING='SIMILAR' SCOPE=SPFILE;
```

## Tuning of PostgreSQL Parameters

PostgreSQL is delivered with default configuration values designed to operate across a wide range of systems with varying hardware resources. These defaults represent conservative baseline settings intended to ensure reliable operation in most environments.

This section highlights selected parameters that may be adjusted in installations where additional system resources—particularly memory—are available.

❗ **Any changes should be made with a clear understanding of the purpose and impact of each parameter. Improper tuning may lead to degraded performance or increased resource consumption.**

Before modifying any PostgreSQL configuration, refer to the official PostgreSQL documentation for detailed descriptions of each setting:

- <https://www.postgresql.org/docs/16/runtime-config-resource.html>
- <https://www.postgresql.org/docs/16/runtime-config-query.html>

## Estimating PostgreSQL settings

The website <https://pgtune.leopard.in.ua/> provides a configuration calculator that can be used as a starting point for estimating suitable values based on the total amount of available system RAM and CPU resources.

Use the following settings for the calculator:

- Set **DB Version** to 16 if you are using the bundled PostgreSQL server included with Orchestra, or to the version that matches your externally installed PostgreSQL instance.
- Set **DB Type** to “Online transaction processing system”.
- For installations using the bundled PostgreSQL server with all Orchestra components enabled, set **Number of connections** to 1692, which corresponds to the default configuration derived from the combined connection pool sizes. This value is generally sufficient for small to medium-sized installations.

❗ In larger environments, particularly those with a centralized Queue Agent, it may be appropriate to increase the number of allowed connections. Conversely, if not all Orchestra components are installed, or in smaller deployments, reducing the number of connections may be advisable to limit unnecessary resource consumption.

## Adjusting PostgreSQL configuration parameters

These are the main parameters to adjust (values are based on a Linux system running PostgreSQL version 16, with 13 GB of available memory, `max_connections` set to 1692, database type set to OLTP, and disk type set to SSD).

❗ Increasing memory-related parameters and cache sizes does not imply that PostgreSQL will consume that amount of memory at all times; actual memory usage depends on workload and runtime behavior.

If PostgreSQL is running co-located with Orchestra (which is not recommended for production installations), keep in mind that not all system memory will be available to the database. The Orchestra application processes will also consume memory resources.

❗ Using mechanical disks instead of SSD or SAN-based storage will result in different recommendations for parameters such as `random_page_cost` and `effective_io_concurrency`.

The default values shown in the table apply to a standard PostgreSQL 16 installation. When using the bundled PostgreSQL server included with Orchestra, the default value for `max_connections` is not 100, but instead depends on which Orchestra components are installed.

The number of database connections allocated per component is as follows:

- Central: 596 (64 for `qp_central`, 32 for `qp_app`, 500 for `qp_agent`)
- Calendar: 64
- Statistics: 1000
- Auditing: 32

If all components are installed, the total number of connections is 1692.

**Table 1: Changes to postgresql.conf.**

Parameter	New value (Default)	Description
<code>max_connections</code>	100 (100 / 1692)	<b>Do not change this parameter without consulting the manual.</b> This parameter works in conjunction with <code>work_mem</code> and other memory-related settings, and incorrect tuning may lead to excessive memory usage. If this value is increased, make sure to reduce <code>work_mem</code> accordingly.
<code>shared_buffers</code>	3328 MB (128 MB)	A general recommendation is to allocate approximately 25% of available system memory to this setting, although some deployments see limited benefit. In many cases, <code>effective_cache_size</code> has a greater impact on performance.
<code>effective_cache_size</code>	9984 MB (4GB)	Estimate this value based on available free RAM and the system's cache usage. Higher values favor index usage and can significantly improve query planning. Some installations set this close to the amount of available free memory.
<code>maintenance_work_mem</code>	832 MB (64 MB)	Controls the memory available for maintenance operations such as VACUUM and CREATE INDEX. These operations occur infrequently and typically only once per session, so a relatively high value can be beneficial. Note that autovacuum may use up to three times this value, controlled by <code>autovacuum_max_workers = 3</code> .

**Table 1: Changes to postgresql.conf.**

Parameter	New value (Default)	Description
wal_buffers	16 MB (-1: calculated automatically)	Controls the amount of memory used for write-ahead log buffers. When set to -1, PostgreSQL automatically determines an appropriate value based on other memory settings.
random_page_cost	1.1 (4)	If a large portion of the database can be cached in memory, and/or if the database uses SSD storage, lowering this value may improve query planning and performance.
effective_io_concurrency	200 (1)	Controls how many concurrent I/O operations PostgreSQL expects the system to handle. Increasing this value can improve performance on systems with fast storage and large databases, but excessively high values may lead to I/O bottlenecks.
work_mem	2 MB (4 MB)	Specifies the amount of memory used by a query operation before temporary files are written to disk. This setting must be tuned carefully, as multiple concurrent queries can multiply memory usage. If <code>max_connections</code> is reduced, this value can often be increased safely.
min_wal_size	2 GB (80 MB)	Specifies the minimum size of the write-ahead log. Increasing this value helps avoid frequent creation and deletion of WAL files in systems with high or bursty workloads, which is common in Orchestra deployments.
max_wal_size	8 GB (1 GB)	Defines the maximum size of the write-ahead log before a checkpoint is triggered. Increasing this value can reduce checkpoint frequency in high-load systems.

**Restart requirements and verification**

After modifying `postgresql.conf`, restart the PostgreSQL service to apply the changes. Reloading the configuration is not sufficient, as several of the parameters described in this section require a full restart to take effect.

To verify the current PostgreSQL configuration, run the following query:

```
SHOW ALL;
```

This command returns all active configuration parameters and their effective values.



# Appendix B - Logging

Topics in this Appendix

Enabling Customer Journey Management Logging ..... 232

Enabling Publish Logging ..... 233

## Enabling Customer Journey Management Logging

Add a new file appender for the Customer Journey Management logging.

In `<orchestra install dir>/conf/logback.xml`, add the following element just above the line with `<logger name="com.sun.enterprise.server.logging.GFFFileHandler" level="INFO"/>`:

```
<appender name="cfmAppender"
class="com.qmatic.qp.logging.QPRollingFileAppender">
 <file>${com.sun.aas.instanceRoot}/../../../../../../logs/
cfm.log</file>
 <rollingPolicy
class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
 <fileNamePattern>${com.sun.aas.instanceRoot}/../../../../
../../../../logs/cfm-%d{yyyy-MM-dd}_%d{HHmmss}.%i.log</fileNamePattern>
 <timeBasedFileNamingAndTriggeringPolicy
class="com.qmatic.qp.logging.StartupTimeAndSizeBasedTriggeringPoli
cy">
 <maxFileSize>10MB</maxFileSize>
 <maxBackupIndex>10</maxBackupIndex>
 </timeBasedFileNamingAndTriggeringPolicy>
 </rollingPolicy>
 <encoder>
 <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level
%logger{36} - %msg%n</pattern>
 </encoder>
 <filter
class="ch.qos.logback.classic.filter.ThresholdFilter">
 <!-- In order to DEBUG change this filter level, along
with required components below -->
 <level>DEBUG</level>
 </filter>
</appender>
```

Around line 160, replace the line:

```
<logger name="com.qmatic.qp.jiql.core.cfm" level="INFO"/>
```

with these lines:

```
<logger name="com.qmatic.qp.jiql.core.cfm" level="DEBUG"
 additivity="false">
<appender-ref ref="cfmAppender"/>
</logger>
```

Save the file, it will take effect within a minute.

## Enabling Publish Logging

Add a new file appender for the publish logging.

In `<orchestra install dir>/conf/logback.xml`, add the following element just above the line with `<logger name="com.sun.enterprise.server.logging.GFFFileHandler" level="INFO"/>`:

```
<appender name="publishAppender"
class="com.qmatic.qp.logging.QPRollingFileAppender">
 <file>${com.sun.aas.instanceRoot}/../../../../../../logs/
publish.log</file>
 <rollingPolicy
class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
 <fileNamePattern>${com.sun.aas.instanceRoot}/../../../../
../../../../logs/publish-%d{yyyy-MM-dd}_%d{HHmmss}.%i.log</
fileNamePattern>
 <timeBasedFileNamingAndTriggeringPolicy
class="com.qmatic.qp.logging.StartupTimeAndSizeBasedTriggeringPoli
cy">
 <maxFileSize>10MB</maxFileSize>
 <maxBackupIndex>10</maxBackupIndex>
 </timeBasedFileNamingAndTriggeringPolicy>
 </rollingPolicy>
 <encoder>
 <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level
%logger{36} - %msg%n</pattern>
 </encoder>
 <filter
class="ch.qos.logback.classic.filter.ThresholdFilter">
 <!-- In order to DEBUG change this filter level, along
with required components below -->
 <level>DEBUG</level>
 </filter>
</appender>
```

Around line 150, replace the this line:

```
<logger name="com.qmatic.qp.jiql.core.cm" level="INFO"/>
```

with these lines:

```
<logger name="com.qmatic.qp.jiql.core.cm" level="DEBUG"
 additivity="false">
<appender-ref ref="publishAppender"/>
</logger>
```

Save the file, it will take effect within a minute.



# Appendix C - Security

## Topics in this Appendix

Introduction .....	236
Content-Security-Policy .....	236
X-Frame-Options .....	236
X-Content-Type-Options .....	237
X-XSS-Protection .....	237
Strict-Transport-Security .....	238
Modifying HTTP Response Headers .....	238

## Introduction

The following HTTP response headers are set by default in Orchestra.

### Content-Security-Policy

The HTTP Content-Security-Policy response header allows web site administrators to control resources the user agent is allowed to load for a given page. With a few exceptions, policies mostly involve specifying server origins and script endpoints. This helps guard against cross-site scripting attacks ([XSS](#)).

#### Syntax

```
Content-Security-Policy: <policy-directive>; <policy-directive>
```

#### Orchestra default configuration

```
Content-Security-Policy: *
```

### X-Frame-Options

The X-Frame-Options HTTP response header can be used to indicate whether or not a browser should be allowed to render a page in a <frame>, <iframe> or <object>. Sites can use this to avoid [clickjacking](#) attacks, by ensuring that their content is not embedded into other sites.

The added security is only provided if the user accessing the document is using a browser supporting X-Frame-Options.

#### Syntax

There are three possible directives for X-Frame-Options:

```
X-Frame-Options: DENY
```

```
X-Frame-Options: SAMEORIGIN
```

```
X-Frame-Options: ALLOW-FROM https://example.com/
```

#### Orchestra default configuration

```
X-Frame-Options: SAMEORIGIN
```

## X-Content-Type-Options

The X-Content-Type-Options response HTTP header is a marker used by the server to indicate that the MIME types advertised in the Content-Type headers should not be changed and be followed. This allows to opt-out of MIME type sniffing.

### Syntax

```
X-Content-Type-Options: nosniff
```

### Orchestra default configuration

```
X-Content-Type-Options: nosniff
```

## X-XSS-Protection

The HTTP X-XSS-Protection response header is a feature of Internet Explorer, Chrome and Safari that stops pages from loading when they detect reflected cross-site scripting (XSS) attacks. Although these protections are largely unnecessary in modern browsers when sites implement a strong Content-Security-Policy that disables the use of inline JavaScript ('unsafe-inline'), they can still provide protections for users of older web browsers that don't yet support CSP.

### Syntax

```
X-XSS-Protection: 0
X-XSS-Protection: 1
X-XSS-Protection: 1; mode=block
X-XSS-Protection: 1; report=<reporting-uri>
```

### Orchestra default configuration

```
X-XSS-Protection: 1
```

Explanation: Enables XSS filtering (usually default in browsers). If a cross-site scripting attack is detected, the browser will sanitize the page (remove the unsafe parts).

## Strict-Transport-Security

The **HTTP** Strict-Transport-Security response header (often abbreviated as HSTS) is a security feature that lets a web site tell browsers that it should only be communicated with using HTTPS, instead of using HTTP.

### Syntax

```
Strict-Transport-Security: max-age=<expire-time>
Strict-Transport-Security: max-age=<expire-time>; includeSubDomains
Strict-Transport-Security: max-age=<expire-time>; preload
```

### Orchestra default configuration

```
Strict-Transport-Security: max-age=31536000; includeSubDomains
```

Explanation: All present and future subdomains will be HTTPS for a max-age of 1 year. This blocks access to pages or sub domains that can only be served over HTTP.

## Modifying HTTP Response Headers

The default response headers are configured in the application server configuration file and can be modified.

### Wildfly

On Wildfly application servers, the configuration file is `<Orchestra>/system/app/wildfly-x.y.z.Final/standalone/configuration/standalone-full.xml`

Locate the following section and modify the *responseHeaderValue* parameters (in bold below).

To disable a response-header, comment out or remove the entire `<response-header>` tag together with the corresponding `<filter-ref>` tag found a few lines above the `<response-header>`.

Save the file and restart Orchestra to apply the changes.

```
<filters>
 <response-header name="server-header" header-name="Server"
header-value="WildFly/8"/>
 <response-header name="x-powered-by-header" header-name="X-
Powered-By" header-value="Undertow/1"/>
 <response-header name="xContentTypeOptions" header-name="X-
Content-Type-Options" header-value="nosniff"/>
 <response-header name="contentSecurityPolicy" header-
name="Content-Security-Policy" header-value="*" />
```

```
<response-header name="xXssProtection" header-name="X-XSS-
Protection" header-value="1"/>
<response-header name="strictTransportSecurity" header-
name="Strict-Transport-Security" header-value="max-age=31536000;
includeSubDomains"/>
<response-header name="xFrameOptions" header-name="X-Frame-
Options" header-value="SAMEORIGIN"/>
<filter name="qmatic-http-method-white-list" class-
name="com.qmatic.httpfilter.UndertowWhiteList" module="qmatic-
httpfilter-lib"/>
</filters>
```

## HTTP Host Referer Guard

The HTTP Host Referer Guard has a parameter called *approvedRefererHostnames* which is a comma separated list of host names that are excluded from the test in the HTTP Host Referer Guard.

The guard has been given a default parameter value of *cf.qmatic.cloud* in the file *standalone-full.xml*. Add to that value, e.g use *cf.qmatic.cloud, anotherhostname* instead of *cf.qmatic.cloud*, if you want to.

The syntax for Wildfly is described below.

## Wildfly

```
<filter name="qmatic-referer-host-http-guard" class-
name="com.qmatic.httpfilter.HttpHostRefererGuard" module="qmatic-
httpfilter-lib">
 <param name="approvedRefererHostnames" value="cf.qmatic.cloud"/>
</filter>
```



# Appendix D - Open LDAP Setup

## Topics in this Appendix

Introduction .....	242
Install LDAP and Helper Utilities .....	242
Reconfigure slapd to Select Better Settings .....	242
Install phpLDAPadmin to Manage LDAP with a Web Interface .....	243
Configure phpLDAPadmin .....	243
Create an SSL Certificate .....	244
Create a Password Authentication File .....	245
Secure Apache .....	245
Modify the phpLDAPadmin Apache Configuration .....	246
Configure the HTTP Virtual Host .....	246
Configure the HTTPS Virtual Host File .....	247
Log in to the phpLDAPadmin Web Interface .....	248
Adding Nodes .....	249
Adding a User .....	251
Adding a Group .....	251
Troubleshooting .....	252

## Introduction

LDAP, or Lightweight Directory Access Protocol, is a protocol designed to manage and access related information in a centralized, hierarchical file and directory structure.

In some ways, it operates similarly to a relational database, but this does not hold true for everything. The hierarchical structure is the main difference in how the data is related. It can be used to store any kind of information and it is often used as one component of a centralized authentication system.

In this appendix, we will discuss how to install and configure an *OpenLDAP* server on an *Ubuntu 14.04* server. We will then install and secure a *phpLDAPadmin* interface to provide an easy web interface.

## Install LDAP and Helper Utilities

Before we begin, we must install the necessary software. The packages are all available in Ubuntu's default repositories. This is our first time using apt in this session, so we will refresh our local package index. Afterwards, we can install the packages we want:

```
sudo apt-get update
sudo apt-get install slapd ldap-utils
```

During the installation, you will be asked to select and confirm an administrator password for LDAP. You can actually put anything here because you will have the opportunity to change it, in just a moment.

## Reconfigure slapd to Select Better Settings

Even though the package was just installed, we are going to reconfigure the defaults that Ubuntu installs with.

The reason for this is that while the package has the ability to ask a lot of important configuration questions, these are skipped over in the installation process. We can gain access to all of the prompts, by telling our system to reconfigure the package:

```
sudo dpkg-reconfigure slapd
```

There are quite a few new questions that will be asked as you go through this process. Let's go over these (and their answers) now:

1. *Omit OpenLDAP server configuration?* **No**
2. *DNS domain name?* This option will determine the base structure of your directory path. Read the message to understand exactly how this will be implemented. This is actually a rather open option. You can select whatever *domain name* value you would like, even if you do not own the actual domain.

However, if you have a domain name for the server, it is probably wise to use that. For this guide, we are going to select *test.com* for our configuration.

3. *Organization name?* This is, again, pretty much entirely up to your preferences. For this guide, we will be using *example* as the name of our organization.
4. *Administrator password?* As mentioned in the installation section, this is your real opportunity to select an administrator password. Anything you select here will overwrite the previous password you used.
5. *Database backend?* **HDB**
6. *Remove the database when slapd is purged?* **No**
7. *Move old database?* **Yes**
8. *Allow LDAPv2 protocol?* **No**

At this point, your LDAP should be configured in a fairly reasonable way.

## Install phpLDAPAdmin to Manage LDAP with a Web Interface

Although it is very possible to administer LDAP through the command line, most users will find it easier to use a web interface. We are going to install *phpLDAPAdmin*, which provides this functionality, to help remove some of the friction of learning the LDAP tools.

The Ubuntu repositories contain the *phpLDAPAdmin* package. You can install it by typing:

```
sudo apt-get install phpldapadmin
```

This should install the administration interface, enable the necessary Apache virtual hosts files, and reload Apache.

The web server is now configured to serve your application, but we will make some additional changes. We need to configure *phpLDAPAdmin* to use the domain schema we configured for LDAP, and we are also going to make some adjustments to secure our configuration a little bit.

## Configure phpLDAPAdmin

Now that the package is installed, we need to configure a few things so that it can connect with the LDAP directory structure that was created during the OpenLDAP configuration stage.

Begin by opening the main configuration file with root privileges in your text editor:

```
sudo nano /etc/phpldapadmin/config.php
```

In this file, we need to add the configuration details that we set up for our LDAP server. Start by looking for the host parameter and setting it to your server's domain name or public IP address. This parameter should reflect the way you plan on accessing the web interface:

```
$servers->setValue('server','host','server_domain_name_or_IP');
```

Next, you will need to configure the domain name you selected for your LDAP server. Remember, in our example we selected *test.com*. We need to translate this into LDAP syntax by replacing each domain component (everything not a dot) into the value of a *dc* specification.

All this means is that instead of writing *test.com*, we will write something like:

```
dc=test,dc=com
```

We should find the parameter that sets the server base parameter and use the format we just discussed to reference the domain we decided on:

```
$servers->setValue('server','base',array('dc=test,dc=com'));
```

We need to adjust the same thing in our login *bind\_id* parameter. The *cn* parameter is already set as *admin*. This is correct. We just need to adjust the *dc* portions again, as we did above:

```
$servers->setValue('login','bind_id','cn=admin,dc=test,dc=com');
```

The last thing that we need to adjust is a setting that controls the visibility of warning messages. By default, *phpLDAPadmin* will throw quite a few annoying warning messages in its web interface about the template files that have no impact on the functionality.

We can hide these by searching for the *hide\_template\_warning* parameter, uncommenting the line that contains it, and setting it to *true*:

```
$config->custom->appearance['hide_template_warning'] = true;
```

This is the last thing that we need to adjust. You can save and close the file when you are finished.

## Create an SSL Certificate

We want to secure our connection to the LDAP server with SSL, so that outside parties cannot intercept our communications.

Since the admin interface is talking to the LDAP server itself on the local network, we do not need to use SSL for that connection. We just need to secure the external connection to our browser when we connect.

To do this, we just need to set up a self-signed SSL certificate that our server can use. This will not help us validate the identity of the server, but it will allow us to encrypt our messages.

The *OpenSSL* packages should be installed on your system, by default. First, we should create a directory to hold our certificate and key:

```
sudo mkdir /etc/apache2/ssl
```

Next, we can create the key and certificate by executing:

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /
etc/apache2/ssl/apache.key -out /etc/apache2/ssl/apache.crt
```

You will have to answer some questions, in order for the utility to fill out the fields in the certificate correctly. The only one that really matters is the prompt that says *Common Name* (e.g. server FQDN or YOUR name). Enter your server's domain name or IP address.

When you are finished, your certificate and key will be written to the */etc/apache2/ssl* directory.

## Create a Password Authentication File

We also want to password protect our *phpLDAPadmin* location. Even though *phpLDAPadmin* has password authentication, this will provide an extra level of protection.

The utility that we need is contained in an Apache utility package. Get it by typing:

```
sudo apt-get install apache2-utils
```

Now that you have the utility available, you can create a password file that will contain a username that you choose and the associated hashed password.

We will keep this in the */etc/apache2* directory. Create the file and specify the username you want to use by typing:

```
sudo htpasswd -c /etc/apache2/htpasswd demo_user
```

Now, we are ready to modify Apache to take advantage of our security upgrades.

## Secure Apache

The first thing we should do is enable the SSL module in Apache. We can do this by typing:

```
sudo a2enmod ssl
```

This will enable the module, allowing us to use it. We still need to configure Apache to take advantage of this, though.

Currently, Apache is reading a file called *000-default.conf* for regular, unencrypted HTTP connections. We need to tell it to redirect requests for our *phpLDAPadmin* interface to our HTTPS interface so that the connection is encrypted.

When we redirect traffic to use our SSL certificates, we will also implement the password file to authenticate users. While we are modifying things, we will also change the location of the *phpLDAPAdmin* interface itself to minimize targeted attacks.

## Modify the phpLDAPAdmin Apache Configuration

The first thing we will do is modify the alias that is set up to serve our *phpLDAPAdmin* files.

Open the file with root privileges in your text editor:

```
sudo nano /etc/phpldapadmin/apache.conf
```

This is the place where we need to decide on the URL location where we want to access our interface. The default is */phpldapadmin*, but we want to change this to cut down on random login attempts by bots and malicious parties.

For this guide, we are going to use the location */superldap*, but you should choose your own value.

We need to modify the line that specifies the Alias. This should be in an *IfModule mod\_alias.c* block. When you are finished, it should look like this:

```
<IfModule mod_alias.c>
 Alias /superldap /usr/share/phpldapadmin/htdocs
</IfModule>
```

When you are finished, save and close the file.

## Configure the HTTP Virtual Host

Next, we need to modify our current Virtual Hosts file. Open it with root privileges in your editor:

```
sudo nano /etc/apache2/sites-enabled/000-default.conf
```

Inside, you will see a rather bare configuration file that looks like this:

```
<VirtualHost *:80>
 ServerAdmin webmaster@localhost
 DocumentRoot /var/www/html
 ErrorLog ${APACHE_LOG_DIR}/error.log
 CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

We want to add information about our domain name or IP address to define our server name and we want to set up our redirect to point all HTTP requests to the HTTPS interface. This will match the alias we configured in the last section.

The changes we discussed will end up looking like this. Modify with your own values:

```
<VirtualHost *:80>
 ServerAdmin webmaster@server_domain_or_IP
 DocumentRoot /var/www/html
 ServerName server_domain_or_IP
 Redirect permanent /superldap https://server_domain_or_IP/
 superldap
 ErrorLog ${APACHE_LOG_DIR}/error.log
 CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Save and close the file when you are finished.

## Configure the HTTPS Virtual Host File

Apache includes a default SSL Virtual Host file. However, it is not enabled by default.

We can enable it by typing:

```
sudo a2ensite default-ssl.conf
```

This will link the file from the sites-available directory into the sites-enabled directory. We can edit this file now by typing:

```
sudo nano /etc/apache2/sites-enabled/default-ssl.conf
```

This file is a bit more complex than the last one, so we will only discuss the changes that we have to make. All of the changes below should go within the *Virtual Host* block in the file.

First, set the *ServerName* value to your server's domain name or IP address again and change the *ServerAdmin* directive as well:

```
ServerAdmin webmaster@server_domain_or_IP
ServerName server_domain_or_IP
```

Next, we need to set the SSL certificate directives to point to the key and certificate that we created. The directives should already exist in your file, so just modify the files they point to:

```
SSLCertificateFile /etc/apache2/ssl/apache.crt
SSLCertificateKeyFile /etc/apache2/ssl/apache.key
```

The last thing we need to do is set up the location block that will implement our password protection for the entire *phpLDAPadmin* installation.

We do this by referencing the location where we are serving the *phpLDAPadmin* and setting up authentication using the file we generated. We will require anyone attempting to access this content to authenticate as a valid user:

```
<Location /superldap>
 AuthType Basic
 AuthName "Restricted Files"
 AuthUserFile /etc/apache2/htpasswd
 Require valid-user
</Location>
```

Save and close the file when you are finished.

Restart Apache to implement all of the changes that we have made:

```
sudo service apache2 restart
```

We can now move on to the actual interface.

## Log in to the phpLDAPadmin Web Interface

We have made the configuration changes we need to the *phpLDAPadmin* software. We can now begin to use it.

We can access the web interface by visiting our server's domain name or public IP address followed by the alias we configured. In our case, this was */superldap*:

*http://<server\_domain\_name\_or\_IP>/superldap*

The first time you visit, you will probably see a warning about the site's SSL certificate.

The warning is just here to let you know that the browser does not recognize the certificate authority that signed your certificate. Since we signed our own certificate, this is expected and not a problem.

Click the *Proceed anyway* button or whatever similar option your browser gives you.

