



**Administrator**

---

# **SAP Mobile Platform 3.0 SP02**

DOCUMENT ID: DC01994-01-0302-01

LAST REVISED: February 2014

Copyright © 2014 by SAP AG or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries. Please see <http://www.sap.com/corporate-en/legal/copyright/index.epx#trademark> for additional trademark information and notices.

# Contents

<b>SAP Mobile Platform Server Overview .....</b>	<b>1</b>
<b>Administrator Overview .....</b>	<b>3</b>
Administrator Overview for Integration Gateway .....	3
<b>Getting Started .....</b>	<b>5</b>
Postinstallation Landscape Setup .....	5
Setting up Back-End Communications .....	5
Adding a SOCKS Proxy or Load Balancer for APNS Connections .....	22
Starting and Stopping SAP Mobile Platform Server .....	24
Starting and Stopping SAP Mobile Platform Server on Windows .....	25
Starting and Stopping SAP Mobile Platform Server on Linux .....	25
Getting Started with Management Cockpit .....	26
Starting and Stopping the Management Cockpit on Windows .....	26
Starting and Stopping the Management Cockpit on Linux .....	27
Using the Home Screen .....	27
Setting Up Browser Certificates for Management Cockpit Connections .....	28
Activating Sample Data for ESPM .....	28
<b>Application Administration .....</b>	<b>31</b>
Deploying Applications .....	31
Deploying from a Preproduction Environment .....	33
Configuring a Production Environment for Deployment .....	34
Production Environment Deployment Tasks By Application Type .....	35
Defining Applications .....	36
Defining Back-end Connections .....	38

Defining Application Authentication .....	58
Defining Client Password Policy .....	59
Defining Push Notifications .....	60
Uploading Client Resources .....	65
Defining Application-specific Settings .....	66
Saving Application Settings .....	83
Managing and Monitoring Applications .....	84
Managing and Monitoring Tasks By Application Type .....	84
Managing Applications .....	84
Managing Registrations .....	88
Managing Users .....	92
Importing and Exporting Applications .....	92
Managing Security Profiles .....	94
Managing Connections .....	98
Reporting Usage Statistics .....	102
Managing Application Logs .....	103
Configuring Agency Application Logs .....	107
Provisioning Applications .....	108
Provisioning with Afaria .....	108
<b>Server Administration .....</b>	<b>113</b>
Starting and Stopping SAP Mobile Platform Server on Windows .....	113
Starting and Stopping SAP Mobile Platform Server on Linux .....	114
Configuring SAP Mobile Platform Server .....	114
Configuring HTTP/HTTPS Port Properties .....	114
Configuring Push Notification Properties .....	115
Configuring Back-end Communications with an HTTP Proxy .....	116
Changing Database Connection Passwords .....	117
Performance Properties .....	118
Reconfiguring the Windows Service After Server Configuration Changes .....	123
Props.ini Reference .....	124

Server Monitoring and Analysis .....	126
Managing Server Logs .....	126
Workload Analysis with Wily Introscope .....	139
Managing SAP Mobile Platform Server Features .....	142
Backing Up Server Configuration Data .....	144
Port Number Reference .....	145
HTTP/HTTPS Port Number Reference .....	145
TCP Port Number Reference .....	147
<b>Security Administration .....</b>	<b>149</b>
Planning Your Security Landscape .....	149
Security Postinstallation Checklist .....	158
Platform Security Quick Start .....	159
SAP Mobile Platform Authentication Quick Start .....	159
SAP Mobile Platform Security Overview .....	161
Component Security .....	161
Communication Security .....	161
Authentication .....	162
Back-End Security .....	163
Common Security Infrastructure in SAP Mobile Platform .....	164
Configuring Security Fundamentals in SAP Mobile Platform .....	164
Security Profiles in SAP Mobile Platform .....	164
Role Mapping .....	171
Authentication in SAP Mobile Platform .....	177
Adding Reverse Proxies .....	209
Server Security .....	215
Securing the Server Infrastructure .....	215
Securing Platform Administration .....	217
Managing Keystore and Truststore Certificates .....	223
Strong Encryption for JVM Security .....	230
Application Security .....	231
Authentication for User Logins .....	232
Enabling a Direct HTTPS Connection to SAP Mobile Platform Server .....	249

Enabling OCSP .....	250
Device Security .....	251
Limiting Application Access .....	252
Securing Sensitive Data On-Device with DataVault .....	253
Agentry Security .....	255
Agentry Security Specifications Reference .....	257
Security Monitoring and Validation .....	258
Checking the Security Log .....	258
Debugging Authentication Errors with CSI Tool ..	258
Migrating Security Components .....	260
Migrating Authentication for Native OData HTTP Clients .....	260
<b>Application Services (Mobiliser) Administration .....</b>	<b>261</b>
Basic Deployment Model .....	261
Standard Reverse Proxy Setup .....	262
Mobiliser Setup .....	263
Mobiliser Universal User .....	265
Creating or Updating Hashed Password for the Universal User .....	265
Creating or Updating the Encrypted Password for Preferences .....	266
Encryption and Keystore Configuration .....	267
Encryption in MOB_PREFERENCES .....	267
Keystore Configuration .....	269
Creating the Keystore for Data Encryption .....	271
Encrypting Preferences Using Operations Dashboard .....	273
On-Device Charging Installation and Configuration ...	274
Provision Secure Element Keys for DIRECT Mode .....	274
Generate Private Keys Used by On-Device Charging .....	276
Performance Considerations .....	277
Event Handling Threads .....	277

Business Logic Configuration .....	278
Framework .....	278
Messaging .....	287
Audit .....	300
Event Handler .....	303
Tasks .....	304
Jobs .....	305
cron Job Task Handler .....	305
MOB_JOBS Table .....	307
Miscellaneous Configuration .....	308
Security Fundamentals .....	312
Prevent Unauthorized Access .....	312
Web Portal Access to Mobiliser .....	312
Hashing Customer Credentials .....	314
Default Mobiliser Web Portal Accounts .....	316
Changing the Mobiliser Web Portal Passwords .....	317
Operations Dashboard Reference .....	317
Logging in to the Operations Dashboard .....	317
Preferences .....	318
Jobs .....	321
Servers .....	322
Trackers .....	332
SOAP/REST Interface Management .....	333
Data Archiving, Retention and Deletion .....	333
Data Archiving .....	333
Data Retention and Deletion .....	334
Deletion Script .....	335
Virus Protection .....	336
Configuring the Virus Scan Adapter for SAP NetWeaver .....	336
Configuring ClamAV on a Linux Server .....	337
cron Expression Reference .....	338
<b>SMS Administration .....</b>	<b>341</b>
Setting Up a Production System .....	342

Hashing the Admin Password .....	342
Configuring New Database Installations .....	342
Enabling Encryption .....	348
Encrypting Property Values .....	349
Configuring Authentication .....	349
Configuring the Event Scheduler JDBC Driver .....	350
Enabling SSL .....	350
Communication Channels .....	351
Configuring SMPP Inbound Channels .....	352
Configuring SMPP Outbound Channels .....	353
Configuring JMS Channels .....	354
Workspaces .....	354
Creating Workspaces .....	356
Opening Workspaces .....	356
Default Menu .....	357
Configuring Default Menus .....	358
Deleting Workspaces .....	359
Users .....	360
Adding Users .....	360
Editing User Properties .....	361
Deactivating Users .....	362
User Roles .....	362
Deleting Users .....	363
Categories .....	364
Subscribers .....	364
Creating Empty Subscriber Sets .....	365
Uploading Subscriber Sets .....	366
Reports .....	366
Generating Traffic Reports .....	367
Generating Subscriber Reports .....	367
Maintenance and Tuning .....	368
Default Ports .....	368
System Configuration Files .....	369
Log Files .....	371

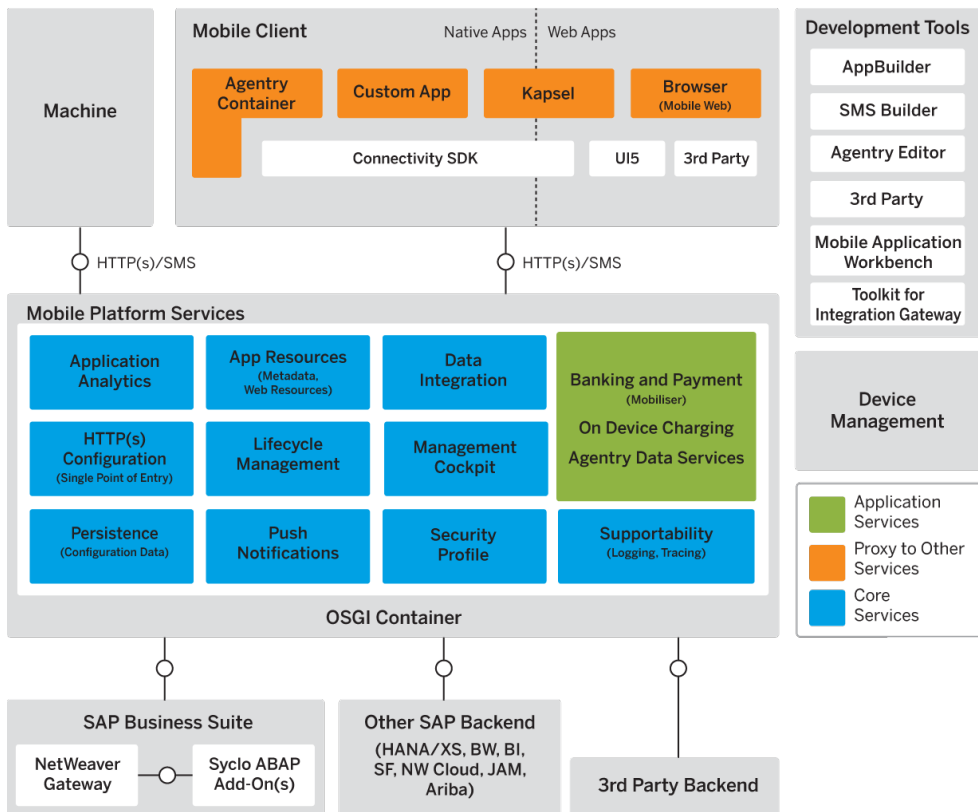


Database Table Maintenance .....	371
Processing Engine .....	372
<b>Troubleshooting .....</b>	<b>377</b>
Quick Fixes to Simple Server Problems .....	377
Management Cockpit Stops Working After	
Enabling Mobiliser Services Feature .....	377
Mobiliser Features Not Included in Enabled	
Features List .....	377
Connection Failure Outside Corporate Network	
.....	378
Unknown Host Exception While Accessing BIS	
.....	378
Browser Caching Issue .....	378
Impersonator Role Missing Error .....	379
Unauthorized Error for First Mutual	
Authentication Request .....	379
HTML Code Appears in Server Log .....	379
Issues Requiring Product Support .....	380
Product Support Engagement Requirements ...	380
Creating an Incident on SAP Service	
Marketplace .....	381
Increasing Server Status Logging Levels .....	381
Server Fails to Initialize .....	382
Statistics Data Loss During Server Crash or	
Shut Down .....	382
<b>Glossary: SAP Mobile Platform .....</b>	<b>383</b>
<b>Index .....</b>	<b>393</b>

# Contents

# SAP Mobile Platform Server Overview

SAP® Mobile Platform Server is a lightweight OSGi based application server that provides both application services and core services. Application services are specialized for a specific application type. Core services can be accessed and used by any application type.



Application services consist of services for Mobiliser and Agentry applications.

Core services consist of the following:

- **Application Analytics** – usage statistics that can be displayed graphically in the Management Cockpit.
- **App Resources** – containers of dynamic configurations, styles, or content that can be downloaded by Native applications.

## SAP Mobile Platform Server Overview

- **Data Integration** – standards-based integration to both SAP and non-SAP back-end systems.
- **HTTP(s) Configuration** – open standards for client communications.
- **Lifecycle Management** – managing and deploying multiple versions of a hybrid application.
- **Management Cockpit** – deploying, managing and monitoring native, hybrid, and Agency applications.
- **Persistence** – server configuration data stored in the database of your choice (ASE, DB2, or Oracle).
- **Push Notifications** – native notifications sent from back-end systems to the server, which forwards them on to the clients.
- **Security Profile** – authentication services based on third party identity management systems.
- **Supportability** – logs for monitoring system health and troubleshooting.

# Administrator Overview

Administrators interact with SAP Mobile Platform to ensure the mobile application production environment works efficiently.

Administrator tasks fall into three main categories:

- Server administration for configuring the server, enabling optional server features, and troubleshooting server issues.
- Application administration for defining application types for deploying to users and monitoring applications in the user community.
- Security administration for setting up secure communications and protecting data.

There are separate administration tasks for Mobiliser applications. SMS applications are administered through a separate server.

There are additional administrator tasks to perform when using Integration Gateway.

## See also

- *Application Administration* on page 31
- *Server Administration* on page 113
- *Security Administration* on page 149
- *Application Services (Mobiliser) Administration* on page 261
- *SMS Administration* on page 341

## Administrator Overview for Integration Gateway

---

Integration Gateway enables you to expose and manage OData services, which can be consumed by SAP Mobile Platform applications. The OData services are based on database and web services data from different data sources (both SAP and non-SAP).

Integration Gateway includes:

- Toolkit for Integration Gateway, which provides an environment to connect to different data sources and to create and deploy artifacts on SAP Mobile Platform Server.
- Gateway Management Cockpit, which is the central UI for managing OData services that have been deployed on SAP Mobile Platform Server.

If you are using Integration Gateway, you need to perform the following administrator tasks:

- Map the logical roles required by Toolkit for Integration Gateway users. Default Integration Gateway roles are installed with SAP Mobile Platform Server.

## Administrator Overview

- Manage the OData services deployed on SAP Mobile Platform Server. Use the Gateway Management Cockpit to register and create destinations for services. For details, see *Gateway Management Cockpit*.
- Configure application connections to Integration Gateway back ends using SAP Mobile Platform Server Management Cockpit. The endpoint URL for the connection should be the same as the service document URL of the registered service in Gateway Management Cockpit.

For an overview of both data model developer and administrator tasks for Integration Gateway, see *Data Integration using Integration Gateway*.

### **See also**

- *Integration Gateway Roles* on page 174
- *Defining a Back-end Connection* on page 99
- *Defining Back-end Connections for Native and Hybrid Apps* on page 38

# Getting Started

Get started with SAP Mobile Platform Server by first performing post installation tasks, then starting the server and logging in to and using Management Cockpit.

## Postinstallation Landscape Setup

After installing SAP Mobile Platform Server you need to perform postinstallation tasks before performing any server or application administration tasks.

The postinstallation tasks you need to perform depend on the landscape you need to support:

- Configure communications between the server and your supported back end systems.
- Set up communications between the server and the Apple Push Notification Service (APNS).
- Set up the security landscape for the server.

### See also

- *Security Postinstallation Checklist* on page 158

## Setting up Back-End Communications

Set up communications between SAP Mobile Platform Server and your back end. Back end refers to your data source, or enterprise information system (EIS).

**Table 1. Back-End Communication Setup Tasks**

Task	Description
Download back end drivers and libraries	(Non-SAP back ends) Download and install all required drivers and libraries for your back-end type.
Download and install Java Connector libraries and utilities	(SAP back ends) Download and install the SAP-CAR utility and SAP cryptographic library.
Configure back-end communications with an HTTP proxy	Configure the proxy host and port. Override for HTTP/HTTPS Authentication provider if required.
Configure host systems for Agency back ends	Configure the host systems that Agency applications connect to and synchronize data with.

If you are connecting to SAP or non-SAP back ends using Integration Gateway you need to expose OData services using Toolkit for Integration Gateway and manage OData

## Getting Started

services.using Gateway Management Cockpit. For more information, see *Data Integration using Integration Gateway*.

If you are using a single sign-on system (SSO) to achieve end-to-end integration across client applications and back-end resources, complete SSO setup as part of your security administration.

### See also

- *Single Sign-on Integration Across Client Applications* on page 232
- *Configuring Back-end Communications with an HTTP Proxy* on page 116

### **Preparing to Connect to SAP using Java Connectors**

SAP Mobile Platform Server can use Java Connectors (JCo) to connect to the SAP back end. With the correct security setup, you can also implement single sign-on (SSO) authentication.

### Prerequisites

You must have an SAP account to access the SAP Web site and download libraries and utilities.

#### *Installing the SAPCAR Utility*

Unzip and install the latest **SAPCAR** utility on your SAP Mobile Platform Server host. You can use **SAPCAR** to extract the contents of compressed SAP files, for example, RFC and cryptographic library files.

The installation package is available to authorized customers on the SAP Service Marketplace. There are different distribution packages for various hardware processors. Select the package that is appropriate for your platform.

1. Go to the SAP Web site at <http://service.sap.com/swdc> (login required).
2. From the SAP Download Center, navigate and log in to **Support Packages and Patches > Browse our Download Catalog > Additional Components**.
3. Select **SAPCAR**.
4. Select the current version, for example, **SAPCAR 7.20**, then download the appropriate **SAPCAR** for your platform.

### See also

- *Installing the SAP Cryptographic Libraries* on page 7



### Installing the SAP Cryptographic Libraries

Configure Secure Network Communications (SNC) for SAP Mobile Platform Server SAP JCo connections. SNC may be required by your SAP back end, if you are using SSO2 tokens or X.509 certificates for connection authentication.

#### **Prerequisites**

Download and install the **SAPCAR** utility, which is required to extract the contents of the cryptographic library.

#### **Task**

Unzip and install the contents of the latest SAP cryptographic archive on your SAP Mobile Platform Server host. There are different distribution packages for various hardware processors.

Make sure you are installing the correct libraries for your environment, and into folders that are based on the architecture of your machine.

1. Go to the SAP Web site at <http://service.sap.com/swdc> (requires login) and download the latest SAP cryptographic library suitable for your platform.
  - a) Navigate to **Installations and Upgrades > Browse our Download Catalog > SAP Cryptographic Software > SAPCryptolib for Installation > SAPCRYPTOLIB <version>**.
  - b) Select and download the platform-specific file.
2. Create a directory into which to unzip the cryptographic file. For example: C : \sapcryptolib.
3. Copy the appropriate Windows cryptographic library for your machine (for example, SAPCRYPTOLIB<version>.SAR) to the C : \sapcryptolib directory.
4. Open a command prompt and navigate to C : \sapcryptolib.
5. Extract the SAR file. For example:
 

```
SAPCAR_4-20002092.EXE -xvf C:\SAPCRYPTOLIB<version>.SAR -R C:\sapcryptolib
```
6. Copy the following into the C : \sapcryptolib directory:
  - For Itanium 64-bit processors, copy the `ntia64` subdirectory contents.
  - For Intel 64-bit processors, copy the `nt-x86_64` subdirectory contents.
  - For Intel 32-bit processors, copy the `ntintel` subdirectory contents.
7. Delete the corresponding subdirectory when files have been moved.
8. If you have installed SAP Mobile Platform SDK, you must add the SECUDIR variable to the following batch file: `SMP_HOME\MobileSDK<version>\Eclipse\MobileWorkspace.bat`.

### See also

- *Installing the SAPCAR Utility* on page 6

### **Setting Up Agentry Application Host System Connectivity**

You must configure the host system to support the connections that are required by Agentry applications.

Within Agentry, "system connection" refers to the connectivity between the Agentry application and the back-end systems with which it synchronizes data. The specifics of the host system configuration in relation to the system connections depends on the types of system connections in use.

There are four different system types that Agentry applications can connect to and synchronize data with. Any application can contain multiple system connections of varying types.

- **SQL Database** – connects the server to either an Oracle or MS SQL Server database. The Agentry application synchronizes data with a database system using ANSI SQL statements.
- **Java Virtual Machine** – connects the server to a system via a Java API. The mobile application contains Java code that calls into this API for the purposes of synchronizing data.
- **HTTP-XML** – connects the server to an HTTP server. The Agentry application makes CGI requests of the HTTP Server. Data returned based on these calls is expected to be in structured XML format. This return data is parsed and processed based on XPath statements within the mobile application.
- **File System** – connects the server to the host operating system. This type of connection allows the server to execute commands on the host system for the purposes of data synchronization and file transfer, and to validate users against the host system where necessary.

Before installing an Agentry application, configure the host system in relation to the system connection types as they pertain to the needs of the application. Additional configuration of the Agentry application itself will be needed after it has been installed. All system connection types support user authentication.

---

**Note:** If any of these resources are already installed on the host system, you do not need to install them again. For example, the SAP Mobile Platform Server installation includes a Java Virtual Machine (*SMP\_HOME\sapjvm7*). This is available for you to use if you do not want to install a separate JVM for your Agentry application.

---

### Establishing Connectivity: Oracle Net Service Names

Create a net service name for use by an Agentry application.