Next, you will see the password prompt that you configured for Apache.

Fill in the account credentials you created with the *htpasswd* command. You will see the main *phpLDAPadmin* landing page.

Click on the *login* link, on the left-hand side of the page.

You will be taken to a login prompt. The login *DN* is like the username that you will be using. It contains the account name under *cn* and the domain name you selected for the server broken into *dc* sections as we described above.

It should be pre-populated with the correct value for the admin account if you configured *phpLDAPadmin* correctly. In our case, this looks like this:

*cn=admin,dc=test,dc=com*

For the password, enter the administrator password that you configured during the LDAP configuration.

You will be taken to the main interface

## Adding Nodes

After installing OpenLDAP, we are ready to create some nodes. When comparing to a relational database, we could compare nodes to a table where we store our records. Of course they are not the same, but this comparison is made so you can have a grasp of what a node is.

We want two nodes:

*Groups node*

*People node*

There is no native application or shell. Instead, *Openldap* comes with a few executables you can use to perform your actions. You feed these executables with files containing the data you want to add. So let's get to it.

Make a file named *add\_nodes.ldif* in your favorite text editor and add the following lines.

```
dn: ou=people,dc=test,dc=com
objectClass: organizationalUnit
ou: People
dn: ou=groups,dc=test,dc=com
objectClass: organizationalUnit
ou: Groups
```

To add these nodes, simply run the following command and give the admin password (that you chose during *slapd* setup) when prompted:

```
ldapadd -x -D cn=admin,dc=test,dc=com -W -f add_nodes.ldif
```

Our base is *dc=test,dc=com* and with this we have just added people and groups nodes.

## Enabling MemberOf

In order to easily and efficiently do queries that enables you to see which users that are part of which groups, we need to set up the feature in *Ldap* that allows us to do this.

To do so, make the following 3 files:

**memberof\_config.ldif**

```
dn: cn=module,cn=config
cn: module
objectClass: olcModuleList
olcModuleLoad: memberof
olcModulePath: /usr/lib/ldap
dn: olcOverlay={0}memberof,olcDatabase={1}hdb,cn=config
objectClass: olcConfig
objectClass: olcMemberOf
objectClass: olcOverlayConfig
objectClass: top
olcOverlay: memberof
olcMemberOfDangling: ignore
olcMemberOfRefInt: TRUE
olcMemberOfGroupOC: groupOfNames
olcMemberOfMemberAD: member
olcMemberOfMemberOfAD: memberOf
```

**refint1.ldif**

```
dn: cn=module{1},cn=config
add: olcmoduleload
olcmoduleload: refint
```

**refint2.ldif**

```
dn: olcOverlay={1}refint,olcDatabase={1}hdb,cn=config
objectClass: olcConfig
objectClass: olcOverlayConfig
objectClass: olcRefintConfig
objectClass: top
olcOverlay: {1}refint
olcRefintAttribute: memberof member manager owner
```

To set up the *memberof* module and configure it, run this command:

```
sudo ldapadd -Q -Y EXTERNAL -H ldapi:/// -f memberof_config.ldif
```

To load and configure the *refint* module:

```
sudo ldapmodify -Q -Y EXTERNAL -H ldapi:/// -f /tmp/refint1.ldif
sudo ldapadd -Q -Y EXTERNAL -H ldapi:/// -f /tmp/refint2.ldif
```

❗ Every group created before this module is enabled has to be deleted and remade, in order for these changes to take effect. LDAP assigns a *member* attribute, behind the scenes, to existing users, when creating a group.

## Adding a User

Before we add a user, we first need to generate its password hash:

```
slappasswd -h {SHA} -s my_secret_password
```

yielding this result:

```
{SHA}M6XDJwA47cNw9gm5kXV1uTQuMoY=
```

We will use this result when creating our user file. Make the following file and name it **`add_user.ldif`**

```
dn: uid=john,ou=people,dc=test,dc=com
cn: John Doe
givenName: John
sn: Doe
uid: john
uidNumber: 5000
gidNumber: 10000
homeDirectory: /home/john
mail: john.doe@example.com
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
loginShell: /bin/bash
userPassword: {SHA}M6XDJwA47cNw9gm5kXV1uTQuMoY=
```

And add it by running:

```
ldapadd -x -D cn=admin,dc=test,dc=com -W -f add_user.ldif
```

## Adding a Group

Create the file:

### **add\_group.ldif**

```
dn: cn=mygroup,ou=groups,dc=example,dc=com
objectClass: groupofnames
cn: mygroup
description: All users
member: uid=john,ou=people,dc=test,dc=com
```

where you add a **`"member : user_dn"`** line for each user you want to add to this group. And then run:

```
ldapadd -x -D cn=admin,dc=test,dc=com -W -f add_group.ldif
```

## Taking it for a test-run

You can run the following command to see if it is all set up properly:

```
ldapsearch -x -LLL -H ldap:/// -b uid=john,ou=people,dc=test,dc=com
dn memberof
```

And it should yield this result:

```
dn: uid=john,ou=People,dc=test,dc=com
memberOf: cn=mygroup,ou=groups,dc=test,dc=com
```

## Troubleshooting

**Problem:** Orchestra fails to find users in the configured OpenLDAP.

**Solution:** If the central database is MSSQL, the key used for user group data needs to be changed to lower case. Run the following query in SQL Server Management studio:

```
UPDATE [qp_central].[qp_central].[system_parameter] SET value = 'cn'
WHERE parameter_definition_id = 'aaa.ldap.roleAttributeName'
```

**Problem:** I am not allowed to enter an ldap server with a hyphen in its DNS name in the ldap system parameter.

**Solution:** This is a known problem in systems that were installed before Orchestra 6, update 2. Run the following query to fix the issue:

```
UPDATE qp_central.system_parameter_definitions SET validation ''
WHERE id = 'aaa.ldap.urls'
```

# Appendix E - Set up SAML

Topics in this appendix

Introduction .....	254
Configure trust between Orchestra and the identity provider ...	254
Assign users different access rights .....	261
Configure single logout .....	266
Troubleshooting .....	268

## Introduction

This appendix contains a detailed instruction for setting up SAML 2.0 Web SSO. For more info about SAML, see “SSO Setup Using SAML 2.0 Web SSO” on page 211.

ⓘ Note that you need an identity provider such as Entra ID ready, and that Orchestra (including distributed agents) needs to be configured for HTTPS before you continue setting up SAML.

ⓘ The Orchestra parameters for SAML are found in System Administration > Parameters > SAML v2 Web Single Sign On Settings. Make sure SAML is enabled here.

ⓘ The different URLs for Orchestra are as follows:

- On prem: <hostname>:port (Default port is 8443 but may vary)
- Managed Services: <clientname>.qmatic.cloud.

## Configure trust between Orchestra and the identity provider

All identity providers have their own instructions on how to create and register an application. Set up the identity provider following those instructions. Below are the specific settings for configuring Orchestra in Entra ID and ADFS.

ⓘ It is important that you configure the parameter Service Provider Entity ID correctly in Orchestra. This is the identifier and can be called different things in different identity providers, such as Audience URI, Entity ID or Client ID.

### Configure Orchestra and Entra ID to trust each other

Follow the steps in this section if you are using Entra ID. This procedure is similar in other identity providers as well so you can use this guide as help.

1. In Entra ID, select Entra ID > Enterprise Applications
2. Click New application and add a Non-gallery application. Select Single Sign On and type SAML.
3. You will be asked to fill in the following fields:

❶ *Identifier (Entity ID)* is what in Orchestra is called Service Provider Entity ID. What is configured as Entity ID in Entra ID and in Orchestra should be the same. In Orchestra, this is found in System Administration > Parameters. In other identity providers, this could be called e.g. Audience URI, Entity ID or Client ID.

Enter the Entity ID both in Orchestra and in Entra ID.

▼ SAML v2 Web Single Sign On Settings

SAML v2 web SSO enabled ☒

Service provider entity ID **1** TestSAML

Max authentication token lifetime 3600

Role attribute identifier

Username attribute identifier

First name attribute identifier

Last name attribute identifier

Locale attribute identifier

RTL attribute identifier

Identifier (Entity ID) (Required) **1**

This value must be unique across all applications in your (Azure Active Directory) tenant. It should follow one of the patterns provided below the textbox.

TestSAML

Reply URL (Assertion Consumer Service URL) (Required) **2**

The reply URL is where the application expects to receive the authentication token. This is also referred to as the "Assertion Consumer Service" (ACS) in SAML. It should follow one of the patterns provided below the textbox.

https://t94.qmaticcloud.com:8443/callback?client\_name=SAML2Client

https://t92.qmaticcloud.com:18443/callback?client\_name=SAML2Client

**2** Reply URL should be the following:

`https://<host>:<port>/callback?client_name=SAML2Client`

In this example the host is t92.qmaticcloud.com and the port is 8443.