#### **Prerequisites**

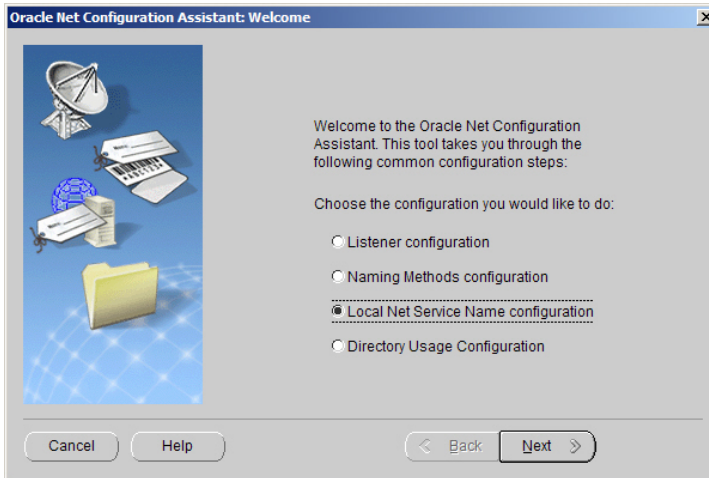
- Verify the proper version of the Oracle client software is installed on the intended host system for the Agentry application.
- Gather the following information:
  - Database service name – the service name or global database name of the database to which the Agentry application is to connect.
  - Communication protocol – the protocol to use for communication between the Agentry Server and the Oracle database. May be one of: TCP, TCPS, ICP, or NMP.
  - Database host network name – the network name of the host system for the Oracle database server. This is needed only if the communications protocol used is TCP, TCPS, or NMP.
  - Database port number – the port number used to communicate with the Oracle database server. This value is only needed if the TCP or TCPS communications protocol is used.
  - Pipe name – the name of the pipe for the database service. This value is only needed if the NMP communications protocol is used.
  - Agentry application login and password – the login and password to the database that used by the Agentry application to connect with the database server.
  - Desired net service name – the Net Service Name by which the connection created is identified on the Agentry application's host system.

#### **Task**

This task uses the Net Configuration Assistant that is included in version 9i of the Oracle client software. If you are not familiar with this tool, review the Oracle documentation for the Net Configuration Assistant and for creating net service names before proceeding.

1. Start the Oracle Net Configuration Assistant, select **Local Net Service Name configuration**, and click **Next**.

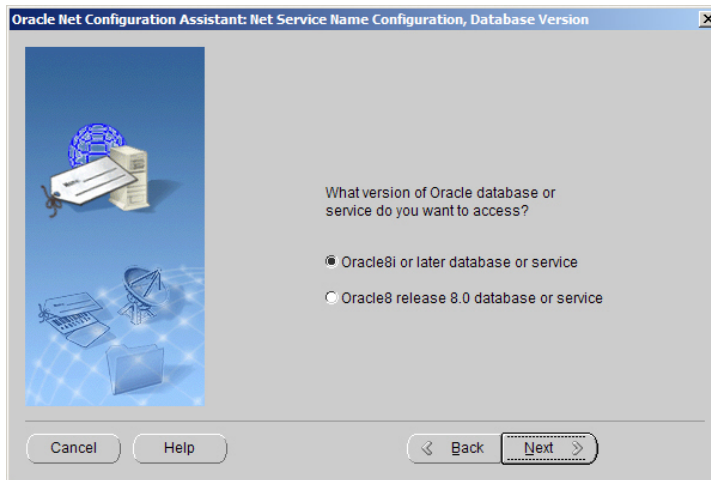
## Getting Started



### 2. Select **Add** and click **Next**.



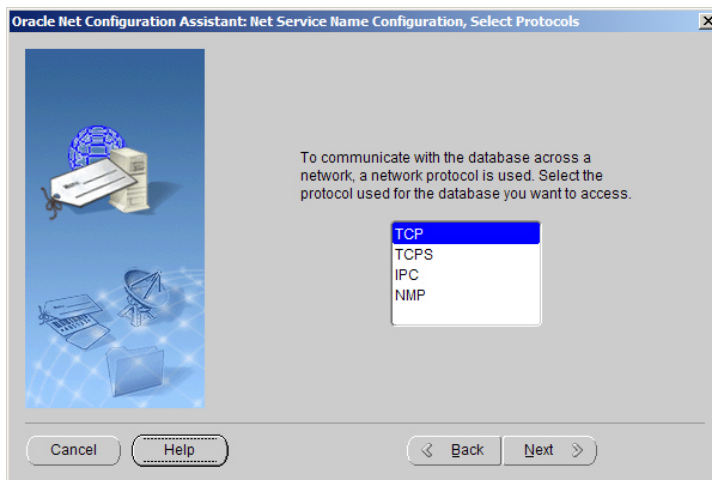
### 3. Select the Oracle database version used in your environment, then click **Next**..



4. Enter the database service name to which the Agentry application will connect, then click **Next**.



5. Select the appropriate protocol for your environment, then click **Next**. Click **Help** for additional information.



6. Enter protocol-specific configuration information:

- TCP or TCPS – the database host computer name for the database to connect to. You must also enter the database port number by entering its value or choosing to use the default value provided.
- IPC – the IPC key value for the local Oracle database service.
- NMP – the database host computer name. You must also enter the database pipe name or choose the default pipe name provided.

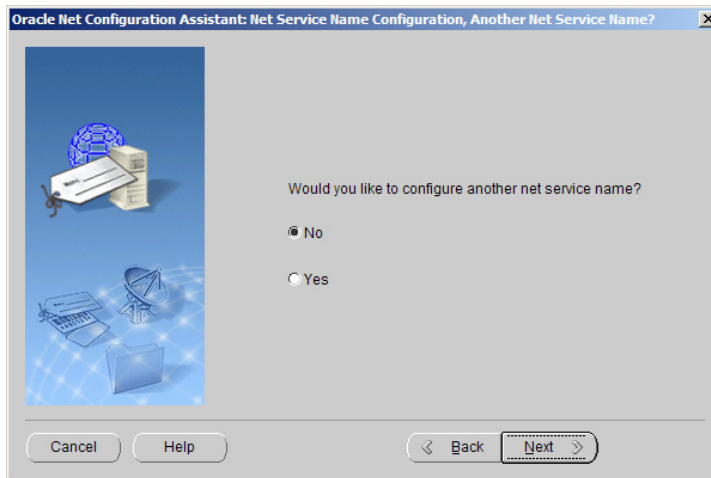
Click **Next**.

7. Test the new net service name to verify all configuration options are correct and to validate the login and password used by the Agentry application application. Follow the instructions on the next screen. As an added test, change the login information to use the login and password intended for use by the Agentry application. When you finish testing, click **Next**.
8. Enter the Net Service Name by which this connection is identified on the system. Make a note of this name, as you will need it to configure the Agentry application to connect to the target database.



Click **Next**.

9. Select **Yes** to add another service name. Select **No** to advance the wizard to the final screen.



Click **Next**, then **Finish** to close the wizard.

An Oracle net service name has now been created on the host system on which the Agency application will be installed. The server uses the net service name to connect to the database with which it will synchronize data for the mobile application.

### Establishing Connectivity: SQL Server ODBC Connections

Create an ODBC data source name (DSN), which is used for connections to a Microsoft SQL Server database system.

#### **Prerequisites**

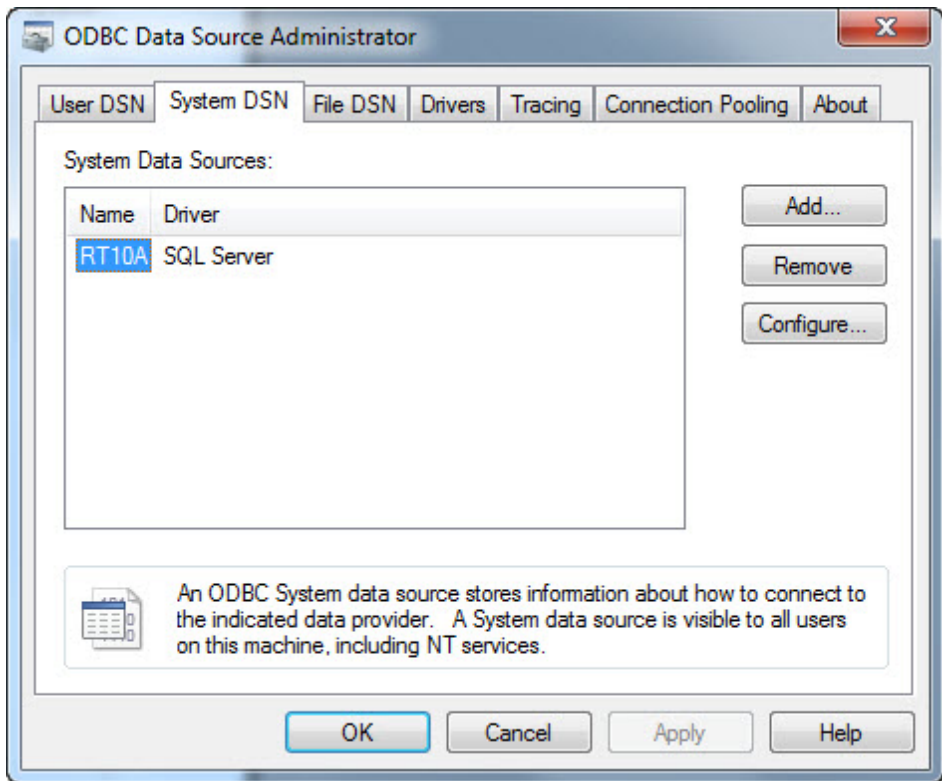
- Install or verify the presence of the proper version of the Microsoft SQL Server ODBC drivers matching the version of the SQL Server database to which the Agency application will connect.
- Gather the following information:
  - DSN name – determine and record the name for the DSN, which is the name by which the ODBC connection for the SQL Server database is identified on the host system.
  - Server network name – the network name of the host system for the SQL Server database.
  - Login authentication method – determine whether a database client’s login and password will be validated using Windows NT authentication, or SQL Server authentication. Determine the proper method before creating the DSN.
  - Login and Password – if the login authentication method will be SQL Server authentication, obtain the login and password of a valid database user.
  - Default database – determine the proper database instance to which a database client connects when using the DSN created in this procedure.
  - Additional options – there are several options available within the Add System DSN wizard that do not usually need to be modified for Agency. However, if special circumstances in an implementation require changes to these options, make such determinations prior to creating the new DSN for the Agency application.

#### **Task**

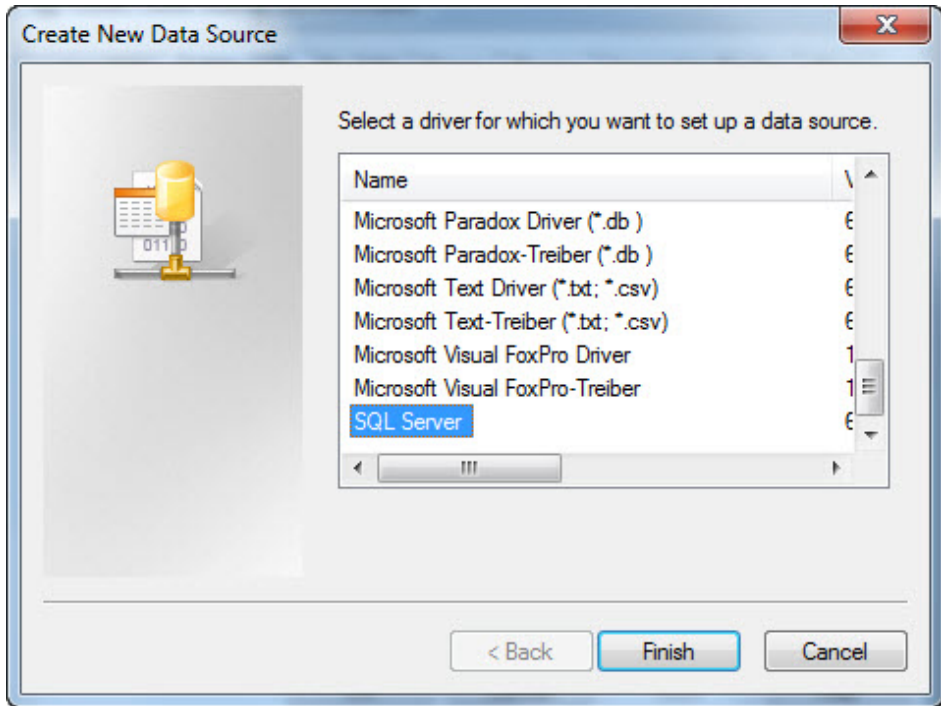
A DSN is created using the Add System DSN wizard provided in the Data Sources (ODBC) utility in Windows. Minimal information is provided here to assist you in completing this task. If you are not familiar with this procedure or the concepts of ODBC, review the Microsoft documentation before proceeding.

1. In Windows, select **Start > Settings > Control Panel > Administrative Tools > Data Sources (ODBC)**. Select the **System DSN** tab.



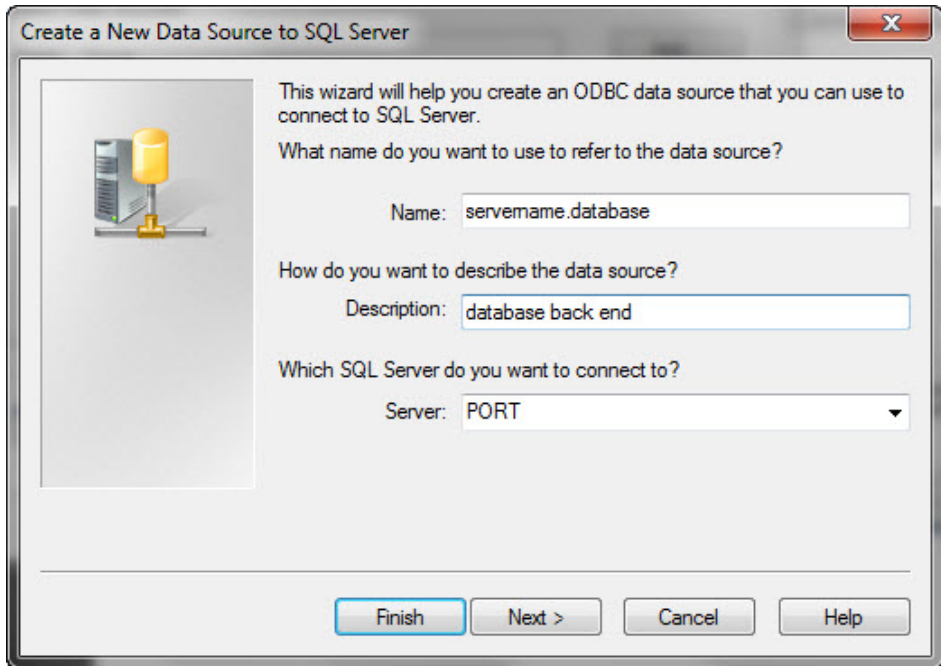


2. Click **Add**.



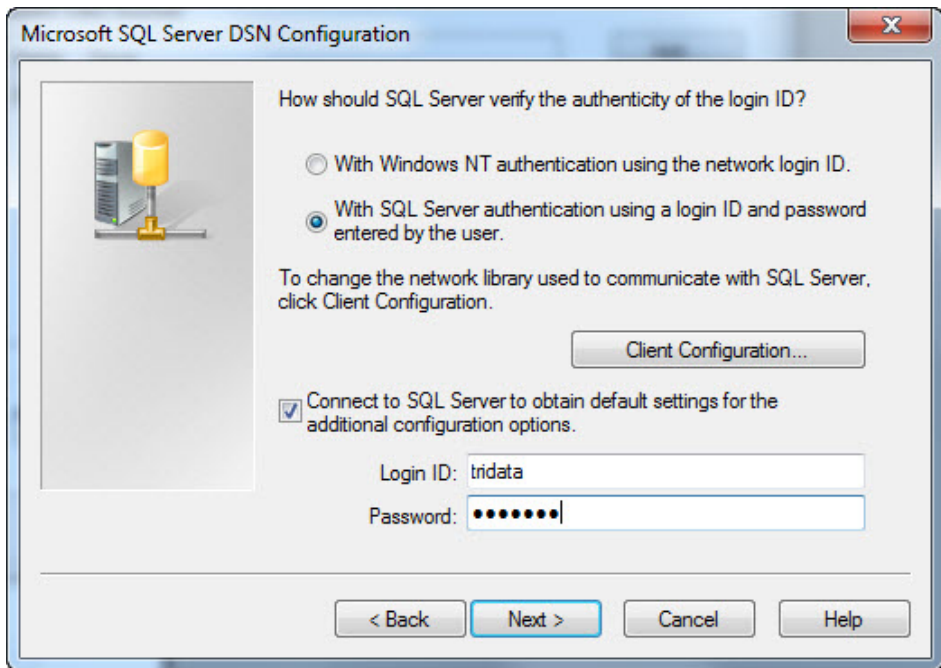
Select the appropriate SQL Server driver and click **Finish**.

3. Enter the DSN name and make a note of it, as you will need it during the Agency application installation. Also enter a description and network name.



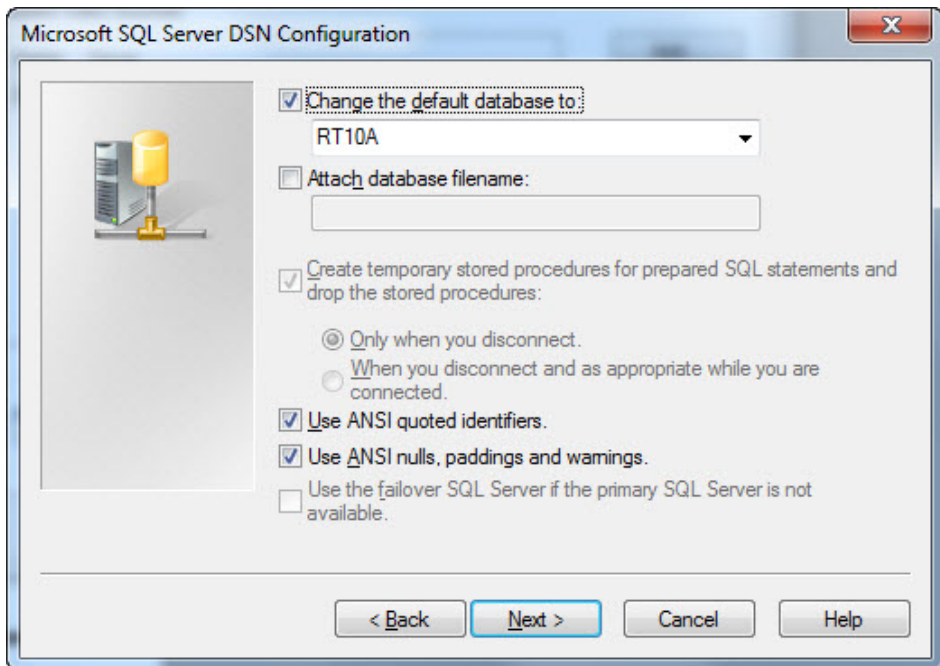
Enter or select the server to which to connect, then click **Next**.

4. Select the authentication method for database clients using this DSN and, if SQL authentication is selected, the login and password for the SQL Server database.



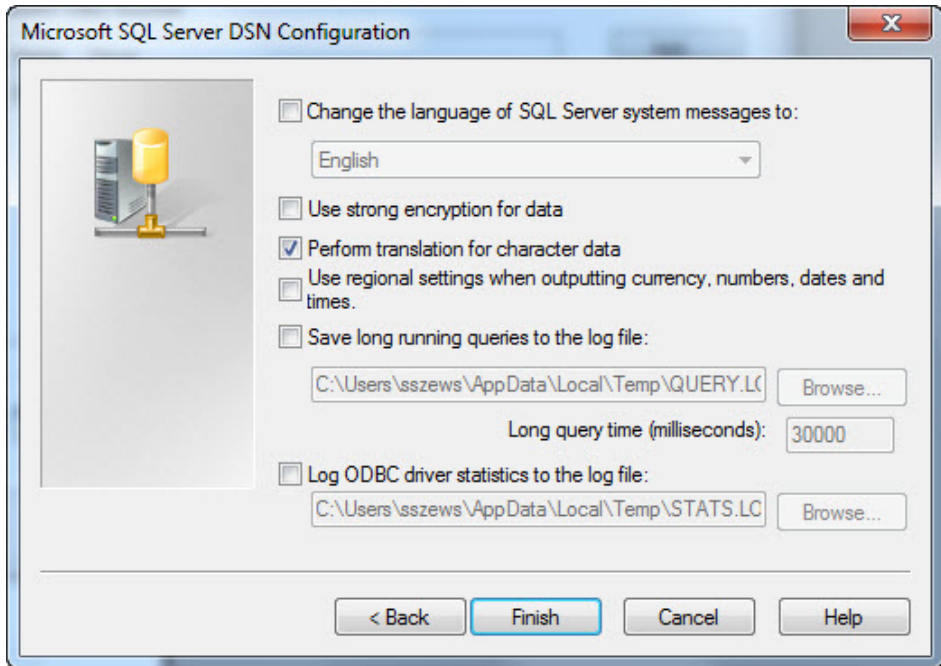
Click **Next**.

5. Set the default database, and, for most situations, leave to the remaining options set to their defaults. Any changes should be made only by an expert user who understands the purpose and resulting behavior of each setting and the overall needs of the implementation environment.



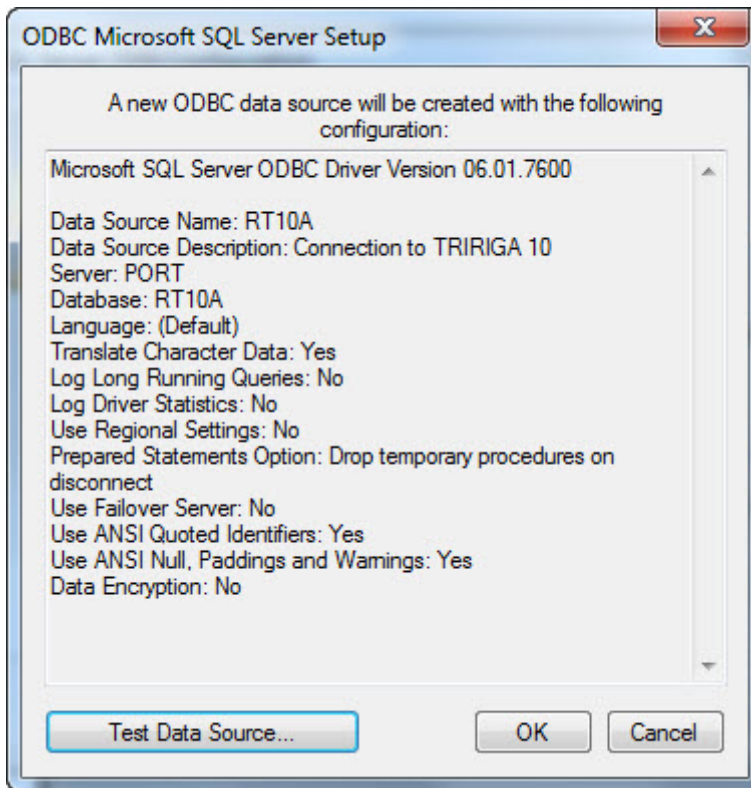
Click **Next**.

6. Set the final options as needed, based on the environment and administrative needs. These settings pertain primarily to locality and logging, and should be changed only by someone who is familiar with their purposes.



Click **Finish**.

7. Review the summary of the DSN configuration and click **Test Data Source**.



Once the test has completed successfully, click **OK** to close this wizard and return to the Data Sources (ODBC) utility.

A new ODBC Data Source Name is created. This DSN will be used by the Agentry application to connect with the MS SQL Server database.

### *Establishing Connectivity: Java Virtual Machine*

When the Agentry application is to synchronize data using a Java Virtual Machine system connection, the requirements of configuring the server's intended host system depend on the needs of the back-end system, and on the nature of the interface with which the server is to communicate.

Due to the wide array of methods of implementing Java systems, it is impossible to provide a single method for all situations. The following task represents the items that always need to be performed, as well as those that may need to be performed for a given implementation.

1. Determine the proper version of Java for the implementation. The minimum supported version of the Agentry Java API is version 1.5. The specific version depends on how the back-end system has been implemented.
2. Obtain the installer for the proper version of Java from Sun's Java Web site:

<http://java.sun.com/javase/downloads/index.jsp>

3. Run the installer, which includes both the Java Development Kit (JDK) and Java Runtime Environment (JRE). Both are required components for the Agentry application. Make a note of the installation locations of both components, as you will need the information when you configure the Agentry application.
4. If the API of the back-end system is exposed via `.class` or `.jar` files, or other similar resources, obtain these files and copy them to the intended host system of the Agentry application. Make a note of the location of these files, as you will need the information when you configure the Agentry application to use the JVM system connection type.
5. Obtain and make a note of the host name, port number, and Java application server or interface name or identifier.

### Establishing Connectivity: HTTP-XML

When the Agentry application is to synchronize data with an HTTP server that returns XML data, little configuration is required prior to installing the Agentry application. The configuration required involves access to the HTTP server. The specific tasks for this vary with each implementation and application.

Most of the tasks involved in connecting the Agentry application to the HTTP server occur during configuration. For details on these post-installation tasks, see *Setting Up the Development Environment for Agentry Toolkit* in *Agentry App Development* and *Agentry Security in Administration*.

### Establishing Connectivity: Host File System

The Agentry application can establish a system connection with the file system of the host computer on which it is installed.

This connection allows the server to read and write files, and transfer those files between clients and the file system. It also enables the Agentry application to execute command line processes on the host system. Finally, the Agentry application can use this connection to read and write text files, using the contents as part of the production data for an application.

To implement a file system connection for the Agentry application, the host system must include a user account under which the Agentry application runs. This account must have appropriate permissions to perform the tasks required by the application. That is, the account must have read/write access to the folders or directories on the file system with which it will be working, and the proper permissions to execute the commands the application has been developed to execute.

Execute all Agentry application operations using this account, including configuring the Windows Service or Linux daemon process to run the server.

## **Adding a SOCKS Proxy or Load Balancer for APNS Connections**

If you have a network-based proxy server between SAP Mobile Platform and your Internet connection to Apple Push Notification Service (APNS), you can enable a path through



network obstacles to the APNS servers by using either a SOCKS proxy in the network proxy server, or a hardware load balancer in the DMZ.

### **Enabling a SOCKS Proxy in a Network Proxy Server**

Specify the proxy server and port number in the SAP Mobile Platform Server `props.ini` file.

#### **Prerequisites**

The network proxy server must have SOCKS capability.

#### **Task**

1. Use a text editor to open `SMP_HOME\Server\props.ini`.

2. Locate these two lines:

```
-DapnsSocksProxyHost=
-DapnsSocksProxyPort=
```

3. Add the information for the network proxy server on which you want to enable SOCKS proxy for APNS connections:

```
-DapnsSocksProxyHost=socksproxy.int.company.corp
-DapnsSocksProxyPort=1080
```

where:

- `socksproxy.int.company.corp` is the network name of the network proxy server.
- `1080` is the network proxy server port number.

Check with your network administrator team to determine the appropriate SOCKS proxy server and port number.

4. Save and close the file.

### **Adding a Load Balancer in the DMZ**

If the DMZ load balancer meets the prerequisite requirement and the firewalls are configured properly, enable the load balancer to support APNS by adding entries in the SAP Mobile Platform Server's `host` file.

#### **Prerequisites**

The DMZ load balancer must:

- Be able to handle SSL connections and pass the client certificate through to the APNS server upon challenge.
- Have firewall rules that allow connection to the APNS server farm. Because APNS uses a load-balancing scheme that results in different IP addresses at different times for the same host name, Apple recommends that you specify the entire `17.0.0.0/8` address block in your

firewall rules. See [https://developer.apple.com/library/ios/technotes/tn2265/\\_index.html#//apple\\_ref/doc/uid/DTS40010376-CH1-TNTAG41](https://developer.apple.com/library/ios/technotes/tn2265/_index.html#//apple_ref/doc/uid/DTS40010376-CH1-TNTAG41).

The steps below have been tested with the two load balancers listed below and should work for other load balancers. For more information on setting up the two load balancers tested, see:

- Citrix Netscaler – <http://support.citrix.com/proddocs/topic/netscaler-ssl-92/ns-ssl-bridging-tsk.html>
- F5 Big IP – [http://support.f5.com/kb/en-us/products/big-ip\\_ltm/manuals/product/lm-implementations-11-1-0/15.html](http://support.f5.com/kb/en-us/products/big-ip_ltm/manuals/product/lm-implementations-11-1-0/15.html)

### Task

1. Log in with administrative privileges to the SAP Mobile Platform Server host machine.
2. Go to the directory where the `hosts` file is located:
  - Windows – `C:\Windows\System32\drivers\etc\`
  - Linux – `/etc/`
3. Back up the `hosts` file.
4. Use a text editor to open the `hosts` file.
5. Add these two lines to the end of the file, mapping IP addresses to the APNS:

```
XXX.XXX.XXX.XXX gateway.sandbox.push.apple.com  
YYY.YYY.YYY.YYY feedback.sandbox.push.apple.com
```

where:

- `XXX.XXX.XXX.XXX` is the gateway push IP address of the load balancer virtual server (LBVS).
- `YYY.YYY.YYY.YYY` is the feedback push IP address of the LBVS.

---

**Note:** Inclusion of ".sandbox" in the domain names is for testing purposes. In a production installation, omit ".sandbox" from each domain name:

```
... gateway.push.apple.com  
... feedback.push.apple.com
```

---

6. Save and close the file.

Now when SAP Mobile Platform Server sends a notification to a device via APNS, the underlying network layers resolve the host address using the `hosts` file, which references the load balancer.

---

## Starting and Stopping SAP Mobile Platform Server

---

Start SAP Mobile Platform Server to access the services that it provides.

## **Starting and Stopping SAP Mobile Platform Server on Windows**

You can start and stop SAP Mobile Platform Server on Windows in different ways.

---

**Note:** If you are using a custom database with SAP Mobile Platform, starting and stopping SAP Mobile Platform Server has no effect on the database. Only the Derby database is automatically stopped and started in sync with SAP Mobile Platform Server.

---

### *Starting SAP Mobile Platform Server*

Use any of these methods to start SAP Mobile Platform Server:

- Desktop shortcut – right-click the **Start SAP Mobile Platform Server** icon on the Windows desktop and select **Run as Administrator**.
- Windows Start menu – select **Start > (All) Programs > SAP Mobile Platform 3.0 > Start SAP Mobile Platform Server**.
- Windows Services Control Panel – start the **SAP Mobile Platform Server** service.

Server start-up may take several minutes. The start-up process is complete when you see:

```
The SMP server has initialized and is ready.
```

### *Stopping SAP Mobile Platform Server*

Use any of these methods to stop SAP Mobile Platform Server:

- Desktop shortcut – double-click the **Stop SAP Mobile Platform Server** icon on the Windows desktop.
- Windows Start menu – select **Start > (All) Programs > SAP Mobile Platform 3.0 > Stop SAP Mobile Platform Server**.
- Windows Services Control Panel – stop the **SAP Mobile Platform Server** service.

## **Starting and Stopping SAP Mobile Platform Server on Linux**

Start and stop SAP Mobile Platform Server on Linux through a terminal window.

---

**Note:** If you are using a custom database with SAP Mobile Platform, starting and stopping SAP Mobile Platform Server has no effect on the database. Only the Derby database is automatically stopped and started in sync with SAP Mobile Platform Server.

---

### *Starting SAP Mobile Platform Server*

You must manually start SAP Mobile Platform Server each time you restart the host system.

1. Open a terminal window.
2. Go to `SMP_HOME/Server/`.
3. Execute `./daemon.sh start`.

Server start-up is complete when you see:

```
The SMP server has initialized and is ready.
```

### *Stopping SAP Mobile Platform Server*

1. Open a terminal window.
2. Go to `SMP_HOME/Server/`.
3. Execute `./daemon.sh stop`.

## **Getting Started with Management Cockpit**

---

Use Management Cockpit to manage server logs and features, and to manage and monitor native, hybrid, and Agentry mobile applications.

### **Starting and Stopping the Management Cockpit on Windows**

You can start and stop Management Cockpit in different ways.

---

**Note:** SAP Mobile Platform Server must be started before you can start Management Cockpit.

---

#### *Starting Management Cockpit from Any Computer on the Network*

On any computer on the network, in a supported browser, enter:

```
https://host_name:https_admin_port/Admin/
```

then log in with the administrative user name and password.

#### *Starting Management Cockpit on the Server Host*

On the server host, you can also use any of these options to start Management Cockpit, then log in with the administrative user name and password:

- Desktop icon – double-click the **SAP Management Cockpit** icon on the Windows desktop.
- Windows Start menu – select **Start > (All) Programs > SAP Mobile Platform version > Management Cockpit**.
- Web browser – in a supported browser, enter:

```
https://localhost:https_admin_port/Admin/
```

#### *Stopping Management Cockpit*

To stop Management Cockpit, click the **Logout** icon, in the upper right corner.

#### **See also**

- *HTTP/HTTPS Port Number Reference* on page 145

## Starting and Stopping the Management Cockpit on Linux

Start and stop Management Cockpit through a browser window.

---

**Note:** SAP Mobile Platform Server must be started before you can start the Management Cockpit.

---

### *Starting Management Cockpit*

- In a browser on any computer on the network, enter:  
`https://host_name:https_admin_port/Admin/`
- Optionally, in a browser on the system where SAP Mobile Platform Server is installed, enter:  
`https://localhost:https_admin_port/Admin/`

### *Stopping Management Cockpit*







To stop Management Cockpit, click the **Logout** icon in the upper-right corner.




### **See also**

- *HTTP/HTTPS Port Number Reference* on page 145

## Using the Home Screen

The Management Cockpit home screen allows the administrator to view and monitor the statistics of all configured applications. It displays the number of applications, users, and registrations. It provides quick links to relevant administration tasks.

Icon	Name	Description
	<b>Applications</b>	Displays the number of configured applications. Click to navigate to the <b>Applications</b> view.
	<b>Users</b>	Displays the number of users using the configured applications. Click to navigate to the <b>Users</b> view.
	<b>Registrations</b>	Displays the number of registrations. Click to navigate to the <b>Registrations</b> view.
	<b>Help</b>	View online documentation.
	<b>Logout</b>	Log out of the application.
<b>Quick links</b> - These icons link to relevant configuration tasks and monitor logs:		
	<b>Configure application</b>	Configure an application. See <i>Defining Applications</i> on page 36.

Icon	Name	Description
	<b>Configure security profile</b>	Configure a security profile. See <i>Managing Security Profiles</i> on page 94.
	<b>Configure back-end connection</b>	Configure back-end connection. See <i>Managing Connections</i> on page 98.
	<b>View logs</b>	View logs. See <i>Setting and Purging Logs</i> on page 103.

**Graphical Representation:**

- **Users/Application:** provides a graphical representation of number of users per application.
- **Registration by Device Type:** displays a pie-chart distribution of registrations by device type.

**Setting Up Browser Certificates for Management Cockpit Connections**

To avoid security exceptions when launching Management Cockpit, set up security certificates correctly.

This task is required when:

- The browser session starts from a host computer that is remote from the Management Cockpit installation.
- The browser session starts on the same computer as Management Cockpit and reports a Certificate Error. The installer automatically sets up a local security certificate, but the certificate installed for https in the web container keystore is a self-signed root certificate, which is not recognized by the client browser.
- The host computer does not have Visual Studio Certificate Manager SDK installed.

Follow the steps below to add the certificate to the Windows certificates store. Alternatively, follow browser-specific instructions to accept the certificate into the Windows certificates store.

**1. Extract the self-signed certificate:**

```
SMP_HOME\JDKX.X.X_XX\bin\keytool.exe -exportcert -alias jetty
-keystore SCC_HOME\services\EmbeddedWebContainer\keystore -file
cert.crt
```

2. Click **Start > Run**, type `mmc`, and then click **OK** to import the `cert.crt` file into the host computer's Windows store with the Windows Certificate Manager. The default password for both the keystore and the alias is "changeit".

**Activating Sample Data for ESPM**

You can use reference data as a test back end for running sample applications. There are sample applications available on SAP GitHub. The Enterprise Sales and Procurement Model

(ESPM) is a simple business model scenario for developing a retail business. It is based on the Enterprise Procurement Model (EPM). In this scenario, the datasource is a JPA model for ESPM Java.

## Prerequisites

- Access to Gateway Management Cockpit for managing the services. For details, see *Gateway Management Cockpit*
- A security profile for the namespace 'sap'.
- Access to SAP GitHub for downloading samples at <http://sap.github.io>

## Task

To activate the sample data for ESPM:

1. Start Gateway Management Cockpit by entering `https://host_name:https_admin_port/gateway/cockpit` in any supported browser.
2. Log in using the administrator user name and password..  
You should see "EspmService" under Registered Services.
3. Configure the JPA destination for the ESPM service:
  - a) Select the **DESTINATIONS** tab.
  - b) Click **Create a New Destination** and enter:

**Table 2. Configuration Values for New Destination**

Field	Description
<Destination Name>	A destination name of your choice for Espmservice, for example, ESPMDestination
<Destination Type>	JPA
<Persistence Unit>	com.sap.espm.model
<Authentication Type>	Basic Authentication
<Username>	gomobile
<Password>	secret

- c) Click **Save**.
4. Assign the destination ESPMDestination for the service or for entity sets. See *Assigning and Removing Destinations* in *Gateway Management Cockpit*.
  5. Activate the Espmservice service:
    - a) Go to the **SERVICES** tab, and verify that ESPMDestination is the default destination for Espmservice.

## Getting Started

- b) Select the `Espmservice` row and click **Activate/Deactivate** to activate the `Espmservice`.

### **See also**

- *Creating and Configuring Security Profiles* on page 165



# Application Administration

Use Management Cockpit and other tools to manage and monitor native, hybrid, and Agentry mobile applications. Managing includes defining and configuring applications; monitoring applications and application usage; viewing statistics and logs; checking system health; and troubleshooting problems.

Native (online and offline), hybrid (Kapsel), and Agentry (offline) applications are developed using a variety of tools and methods. SAP development tools help facilitate development of mobile apps, with modularized methods for download, logon, push notification, and error reporting. During the development process, a unique application identifier is generated for each application, and the application is deployed to an application download site or SAP Mobile Platform Server.

The administrator creates an application definition in Management Cockpit, which includes the unique application identifier, plus the connection to its back-end data source in the production system, the security configuration, and application-specific entries. For Agentry applications, settings are also made in configuration files. This definition enables applications to communicate with SAP Mobile Platform Server.

The administrator provisions applications to devices through native application stores, through enterprise Web site downloads, or through Afaria. When a user logs in to an application (or accesses the application as an anonymous user), the application+user+device combination is registered in Management Cockpit. This registration enables you to manage and monitor device applications in the field using Management Cockpit, and to take advantage of individual and aggregate usage statistics. For Agentry applications, the user enters an Agentry application URL from the device client to access the application. Use Agentry logs to monitor and manage the application in the field.

## See also

- *Server Administration* on page 113
- *Security Administration* on page 149
- *Application Services (Mobiliser) Administration* on page 261
- *SMS Administration* on page 341

## Deploying Applications

---

Create an application definition that enables you to manage the application using Management Cockpit. The application definition includes a unique application identifier,

connections to the back-end data source, security profile settings, and additional application-specific options.

### 1. *Defining Applications*

Create a new native, hybrid, or Agentry application definition, which enables you to use Management Cockpit to manage the application.

### 2. *Defining Back-end Connections*

Define the back-end connection for the selected application. The back-end connection refers to the connection to the data source, or enterprise information system (EIS).

### 3. *Defining Application Authentication*

Assign a security profile to the selected application. The security profile defines parameters that control how the server authenticates the user during onboarding, and request-response interactions with the back end.

### 4. *Defining Client Password Policy*

(Applies only to native and hybrid) Define the client password policy used to unlock the DataVault, for the selected application. The application developer must have added enforcement code to the application DataVault to enforce the policy. The administrator enters the application password policy used to unlock the DataVault during application initialization. Note that this policy does not apply to Agentry or Mobiliser clients, which do not use DataVault.

### 5. *Defining Push Notifications*

(Optional) Configure push-related settings for the selected application. The push listener service provided with SAP Mobile Platform Server allows back-end systems to send native notifications to devices. The application developer must have enabled push notification code in the application to use this option.

### 6. *Uploading Client Resources*

(Optional, applies only to native) Upload client resources, or resource bundles, for the selected application. Resource bundles are containers used by applications to download dynamic configurations, styles, or content from the SAP Mobile Platform Server. The application developer must set up the capability to use client resource bundles through the OData SDK or REST API SDK, before the administrator can modify settings in Management Cockpit.

### 7. *Defining Application-specific Settings*

(Optional, applies only to hybrid and Agentry) Configure application-specific settings for the selected application, using Management Cockpit, configuration files, or other tools.

### 8. *Saving Application Settings*

Save the application settings.

## **Deploying from a Preproduction Environment**

The developer and administrator must coordinate to deploy applications from the development or test environment to the production environment.

### *Hybrid Apps*

To deploy hybrid apps to the production environment:

1. (Administrator) on the preproduction server, select the app in Management Cockpit, and choose **Export**. This generates a ZIP file that contains the app, and retains many of the app settings.
2. (Administrator) in Management Cockpit, import the ZIP file to SAP Mobile Platform production server, and define the hybrid application. If the hybrid app uses the **AppUpdate** plugin, use the **App Specific Settings** tab to manage the app version.
3. (Administrator) continue using Management Cockpit to complete hybrid app configuration tasks in the production environment; for example, security provider settings are not retained as part of the export/import process. Once the application is defined and configured on the server, application users can log in to the client-side application to register the application with the server, and start using the application and resources from the server.

### *Native Applications*

To deploy native applications to the production environment:

1. (Developer) deploy the application to the devices.
2. (Administrator) use Management Cockpit to complete native application configuration tasks in the production environment. Once the application is defined and configured on the server, application users can log in to the client-side application to register the application with the server, and start using the application and resources from the server.