The URL is different for every Orchestra agent which means you might have to add several URLs. The URL for your agent(s) can be found in System Administration > Queue Agents > Agents.

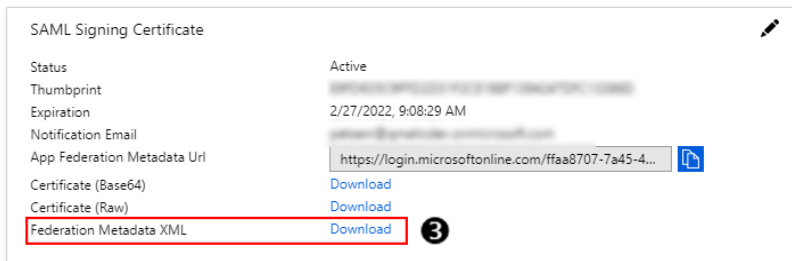
- Once the Entity ID and the reply URL are configured, you need to handle the **federation metadata**. There are two options - either delegate to federation metadata URL (option A), or download federation metadata file (option B).

Option A means less manual steps, and can be used even if you don't have access to the file directory where Orchestra is installed. However, option A requires that all agents have network access.

**Option A:** To use the URL, see “Option A – Delegate to Federation Metadata URL” on page 259.

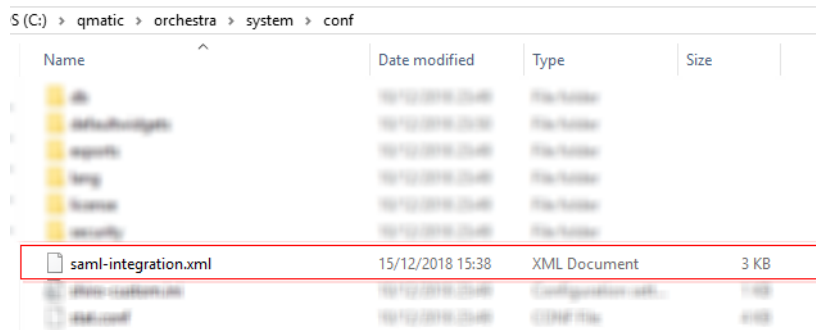
**Option B:** To use the metadata file follow these steps:

- a) Download the identity provider federation metadata file from the identity provider. For Entra ID, that can be downloaded from the link below (3).

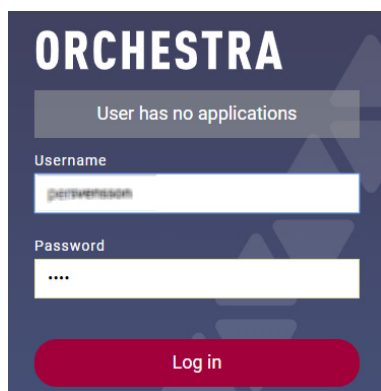


- b) Rename the downloaded file to saml-integration.xml and do the following:

- For Orchestra Central, place the file in orchestra/system/conf folder.
- For distributed queue agents, place the file in orchestra/system/media/agentProfile/ <wanted agent profile> /conf. Then do a remote upgrade.



- c) Verify that the field **Identity Provider Metadata URL** in Orchestra System Administration > Parameters is empty and then restart Orchestra for changes to go into effect.
5. At this point, you should make sure that the SAML flow between Orchestra and the Entra ID is working.
- Create a test user in Entra ID that doesn't have any roles assigned.
  - Log out the test user from Entra ID.
  - Go to Orchestra and see if you are redirected to the login page in Entra ID.
  - If the login is successful, you should be redirected to the following screen.



As you can see, it says *User has no applications*, which is correct because you have not yet configured any roles for this test user. However, it shows that the SAML flow between Orchestra and Entra ID is working. If it is not working, see “Troubleshooting” on page 268

If everything seems to be working properly, the next step is to assign roles and branch access to users. See “Assign users different access rights” on page 261.

## Configure Orchestra and ADFS to trust each other

Follow the steps in this section if you are using ADFS as your identity provider.

### Use the wizard “Add Relying Party Trust”

1. In the Server Manager and go to Tools > AD FS Management.
2. In the Actions toolbar, click **Add Relying Party Trust** to start the wizard:
  - a) Welcome: Select **Claims aware** and click **Start**.
  - b) Select Data Source: Select **Enter data about the relying party manually** and click **Next**.
  - c) Specify Display Name: Enter your **Orchestra https URL** as display name, e.g. `https://<orchestra_host>:port`, and click **Next**.
  - d) Configure Certificate: Browse and select your Orchestra certificate file. Obtain it first by exporting the certificate chain from Orchestra’s public https key.  
We recommend using a tool such as Keystore Explorer, <http://keystore-explorer.org/>, for exporting the key chain from your Orchestra keystore.
  - e) Configure URL: Check **Enable support for the SAML 2.0 WebSSO protocol** and enter the callback URL for the Orchestra client, e.g. `https://<orchestra_host>:port/callback?client_name=SAML2Client`
  - f) Configure Identifiers: Add the callback URL for Orchestra, e.g. `https://<orchestra_host>:port/callback`

- g) Choose Access Control Policy: Select **Permit everyone**.
- h) Ready to Add Trust: Click **Next**.
- i) Finish: Click **Close**.

**Adjust signature certificate**

1. Select the trust created by the wizard and select **Properties**
2. Select **Signature**
3. Select **Add** and select your ADFS certificate file.  
(You can export this by going to Service > Certificates in the left-hand folder structure, double click the certificate in Token-signing, open the Details tab and click Copy to File.)
4. Edit Claim Issuance Policy: Select the created trust and select **Edit Claim Issuance Policy...**
5. Issue Transform Rules: Add the following rules (use rule template "Send LDAP Attributes as Claims"):

**Table 1:**

Claim rule name	LDAP Attribute	Outgoing Claim Type
Surname as Surname	Surname	Surname
SAM-Account-Name as Name	SAM-Account-Name	Name
SAM-Account-Name as Name ID	SAM-Account-Name	Name ID
Member-Of as Role	Token-Groups - Unqualified Names	Role
Given-Name as Given Name	Given-Name	Given Name

**Orchestra configuration**

1. Configure the Orchestra Service provider entity ID.
  - a) Go to Orchestra and open **System Administration > Parameters**.
  - b) Check the **SAML v2 web SSO enabled** check box.
  - c) In SAML v2 Web Single Sign On Settings, in the **Service provider entity ID** field, enter the **callback URL** for Orchestra, e.g. `https://<orchestra_host>:port/callback`
  - d) Save

ⓘ Note that the **Service provider entity ID** must be an exact match of the callback URL for Orchestra that was used in Wizard under in **Configure Identifiers**.

## 2. Handle Federation Metadata

There are two options - either delegate to federation metadata URL (option A), or download federation metadata file (option B). Option A means less manual steps, and can be used even if you don't have access to the file directory where Orchestra is installed. However, option A requires that all agents have network access.

### ***Option A – Delegate to Federation Metadata URL***

- a) Acquire the URL for the Federation Metadata for your ADFS. Normally it is `https://<ADFS-server-host>/FederationMetadata/2007-06/FederationMetadata.xml`
- b) Add that URL to the field **Identity Provider Metadata URL** in Orchestra System Administration > Parameters.
- c) Save.

ⓘ **Important:** If the certificate of the ADFS server for securing its service communication is not signed by a trusted CA, you will also need to import that certificate to the trust store of Orchestra, similar to the instruction in “1a & 2a” section of “Secure WebSocket Between Distributed Queue Agent and Central”. To create the cer file:

1. Using the Server Manager, issue **Tools > ADFS Management**.
2. Select the certificate in ADFS > Service > Certificates > Service communications.
3. Select **View Certificate...** In the Details tab, use button **Copy to File....**
4. Select “Next” three times.
5. Choose to save to a file, and use the file to import to the trust store in Orchestra.

### ***Option B – Download Federation Metadata file***

- a) Acquire the URL for the Federation Metadata for your ADFS. Normally it is `https://<ADFS-server-host>/FederationMetadata/2007-06/FederationMetadata.xml`
- b) Download the file from that URL.
- c) Rename the downloaded file to `saml-integration.xml` and store it in `orchestra/system/conf` folder. For distributed queue agents, put the file in `orchestra/system/media/agentProfile/ <wanted agent profile> /conf`. Then do a remote upgrade.
- d) Verify that the field **Identity Provider Metadata URL** in Orchestra System Administration > Parameters is empty.

e) Restart Orchestra for changes to go into effect.

At this point, you should make sure that the SAML flow between Orchestra and the ADFS is working:

1. Create a test user in ADFS that doesn't have any roles assigned.
2. Log out the test user from ADFS.
3. Go to Orchestra and see if you are redirected to ADFS
4. Log in to the ADFS with your test user credentials.
5. If it is successful, you should be redirected to the Orchestra login screen.