### *Agency Applications*

To deploy Agency applications to the production environment:

1. (Administrator) in Management Cockpit, define the Agency application. If you are publishing an update to an existing application, you do not need this step.
2. (Developer) use Agency Editor to publish the application. The publishing process creates the ZIP file, which contains production definitions and any other files the developer included.
3. (Administrator) in Management Cockpit, complete any additional Agency application configuration tasks for the production environment; use the **App Specific Settings** tab to publish the ZIP file.

The ZIP file is uploaded through the browser to the SAP Mobile Platform Server, and is published. You are notified of success or failure.

4. (Administrator) once the application is defined and configured on the server, application users can log on to the client-side application to register the application with the server, and start using the application and resources from the server.

### See also

- *Defining Applications* on page 36
- *Uploading and Deploying Hybrid Apps* on page 66
- *Publishing Agentry Apps* on page 71
- *Importing an Application* on page 93

## Configuring a Production Environment for Deployment

The administrator must complete several prerequisite configuration tasks before deploying applications into the production environment.

---

**Note:** Work with back-end system and security administrators as needed to complete prerequisite configuration tasks.

---

### *Server Configuration*

Configure SAP Mobile Platform to support application deployment. Tasks include setting up back-end communications, and push notification.

Task	Task Overview
Set up communications between SAP Mobile Platform Server and your back-end systems.	<i>Setting up Back-end Communications</i> on page 5
Configure push notification properties to enable back-end systems to send notifications to SAP Mobile Platform Server.	<i>Configuring Push Notification Properties</i> on page 115

### *Security Configuration*

Plan and implement platform security before deploying applications into the production system. Tasks include planning your security strategy, setting up levels of security from server to device, and setting up security profiles.

Task	Task Overview
Plan your SAP Mobile Platform security landscape, using the security support matrices to plan your strategy and make decisions.	<i>Planning Your Security Landscape</i> on page 149

Task	Task Overview
Secure your SAP Mobile Platform Server installation as soon as possible after installation, using the postinstallation checklist and quick starts as a guide.	<i>Security Postinstallation Checklist</i> on page 158 <i>Platform Security Quick Start</i> on page 159 <i>SAP Mobile Platform Authentication Quick Start</i> on page 159
Configure security profiles, and map to logical roles.	<i>Security Profiles, Roles, and Authentication in SAP Mobile Platform</i> on page 164

**See also**

- *Defining Applications* on page 36

**Production Environment Deployment Tasks By Application Type**

Many application deployment tasks are the same for all application types. However, there are some tasks that are different or do not apply.

---

**Note:** Work with developers to complete prerequisite deployment tasks.

---

Task	Native	Hybrid	Agentry
Define applications	<i>Defining Applications</i> on page 36	<i>Defining Applications</i> on page 36	<i>Defining Applications</i> on page 36
Define back-end connections	<i>Defining Back-end Connections for Native and Hybrid Apps</i> on page 38	<i>Defining Back-end Connections for Native and Hybrid Apps</i> on page 38	<i>Defining Back-end Connections for Agentry</i> on page 41
Define application authentication	<i>Defining Application Authentication</i> on page 58	<i>Defining Application Authentication</i> on page 58	<i>Defining Application Authentication</i> on page 58
Define client password policy	<i>Defining Client Password Policy</i> on page 59	<i>Defining Client Password Policy</i> on page 59	Not applicable
Define push notifications	<i>Configuring Push for Native and Hybrid</i> on page 61	<i>Configuring Push for Native and Hybrid</i> on page 61	<i>Configuring Push for Agentry</i> on page 64
Upload client resources	<i>Uploading Client Resources</i> on page 65	Not applicable	Not applicable

<b>Task</b>	<b>Native</b>	<b>Hybrid</b>	<b>Agentry</b>
Define application-specific settings	Not applicable	<i>Uploading and Deploying Hybrid Apps</i> on page 66	<i>Publishing Agentry Apps</i> on page 71 <i>Configuring Agentry Settings</i> on page 72 <i>Localizing Agentry Applications</i> on page 76
Save application settings	<i>Saving Application Settings</i> on page 83	<i>Saving Application Settings</i> on page 83	<i>Saving Application Settings</i> on page 83

## **Defining Applications**

Create a new native, hybrid, or Agentry application definition, which enables you to use Management Cockpit to manage the application.

1. In Management Cockpit, select **Applications**, and click **New**.
2. In the New Application window, enter:

Field	Value
ID	<p>Unique identifier for the application, in reverse domain notation. This is the application or bundled identifier that the application developer assigns or generates during application development. The administrator uses the Application ID to register the application to SAP Mobile Platform Server, and the client application code uses the Application ID while sending requests to the server. Reverse domain notation means reversing a registered domain name; for example, reverse domain notation for the object <code>MyApp.sap.com</code> is <code>com.sap.MyApp</code>.</p> <p>The application identifier:</p> <ul style="list-style-type: none"> <li>• Must be unique.</li> <li>• Must start with an alphabetic character.</li> <li>• Can contain only alphanumeric characters, underscores (<code>_</code>), and periods (<code>.</code>).</li> <li>• Cannot include spaces.</li> <li>• Can be up to 64 characters long.</li> </ul> <hr/> <p><b>Note:</b> The keywords that are not allowed to be entered as application identifiers include: <code>Admin</code>, <code>AdminData</code>, <code>Push</code>, <code>smp_cloud</code>, <code>resource</code>, <code>test-resources</code>, <code>resources</code>, <code>Scheduler</code>, <code>odata</code>, <code>applications</code>, <code>Connections</code>, <code>public</code>. These keywords are case sensitive.</p> <hr/> <p>Formatting guidelines:</p> <ul style="list-style-type: none"> <li>• SAP recommends that application IDs contain a minimum of two periods (<code>.</code>). For example, this ID is valid: <code>com.sap.mobile.appl</code>.</li> <li>• Application IDs cannot start with a period (<code>.</code>). For example, this ID is invalid: <code>.com.sap.mobile.appl</code>.</li> <li>• Application IDs cannot include two consecutive periods (<code>.</code>). For example, this ID is invalid: <code>com..sap.mobile.appl</code>.</li> </ul>
Name	<p>Application name. The name:</p> <ul style="list-style-type: none"> <li>• Can contain only alphanumeric characters, spaces, underscores (<code>_</code>), and periods (<code>.</code>).</li> <li>• Can be up to 80 characters long.</li> </ul>
Vendor	<p>(Optional) Vendor who developed the application. The vendor name:</p> <ul style="list-style-type: none"> <li>• Can contain only alphanumeric characters, spaces, underscores (<code>_</code>), and periods (<code>.</code>).</li> <li>• Can be up to 255 characters long.</li> </ul>

Field	Value
Type	Application type. <ul style="list-style-type: none"> <li>• Native – native applications, including Android, BlackBerry, iOS, Windows Mobile 8, and Windows 8.</li> <li>• Hybrid – Kapsel container-based applications.</li> <li>• Agentry – metadata-driven application. You can configure only one Agentry application per SAP Mobile Platform Server; after that, Agentry no longer appears as an option.</li> </ul>
Description	(Optional) Short description of the application. The description: <ul style="list-style-type: none"> <li>• Can contain alphanumeric characters.</li> <li>• Can contain most special characters, except percent signs (%) and ampersands (&amp;).</li> <li>• Can be up to 255 characters long.</li> </ul>

3. Click **Save**. You see application-related tabs, such as Back End, Authentication, Push, and so forth. The tabs you see differ by application type. You are ready to configure the application, based on the application type.

---

**Note:** These tabs appear in Management Cockpit only after you define or select an application. The steps that follow assume you have selected the application, and are working through each of the relevant tabs for your selected application. The steps use a "navigational shorthand"—such as "select **Applications > Back End**—to indicate the tab where tasks are performed, relative to that selected application, rather than repeating the entire navigation instruction.

---

#### See also

- *Uploading and Deploying Hybrid Apps* on page 66
- *Publishing Agentry Apps* on page 71
- *Importing an Application* on page 93
- *Deploying from a Preproduction Environment* on page 33

## Defining Back-end Connections

Define the back-end connection for the selected application. The back-end connection refers to the connection to the data source, or enterprise information system (EIS).

#### See also

- *Defining Application Authentication* on page 58

### Defining Back-end Connections for Native and Hybrid Apps

Define back-end connections for the selected native or hybrid application. SAP Mobile Platform supports one primary endpoint per application ID. However, the administrator can create multiple secondary endpoints for other services used by the application; SAP Mobile Platform treats these additional endpoints as proxy connections.



1. From Management Cockpit, select the **Home** tab, and then **Configure Application**. Alternatively, select the **Applications** tab.
2. On the Applications tab, select one of the applications.  
You see application-related tabs, such as Back End, Authentication, Push, and so forth. The tabs you see differ by application type. You are ready to configure the application, based on the application type.

---

**Note:** These tabs appear in Management Cockpit only after you define or select an application. The steps that follow assume you have selected the application, and are working through each of the relevant tabs for your selected application. The steps use a navigational shorthand—such as "select **Applications** > **Back End**"—to indicate the tab where tasks are performed, relative to that selected application, rather than repeating the entire navigation instruction.

---

3. Enter values for the selected application:

Field	Value
Connection Name	<p>(Appears only when adding a connection under Back-End Connections.) Identifies the back-end connection by name. The connection name:</p> <ul style="list-style-type: none"> <li>• Must be unique.</li> <li>• Must start with an alphabetic character.</li> <li>• Can contain only alphanumeric characters, underscores (_), and periods (.).</li> <li>• Cannot include spaces.</li> </ul>
Endpoint	<p>The URL (back-end connection, or service document) the application uses to access business data on the back-end system or service. The service document URL is the document destination you assigned to the service in Gateway Management Cockpit. Include a trailing slash to avoid triggering a redirection of the URL, and losing important HTTP header details. This is especially important when configuring the application with security, such as SSOToken and Certificates, and when Rewrite URL is enabled. Typical format:</p> <pre>http://host:port/gateway/odata/namespace/Connection_or_ServiceName.../</pre> <p>Examples:</p> <pre>http://testapp:65908/help/abc/app1/opg/sdata/TEST-FLIGHT/</pre> <pre>http://srv3333.xyz.com:30003/sap/opu/odata/RMTSAMPLE/</pre>
Use System Proxy	<p>(Optional) Whether to use system proxy settings in the SAP Mobile Platform <code>props.ini</code> file to access the back-end system. This setting is typically disabled, because most back-end systems can be accessed on the intranet without a proxy. Enable this setting only in unusual cases, where proxy settings are needed to access a remote back-end system outside of the network. When enabled, this particular connection is routed via the settings in <code>props.ini</code> file.</p>

Field	Value
Rewrite URL	(Optional) Whether to mask the back-end URL with the equivalent SAP Mobile Platform Server URL. Enable this setting to ensure the client makes all requests via SAP Mobile Platform Server, and directly to the back end. Rewriting the URL also ensures that client applications need not do any additional steps to make requests to the back end via SAP Mobile Platform Server. If enabled, the back-end URL is rewritten with the SAP Mobile Platform Server URL. By default, this property is enabled.
Allow Anonymous Access	<p>(Optional) Whether to enable anonymous access, which means the user can access the application without entering a user name and password. However, the back-end system still requires login credentials for data access, whether it is a read-only user, or a back-end user with specific roles.</p> <ul style="list-style-type: none"> <li>If enabled and the back end requires it, enter the login credential values used to access the back-end system: <ul style="list-style-type: none"> <li><b>User name</b> – supply the user name for the back-end system.</li> <li><b>Password</b> – (required if you set a user name) supply the password for the back-end system.</li> </ul> </li> <li>If disabled (the default value) or the back end does not require it, you need not provide these credentials.</li> </ul> <p><b>Note:</b> If you use Allow Anonymous Access for a native OData application, do not also assign the No Authentication Challenge security profile to the application; anonymous OData requests are not sent, and Status code: 401 is reported.</p>
Maximum Connections	<p>The number of back-end connections that are available for connection pooling for this application. The larger the pool, the larger the number of possible parallel connections to this specific connection. The default and minimum is 500 connections. Factors to consider when resetting this property:</p> <ul style="list-style-type: none"> <li>The expected number of concurrent users of the application.</li> <li>The load that is acceptable to the back-end system.</li> <li>The load that the underlying hardware and network can handle.</li> </ul> <p>Increase the maximum number of connections only if SAP Mobile Platform Server hardware can support the additional parallel connections, and if the underlying hardware and network infrastructure can handle it.</p>
Certificate Alias	If the back-end system has a mutual SSL authentication requirement, supply the certificate alias name given to the private key and technical user certificate that is used to access the back-end system. The alias is located in <code>smp_keystore</code> . Otherwise, leave the entry blank.

4. (Optional) Under Back-end Connections, view additional connections, or add new connections.
  - a) Click **New**, to add additional back-end connections in the server.
  - b) Enter values for the new back-end connection, using the values shown above.
  - c) Click **Save**. The new back-end connection is added to the list.

You can maintain the list of server-level back-end connections (including all the connections in SAP Mobile Platform Server), and of application-specific back-end connections. Application-specific back-end connections are the secondary connections that are enabled for an application; by default, no secondary connections are enabled. You must explicitly enable additional back-end connections for an application. Users who are registered to an application can access only these back-end connections. If a user attempts to access a back-end connection (request-response) that is not enabled for an application, it is not allowed and a 403, Forbidden error is thrown.

5. Select **Application-specific Connections** from the drop-down to show the back-end connections that are enabled for the application.

Select **Server-level Connections** from the drop-down to show all available connections for the server. Use the checkbox to enable additional connections for the application.

---

**Note:**

- You can authenticate multiple back ends using various authentication provider options in the back-end security profile.
  - If the back-end system issues a “302 Redirect” response, which means it is redirecting the request to a different URL, then you must also add the target URL to the list of application-specific connections.
- 

**See also**

- *Certificate Alias* on page 223
- *Managing Keystore and Truststore Certificates* on page 223
- *Configuring Back-end Communications with an HTTP Proxy* on page 116
- *Defining a Back-end Connection* on page 99
- *HTTPS Back-End Connection Properties* on page 118
- *Administrator Overview for Integration Gateway* on page 3

**Defining Back-end Connections for Agentry**

Define back-end connections for the selected Agentry application after publishing the application. Before then, the Back End tab is blank. The application developer creates initial back-end connection settings during development, and then publishes the application to SAP Mobile Platform Server. Once you publish the application, you must modify the initial back-end connection values for the SAP Mobile Platform environment.

Communication between SAP Mobile Platform Server and the back-end system is represented within the application by the back-end connection type. Each type of back end has its own configuration options. Back-end connection types include:

- SQL database connections (both Microsoft SQL Server and Oracle) – between the Agentry application and a single SQL database
- Java Virtual Machine connections – between the Agentry application and custom code (use this option to connect to SAP from Agentry via custom Java code)

- HTTP-XML Server (Web Service) connections – between the Agentry application and an XML-based Web service (use this option to connect to SAP from Agentry via web services)
- File system connections – between the Agentry application and a file system

---

**Note:** Before your Agentry application can communicate with some back-end systems, you need to set up Agentry application host server connectivity.

---

### Configuring SQL Back-end Connections for Agentry Applications

The first time you publish an application from Agentry Editor, the publication process creates initial back-end configuration information, which you must modify using Management Cockpit. Before modifying the configuration information for a SQL database back-end connection, you need to research and gather information about the database.

### **Prerequisites**

- Obtain and record the numeric identifier of the SQL database system connection definition from the Agentry Editor.
- Obtain or create the user ID and password to be used to connect SAP Mobile Platform Server to the database system.
- Obtain information about, or create and configure the necessary connection settings on the host system for, SAP Mobile Platform Server application. This may be the System Data Source Name within ODBC for a SQL Server database, or the Local Net Service Name for an Oracle database.
- Determine whether to authenticate users during synchronization via security information provided by the database.

### **Task**

After modifying the configuration information in Management Cockpit, you may need to perform additional configuration tasks. Certain tasks may or may not pertain to a particular implementation environment and you should perform a careful review of the application's requirements and functionality before proceeding.

1. From Management Cockpit, select **Applications > Back End**.
2. Under *SQL- $n$* , edit SQL-related settings. These configuration options are for a single SQL database system connection, and control the connection behavior between the server and a database system. There may be multiple *SQL- $n$*  sections within the `Agentry.ini` file. There must be one such section for each SQL database system connection definition within the application. Replace the  $n$  portion of the section name with the ID value generated by the Agentry Editor for the system connection definition.

Definitions	Values	Descriptions
Allow User to Change Password	Boolean value of true or false. Default: true.	Whether users can change expiring passwords on the client. If Enable Authentication is false, this setting has no affect.
Database Connection Name	Text value with valid system connection name. Default: none.	The TNS name (Oracle) or ODBC DSN name (such as SQL Server, or any kind of database using an ODBC driver) the server uses to connect with the database.
Database Connection Password	Text value with valid password. Default: set by installer.	The password for the user ID.
Database Connection Type	Oracle or ODBC. Default: set by installer.	Whether the server is connecting to the database using an ODBC or Oracle Net Service Name.
Database Connection User ID	Text value with valid user ID.	The user ID the server uses to log in to the database.
Disconnect from Database	Boolean value of true or false. Default: false.	Whether the server should disconnect from the database at the time specified in Disconnect Time. The value is set to true when the database performs periodic batch or other processing, and when such processing dictates that no connections should be made to the database system.
Disconnect Time	24-hour clock value in hours and minutes. Default: 2:30.	A specified time when the server disconnects from the database. This setting has no effect if Disconnect from Database is false.
Enable Authentication	Boolean value of true or false. Default: true.	Whether users are authenticated against the back-end system for this system connection. At least one system connection within the application must perform user authentication.
Enable Previous User Authentication	Boolean value of true or false. Default: true.	Whether previous users are authenticated against the back-end system for this system connection. This authentication occurs when a user change occurs on the client.
Name	Text value. Default: null.	Any text value used to identify the system connection in log files and other areas. This should be set to a unique value.
Prefetch Rows	1000	Number of rows to retrieve in a single batch from a database query (can be used to optimize speed).

Definitions	Values	Descriptions
Query Constant File	Valid file name or path and file name. Default: none.	Name and location of the query constants file, relative to the server. The default for this setting depends on the type of database selected during installation: <code>Oracle_sd.ini</code> or <code>SqlServer_sd.ini</code> .
Query Init File	Valid file or path and file name to properly formatted query initialization file. Default: <code>SqlBe.ini</code> .	Name and location for the query initialization file used by this system connection. The initialization file contains multiple sections related to various server events that use a database system connection. The file must contain all of these sections in the proper format.
Reconnect Time	24-hour clock value in hours and minutes. Default: 4:00.	The time at which the server reconnects to the database; it must be later than Disconnect Time. This setting has no effect if Disconnect from Database is false.
Shared Connections	Zero or positive integer value. Default: 0.	Number of connections created by the server upon start-up. If greater than 0, a connection pool is created and data from a single client may be processed on any of these connections, including multiple different connections during a single transmission. If set to 0, each client transmission uses a single database connection from start to finish.
Time Zone Name	Valid time zone name.	The time zone to which date and time values are converted when received from clients; or from which they are converted when sent to clients. This conversion is based on the application definitions. This setting is used only to specify a time zone that differs from the back-end system.
User Database Authorization Connection	Boolean value of true or false. Default: true.	Whether the connection that authenticates the user is the same one that processes synchronization for that user. If false, the server uses a different connection.

### Next

If required:

- Make application-specific modifications to the `SqlBe.ini` configuration file.
- Make application-specific additions or modifications to the query constants file, or create additional constants files.

### *Configuring the `SqlBe.ini` Query Initialization File for Agency Applications*

After configuring the SQL database backend connections in Management Cockpit, you may need to make modifications to the `SqlBe.ini` configuration file. The `SqlBe.ini` is the

default query initialization file that is provided by the server installer. You can modify the file for a single application, or use it as a template to create other query initialization files for applications that have multiple SQL database system connection types.

### *Application-Specific Changes*

Query initialization files are generally set up initially by application developers, or as the result of development efforts and standard application needs. Before making any changes to the settings, you should have a full understanding of how your changes might impact application behavior. SAP recommends that, rather than changing the Query initialization file, you instead make your changes to the scripts it references.

Any changes you make, either to the settings in this file or to those scripts it references, are generally the result of specific implementation requirements. Items such as user validation or auditing requirements can dictate when and what modifications may be necessary.

### *SQL Script Locations*

The scripts referenced in the `SqlBe.ini` file must be located in one of two directories, both of which are subdirectories of the Agentry Server within the SAP Mobile Platform installation location (for example, `SMP_HOME\Server\Configuration\com.sap.mobile.platform.server.agentry.application\`):

```
sql
sql\custom
```

These two separate folders allow for both standard and customized processing. For an application implementation, the `sql` folder contains the standard scripts that are provided with the application installer. Make a copy of each script before you make any changes to it, and then save the new version in the `sql\custom` folder.

When processing scripts listed in the `SqlBe.ini` file, SAP Mobile Platform Server looks first in the `sql\custom` folder. If a script is not found in the `\custom` folder, SAP Mobile Platform Server then looks for the same file in the `sql` folder.

To return to default processing, simply move or rename the script file. The next time SAP Mobile Platform Server looks for the script, it finds the default version, and executes it.

### *Sections Listing*

The following is a list of the sections within this file and the events to which they relate:

- `[Misc]` – do not modify this section; it contains two queries, used for internal purposes, that are executed by SAP Mobile Platform Server.
- `[DbConnect]` – a list of SQL scripts to be run when SAP Mobile Platform Server connects to the database. The connection is made when SAP Mobile Platform Server is started, and also when SAP Mobile Platform Server reconnects to the database based on the Disconnect from Database setting in Management Cockpit.
- `[DbDisconnect]` – a list of SQL scripts to be run when SAP Mobile Platform Server disconnects from the database. This occurs as a part of the SAP Mobile Platform Server

shutdown process, and also when SAP Mobile Platform Server disconnects from the database based on the Disconnect from Database setting in Management Cockpit.

- [DbConnectionInit] – a list of SQL scripts that are run when SAP Mobile Platform Server creates a connection to the database. This differs from the DbConnect section, in that DbConnectionInit scripts are run whenever SAP Mobile Platform Server creates a new connection to the database, not just during start-up and reconnect. New connections may be made whenever a user synchronizes the Agentry client, or when SAP Mobile Platform Server adds a connection to the shared connections pool.
- [ValidateUser] – a list of SQL scripts that validate a user against the database during client-server synchronization. These are the primary processing steps for user authentication with a SQL database system connection. These scripts are run after those listed in [DbConnectionInit] and before any step definitions for the application are processed. If the scripts listed here return one of the failed validation values, the user is logged out of SAP Mobile Platform Server, and no application data is synchronized.
- [ValidatePreviousUser] – a list of SQL scripts that validate a previous user against the database during client-server synchronization when a user change has occurred on the Agentry client. These scripts are run after those listed in [DbConnectionInit] and before those listed in [ValidateUser]. If the scripts listed here return one of the failed security validation values, the previous user's data (pending transactions on the Agentry client) is not synchronized. A user change cannot be completed until the previous user's data has been successfully synchronized.
- [LoggedIn] – a list of SQL scripts to be run after a validated the user has logged in successfully.
- [LoggedOut] – a list of SQL scripts to be run just prior to a user logging out of the server, when client-server synchronization has completed.
- [UserInfo] – a list of SQL scripts to be run after those listed in the [LoggedIn] section, and before the step definitions for the application are processed. These scripts are expected to return values, including SQL step definitions within the application, that are then accessible to all other SQL scripts run against the system connection for a user. The values are available via the Syclo Data Markup Language using the data tag `<<user.info.valueName>>`.
- [ApplicationGlobals] – a list of SQL scripts that retrieve override values for global definitions within the application. These values are for the application in general; that is, not user-specific.
- [UserGlobals] – a list of SQL scripts that retrieve override values for global definitions within the application. These values are for a specific user or user group, and allow for overrides on a per-user basis.
- [LoginFailed] – a list of SQL scripts to be run when scripts in [ValidateUser] or [ValidatePreviousUser] indicate the user has failed validation. This section is processed when validation fails for any reason, including the user being blocked.
- [LoginBlocked] – a list of SQL scripts to be run when a user fails validation because he or she has been blocked. The scripts in [ValidateUser] or



[ValidatePreviousUser] must explicitly indicate the user has been blocked. In this case, these scripts are run after those in [LoginFailed].

- [ChangePassword] – a list of SQL scripts that process a password change by the user. The scripts, which perform the necessary processing to change a user's password, are run after a user receives notification of an expired password, or when a password is about to expire. The scripts are expected to return a value indicating success or failure in processing the password change.
- [ChangePasswordFailed] – a list of SQL scripts to run when a user password change fails. The scripts should provide the user with information about why the new password was rejected.
- [EnablePush] – a list of SQL scripts to run when a user performs a transmit and stays logged in for push processing. Specifically, these scripts run when both of the following conditions are met:
  - The Agentry client remains connected after synchronization, as the transmit configuration definition is defined to enable push functionality, and
  - A push definition exists within the application.
- [DisablePush] – a list of SQL scripts to run when a client previously connected to SAP Mobile Platform Server to receive push data is about to disconnect. These scripts are run prior to those listed in the [LoggedOut] section.
- [TimeZone] – a list of SQL scripts that determine the time zone of the database. The time zone returned by these scripts is be used when the application converts date and time values from one time zone to another, based on different settings for database and Agentry client time zones.

### *Configuring Query Constant Files for Agentry Applications*

Query constant files are configuration files containing constant values that may be referenced by any SQL script run by SAP Mobile Platform Server. Each SQL database system connection can have its own query constant file.

By default, there are two such files installed in the SAP Mobile Platform installation directory (SMP\_HOME\Server\Configuration\com.sap.mobile.platform.server.agentry\):

- Oracle\_sd.ini – used by a system connection to an Oracle database systems
- SqlServer\_sd.ini – used by a system connection to an SQL Server database systems

---

**Note:** These default files should not be modified. Instead, copy the files to com.sap.mobile.plaform.server.agentry.application, and modify the copies. Otherwise changes will be lost if the server is upgraded; files in com.sap.mobile.platform.server.agentry could be overwritten by the installer.

---

The values in these scripts are intended to make SQL commands more portable when executed, either from definitions within the application, or from the query initialization file

(`SqlBe.ini`). You can use the values in these files to write a single SQL statement that executes properly throughout the applicable database system.

SDML data tags make the values within these files accessible to SQL statements that are run by the server. Like all other configuration files for the server, the query constant files can contain multiple sections. By default, each contains the single section `[Database]`, which includes these values:

- `name` – do not change this value, which represents the database type.
- `getSystemTime` – the database function that is called to return the current date and time.
- `timeStampFormat` – do not change this value, except in exceptional circumstances. The value represents the format of a date and time value that is inserted into a table column for a date and time data type. This is also the format used by the SDML processor within SAP Mobile Platform Server when a date and time SDML data tag is expanded.
- `dateFormat` – do not change this value, except in exceptional circumstances. The value represents the format of a date value that is inserted into a table column of a date or date and time data type. This is also the format used by the SDML processor within SAP Mobile Platform Server when a date data tag is expanded.
- `timeFormat` – do not change this value, except in exceptional circumstances. The value represents the format of a time value such that is inserted into a table column of a time or date and time data type. This is also the format used by the SDML processor within SAP Mobile Platform Server when a time data tag is expanded.
- `tempdate` – a temporary date value of 1/1/1901 12:00:00, which represents the proper format for the database type to use as a date and time value. The default value is assumed to be unlikely for real data; if it is a valid date and time for the system, change it to reflect an invalid date and time.
- `substring` – the database function that is called to return a range of characters from within a larger string.
- `stringcat` – the database function or operator that concatenates two string values.
- `charFunction` – the database function that converts a value to a character data type.
- `nullFunction` – the database function that determines whether a value is null.
- `singleRow` – Oracle databases require every **SELECT** statement to include a **FROM** clause. When you select literal values, for example, **SELECT 'X'...**, the table object DUAL is used. The `singleRow` value in the Oracle constant file contains the text "FROM DUAL." In the SQL Server script, this value contains an empty string.
- `unicodePrefix` – the value that is prefixed to a Unicode value, that identifies it as such to the database.
- `terminalErrorCodes` – a semicolon-delimited list of return values that are treated as terminal error codes by SAP Mobile Platform Server. When such an error is returned, the server ends the synchronization with the client, and returns an error message.
- `retryWithChangeErrorCodes` – a semicolon-delimited list of return values that may be received from the database system as the result of processing a SQL statement. The

values listed here are treated as a retry-with-change error for the purposes of transaction failure handling. If this functionality is not enabled, this setting has no affect on server behavior.

- `retryWithoutChangeErrorCodes` – a semicolon-delimited list of return values that may be received from the database system as the result of processing a SQL statement. The values listed here are treated as a retry-without-change error for the purposes of transaction failure handling. If this functionality is not enabled, this setting has no affect on server behavior.
- `fatalWithMessageErrorCodes` – a semicolon-delimited list of return values that may be received from the database system as the result of processing a SQL statement. The values listed here are treated as a fatal-with-message error for the purposes of transaction failure handling. If this functionality is not enabled, this setting has no affect on server behavior.
- `fatalWithoutMessageErrorCodes` – a semicolon-delimited list of return values that may be received from the database system as the result of processing a SQL statement. The values listed here are treated as a fatal-without-message error for the purposes of transaction failure handling. If this functionality is not enabled, this setting has no affect on server behavior.

### *Additional Query Constant File Sections*

You can add other sections to the query constants files, which includes values or functionality for application-specific purposes. The contents of this file are usually modified by, or at the direction of, the application developer, as the values are referenced as a part of the application's business logic. Before making any changes, understand the effect that such changes will have, and how the configuration options are used. The default values provided with SAP Mobile Platform Server should only be changed in the rarest of situations, particularly those related to date and time formats.

Examples of application-specific changes:

- The application extends a back-end system that may be deployed on different database systems, and certain values or functionality is different within those systems.
- The application extends a back-end system that is deployed in multiple languages. As a result, certain constant values, such as Y and N flags, are different in different languages. You can represent such values as options in an application-specific section, to become the items referenced via SDML in the SQL statements. When the application is deployed to different languages, you can update the constant file with the proper language-specific value.
- Additional functions may be needed beyond those in the standard version of the query constants file. It may be more straightforward to add a separate section that differentiates between the functions that are provided by default, and those that are added later.

### *Configuring the Query Constant Files for a System Connection*

A single SQL database system connection might use multiple query constant files. You can also change the name of the file that is referenced for the connection.

If a different file name is to be used, or if multiple constant files are needed for a system connection, modify the `queryConstantFiles` configuration option, in the `[SQL-n]` section, of the `Agentry.ini` file. For multiple file listings, each file name is separated by a semicolon.

```
-- different file name
[SQL-1]
...
queryConstantFiles=MyConstantFile_sd.ini
...
--- multiple constant files
[SQL-1]
...
queryConstantFiles=Oracle_sd.ini;MyApplicationConstantFile_sd.ini
```

If you use a query constant file other than the default, the file you use must contain the `[Database]` section, including all configuration options.

### *Configuring Java Virtual Machine Back-end Connections for Agentry Applications*

The first time you publish an application from Agentry Editor, the publication process creates initial back-end configuration information, which you must modify using Management Cockpit. Use the Java Virtual Machine option to connect from an Agentry application to custom code (for example, you could connect to SAP from Agentry via custom Java code). Before modifying the configuration information for a Java Virtual Machine database back-end connection, you need to research and gather information about the database. Certain tasks may or may not pertain to a particular implementation environment and you should perform a careful review of the application's requirements and functionality before proceeding.

### **Prerequisites**

- Obtain and record the numeric identifier for the Java Virtual Machine system connection definition from the Agentry Editor.
- Obtain and record the host system network name of the Java application server to which SAP Mobile Platform Server is to connect.
- Obtain and record the application server name of the Java interface, where applicable.
- Identify any `.jar` or `.class` files that contain business objects that the application being implemented may use, and make them accessible to SAP Mobile Platform Server. These files are normally provided with the back-end system with which the SAP Mobile Platform Server is synchronizing.
- Determine whether users are authenticated using the JVM system connection. If they are, they cannot synchronize unless they have first been successfully authenticated. Proper user credentials must be established within the back-end system before users are allowed access. This may be done before or after configuring the back-end connection in Management Cockpit.

**Task**

1. From Management Cockpit, select **Applications > Back End**.
2. Under JAVA, edit JAVA-related settings. These configuration options are for a single Java Virtual Machine system connection, and control the connection behavior between the server and a Java interface. There may be multiple Java-*n* sections within the `agentyr.ini` file. One must exist for each Java Virtual Machine system connection definition within the application. The *n* portion of the section name must be replaced with the ID value generated by the Editor for the system connection definition.

Definitions	Values	Descriptions
Class Path	Valid fully qualified paths.	Contains multiple path values that specify the location of different Java resources. This can include <code>.jar</code> files used by the application. Each path must be fully qualified. Each entry must be separated by a semicolon. All paths for this setting are added to the system variable CLASSPATH of the server's host system. Requires server restart.
Constants File	Valid file name or path and file name. Default: none.	Name and location of the <code>.ini</code> file, relative to the server, that contains constants that can be accessed from the Java code via the SessionData.
Delete Source (Deprecated)	Boolean value of true or false. Default: false	Controls whether the files stored in Source Directory are deleted after being successfully compiled by the JVM. This may be necessary in a development environment if the Java source files are modified outside of the Agentry Editor. In development mode, the Agentry Editor can push Java source files to the server for the server to compile. Requires server restart.
Enable Authentication	Boolean value of true or false" Default: true	Whether users are authenticated against the back-end system for this system connection. At least one system connection within the application must perform user authentication.
Enable Previous User Authentication	Boolean value of true or false. Default: true	Whether previous users are authenticated against the back-end system for this system connection. This authentication occurs when a user change occurs on the client.
Name	Text value.	Any text value that identifies the system connection in log files and other areas. Set each connection name to a unique value. Requires server restart.

Definitions	Values	Descriptions
Output Directory (Deprecated)	Valid path value, relative to the server.	The location where the compiled .class files produced by the JVM are to be stored for execution. The standard is to use the Java subfolder in the server's installation folder; however, you can use another location. This path value you enter here must also be a part of the system's CLASSPATH (either using a system variable, or by adding the value to the classPath configuration option). This only applies to applications running in development mode; in development mode the Agency Editor can push Java source files to the server for the server to compile. Requires server restart.
Perform Compile (Deprecated)	Boolean value of true or false. Default: true	Whether to compile the .java files into .class files for the Steplet, ComplexTable, and DataTable classes written for the application. A value of false may improve performance in a production environment; the most recently built .class files are reused each time such a class is called within the application. If set to true, each class of the above type is recompiled prior to execution for each user. This only applies to applications running in development mode; in development mode the Agency Editor can push Java source files to the server for the server to compile. Requires server restart.
Print Business Logic Stack Trace	Boolean value of true or false. Default: false	Setting this option to true results in messages generated by the AJ-API exception class <code>JavaBusinessLogicError</code> being printed to the <code>events.log</code> file produced by the server. Requires server restart.
Print Stack Trace	Boolean value of true or false. Default: false	Setting this option to true results in messages generated by the exception stack trace being printed to the <code>events.log</code> file produced by the server. Requires server restart.
Scripts Path	Valid file name or path and file name. Default: none.	Name and location of the scripts file, relative to the server. This is the directory in which Agency Editor places Java source code, when the source code for a Java class is entered directly into the Editor to define a steplet or similar. Requires server restart.

Definitions	Values	Descriptions
Server Class	Valid Java Server Class extension.	The application-specific extension of the AJ-API <code>sync- lo.agentry.server</code> class. This option is automatically set when this API class has been extended by a server class for the application. If no such class exists, the default <code>sync- lo.agentry.server</code> class is assumed and should not be set for this configuration option. Requires server restart.
Source Directory (Deprecated)	Valid path value, relative to the server.	This directory can be used for all other source code on which the Scripts Path might depend. This is the location of the <code>.java</code> files for the <code>Steplet</code> , <code>ComplexTable</code> , and <code>DataTable</code> classes. Files are placed here before they are compiled by the JVM. This is not where the Java project files reside that are related to the application. The <code>.java</code> files are written to this location by the server at runtime. Their contents are based on the definitions in which they are contained. This only applies to applications running in development mode; in development mode the Agentry Editor can push Java source files to the server for the server to compile.
Time Zone Name	Valid time zone name. Default: null	Identifies the time zone to which date and time values are converted when received from clients; or from which they are converted when sent to clients. This conversion is based on the application definitions. This setting is used only to specify a time zone other than the one for the back-end system.

- Restart SAP Mobile Platform Server if you changed any setting that requires it.

### Configuring HTTP-XML Backend Connections for Agentry Applications

The first time you publish an application from Agentry Editor, the publication process creates initial back-end configuration information, which you must modify using Management Cockpit. Use the HTTP-XML option to connect from an Agentry application to an XML-based Web service (for example, you could connect to an SAP Web service back end). After modifying the configuration information in Management Cockpit, you may need to perform additional configuration tasks. Certain tasks may or may not pertain to a particular implementation environment and you should perform a careful review of the application's requirements and functionality before proceeding.

### Prerequisites

- Obtain and record the numeric identifier for the HTTP-XML system connection definition from the Agentry Editor.

- Obtain and record the base URL of the HTTP-XML server to which the SAP Mobile Platform Server will make CGI or HTTP requests. This value is prefixed to the paths of specific requests within the application, as defined in the Agentry Editor.
- Make a note of any specific name spaces that exist in the XML schema to be used.
- Determine the back-end user authentication needs of the application.
- Obtain and record other relevant communication information for the back-end system, including port numbers and network timeout durations.

### Task

1. From Management Cockpit, select **Applications > Back End**.
2. Under HTTPXML, edit HTTPXML-related settings. These configuration options are for a single HTTP-XML system connection, and control the connection behavior between the server and an HTTP server. There may be multiple HTTPXML-*n* sections within the `Agentry.ini` file. One must exist for each HTTP-XML system connection definition within the application. The *n* portion of the section name must be replaced with the ID value generated by the Editor for the system connection definition.

Definition	Values	Description
Authentication Certificate Store	MY	Specifies the location of the authentication certificate for the server. This file may be a Certificate file (.cer), Certificate store file (.sst), or Personal Information Exchange file (.pfx). Used only if Listen On is set to a valid port. Requires server restart.
Authentication Certificate Store Password	None	Specifies the location of the authentication certificate for the server. This file may be a Certificate file (.cer), Certificate store file (.sst), or Personal Information Exchange file (.pfx). Used only if Listen On is set to a valid port. Requires server restart.
Base URL	Valid URL and optional port number (format: <code>http://localhost:81</code> ). Default: none	The base URL address of HTTP requests made by the server for this system connection. Set this option before starting the server.
Basic Authentication Password	None	The basic authentication password.



Definition	Values	Description
Basic Authentication User ID	None	The basic authentication user identification.
Constants File	Valid file name or path and file name to constants file. Default: httpxml_sd.ini	The constants file used for this system connection. This file contains constant named values that may be referenced by the HTTP-XML application components.
Enable Authentication	Disabled	Whether users are authenticated against the back-end system for this system connection. At least one system connection within the application must perform user authentication.
Enable Previous User Authentication	Disabled	Whether previous users are authenticated against the back-end system for this system connection. This authentication occurs when a user change occurs on the client.
HTTP Connect Timeout	60	Time out period, in milliseconds, for this system connection. When the connection attempt exceeds this value, the connection terminates. The timer resets when data is exchanged between server and the back-end system during client synchronization.
HTTP Receive Timeout	300	Time out for receiving the response from the remote Web server.
HTTP Resolve Timeout	60	Time out for resolving the DNS name for a remote connection into an IP address.
HTTP Send Timeout	300	Time out for sending a request to the remote Web server.
Listen On	Disabled	Port to use for connections to the HTTPXML-BE by remote systems to trigger system events in Agency (used for Web service calls). Requires server restart.
Name	Text Value	Any text value used to identify the system connection in log files and other areas. This should be set to a unique value especially when working with multiple system connections. Requires server restart.

Definition	Values	Description
Time Zone Name	None	The time zone to which date and time values are converted when received from clients; or from which they are converted when sent to clients. This conversion is based on the application definitions. This setting is used only to specify a time zone that differs from the back-end system.
Timeout	300 (seconds)	Specifies how long the server will keep a connection open with a client when no messages or responses are received from the client.
Trusted Certificate Store	None	Specifies the location of the trusted certificate store for the server. This store is only used when clients are required to provide authentication certificates during synchronization. Used only if Listen On is set to a valid port. Requires server restart.
Use SSL	None	Whether to use SSL when accepting incoming Web connections. This is used for system event processing for the HTTP-XML back end, since system events for this back end are connections to Agentry's built-in Web server from remote systems. Used only if Listen On is set to a valid port. Requires server restart.
XML Namespaces	None	Registers new XML prefixes (the " <i>localNames</i> ") for XML namespaces that may appear in documents retrieved from remote servers; these prefixes can then be used in XPath expressions that are defined in the Agentry Editor for processing the documents. The format is <code>&lt;localName&gt;   &lt;URI&gt; [   &lt;localName&gt;   &lt;URI&gt; ] *</code> , with as many pairs as needed to define the needed namespaces. Separate each entry using a pipe (   ) symbol.

3. If you modified any values that require a server restart, you must restart SAP Mobile Platform Server before the change takes effect.

### Next

If required, make application-specific additions or modifications to the XML constants file, or create additional constants files. The XML Constants file is used to define values that can be referenced from Agentry Editor XML definitions for Server Data Markup Language (SDML) tags. The structure of the file is similar to the SQL query constants files.

### Configuring File System Back-end Connections for Agentry Applications

For Agentry applications that communicate over a file system connection, use Management Cockpit to configure the SAP Mobile Platform Server to connect to the host system. Certain tasks may or may not pertain to a particular implementation environment and you should perform a careful review of the application's requirements and functionality before proceeding.