The redirection to Orchestra login screen shows that the SAML flow between Orchestra and ADFS is working. If it is not working, see "Troubleshooting" on page 268.

If everything seems to be working properly, the next step is to assign roles and branch access to users.

## Assign users different access rights

### Configure attribute identifiers

Configure Orchestra and the identity provider to have the same attribute identifiers. This step is needed to provide roles, language settings and branch access for each user.

In Orchestra you can see the settings in System Administration > Parameters > SAML v2 Web Single Sign On Settings:

▼ SAML v2 Web Single Sign On Settings	
SAML v2 web SSO enabled	<input checked="" type="checkbox"/>
Service provider entity ID	TestSAML
Max authentication token lifetime	3600
Role attribute identifier	http://schemas.microsoft.com/ws/
Username attribute identifier	http://schemas.xmlsoap.org/ws/20
First name attribute identifier	http://schemas.xmlsoap.org/ws/20
Last name attribute identifier	http://schemas.xmlsoap.org/ws/20
Locale attribute identifier	http://schemas.xmlsoap.org/ws/20
RTL attribute identifier	http://schemas.xmlsoap.org/ws/20

- The default values are selected to match Entra ID. If you have another identity provider, make sure all the attribute identifiers match.
- The attribute for **Locale attribute identifier** should match a language previously added to Orchestra (see Localisation chapter). If the identity provider uses a combined format such as locale+country (e.g. sv\_SE or sv-SE), only the first characters ("sv") are used to match against the Orchestra languages. Users with an unknown language code, or without the locale attribute will get English.
- The attribute for RTL controls whether the user should have a right-to-left layout. If this attribute contains "true" or "rtl" a right-to-left layout will be used. If the attribute is missing or contains some other value, left-to-right will be used.

Once this is done, the identity provider and Orchestra can exchange user attributes.

### Configure role and branch mapping

The last step is to configure the **role and branch mapping** in to give access rights to users.

- In the identity providers, this is done differently depending on the identity provider. They have their own instructions on how to define role claims. Please

see the documentation for your specific identity provider. Below are some examples, from ADFS and Entra ID.

- In Orchestra, this is configured in User management > LDAP/SAML. See “Role and branch mapping in Orchestra” on page 266.

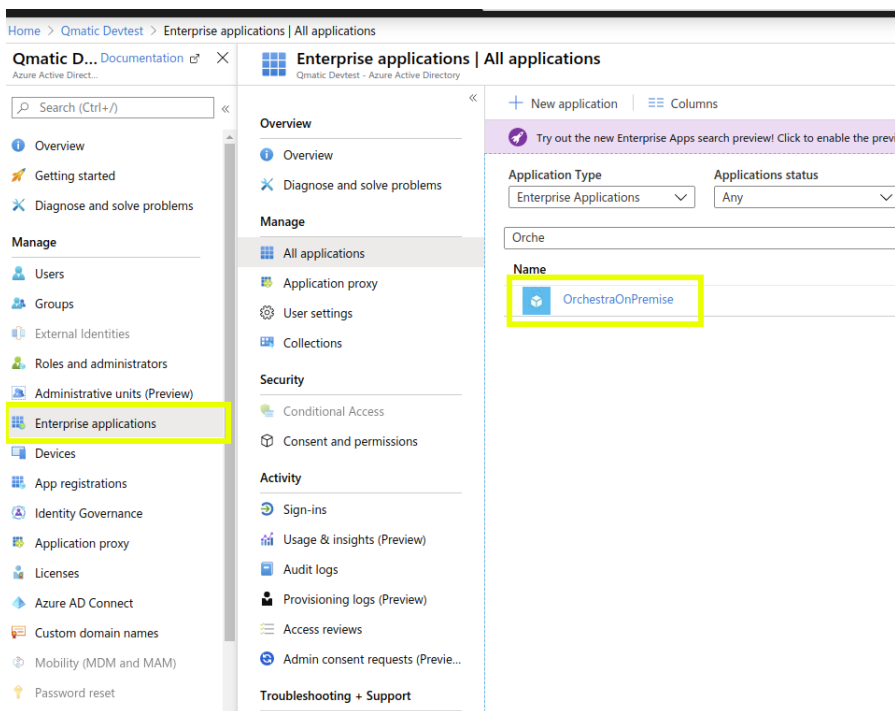
### Role and branch mapping in ADFS

For **ADFS** integration, the regular Active Directory management tools are used to define roles/groups and their members.

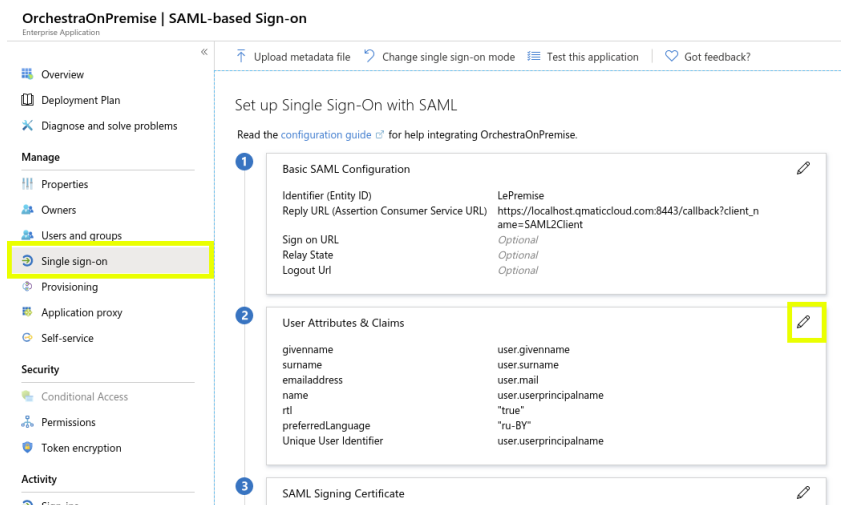
### Role and branch mapping in Entra ID

For **Entra ID** where the users and group information is synchronized from an on-premise Active Directory, the following set up can be used to propagate group membership information from the Active Directory on-premise system as SAML role claims:

1. In the Entra ID management console, select Enterprise Applications > <Name of enterprise application>



2. Select Single Sign On in the left menu, then select the pen in section (2) User Attributes And Claims to edit.



3. Select Add a group claim at the top
4. Select Security Groups as the groups to associate
5. Source attribute should be set to: sAMAccountName
6. In the advanced options section set these values:
  - Customize the name of the group claim: true
  - Emit groups as role claims: true

**Group Claims**

Manage the group claims used by Azure AD to populate SAML tokens issued to your app

Which groups associated with the user should be returned in the claim?

☐ None

☐ All groups

☒ Security groups

☐ Directory roles

☐ Groups assigned to the application

Source attribute \*

sAMAccountName

⚠ This source attribute only works for groups synchronized from an on-premises Active Directory using AAD Connect Sync 1.2.70.0 or above. [Learn More](#)

**Advanced options**

☒ Customize the name of the group claim

Name (required)

Namespace (optional)

☒ Emit groups as role claims ⓘ

7. See this page for more details:

<https://learn.microsoft.com/en-us/entra/identity/hybrid/connect/how-to-connect-fed-group-claims>

For Entra ID (without a connection to an on-premise Active directory) the configuration of groups/roles is done in a similar manner with some distinct differences.

- Group claims are configured in the same place as described in the section above (Entra ID with users/groups synchronized from an on-premise Active Directory).
- However, as the group/role information does not originate from an on-premise Active Directory, the claims issued will not be in a “human-friendly” format, but will instead be issues as UUID:s (e.g. 317ddc14-6e7d-0943-32a7-777190d0c3bb). These IDs need to be used when mapping role / branch / branch group access in Orchestra.
- In order to be able to use “human-friendly” names when mapping access and branch rights, the enterprise application in Entra ID must be configured with specific application roles to support this. The application roles can then be mapped to specific directory groups and/or users. This way the regular organizational groups defined in Entra ID can be used to map specific Orchestra access rights.

- The steps involved require you to edit the metadata information for the application registration and add the needed application roles.
- The second step is to tie a group in the Entra ID to the application role, thus all users belonging to that Entra ID group will be assigned that application role.
- Both steps are described here: <https://learn.microsoft.com/en-us/entra/identity-platform/howto-add-app-roles-in-apps>

⚠ This document describes that the IDs for each role must be unique, for a generator of unique IDs, see: <https://www.guidgenerator.com/>

⚠ It is the value field given when editing the appRoles list in the manifest that will be the mapping sent to Orchestra, thus the value field should not be left blank.

⚠ If using application roles in this way, you should NOT add a group claim to the enterprise application.