#### Prerequisites

- Obtain and record the numeric identifier for the file system connection definition from the Agentry Editor.
- Determine whether users are authenticated during synchronization by logging them in to the host system.
- Obtain and record the Windows User Domain for user authentication.
- Determine the location in which SAP Mobile Platform Server stores temporary script files for execution. The default location is the Windows TEMP folder as configured for the system.
- Determine whether the previous user should be connected to the Windows system when a user change occurs on the clients. You might want to omit this portion of synchronization to expedite the file transfer process.
- Create a user account for SAP Mobile Platform Server to connect to the host system.

#### Task

1. From Management Cockpit, select **Applications > Back End**.
2. Under File, edit file-related connection settings. These configuration options are for a single NT File system connection, and control connection behavior between the server and the server's host system. There may be only one File-*n* section within the `agentry.ini` file. This section must exist for an NT File system connection definition within the application. The *n* portion of the section name must be replaced with the ID value generated by the Editor for the system connection definition.

Definitions	Values	Descriptions
Enable Authentication	Boolean value of true or false. Default: false	Whether users are authenticated against the host system for this system connection. At least one system connection within the application must perform user authentication.
Enable Previous User Authentication	Boolean value of true or false. Default: false	Whether previous users are authenticated against the host system for this system connection. This authentication occurs when a user change occurs on the client. If Allow Previous User Login is false, this setting has no affect.

Definitions	Values	Descriptions
User Domain	Text value of a configured user domain.	The domain under which all users are logged in to the host system. It is only used on Windows, to control which Active Directory domain users are authenticated against by Agentry, when the file back end is configured to do authentication.
Temporary Path	Valid full path.	The location where temporary files produced by the server for file system processing are stored (normally the batch file defined in the application). If this option is null or not included, the configured Windows temporary folder is used.
Allow Previous User Login	Boolean value of true or false. Default: true	Whether previous users are logged in to the file system when a user change occurs on the client. If not logged in, no file system connection processing defined in the application is performed for the previous user.
Name	Text value.	This setting can contain any text value used to identify the system connection in log files and other areas. Set this to a unique value, especially when working with multiple system connections.
PAM Service Name	agentry	Name of the Pluggable Authentication Modules (PAM) service that Agentry should use to authenticate users on UNIX platforms.

## Defining Application Authentication

Assign a security profile to the selected application. The security profile defines parameters that control how the server authenticates the user during onboarding, and request-response interactions with the back end.

### Prerequisites

Configure security profiles for application authentication.

### Task

Security profiles are made up of one or more authentication providers. These authentication providers can be shared across multiple security profiles, and can be modified in Management Cockpit. For more information on authentication providers, see *Authentication in SAP Mobile Platform*.

You can stack multiple providers to take advantage of features in the order you chose; the Control Flag must be set for each enabled security provider in the stack.

1. From Management Cockpit, select **Applications > Authentication**.

2. Click **Existing Profile**.

---

**Note:** You can also create a new profile.

---

3. Select a security profile name from the Name list.

The name appears under Security Profile Properties, and the providers that are associated with the security profile appear under Authentication Providers.

4. Under Security Profile Properties, enter values.

Field	Value
Name	A unique name for the application authentication profile.
Check Impersonation	(Optional) In token-based authentication, whether to allow authentication to succeed when the user name presented cannot be matched against any of the user names validated in the login modules. By default the property is enabled, which prevents the user authentication from succeeding in this scenario.

5. Under Authentication Providers, you can select a security profile URL to view its settings. To change its settings, you must modify it using **Settings > Security Profiles**.

**See also**

- *Defining Back-end Connections* on page 38
- *Stacking Providers and Combining Authentication Results* on page 169
- *Creating and Configuring Security Profiles with Authentication Providers* on page 165
- *Security Profiles in SAP Mobile Platform* on page 164
- *Authentication in SAP Mobile Platform* on page 177

## **Defining Client Password Policy**

(Applies only to native and hybrid) Define the client password policy used to unlock the DataVault, for the selected application. The application developer must have added enforcement code to the application DataVault to enforce the policy. The administrator enters the application password policy used to unlock the DataVault during application initialization. Note that this policy does not apply to Agentry or Mobiliser clients, which do not use DataVault.

The client password policy applies only to the application password used to unlock the DataVault during application initialization, and has nothing to do with SAP Mobile Platform security profiles, or the back-end security systems with which they integrate. Password policies for back-end security systems are administered by customer information technology departments using their native security administration tools.

1. From Management Cockpit, select **Applications > Client Password Policy**.
2. Click Enable Password Policy to display additional fields.
3. Enter values:

Property	Default	Description
Expiration Days	0	The number of days a password is valid before it expires.
Minimum Length	8	The minimum password length required.
Retry Limit	20	The number of retries allowed when entering an incorrect password. After this number of retries, the client is locked out, and the DataVault and all its contents is permanently deleted, rendering the application permanently unusable and all encrypted application data un-accessible permanently.
Minimum Unique Characters	0	The minimum number of unique characters required in the password.
Lock Timeout	0	The length of time in seconds the DataVault may remain unlocked within the application, before the user must re-enter their default password to continue using the application (similar to a screen-saver feature).
Password Properties:	See below	Required password policies.
Default Password Allowed	Disabled	Indicates whether a default password can be generated by the DataVault; from the user's point of view this policy turns off the password.
Has Digits	Disabled	Indicates whether the password must include digits.
Has Lower	Disabled	Indicates whether the password must include lower case letters.
Has Upper	Disabled	Indicates whether the password must include upper case letters.
Has Special	Disabled	Indicates whether the password can include special characters.

**See also**

- *Securing Sensitive Data On-Device with Data Vault* on page 253
- *Device Data Security* on page 252
- *Client Password Policy for Data Vault Logins* on page 254

**Defining Push Notifications**

(Optional) Configure push-related settings for the selected application. The push listener service provided with SAP Mobile Platform Server allows back-end systems to send native

notifications to devices. The application developer must have enabled push notification code in the application to use this option.

### See also

- *Uploading Client Resources* on page 65

### **Configuring Push for Native and Hybrid**

(Optional) Configure push-related settings by device type for the selected native or hybrid application. The application developer must have enabled push notification code for the application, and SAP Mobile Platform Server must be configured to support push.

### Prerequisites

SAP Mobile Platform Server notification properties must be configured:

- A base notification URL. The default value is `http://SMPHostName:8080/`. The host name set here should be accessible from the back end which triggers the notifications.
- (Optional) Proxy host and port used by SAP Mobile Platform Server to send notifications to APNS.

### Task

Use native notifications to allow third-party applications, or a back end to deliver native notifications, directly through the SAP Mobile Platform HTTP notification channel to the device, for example, BlackBerry (BES/BIS), Apple (APNS), or Android (GCM).

1. From Management Cockpit, select **Applications > Push**.
2. Under Apple, BlackBerry, Android, and Windows, make push notification settings for the device or devices you choose.

### See also

- *Configuring Push Notification Properties* on page 115

### **Android Push Notifications**

Configure Android push notifications for the selected application, to enable client applications to receive Google Cloud Messaging (GCM) notifications.

1. From Management Cockpit, select **Applications > Push**.
2. Under Android, enter the access key for API key. This is the access key you obtained for your Google API project (<http://developer.android.com/google/gcm/gs.html>).
3. Enter a value for Sender ID. This is the project identifier.
4. (Optional) Configure push notifications for each device type supported.

### Apple Push Notifications

Configure Apple push notifications for the selected application, to enable client applications to receive APNS notifications.

1. From Management Cockpit, select **Applications > Push**.
2. Under Apple, select **APNS endpoint**. "None" is the default endpoint value for all the applications.
3. Select **Sandbox** to configure APNS in a development and testing environment, or **Production** to configure APNS in a production environment.
  - a) Click **Browse** to navigate to the certificate file.
  - b) Select the file, and click **Open**.
  - c) Enter a valid password.

---

**Note:** The default URL is for a production environment; for a development and testing environment, change the URL to [gateway.sandbox.push.apple.com](http://gateway.sandbox.push.apple.com).

---

4. (Optional) Configure push notifications for each device type supported.

### See also

- *Configuring Push Notification Properties* on page 115

### BlackBerry Push Notifications

Configure BlackBerry push notifications for the selected application, to enable client applications to receive BES/BIS notifications.

### Prerequisites

If you intend to use push synchronization with BlackBerry devices, enable push synchronization in the BlackBerry server. See the BlackBerry server documentation.

### Task

1. From Management Cockpit, select **Applications > Push**.
2. Under Blackberry, select the push type.
  - Select None if you do not want to configure Blackberry push notification.
  - Select BES to configure Blackberry Enterprise Server (BES) native notification properties.

Property	Description
Server URL	Address in the form <code>http://domain_name or IP_address:port_Number/pap</code> .
User	(Optional) User who is accessing the URL.



Property	Description
Password	User password to connect to the URL. If you set a user name, then you are required to enter a password.

- Select BIS to configure Blackberry Internet Server (BIS).

Property	Description
Server URL	Address in the form <code>https://cpXXXX.pushapi.eval.blackberry.com/mss/PD_pushRequest</code>
Listener Port	The push listener port for BIS notifications
Application ID	The unique identifier assigned to the registered push application service
Password	The configuration property provided by BlackBerry for BIS push.

3. (Optional) Configure push notifications for each device type supported.

#### Windows Push Notifications

Configure Windows push notification services (WNS) for the selected application, to enable the back-end servers connected with SAP Mobile Platform to send toast, tile, badge, and raw updates to Windows desktop and tablet application users.

1. From Management Cockpit, select **Applications > Push**.
2. Under **Windows > WNS**, enter the application credentials provided by the application developer.

Property	Description
Package SID	Package security identifier
Client Secret	Client secret information

3. (Optional) Configure push notifications for each device type supported.

#### Windows Phone Push Notifications

Configure Microsoft push notification services (MPNS) for the selected application, to enable the back-end servers connected with SAP Mobile Platform to send toast, tile, badge, and raw updates to Windows phone users running mobile applications.

---

**Note:** Only unauthenticated push notification is supported; authenticated push notification for MPNS is not supported.

---

1. From Management Cockpit, select **Applications > Push**.

2. Under **Windows > MPNS**, select the Enable MPNS Http Push check box to send HTTP push notifications to the device.
3. (Optional) Configure push notifications for each device type supported.

### **Configuring Push for Agentry**

Configure Apple (APNS) and Android (GCM) push-related settings for the selected Agentry application, after publishing the application and restarting SAP Mobile Platform Server. Before then, the Push tab is blank. The Agentry application developer must have configured push for the application via Agentry Editor, and enabled or registered push notification as part of application branding. No SAP Mobile Platform Server configuration settings are required for Agentry push.

For Agentry Editor and Agentry app development information, see SAP Mobile Platform SDK product documentation.

1. From Management Cockpit, select **Applications > Push**.
2. Under Push, edit push-related settings for the Agentry application.

<b>Properties</b>	<b>Values</b>	<b>Description</b>
APNS Certificate Directory	apnsCertificates	SAP Mobile Platform directory used for the Apple Push Notification certificate. Requires server restart.
APNS Certificate Password	None	Password associated with the Apple Push Notification certificate. Requires server restart.
APNS Certificate Password Encoded	Enabled	Whether the Apple Push Notification certificate password should be encrypted. Requires server restart.
APNS Enabled	Enabled	Whether Apple Push Notification is enabled. Requires server restart.
GCM Enabled	Enabled	Whether Google Cloud Messaging is enabled. Requires server restart.
GCM Server Authorization Key	Authorization:key	Google Cloud Messaging server authorization key, if GCM is enabled. Requires server restart.
GCM Server Link	http://android.googleapis.com/gcm/send	Google Cloud Messaging server link URL, if GCM is enabled. Requires server restart.

3. Restart SAP Mobile Platform Server if you changed any setting that requires it.

## Uploading Client Resources

(Optional, applies only to native) Upload client resources, or resource bundles, for the selected application. Resource bundles are containers used by applications to download dynamic configurations, styles, or content from the SAP Mobile Platform Server. The application developer must set up the capability to use client resource bundles through the OData SDK or REST API SDK, before the administrator can modify settings in Management Cockpit.

Keep in mind these resource bundle guidelines:

- **Supportability** – the resource bundle can be of any type (.pdf, .xls, .xml, or any other extension), with no restrictions.
- **Size** – the resource bundle can be of any size, with no restrictions. For best performance, a maximum of 1MB is recommended. For sizes above that, work with the application developer on any performance issues.
- **Default resource bundle** – the first resource bundle uploaded is considered to be the default. After that, you can upload additional versions of the bundle, but only one can be the default. You can delete obsolete resource bundle versions.
- **URL for the default resource bundle** – `http://ServerIP:Port/bundles/ApplicationName/`
- **URL to access other resource bundles** – `http://ServerIP:Port/bundles/ApplicationName/BundleName:BundleVersion`

1. From Management Cockpit, select **Applications > Client Resources**.
2. Under Upload Resource Bundle, enter values.
  - a) Enter the customization resource bundle name in Bundle Name.
  - b) Enter the customization resource bundle version in Version.
  - c) Click **Browse** to upload resource bundle. Select the file to be uploaded, and confirm. The resource bundle name appears under Existing Resource Bundles.
3. Under Existing Resource Bundles, select the resource bundle to make it the default.

### **See also**

- *Defining Push Notifications* on page 60
- *Defining Application-specific Settings* on page 66

### Updating a Resource Bundle

Upload a newer version of a resource bundle. You can make the new version the default, or wait to make the change. Note that you cannot assign a new default resource bundle, and delete an old resource bundle in the same step; you must perform the operation in separate steps.

1. From Management Cockpit, select **Applications > Client Resources**.
2. Select a new resource bundle to upload. It is added to the list.

3. Under Existing Resource Bundles, select the new resource bundle to make it the default. You can leave the old resource bundle in the list, until you are ready to delete it.
4. Click **Save**.

### **Deleting a Resource Bundle**

Delete a resource bundle from the application definition. This may be required if an error is discovered, or a newer resource bundle has been uploaded. Note that you cannot assign a new default resource bundle, and delete an old resource bundle in the same step; you must perform the operation in separate steps.

1. From Management Cockpit, select **Applications > Client Resources**.
2. Select a bundle.
3. Click **Delete**, and confirm. The resource bundle is removed from the list.

## **Defining Application-specific Settings**

(Optional, applies only to hybrid and AGENCY) Configure application-specific settings for the selected application, using Management Cockpit, configuration files, or other tools.

### **See also**

- *Uploading Client Resources* on page 65
- *Saving Application Settings* on page 83

### **Uploading and Deploying Hybrid Apps**

If the selected hybrid app uses the **AppUpdate** plugin, activate the new version from this screen. If the hybrid app does not use the **AppUpdate** plugin, the application-specific settings are not applicable.

### **Prerequisites**

The application developer creates the hybrid app package that:

- Contains the contents of the application's `www` folder and `config.xml` of the project, with a separate folder in the archive for each mobile platform (`android/www` and/or `ios/www` in all lower case). Format structure for hybrid apps:

```
| - android
|   | - config.xml
|   | - www
| - ios
...

```

- Is compressed into a standard `.zip` file for upload.

### **Task**

1. From Management Cockpit, select **Applications > App Specific Settings**.

2. (Optional) If you imported a new application, skip this step. If you are updating the version, click **Browse** to upload another version of the application.

- a) In the dialog, navigate to the directory.
- b) Select the hybrid app package, and confirm.

New version information appears for the uploaded Kapsel app for each mobile platform. You cannot change this information.

Property	Description
Required Kapsel Version	Identifies the Kapsel SDK version used to develop the Kapsel app, for example 3.0.0.  <b>Note:</b> This version attribute is informational only, and is not used by SAP Mobile Platform Server to determine whether device clients should receive the Web application update.
Development Version	Identifies the internal development version used to develop the Kapsel app.
Description	Describes the Kapsel app.
Revision	Identifies the production version revision. For a newly uploaded Kapsel app, this is blank.  <b>Note:</b> When the Kapsel app is deployed, the revision number is incremented.

3. When ready, deploy the application:

- a. Click **Deploy** and confirm to deploy an application to one mobile platform.
- b. Click **Deploy All** and confirm to deploy the application to all available mobile platforms.

Deployed Kapsel app information appears as the current version and the revision number is incremented.

For device application users:

- When a device user with a default version (revision = 0) of the Kapsel app connects to the server, the server downloads the full Kapsel app.
- When a device user with a version (revision = 1 or higher) of the Kapsel app connects to the server, the server calculates the difference between the user's version and the new version, and downloads a patch containing only the required changes.
- If the application implements the **AppUpdate** plugin, the server checks for updates when the application starts-up or is resumed. If the developer has made changes, **AppUpdate** detects them using contents in the `www` folder (that is, the HTML based content), and not with native plugins or changes made outside of that folder. For

changes made outside the `www` folder, the developer needs to post a new copy of the app to the application download site, or use Afaria to push the new app to all users.

4. Remove application version that have been imported, but not yet deployed:
  - a. Click **Remove** and confirm to remove an application from one mobile platform.
  - b. Click **Remove All** and confirm to remove an application from all available mobile platforms.

### See also

- *Defining Applications* on page 36
- *Publishing Agentry Apps* on page 71
- *Importing an Application* on page 93
- *Deploying from a Preproduction Environment* on page 33

### Managing Application Versions Using REST APIs

You can automate application version management by integrating it into your application build or application artifact management processes: use REST APIs to deploy, promote, delete, and view information for application versions.

---

**Note:** The REST API calls must be made using a secure port and the HTTPS protocol. Please see your client documentation for details on how to submit a request over HTTPS to satisfy the server's security requirements.

---

### *Deploying Hybrid Apps Using the REST API*

Deploy a new or updated hybrid app to SAP Mobile Platform Server using the deploy application REST API.

Once the application is deployed, it is considered to be a new version. You can make it the current version using the promotion REST API. After the application is promoted, users can download a patch to upgrade the application on their devices.

---

**Note:** It is not possible to deploy a hybrid app for a specific platform: everything in the file is deployed. Once the application is deployed, you can promote or delete hybrid apps for specific platforms as needed.

---

### Syntax

Perform a POST request to the following URI:

```
https://<host>:<admin_port>/Admin/kapsel/jaxrs/KapselApp/{APP_ID}
```

### Parameters

- **file** – The file that contains the application archive, sent as multipart/form-data.

## Returns

A response providing information about the new and current version of the application. For example:

```
{ "newVersion":
  { "requiredKapselVersion": "1.5",
    "developmentVersion": "1.2.5",
    "description": "An update for the sample app.",
    "revision": -1},
  "currentVersion":
  { "requiredKapselVersion": "1.5",
    "developmentVersion": "1.2.4",
    "description": "A sample app.",
    "revision": 2}
}
```

On successful deployment, the client receives a 201 status code; otherwise, an HTTP failure code and message.

## Examples

---

**Note:** This example uses the `curl` command line client and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

---

- **Deploy application to all platforms**

```
curl --user <user>:<password> --cacert <your-server.pem> --form
"file=@C:\work\appl.zip" https://localhost:8083/Admin/kapsel/
jaxrs/KapselApp/MyTestAppId
```

### *Promoting Hybrid Apps Using the REST API*

Promote a new hybrid app to make it the current version of the application using the promote application REST API.

---

**Note:** You can promote a hybrid app for a specific platform or for all platforms

---

## Syntax

To promote for all platform, perform a PUT request to the following URI:

```
https://<host>:<admin_port>/Admin/kapsel/jaxrs/KapselApp/{APP_ID}
```

To promote for a specific platform, perform a PUT request to the following URI::

```
https://<host>:<admin_port>/Admin/kapsel/jaxrs/KapselApp/{APP_ID}/
{PLATFORM}
```

Include the user name and password in the request to provide authentication for the URI.

After the application is promoted, users can download a patch to upgrade the application on their device.

### **Parameters**

None.

### **Returns**

On successful promotion, the client receives a 200 status code. Otherwise, an HTTP failure code and failure message are returned.

### **Examples**

---

**Note:** This example uses the `curl` command line client and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

---

- **Promote Application**

```
curl --user <user>:<password> --cacert <your-server.pem> --X PUT -  
i https://localhost:8083/Admin/kapsel/jaxrs/KapselApp/MyTestAppId
```

#### *Retrieving Hybrid App Details Using the REST API*

Retrieves details on any new or current versions of a hybrid app using the retrieve application details REST API.

### **Syntax**

Perform a GET request to the following URI:

```
https://<host>:<admin_port>/Admin/kapsel/jaxrs/KapselApp/{APP_ID}
```

Include the user name and password in the request to provide authentication for the URI.

### **Parameters**

None.

### **Returns**

On successful retrieval of details, the client receives a 200 status code. Otherwise, an HTTP failure code and failure message are returned.

### **Examples**

---

**Note:** This example uses the `curl` command line client and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

---

- **Retrieve Application Details**

```
curl --user <user>:<password> --cacert <your-server.pem> --X GET -  
i https://localhost:8083/Admin/kapsel/jaxrs/KapselApp/MyTestAppId
```



### *Deleting Hybrid App Details Using the REST API*

Delete a hybrid app using the delete application REST API.