### Give the test user access rights

When you have managed to create roles, give a test user in the identity provider at least one role and one branch. This is done differently for all identity providers.

This picture shows the group membership for a test user in ADFS:



## Role and branch mapping in Orchestra

In Orchestra, go to User Management > LDAP/SAML. Here, you need to create mappings for roles, branches and branch groups.

1. To create a new mapping, click the *Create New Mapping* button.
2. Create a mapping for the test user in Orchestra.

In the example below we have a role called orchestra-role-counter mapped to the Orchestra CounterUser role. We also have a branch group mapping called all-orchestra-branches.

The screenshot shows the 'User Management' interface with the 'LDAP / SAML' tab selected. Below the tab, there are buttons for 'Save', 'Discard changes', and 'Create New Mapping'. A table displays the current mappings:

Name	Type	Mapping
orchestra-role-counter	Role	CounterUser
all-orchestra-branches	Branch Group	REGIONS

**Name** – name of the role/group that should be mapped, matched with the name in the identity provider.

**Type** – type of mapping, *role*, *branch* or *branch group*.

**Mapping** – available roles, branches or branch groups

3. When done creating mapping for the test user in Orchestra, click **Save**.
4. Try to log in with the test user. You should automatically be logged in and have access to the role(s) that you configured.
5. The last step is to assign correct mapping to all users that should have access to Orchestra.

## Configure single logout

To configure single logout, a logout URL needs to be provided from the identity provider. Not all identity providers have a logout URL, but some of them can be manually configured like Entra ID.

To configure a logout URL for Entra ID (the procedure is similar for other identity providers):

1. Open the enterprise application that Orchestra is connected to.
2. Go to *Set up single sign on*.



## 2. Set up single sign on

Enable users to sign into their application using their Microsoft Entra credentials

[Get started](#)

- Copy the logout URL in the section Set up <Name of the enterprise application>

4

### Set up Orchestra

You'll need to configure the application to link with Microsoft Entra ID.

Login URL	<a href="https://login.microsoftonline.com/ffaa8707-7a45-...">https://login.microsoftonline.com/ffaa8707-7a45-...</a>
Microsoft Entra Identifier	<a href="https://sts.windows.net/ffaa8707-7a45-44db-9152...">https://sts.windows.net/ffaa8707-7a45-44db-9152...</a>
Logout URL	<a href="https://login.microsoftonline.com/ffaa8707-7a45-...">https://login.microsoftonline.com/ffaa8707-7a45-...</a>

- Go to *System Administration > Settings > Parameters > Orchestrated Logout Settings* and paste the logout URL in the parameter *Logout URLs*.

▼ Orchestrated Logout Settings

Redirect URL

/login.jsp

Logout URLs

<https://login.microsoftonline.com/ffaa8707-7a45-...>

- Save

## Troubleshooting

ⓘ Note that excessive stack traces have been pruned from log examples.

### Problem: The SAML flow between Orchestra and the identity provider isn't working.

Enable debug logs in Orchestra for SAML. That is done by adding DEBUG to the following lines in orchestra/system/conf/logback.xml. No restart is required.

```
<!-- SAML -->
<logger name="org.opensaml.xmlsec.algorithm.AlgorithmRegistry" level="DEBUG" />
<logger name="org.pac4j.saml.crypto.DefaultSignatureSigningParametersProvider" level="DEBUG" />
<logger name="org.pac4j.saml.sso.impl.SAML2DefaultResponseValidator" level="DEBUG" />
<logger name="org.opensaml.saml.metadata.resolver.impl.AbstractReloadingMetadataResolver" level="DEBUG" />
<!-- End of SAML configuration -->
```

#### First possible error in log could be:

```
2018-12-18 15:18:45.433 [default task-25] WARN
c.q.q.c.a.s.saml.SamlCallbackFilter - Unable to log in using SAML
integration. error: Assertion audience [okta2] does not match SP
configuration okta
```

```
2018-12-18 15:19:44.503 [default task-43] ERROR
o.p.s.s.i.SAML2DefaultResponseValidator - Current assertion
validation failed, continue with the next one
```

```
org.pac4j.saml.exceptions.SAMLErrorAssertionException: Assertion
audience [okta2] does not match SP configuration okta
```

As can be seen, the okta2 does not match Orchestra configuration of Okta (the selected identity provider in this case), i.e. Okta is incorrectly configured.

#### Second log example:

```
2019-11-08 10:26:04.302 [default task-73] INFO
o.p.s.s.i.SAML2DefaultResponseValidator - Constructed SAML2
credentials:
SAML2Credentials{nameId=SAMLNameID{format='null',
nameQualifier='null', spNameQualifier='null',
spProviderId='null', value='test-adfs'},
sessionIndex='_eba52f30-2933-4587-b09f-feled75dc6f5',
attributes=[SAMLAttribute{friendlyName='null', name='http://
schemas.xmlsoap.org/ws/2005/05/identity/claims/name',
nameFormat='null', attributeValues=[test-adfs]},
SAMLAttribute{friendlyName='null', name='http://
schemas.microsoft.com/ws/2008/06/identity/claims/role',
nameFormat='null', attributeValues=[Domain Users, all-
orchestra-branches, orchestra-role-counter]}],
```

```

SAMLAttribute{friendlyName='null', name='http://
schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname',
nameFormat='null', attributeValues=[test-adfs]],
conditions=SAMLConditions{notBefore=2019-11-
08T09:23:15.440Z, notOnOrAfter=2019-11-08T10:23:15.440Z},
issuerId='http://adfs.test.q-matic.net/adfs/services/
trust',
authnContexts=[urn:oasis:names:tc:SAML:2.0:ac:classes:Pass
wordProtectedTransport]}

```

Inspect the attribute values for `http://schemas.microsoft.com/ws/2008/06/identity/claims/role` and compare those values with the SAML mappings in Orchestra.

Another way to debug would be to open the browser developer tools (ctrl+shift I) on Chrome. Open the network tab and enable preserve logs. Then try to access Orchestra again. The log should look like this.

Name	Method	Status	Protocol	Domain	Type	Initiator
t34.qmaticcloud.com	GET	200	h2	t34.qmaticcloud.com	document	Other
home.html	GET	200	h2	t34.qmaticcloud.com	document	:8443/?
saida-dialog.html	GET	200	chrome-ex...	limnpfdlpiifbklhmjmapnlpghbkalf	xhr	chrome-extension
saml	POST	200	http/1.1	qmatic.okta.com	document	:8443/home.html
jquery-1.12.4.min.e93c5a2265f8e2a3e96fe19159f...	GET	200	h2	ok7static.oktacdn.com	script	qmatic.okta.com/
interstitial.c22f9cc988086c3b50de2f50b83669e5...	GET	200	h2	ok7static.oktacdn.com	stylesheet	qmatic.okta.com/
new_interstitial_static.9481d4731547cec09b26be...	GET	200	h2	ok7static.oktacdn.com	png	qmatic.okta.com/
new_interstitial.c41c3b6f3a8458aca9a5919f238f...	GET	200	h2	ok7static.oktacdn.com	gif	qmatic.okta.com/
okta_watermark.4a7f2cc7d0a787cfe6f59fb67f728...	GET	200	h2	ok7static.oktacdn.com	png	qmatic.okta.com/
interstitial.474dce61acf4c4a4d016921943cf2a68j...	GET	200	h2	ok7static.oktacdn.com	script	qmatic.okta.com/
callback?client_name=SAML2Client	POST	302	h2	t34.qmaticcloud.com	x-www-form-urlencoded	ok7static.oktacdn
login.jsp?failMessage=error.login.failed	GET	200	h2	t34.qmaticcloud.com	document	:8443/callback?cli
fonts.css	GET	200	h2	t34.qmaticcloud.com	stylesheet	login.jsp?failMess
login.css	GET	200	h2	t34.qmaticcloud.com	stylesheet	login.jsp?failMess
...	...	...	...	...	...	...

Here you can see that both the okta.com and my local domain is involved and that Orchestra adds an error.login.failed error message. Check the log file to see what went wrong.