---

**Note:** You can delete a hybrid app from a specific platform or from all platforms

---

#### **Syntax**

To delete from all platforms, perform a DELETE request to the following URI:

```
https://<host>:<admin_port>/Admin/kapsel/jaxrs/KapselApp/{APP_ID}
```

To delete from a specific platform, perform a DELETE request to the following URI:

```
https://<host>:<admin_port>/Admin/kapsel/jaxrs/KapselApp/{APP_ID}/  
{PLATFORM}
```

Include the user name and password in the request to provide authentication for the URI.

#### **Parameters**

None.

#### **Returns**

On successful deletion, the client receives a 200 status code. Otherwise, an HTTP failure code and failure message are returned.

#### **Examples**

---

**Note:** This example uses the `curl` command line client and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

---

- **Delete application from all platforms**

```
curl --user <user>:<password> --cacert <your-server.pem> --X  
DELETE -i https://localhost:8083/Admin/kapsel/jaxrs/KapselApp/  
MyTestAppId
```

#### **Publishing Agency Apps**

To publish a new or updated Agency application, upload the Agency application ZIP file to SAP Mobile Platform Server. The application developer uses Agency Editor to create the application ZIP file, which contains production definitions, and any other files the developer included. You can configure additional Agency settings, and change the Agency production definition settings made by the developer, once the ZIP file has been uploaded.

1. From Management Cockpit, select **Applications > App Specific Settings**.
2. Under Publish, click **Browse** to locate the Agency application ZIP file created by the application developer. For example, `Northwind-app.zip`.

3. Configure additional Agency settings as needed. Changes to some configuration values may require you to restart SAP Mobile Platform Server for the changes take effect.

**See also**

- *Defining Applications* on page 36
- *Uploading and Deploying Hybrid Apps* on page 66
- *Importing an Application* on page 93
- *Deploying from a Preproduction Environment* on page 33

**Configuring Agency Settings**

Once you upload the Agency application, you can change the Agency production definition settings made by the developer, and configure additional Agency settings..

---

**Note:** The sections that appear and their order vary, depending on how the Agency application has been developed.

---

1. From Management Cockpit, select **Applications > App Specific Settings**.
2. Under Agency, edit Agency application settings.

Property	Default	Description
Keep Alive Delta	3600 (seconds)	Interval during which the server writes something to the various log files, if nothing else has been written during that time period to indicate that the server is not dead. Changes to this property require a server restart.
Shut Down Wait	15000 (milliseconds)	Length of time Agency waits for all of its threads to exit while shutting down, before it stops them.
System Name	Agency	Name of the Agency server, which appears in the title bar of the server's GUI interface on the console. Changes to this property require a server restart.

3. Under Time Zone Alias, map time zone names on remote systems (client devices and back-end systems) to time zone names on this server. Create a list of key-values, one key-value per line, in the format: "key=value". The keys are the time zone names on the remote systems; the values are the equivalent time zone names for the server's host operating system. For example: Central Standard Time=America/Chicago.
4. Under Server, edit administrator contact information and client timeout and notification settings.

Properties	Defaults	Descriptions
Administrator Email	admin@yourcompany.com	Administrator e-mail address.
Administrator Name	Your Name Here	Administrator name.

Properties	Defaults	Descriptions
Administrator Phone	???-???-????	Telephone number for contacting the administrator.
Allow Region	Disabled	Whether to allow regions.
Auto Register	Disabled	Whether to allow automatic registration.
Inactive Timeout	7200 (seconds)	Length of time that an idle user can remain connected to the server before the connection is dropped.
Notify User on Logout	Enabled	Whether to notify the user on log out.
Notify User on Shutdown	Enabled	Whether to notify the user on shut down.

5. Under Configuration, edit settings for the Agency directory and path location for configuration-related files.

Properties	Defaults	Descriptions
Application Globals File	Globals.ini	Name and location of the globals localization file. This setting can be changed when the name of the file is different than the default.
Application Strings File	Application-Text.ini	Name and location of the application strings localization file. This setting can be changed when the name of the file is different than the default.
Client String Names File	ClientString-Names.ini	Name and location of the client string names localization file. This setting can be changed when the name of the file is different than the default. Changes to this property require a server restart.
Client Strings File	ClientText.ini	Name of the client strings localization file. This setting can be changed when the name of this file is different from the default.

Properties	Defaults	Descriptions
Enable Failed Transaction Logging	True	<p>Whether to enable the failed transaction queue, which is a part of the transaction failure handling functionality. By default, when transaction failure handling is enabled, transactions that fail with a fatal error code result in the creation of a failed transaction file on the server containing the data from that transaction. If this option is set to false, the file is not generated.</p> <p>All remaining transaction failure handling behaviors are carried out, including the removal of the failed transaction from the client.</p> <hr/> <p><b>Note:</b> Setting this option to false might result in lost data for fatal errors that are related to transaction processing. This option has no affect if transaction failure handling is disabled (that is, <b>enableTransactionFailureHandling</b> is false).</p>
Enable Override File	Enables.ini	Name and location of the enables localization file. Change this setting only if the file name is different from the default.
Enable Transaction Failure Handling	False	Whether to enable the transaction failure handling functionality within the Agentry platform. This option must be true for all aspects of this functionality to be enabled.
Failed Transaction Filename Format	Server-generated XML file name	Format of the file name for each failed transaction queue file generated by the server. The format strings <code>{userid}</code> , <code>{transaction_name}</code> , <code>{date}</code> , <code>{time}</code> , and <code>{count}</code> may be used. The file generated is always be in XML format, and the file extension provided ( <code>.xml</code> ) should reflect this content type.
Failed Transaction Queue	FailedTransactionsQueue	Location of the failed transaction queue files. This path is relative to the installation folder of the server.
Images Path	\Application\Development_or_Production\Images	Location of image files in either the development or production environment.

Properties	Defaults	Descriptions
Localization Path	localizations	Location of localization files for use in the application. This path is relative to the server's installation folder. If the default folder "localizations" does not exist, it must be created.
Localizations	None	One or more local names, according to the ISO 639-1 standard. This option lists the supported languages, which also require corresponding override localization files, for an application. Use a semi-colon to separate each name.
Posted Transaction Directory	posted	Directory where information about pending transactions is stored temporarily, so it is not lost if the server fails. The default usually does not need to be changed.
Posted Transaction File Name Pattern	%{device}.pli	Pattern that constructs the names of files in the posted transaction directory. The default usually does not need to be changed.
Scripts Path	Application \Development \Scripts	If you use an Server Data Markup Language (SDML) tag in the Agentry application that includes content from a file using a relative path, this is the directory under which the file in question needs to be located.
Spin Doc INI File	None	If this value is specified, then the [SpinDoc] configuration is loaded from the named .ini file instead of from Agentry.ini.
Transmit Configuration File	TransmitConfigurations.ini	Name and location of the transmit configurations override file. Change this setting if the file name is different from the default.
Trusted Certificate Store	None	Location of the trusted certificate store for the server. This store is used only when clients are required to provide authentication certificates during synchronization.
urlPath	None	Overrides the path portion of the WebSockets URL (for example, if the URL is <code>https://server:8081/MyApp</code> , this value overrides <i>MyApp</i> , which is the default application name.

- Under SPINDOC, edit the values that are exposed to the Agentry application as variables [Server Data Markup Language (SDML)], and the "Face Path" setting.

Properties	Defaults	Descriptions
Face Path	sql;sql \custom	The Face Path setting defines the search path used by the SQL back end to locate the query files, which are used for database initialization, authentication, password changes, and so forth; and the SQL script files, which are used for SQL back-end processing. If this setting is missing, there is no default to use, so processing may fail.

- If you modified any values that require a server restart, you must restart SAP Mobile Platform Server before the changes take effect.

### See also

- *Localizing Agency Applications* on page 76
- *ClientText.ini* on page 78
- *Globals.ini Format* on page 79
- *ApplicationText.ini Format* on page 80
- *Enables.ini Format* on page 82
- *Localization Files* on page 77

### Localizing Agency Applications

Agency applications provide full support for localizing mobile applications without directly modifying the application project definitions. You can localize the application for one or more languages using localization override files, which replace the display values that are defined for the application. You can also use localization overrides for industry- or deployment-specific terminology.

Localization base files are created by the Agency Editor when an application is published to the server. The initial contents of these files are based on the application definition:

- `ClientTextBase.ini`
- `GlobalsBase.ini`
- `ApplicationTextBase.ini`

To translate an application's client UI into a language other than the one in which it was developed, the translator copies and modifies the localization base files to create override files that contain a localized version of the display values. These files are read and processed by the server during startup. The override values are then sent to clients during synchronization, after application definitions are synchronized but before production data is processed.

For single-language support, translators need to create only one set of override files. The single-language override files, which are specified in Management Cockpit, are used as the default override files if multiple languages are supported.

For multiple-language support, the translator creates additional override files, one for each locale that is supported by the application. The override file name must use the format `<default_override_filename>.<locale_name>.ini`. Copy the override files

to the localization folder for the application. Specify the locales supported by the application and the localization folder for the override files in Management Cockpit.

Restart the server to load the new override values. For example, if:

- Application Globals File = AppGlobals.ini
- Application Strings File = AppStrings.ini
- Client Strings File = ClientTextStrings.ini
- Localization Path = localizations
- Localizations = en\_US;es\_ES

The server loads and processes these files on startup:

- AppGlobals.ini (default)
- AppStrings.ini (default)
- ClientTextStrings.ini (default)
- localizations/AppStrings.en\_US.ini (en\_US)
- localizations/ClientTextStrings.en\_US.ini (en\_US)
- localizations/AppGlobals.es\_ES.ini (es\_ES)
- localizations/AppGlobals.es\_ES.ini (es\_ES)
- localizations/AppStringss.es\_ES.ini (es\_ES)
- localizations/ClientTextStrings.es\_ES.ini (es\_ES)

The server uses the locale information provided by the Agentry clients to determine the proper override values, if any, to use. If the Agentry client locale setting does not match the supported locales for the application, the values from the default localization files are used. If no localization override files are specified, then the values that are defined in the published application project are sent to the client.

### See also

- *ClientText.ini* on page 78
- *Globals.ini Format* on page 79
- *ApplicationText.ini Format* on page 80
- *Enables.ini Format* on page 82
- *Localization Files* on page 77
- *Configuring Agentry Settings* on page 72

### Localization Files

Application localization uses two file types: localization base files are created automatically by Agentry Editor during a publish to the server, and localization override files override the values in the base files with localized values. Both file types are formatted in plain text; you can use a standard text editor to open and edit them.

When creating the localization override files:

- Create override files only for the files that contain values you want to localize.
- You can have a mix of localized and unlocalized values within each file.
- Values in, and the behavior dictated by both file types always defaults back to the application definitions within the Agentry application project, if they are not defined in the override files.

### See also

- *Localizing Agentry Applications* on page 76
- *ClientText.ini* on page 78
- *Globals.ini Format* on page 79
- *ApplicationText.ini Format* on page 80
- *Enables.ini Format* on page 82
- *Configuring Agentry Settings* on page 72

### *ClientText.ini*

The `ClientText.ini` file contains display values that are part of every Agentry client, regardless of the application.

This is an example of the contents of the `ClientText.ini` and `ClientTextBase.ini` files.

```
[Strings]
AG3_ABOUT_BMP_FILE=about.bmp
AG3_ABOUT_DIALOG_TEXT=About %1
AG3_ABOUT_MENU=&About %1...
AG3_ABOUT_TXT=about.txt
AG3_ABOUT_NAME=Agentry Client
AG3_ABOUT_OK=OK
AG3_ABOUT_PVERSION=Published As: v
```

The first line is the section name `[Strings]`, the only section in this file. The same `ClientText` and `ClientTextBase.ini` files are used by all applications that are deployed on the same version of the SAP Mobile Platform Server for a given language. The values here override the built-in components of the Agentry client and are not application-specific.



The items in the above example are those that appear when the user selects **Help > About** on the client. The keys for all items begin with `AG3_`, followed by additional text that identifies the item to which the keys pertain.

The general naming convention of the keys is:

```
AG3_CATEGORY_COMPONENT
```

`CATEGORY` represents general resource (such as a built-in screen), or definition type, (such as items related to actions). `COMPONENT` describes the specific component identified by the `CATEGORY`. So, in the above examples, `ABOUT` refers to the About built-in screen, and `NAME` refers to the name value that appears in the About screen.

The values (that is, the text that follows the equal signs in each key) are the built-in display values or resources used by the client. Changing the value in the file changes the item that appears on the client. For example, `AG3_ABOUT_NAME` has a value of `Agentry Client`, which appears in the About screen. You might want to change this value if you rebrand your application.

### See also

- *Localizing Agentry Applications* on page 76
- *Globals.ini Format* on page 79
- *ApplicationText.ini Format* on page 80
- *Enables.ini Format* on page 82
- *Localization Files* on page 77
- *Configuring Agentry Settings* on page 72

### *Globals.ini Format*

The `Globals.ini` file contains the group, name, and defined values of all global values defined within the application.

This is an example of the contents of the `Globals.ini` and `GlobalsBase.ini` files:

```
[STRINGLENGTHS]
```

```
; AddressMax: This is the maximum length for an address value.
```

```
AddressMax=40
```

```
; AddressMin: This is the minimum length for the Address value.
```

```
AddressMin=1
```

```
; CityMax: This is the maximum length for the City value.
```

```
CityMax=12
```

```
; CityMin: This is the minimum length for the City value.
```

```
CityMin=1
```

```
; PhoneMax: This is the maximum length for the Phone value.
```

```
PhoneMax=22
```

```
; PhoneMin: This is the minimum length for the Phone value.
```

```
PhoneMin=1
```

The first line is the section name [STRINGLENGTHS], which corresponds to a defined global group within the application project. Each global group appears as a section within this file. All global values defined within a group are listed within the appropriate section.

The second line is a comment line. All comments are preceded by a semicolon, and are automatically generated by the Agentry Editor when the file is created. The comments are the description values of each global definition. You can add comments to the base file and override file as needed.

The third line, AddressMax=40, is the first global definition within the StringLengths group. The name of the global definition to which this setting pertains is AddressMax. Its defined value within the application project is 40, which means that any client Address field cannot exceed 40 characters.

Change these values to override the defined behavior for the application. More specifically, the value entered in the override file is the one for the global on the client, affecting any other definitions that reference the global value.

### See also

- *Localizing Agentry Applications* on page 76
- *ClientText.ini* on page 78
- *ApplicationText.ini Format* on page 80
- *Enables.ini Format* on page 82
- *Localization Files* on page 77
- *Configuring Agentry Settings* on page 72

### *ApplicationText.ini Format*

The ApplicationText.ini file contains one section for each module defined within the application, and an additional section for the application definition itself. Every module

definition includes a display name, label, or caption attribute. The key portion identifies the module definition, and the value contains the display name text.

This is an example of the contents of the `ApplicationText.ini` and `ApplicationTextBase.ini` files:

```
[Customers]

module=Customers

object.MainObject=Main Object

property.MainObject.Customers=Customers

object.Customer=Customer

property.Customer.CustomerID=ID

property.Customer.CompanyName=Company

property.Customer.ContactName=Contact
```

---

**Note:** The contents of this file are application-specific. The above example is for a mobile application built for the Northwind demonstration database in SQL Server.

---

This override file contains one section for each module defined within the application, named [*ModuleName*], and one section named [*Misc*] for application-level definitions.

The second line in the above example, `module=Customers`, represents the display name of the module. Each override item's key follows a defined format:

```
defType.ModuleLevelParent.parent.parent.definitionName
```

- `defType` – type of definition that is overridden by the item. Examples include:
  - `property`
  - `object`
  - `platform`
  - `listscreencaption`
- `ModuleLevelParent` – name of the module-level parent definition, of the definition whose display value is overridden.
- `Parent.Parent` – name of each ancestor definition, between the module level ancestor, and the definition that is overridden.
- `definitionName` – definition name, which is overridden.

The last line in the above example, `property.Customer.ContactName`, contains the override value for the display name of the property in the `Customer` object named `ContactName`. This file includes definitions within the application that have a display name, label, or caption attribute.

### See also

- *Localizing Agentry Applications* on page 76
- *ClientText.ini* on page 78
- *Globals.ini Format* on page 79
- *Enables.ini Format* on page 82
- *Localization Files* on page 77
- *Configuring Agentry Settings* on page 72

### *Enables.ini Format*

The `Enables.ini` file contains one section for each module defined within the application. Each section lists all actions, pushes, detail screen fields, and list screen columns that are defined within the module. The value of each item is either `true` or `false`. It is likely that most of the values are set to `true`, unless the definition is disabled in the application.

The override file created from this base file can contain items with a value of `false` to disable the corresponding definition, or `true` to enable it. The effect of disabling a definition depends on the definition type. Disabled fields and columns do not appear on the client. Users cannot execute disabled actions. A disabled push results in all behaviors related to that push being disabled. If there are no enabled pushes, and background sending is not enabled, users do not remain logged in to the server, regardless of the transmit configuration definition.

This is an example of the contents of the `Enables-` and `EnablesBase.ini` files:

```
[Customers]

screencolumn.ShowCustomers.ShowCustomers_List_PPC.CustomerID=true

screencolumn.ShowCustomers.ShowCustomers_List_PPC.CompanyName=true

action.AddCustomer=true

action.EditCustomer=true
```

The contents of this file are application-specific. The above is an example for a mobile application built for the Northwind demonstration database in SQL Server.

This override file contains one section name, `[ModuleName]`, for each module defined within the application.

The following line:

```
screencolumn.ShowCustomers.ShowCustomers_List_PPC.CustomerID=true
```

represents the screen column (screencolumn) CustomerID in the ShowCustomers\_List\_PPC list screen, which is a child definition to the ShowCustomers screen set. Each override item's key follows a defined format:

```
defType.ModuleLevelParent.parent.parent.definitionName
```

- defType – definition type being overridden by the item. Examples include:
  - action
  - screencolumn
  - field
- ModuleLevelParent – name of the module-level parent definition, of the definition whose enabled state is being overridden.
- Parent.Parent – name of each subsequent ancestor definition, of the definition being overridden is a part of the key.
- definitionName – name of the definition being overridden.

### See also

- *Localizing Agentry Applications* on page 76
- *ClientText.ini* on page 78
- *Globals.ini Format* on page 79
- *ApplicationText.ini Format* on page 80
- *Localization Files* on page 77
- *Configuring Agentry Settings* on page 72

## Saving Application Settings

Save the application settings.

1. Click **Save** after you enter all mandatory fields mentioned under application Status, and any additional configuration settings.  
The application status turns green, which indicates the application is saved and consistent.
2. Verify the application settings in the **Application > About** section. If successful, the application status is green.
3. After the initial save, you can configure additional settings for the selected application. Also, once the application has registered users, a new Overview tab appears, which provides application usage information.

### See also

- *Defining Application-specific Settings* on page 66

## Managing and Monitoring Applications

Use Management Cockpit to manage applications, registrations, users, back-end connections to the data source; manage security profiles; view application usage statistics; and manage and view application reports.

### Managing and Monitoring Tasks By Application Type

Many of the application managing and monitoring tasks are the same for all application types. However, there are some tasks that are different or do not apply for a particular application type. Most of these differences are for Agentry applications.

Task	Native	Hybrid	Agentry
Managing applications	<i>Managing Applications</i> on page 84	<i>Managing Applications</i> on page 84	<i>Managing Applications</i> on page 84
Managing registrations	<i>Managing Registrations</i> on page 88	<i>Managing Registrations</i> on page 88	Not applicable
Managing users	<i>Managing Users</i> on page 92	<i>Managing Users</i> on page 92	Not applicable
Importing and exporting applications	<i>Importing and Exporting Applications</i> on page 92	<i>Importing and Exporting Applications</i> on page 92	<i>Importing and Exporting Applications</i> on page 92
Managing security profiles	<i>Managing Security Profiles</i> on page 94	<i>Managing Security Profiles</i> on page 94	Not applicable – Agentry includes an additional security level
Managing Application Logs	<i>Managing Application Logs</i> on page 103	<i>Managing Application Logs</i> on page 103	<i>Viewing Agentry Logs</i> on page 133 (Agentry logs are viewed under server logs)
Reporting usage statistics	<i>Reporting Usage Statistics</i> on page 102	<i>Reporting Usage Statistics</i> on page 102	Not applicable

### Managing Applications

Manage multiple native, hybrid, and Agentry applications from a single location. You can add, edit, or delete applications; and ping a selected application for back-end connection and authorization URL status. View information for all applications, or retrieve a filtered subset of applications. From the list of applications, you can drill down to see application ID and application registration details; and a sequential log of request execution details.

1. From Management Cockpit, select **Applications** on the Home screen to view applications. Alternatively, in the **Applications** tab, click **Applications**.

Field	Value
Application ID	Unique identifier for the application, in reverse domain notation. This is the application or bundled identifier that the application developer assigns or generates during application development. The administrator uses the Application ID to register the application to SAP Mobile Platform Server, and the client application code uses the Application ID while sending requests to the server.
Name	Application name.
Type	Application type.
Vendor	Vendor who developed the application.
Status	Indicates whether the application is consistent, meaning its back-end connection is valid.
Registrations	The number of current user registrations to the application. Registered applications are associated with a user, anonymous user, or nossec_identity on one or more devices. When the user accesses the application from a device the application is registered. A zero (0) indicates there are no current user registrations to the application.
Logs	A list of log statements captured during request execution (such as GET, SET, PUT, and so forth). The data is collected and persisted in the database depending on the Application log level setting, and is a useful troubleshooting tool for debugging an application problem.