### A third log example

```

2020-04-28 15:56:59.455 [default task-15] DEBUG
 o.p.s.transport.Pac4jHttpPostDecoder - Decoded SAML message
2020-04-28 15:56:59.512 [default task-15] ERROR
 o.o.x.encryption.support.Decrypter - Error decrypting encrypted key
org.apache.xml.security.encryption.XMLEncryptionException:
 Unwrapping failed
...
Caused by: java.security.InvalidKeyException: Unwrapping failed
...
Caused by: javax.crypto.BadPaddingException: Decryption error
...
2020-04-28 15:56:59.512 [default task-15] ERROR
 o.o.x.encryption.support.Decrypter - Failed to decrypt
 EncryptedKey, valid decryption key could not be resolved

```

```
2020-04-28 15:56:59.512 [default task-15] ERROR
o.o.x.encryption.support.Decrypter - Failed to decrypt
EncryptedData using either EncryptedData KeyInfoCredentialResolver
or EncryptedKeyResolver + EncryptedKey KeyInfoCredentialResolver
2020-04-28 15:56:59.512 [default task-15] ERROR
o.o.saml.saml2.encryption.Decrypter - SAML Decrypter encountered an
error decrypting element content
org.opensaml.xmlsec.encryption.support.DecryptionException: Failed
to decrypt EncryptedData
...
2020-04-28 15:56:59.513 [default task-15] ERROR
o.p.s.s.i.SAML2DefaultResponseValidator - Decryption of assertion
failed, continue with the next one
org.opensaml.xmlsec.encryption.support.DecryptionException: Failed
to decrypt EncryptedData
...
2020-04-28 15:56:59.513 [default task-15] WARN
c.q.q.c.a.s.saml.SamlCallbackFilter - Unable to log in using SAML
integration. error: No valid subject assertion found in response
```

**Possible cause 1:** The certificate exchange between Orchestra and the identify provider has not been set correctly.

**Solution 1:** Ensure that the certificate exchange is properly set up.

Generally, the public key from the https certificate of Orchestra is used. This can be obtained from Orchestras keystore using a tool such as Keystore Explorer, (<http://keystore-explorer.org>).

Note that if a load balancer is used in front of Orchestra, the certificate of the load balancer should NOT be used (as Orchestra generally does not know the matching private key for that certificate).

**Possible cause 2:** The key exchange between Orchestra and the Identity provider is correct, but Orchestra has several private keys in it's key store.

**Solution 2:** If the key store used in Orchestra contains several private keys, the SAML implementation will by default pick the first key to use when decrypting the SAML payload. In this case, you can specify which of the keys to use by adding the alias of the relevant key to the system parameter **SAML Keystore alias** in the section **SAML v2 Web Single Sign On Settings**.

#### A fourth log example

```
2020-04-21 08:17:57.782 [default task-24] WARN
c.q.q.c.a.s.s.OrchestraConfiguration - URL [https://
adfs.somecompany.com/FederationMetadata/2007-06/
FederationMetadata.xml] does not exist
javax.net.ssl.SSLHandshakeException:
sun.security.validator.ValidatorException: PKIX path building
failed:
```

```
sun.security.provider.certpath.SunCertPathBuilderException: unable
to find valid certification path to requested target
...
Caused by: sun.security.validator.ValidatorException: PKIX path
building failed:
sun.security.provider.certpath.SunCertPathBuilderException: unable
to find valid certification path to requested target
...
Caused by:
sun.security.provider.certpath.SunCertPathBuilderException: unable
to find valid certification path to requested target
...
2020-04-21 08:17:57.782 [default task-24] WARN
c.q.q.c.a.s.s.OrchestraConfiguration - SAML configuration is set to
enabled, but the specified identity provider url does not exist.
SAML support will be disabled
```

**Cause:** Orchestra is set up to fetch the Identity Provider metadata file at start up, but it does not trust the certificate that the Identity Provider uses to serve the metadata file over https.

**Solution:** Import the root/ca certificate that has signed the certificate the identity provider is using into Orchestras trust store.

**Problem: SAML user has no roles assigned.**

The user that attempts to log in into Orchestra using SAML is greeted with the error message “User has no roles assigned.”

**Cause:** Misspelling of role mapping name in User Management, LDAP/SAML in Orchestra

**Solution:** The role mapping name in User Management in Orchestra should be spelled as the group is spelled in the identity provider, for example Windows AD. See Orchestra reference manual section “D. Set up attribute identifiers and assign users with different access rights – Using ADFS”.

**Problem: SAML user has no branches assigned.**

The user that attempts to log in into Orchestra using SAML is greeted with the error message “User has no branches assigned”.

**Cause:** Misspelling of branch mapping name in User Management, LDAP/SAML in Orchestra

**Solution:** The branch mapping name in User Management in Orchestra should be spelled as the group is spelled in the identity provider, for example Windows AD. See Orchestra reference manual section “D. Set up attribute identifiers and assign users with different access rights – Using ADFS”.

### **Problem: SAML users has no applications**

The user that attempts to log in into Orchestra using SAML from ADFS will be greeted with the error message "User has no applications".

This problem may have one of the following causes:

**Possible cause 1:** Misspelling of role mapping name and branch mapping name in User Management, LDAP/SAML in Orchestra

**Solution:** The role mapping name and the branch mapping name in User Management in Orchestra must both be spelled as their groups are spelled in the identity provider, for example Windows AD. See Orchestra reference manual section "D. Set up attribute identifiers and assign users with different access rights - Using ADFS".

**Possible cause 2:** Misspelling of role attribute identifier in System Administration -> Parameters in Orchestra

**Solution:** Ensure that the Role attribute identifier is correctly spelled and matches the corresponding name in ADFS, or whichever identity provider you are using. Note that this name is not by default displayed in ADFS, but can be seen in the Orchestra debug messages. The default value is <http://schemas.microsoft.com/ws/2008/06/identity/claims/role>. This is likely the correct value.

### **Problem: Entra ID login reports an error AADSTS50105**

After being redirected to the identity provider (Entra ID) and logging in an error page with a message like this is displayed:

AADSTS50105: The signed in user 'myuser@myenterprise.com' is not assigned to a role for the application '9999cccc-dddd-4444-3333-8888aaaa2222'(MyApplication).

**Cause:** Users are not assigned to the enterprise application.

**Solution:** Either assign explicit users and groups (Enterprise Applications -> <application name> -> Users and Groups -> Add User) or allow all users (Enterprise Applications -> <application name> -> Properties -> User assignment required = false)

# Index

## A

- AD/LDAP server certificate 135, 137
- Agent Profile 103, 109, 110
- Agent Profile Zip File 105
- Application Parameters 112, 120
- Appointment Management Settings 112

## B

- Bandwidth Requirements 5
- Blocking Websocket Port 58
- Branch Agnosticism 4
- Branch Hub 53, 99
- Browser Settings 112
- Browser Specific Settings 158

## C

- Central 26, 63, 156
- Central Access Parameters 112
- Central Queue Agent 156
- Central WebSocket Server Settings 112
- Certificate and key store Settings 112
- Client Area 206
- Cloud Services 112
- Connectors 112
- Customer 112
- Customer Journey Management 2

## D

- default keystore 168
- default truststore 168
- Deployment Scenarios 7
- Distributed operation 5
- Distributed Queue Agent 158

## E

- Event Manager Settings 112
- Export 90
- export 91

## F

- Failed events 153

## G

- Gateway 1745 53
- General Parameters 112, 113

## H

- hardware licensing 88
- Home 3
- HTTP Settings 112
- HTTPS Parameters 112
- HTTPS server settings 112
- Hub 53, 99

## I

- Import 90, 92
- Installation Wizard 53

## K

- Keystore Explorer 172, 173, 176, 180

## L

- LDAP 134
- LDAP Certificate Handling 135
- LDAP Settings 112
- Licensing 84
- Linux 51, 69, 74

## M

- Mark Types 138
- Mobile API (Central) 112
- Multiple Queue Agents 101

## O

- Off-line activation 86
- On-line activation 85
- OpenLDAP 242

## P

- Parameters 112
- Port and Protocol Handling 169
- Port Numbers 14
- Pre Authentication and Authorization Settings 113

## Q

Queue Agent 48, 74, 97, 100, 111  
Queue Agent Media Settings 112

## R

Recycle Settings 112  
Redeploying Applications 53  
Regression Test 76  
Remote Update 74  
Remote Upgrade 75

## S

SAML single logout URL 266  
SAML v2 Web Single Sign On Settings 112  
Server Area 207  
Sorting Policy Settings 112  
SSO Settings 112  
Statistics Settings 112  
Suggested Upgrade Order 62  
Supported Environments 9  
Synchronizing 111  
System 112

## T

Tuning of Oracle Parameters 224, 232, 236  
Tuning of PostgreSQL Parameters 225, 233

## U

Ubuntu 14.04 server. 242  
Unit Type Files 78  
Unit Type Templates 159  
Unit Types 77, 82  
User Settings 112  
utt 159

## W

Weekend Settings 112  
Widget Administration 93  
Widget Mapping 96  
Widget Whitelist 95  
Widgets 79, 159  
Windows 65, 74  
Windows Authentication 19, 23, 31, 41



# Q-MATIC

Q-MATIC AB,  
Box 198,  
SE-431 23 Mölndal,  
Sweden.

VAT No SE556212749701

Phone: +46 31 756 46 00

Email: [info@qmatic.com](mailto:info@qmatic.com)

Web: [www.qmatic.com](http://www.qmatic.com)