2. (Optional) Use filtering and sorting options to display a subset of applications:
  - In **Filter By**, select ID, Name, Type, or Vendor, and enter a name or number to search for a specific entry.
  - Use column sorting and filtering to display only records that meet your criteria.
  - Click **Refresh** to refresh the data.
3. (Optional) View details about a specific application registration:
  - Select an Application ID URL to view application usage details for the application. The **Applications > Overview** screen appears, if the application is registered; otherwise **Applications > Backend** appears.
  - Select a Registrations URL to view more details about application registrations. The **Applications > Registrations** screen appears if there are valid registrations.
  - Select the Logs icon to view the list of request executions for the application (if enabled). The Logs dialog appears, from which you can view the execution requests.

### **Editing an Application**

Edit an existing application from the application list.

1. From Management Cockpit, select **Applications**.
2. Select an existing application. The application tabs appear, such as Overview, Backend, Authentication, Push, and so forth.

---

**Note:** The Overview tab appears only for registered applications.

---

3. Select a tab, and make changes.
4. Save your changes.

### **Deleting an Application**

Delete the application from the application list.

1. From Management Cockpit, select **Applications**.
2. Select the application to be deleted.
3. Click **Delete**.
4. Click **Yes** to confirm.

---

**Note:** Once the application has been deleted, users cannot use it. All existing logs and traces are deleted and cannot be retrieved.

---

### **Pinging a Back-end Connection**

Test the back-end connection for a selected application, and the authorization URL status.

1. From Management Cockpit, select **Applications**. Alternately, select **Settings > Connections**.
2. Select the application.
3. Click **Ping** to view the back-end connection status, and the authorization URL status.

**Table 3. Ping Status**

<b>Field</b>	<b>Description</b>
URL	The authorization URL
Profile	The back-end connection profile
Status	The current state of the back-end connection
Status Description	Additional information about the back-end connection

**Table 4. Status Descriptions**

<b>Message</b>	<b>Description</b>
Back-end system reached successfully	The application back-end connection to the datasource is available.
Back-end system cannot be reached	The application back-end connection to the datasource is not available.
Request Failed	An unexpected error has occurred in the server. Check the server logs for more information.



Message	Description
Application inconsistent	You can only ping applications that are consistent, meaning they have valid back-end connections.

4. Click **OK**.

### Viewing Logs for an Application

View a list of request executions (such as GET, SET, PUT, and so forth) associated with a native or hybrid application. Detail depends on the log level that is currently set. Use the information to troubleshoot application processing problems.

During execution of application request executions, statements are persisted in the database. When you view the application log, you are viewing log statements for a request consolidated in a sequence. Application log level is set in **Settings > Log Settings**. In a troubleshooting situation, you may want to increase the log level, and then check the application log.

1. From Management Cockpit, select **Applications**.
2. Identify the application.
3. In the **Logs** column, click the Logs icon.

In the Logs dialog, you can view the extracted log entries.

Column	Description
Status	Status associated with the application request.
Registration ID	System-generated application registration ID.
User Name	Name of a person using the application registration.
Created Time	Date and time stamp for the application request, in the format MMM DD YYYY HH:MM:SS..
Request Type	The type of request that was being executed, including GET, SET, PUT, POST, DELETE, and MERGE.
Response Comment	Code or comment that provides additional status, such as 404.
Backend Response	An icon indicates there is a response from the back-end system. Click the icon to view the response, for example: <pre>&lt;?xml version="1.0" encoding="utf-8"?&gt;&lt;error xmlns="http://schemas.microsoft.com/ado/2007/08/data-services/metadata"&gt;&lt;code&gt;SY/530&lt;/code&gt;&lt;message xml:lang="en"&gt;Duplicate resource&lt;/message&gt;&lt;innererror&gt;&lt;code&gt;73F624E344ACF 163833C005056B43CF0&lt;/code&gt;&lt;/innererror&gt;&lt;/error&gt;</pre>

4. Click **X** to close the dialog.

## Managing Registrations

Manage multiple native and hybrid application registrations from a single location. Registered applications are associated with a user, an anonymous user, or nosec\_identity on one or more devices. When the user accesses the application from a device the application is registered. View information for all application registrations, or retrieve a filtered subset of registrations. From the list of registrations, you can drill down to see application and user details. For hybrid applications, if the developer has implemented Logger code in the application, you can drill down to upload and view client logs. You can also delete registrations.

Registration is handled on the client side through Logon Manager. When a user logs in from the device (or is logged in anonymously by the client application), the user is authenticated on SAP Mobile Platform Server using the security profile configured for the application. If authentication is successful, SAP Mobile Platform generates a registration identifier for the application+user+device combination, and creates a record in the SAP Mobile Platform database, indicating the device allows the application to consume data and services of SAP Mobile Platform.

SAP Mobile Platform sends the registration identifier to the device application. For all subsequent requests, such as accessing data in the back-end data source, the device client sends the registration ID to SAP Mobile Platform Server.

1. From Management Cockpit, select **Registrations** on the Home screen to view application connections. Alternatively, in the **Applications** tab, click the **Registrations** tab.

You see information for as many as 200 registered applications.

Field	Value
Registration ID	System-generated application registration ID.
Application ID	Unique identifier for the application, in reverse domain notation. This is the application or bundled identifier that the application developer assigns or generates during application development. The administrator uses the Application ID to register the application to SAP Mobile Platform Server, and the client application code uses the Application ID while sending requests to the server.
Device Type	The parameter value, such as Android and iPhone, sent by the device during registration/onboarding. "Unknown" indicates the device type cannot be detected.

Field	Value
User Name	<p>Name of a person using the application registration. If the application is configured to "Allow Anonymous Access", the user name "anonymous" is used for a successful registration, and can indicate:</p> <ul style="list-style-type: none"> <li>• The user has not provided the credentials.</li> <li>• The user has provided the wrong credentials.</li> <li>• The URL is prefixed with /public/.</li> </ul> <p>If the security profile is "No Authentication Challenge", the user name "nosec_identity" is used for a successful registration.</p>
Registration Time	The date, time, and time zone the application was registered, in the format MMM DD YYYY HH:MM:SS TZ.
Client Log	<p>If client logging is enabled for the application and data is available, click to upload the client log to the server. You can specify the level of verbosity.</p> <hr/> <p><b>Note:</b> Client logs are available only for hybrid applications.</p>

2. (Optional) Under Search Criteria, select filtering options to display a subset of application registrations:
  - a) Under Application, select the Application ID or All to narrow the scope of applications.
  - b) Under Time Frame, use the date picker to identify a specific date and time range.
  - c) Under User, type a specific user name to search only for that name.
  - d) Click **Search**.  
Registrations that meet your search criteria appear under Registrations.
3. (Optional) Use filtering and sorting options to display a subset of the registration results.
  - In **Filter By**, select Registration ID, Application ID, Device Type, or User Name, and enter a name to search for a specific entry.
  - Use column sorting and filtering to show only records that meet your criteria.
4. (Optional) View details:
  - Select an Application ID URL to view more details about a specific application.
  - Select a User Name URL to view more details about a specific user.

### **Deleting a Registration**

Delete an individual registration, which includes Application ID, Device Type, and User Name. This is useful for deleting orphaned registrations, which can occur when a user is no longer using an application on a device, or has obtained a new mobile device, which requires its own registration. It can also be useful for clearing a registration while troubleshooting

configuration issues. Once the registration is deleted, the user will not be able to use the application on the device unless they reregister the application.

1. From Management Cockpit, select **Registrations** on the Home screen to view application connections. Alternatively, in the **Applications** tab, click the **Registrations** tab.  
(Information for up to 200 registered applications appears.)
2. Select the application, and click **Delete**.

### **Viewing Client Logs**

(Applies only to hybrid) Download and view a client log associated with the selected application registration. The developer must have implemented Logger code in the application code, and the application must be registered and collecting data. Client requests to upload a log file are authenticated based on the application security profile. The log content varies by device type and operating system.

1. From Management Cockpit, select **Registrations** on the Home screen to view application connections. Alternatively, in the **Applications** tab, click the **Registrations** tab.  
Information for up to 200 registered applications appears.
2. Use the search, sorting, and filtering options to locate the registration in which you are interested:
3. Click the red client log icon to display the Client Logging dialog. In some cases, a list of client logs appears.
  - a) Click **Enable Log Upload**.
  - b) Select the log level in **Log Type**.
  - c) Click **Save** to save the modified setting of whether to allow uploading of client logs, and the level at which the client should log.  
The red client log icon turns green.
  - d) Click a log file name to download the log and open it in your selected viewer.

### **Client Logs**

(Applies only to hybrid) If client logging has been enabled for a hybrid application and log data is available, you can view the client long for the selected application registration.

---

**Note:** The log format varies by device type and operating system. Following are example log excerpts for Android and iOS.

---

#### *Hybrid Client Log - Android Example*

```
1377125811306
Debug Tag
Debug Message
null (com.sap.mp.cordova.plugins.logger.Logger:execute:54)
1893

1377125813056
Info Tag
```

```

Info Message
null (com.sap.mp.cordova.plugins.logger.Logger:execute:61)
1893

1377125814165
Warn Tag
Warn Message
null (com.sap.mp.cordova.plugins.logger.Logger:execute:68)
1893

1377125815157
Error Tag
Error Message
null (com.sap.mp.cordova.plugins.logger.Logger:execute:75)
1893

```

### *Hybrid Client Log - iOS Example*

```

ASLMessageID = 142697
Time = Aug 20, 2013, 4:37:22 PM
TimeNanoSec = 274834000
Level = 3
PID = 4823
UID = 966313393
GID = 1824234391
ReadGID = 80
Host = PALM00545086A
Sender = KAPSEL326
Facility = com.sap.sdmlogger
Message = MAFLogon MCIM is available: NO -[MAFMCIMManager
isAvailable] Line:48 thread:<NSThread: 0x9d57c10>{name = (null), num
= 1}

ASLMessageID = 142696
Time = Aug 20, 2013, 4:37:22 PM
TimeNanoSec = 234589000
Level = 4
PID = 4823
UID = 966313393
GID = 1824234391
ReadUID = 966313393
Host = PALM00545086A
Sender = KAPSEL326
Facility = com.sap.kapsel326
Message = Finished load of: file:///Users/i834381/Library/
Application%20Support/iPhone%20Simulator/6.1/Applications/9EC4E7F3-
C156-476F-850B-56EE001EEAB2/KAPSEL326.app/www/index.html
CFLog Local Time = 2013-08-20 16:37:22.234
CFLog Thread = c07

ASLMessageID = 142695
Time = Aug 20, 2013, 4:37:22 PM
TimeNanoSec = 194786000
Level = 4
PID = 4823

```

## Application Administration

```
UID = 966313393
GID = 1824234391
ReadUID = 966313393
Host = PALM00545086A
Sender = KAPSEL326
Facility = com.sap.kapsel326
Message = Resetting plugins due to page load.
CFLog Local Time = 2013-08-20 16:37:22.194
CFLog Thread = c07

ASLMessageID = 142694
Time = Aug 20, 2013, 4:37:22 PM
TimeNanoSec = 152145000
Level = 4
PID = 4823
UID = 966313393
GID = 1824234391
ReadUID = 966313393
Host = PALM00545086A
Sender = KAPSEL326
Facility = com.sap.kapsel326
Message = Multi-tasking -> Device: YES, App: YES
CFLog Local Time = 2013-08-20 16:37:22.151
CFLog Thread = c07
```

## Managing Users

Retrieve application and device details for a specific user, for native and hybrid applications.

1. From the Management Cockpit, select **Users** on the Home screen to view applications and devices details for a specific user. Alternatively, in the Applications tab, click the **Users** tab.
2. Under Search User Details, enter user name.
3. Click **Search**.
4. The Application and Device Details section appears for the selected user, showing the applications, devices, and number of connections.

## Importing and Exporting Applications

Import and export applications to copy them from one server environment to another, for example, from the testing environment to the production environment.

### Exporting an Application

Export the application configuration ZIP file to your local system, retaining many of the settings. You can use the export feature for creating a back-up of the application, and as a prerequisite for importing the application to another SAP Mobile Platform Server.

### **Prerequisites**

Ensure the application status is consistent (marked in green) before exporting it to your local system.

## Task

1. From Management Cockpit, select **Applications**.
2. Select the application, and click **Export**.
3. Select the relevant configuration to be exported. For all types of application, some configuration settings, such as application information, authentication profile name, and application-specific configurations are automatically included:
  - For Native applications, client resource configuration settings are also automatically exported, and you can select to export **Backend Configuration, Push Configurations** and **Client Password Policy**.
  - For Agentry applications, you can also select to export **Backend Configuration**.
  - For Hybrid applications, you can also choose to export **Backend Configuration, Push Configurations** and **Client Password Policy**.

The application configuration file downloads (as `<appid_version>.ZIP`) to the default (**Downloads** folder) location, which is specified in the browser.

- The downloaded Native application configuration ZIP file contains two subfolders: `Manifest.MF` and `Common.Manifest.MF`. `Manifest.MF` contains application ID and application version. `Common` includes an `*.smconfig` file, which contains all the information related to the application in the encrypted form. The Agentry and hybrid ZIP files contain an additional subfolder with application-specific settings.
- Passwords are not exported. For example, APNS and BES passwords for push configuration, and the password for anonymous user credentials.
- Only authentication profile names are exported, and not the complete authentication profile.

## Importing an Application

Import the application from one SAP Mobile Platform Server environment to another. Many configuration settings are retained, but you must reconfigure some application settings for the target server environment.

## Prerequisites

Before importing the ZIP file, ensure that it is available on the SAP Mobile Platform network.

## Task

- You can define or update an application during import. If the mapped authentication profile does not exist, the application is mapped to the “default” authentication profile. If the back-end configuration or mandatory passwords are missing, the application is marked as inconsistent, which means it is not ready for use.

- Importing an application does not necessarily make it ready for use. Administrators must review the imported application, and make any necessary adjustments to make it consistent and ready to use.
- For hybrid apps that use the **AppUpdate** plugin, you still need to manage the application version using **Applications > App Spec Setting**.
- For Agency applications, the imported application overwrites an existing version of the application. You must configure the imported application for the target server environment.

1. From Management Cockpit, select **Applications**.
2. Click **Import**.
3. In Import Configuration, enter a file location. Click **Browse**, select the application configuration ZIP file, and click **Open**.
  - If any manual changes are made to an exported ZIP file, it cannot be successfully imported. A failure message appears during import.
  - Importing overwrites any existing application with the same ID.
  - The URL information, for example, back end or BES URLs, is not overwritten during import.

Upon completion of a successful import, you see a summary message indicating the result for each imported component. Use the information to finish configuring the application for the target server environment.

### See also

- *Defining Applications* on page 36
- *Uploading and Deploying Hybrid Apps* on page 66
- *Publishing Agency Apps* on page 71
- *Deploying from a Preproduction Environment* on page 33

## Managing Security Profiles

Manage multiple security profiles for native and hybrid applications from a single location. You can define new security profiles, and edit or delete existing security profiles. Security profiles include configuration properties, and one or more authentication providers.

1. From Management Cockpit, select **Configure Security Profiles** on the **Home** screen to view application connections. Alternately, in the **Settings** tab, click the **Security Profile** tab. Information for all security profiles appears (up to 200 rows)

There are three default security profiles, which you cannot delete:

Security Profile	Description
admin	The Admin security profile is used to authenticate and authorize administrative users.



Security Profile	Description
default	The Default security profile is used to authenticate access to SAP Mobile Platform Server if a client application did not specify one explicitly. This profile is also used by any container managed authentication if a Web application is deployed to SAP Mobile Platform Server.
Notification	The Notification security profile is to enable push notifications using the Notification User role.

- (Optional) Select a security profile URL to view more details about a specific security profile.

### See also

- Security Profiles in SAP Mobile Platform* on page 164

### Creating and Configuring Security Profiles

Create and configure security profiles to define parameters that control how the server authenticates the user during onboarding, and request-response interactions with the back end. You can also define additional SAP Mobile Platform administrator users by creating a new security profile and configuring the required authentication provider.

Guidelines:

- Agentry applications have an additional authentication layer that is configured elsewhere; for Agentry applications, a common scenario is to configure No Authentication Challenge as the security provider, so that only Agentry performs authentication.
- You can stack multiple security providers to leverage security features in complex systems. You can order stacked security providers to take advantage of features in the order you chose. The Control Flag must be set for each enabled security provider in the stack.
- You must map logical roles to physical roles, as required by the application.
- When you create a new security profile, a corresponding XML file is created in the `SMP_HOME\configuration\com.sap.mobile.platform.server.security\CSI\` directory. When the security profile is updated, a copy of the XML file is saved to allow you to recover the security profile.

- From Management Cockpit, select the **Settings** tab, and select **Security Profiles**.
- Click **New**.
- Under Security Profile Properties, enter values.

Field	Value
Name	A unique name for the application authentication profile.

Field	Value
Check Impersonation	(Optional) In token-based authentication, whether to allow authentication to succeed when the user name presented cannot be matched against any of the user names validated in the login modules. By default the property is enabled, which prevents the user authentication from succeeding in this scenario.

4. Under Authentication Providers, set up one or more providers for the application.
  - a) Click **New**.
  - b) In the Add Authentication Provider dialog, select a provider from the list, and click **Create**.

Authentication Provider	Description
No Authentication Challenge	Provider that always authenticates the supplied user. The provider offers pass-through security for SAP Mobile Platform Server, and should typically be reserved for development or testing. SAP strongly encourages you to avoid using this provider in production environments—either for administration or device user authentication.
System Login (Admin Only)	Provider that is configured by the installer with the initial administrator credentials only to give platform administrator access to Management Cockpit, so that SAP Mobile Platform Server can be configured for production use. Administrators are expected to replace this authentication provider immediately upon logging in for the first time. SAP encourages you to avoid using this provider in production environments.
Populate JAAS Subject From Client	<p>Provider that enables administrators to add client values as named credentials, name principals, and role principals to the authenticated subject. This provider copies values from the client's HTTP request into the JAAS subject as:</p> <ul style="list-style-type: none"> <li>• Principals - identifies the user</li> <li>• Roles - grants access rights to SAP Mobile Platform protected resources</li> <li>• Credentials - provides single-sign-on material to use when connecting to back-end systems</li> </ul> <p>Adding client values as named credentials allows them to be used for single sign-on.</p>

Authentication Provider	Description
X.509 User Certificate	<p>Provider to use when the user is authenticated by certificates. This provider can be used in conjunction with other authentication providers that support certificate authentication [for example, Directory Service (LDAP/AD)], by configuring X.509 User Certificate before the authentication providers that support certificate authentication. You can only use this provider to validate client certificates when HTTPS listeners are configured to use mutual authentication.</p> <p><b>Note:</b> Agency clients on iOS and Android do not support client/user certificates. Agency clients on Windows and Windows CE support client-side certificates, but Agency cannot use these certificates for user identification; Agency requires separate user name and password authentication as well.</p>
HTTP/HTTPS Authentication	<p>Provider that authenticates the user with given credentials (user name and password, or SSO tokens from your SSO system) against a back end that is integrated to the your management or SSO systems. Optionally this provider may retrieve a cookie that represents additional SSO credentials to use for back-end systems that are also integrated with your SSO system.</p>
Directory Service (LDAP/AD)	<p>Provider that integrates with the your Active Directory or other Directory Server identity management system using LDAP. It first connects to your Directory Server using a technical user identity so it can perform an LDAP search to discover the fully qualified distinguished name (DN) of the current user in the directory. It then performs a bind to that DN with the provided password. When the bind succeeds, the user is considered authenticated. The provider then performs an LDAP search to see which groups the user is a member of. These group names are then considered as physical roles in the role mapping definitions that are used later for access controls.</p> <p>This provider is particularly useful in the Admin security profile to grant existing enterprise users usage of the Management Cockpit, and also any custom security profiles used for authenticating enterprise users for SAP Mobile Platform application usage.</p>

- c) Enter values based on the selected authentication provider.
- d) (Optional) Click **New** to add additional security providers on the stack. Use the up and down arrow icons to move the security providers into the desired order.

5. Click **Save**, and confirm.

### Next

You must configure role mapping (map logical roles to physical roles) as required by the application.

### Editing a Security Profile

Change the settings for an existing application security profile.

1. Select the **Settings** tab, and select **Security Profiles**.
2. Under **Security Profiles**, click an existing profile.
3. Modify the settings as needed.
4. Click **Save**.

### Deleting a Security Profile

Delete an existing application security profile.

1. Select the **Settings** tab, and select **Security Profiles**.
2. Select a security profile.
3. Click **Delete**, and confirm. The profile is removed from the list, and any application using it.

## Managing Connections

Manage multiple back-end connections for native and hybrid applications from a single location. You can define, edit, and delete back-end connections; test a connection using **Ping**, and filter connections based on connection name and connection URL.

1. From Management Cockpit, select **Configure Back-end Connections** on the **Home** screen to view application connections. Alternatively, in the **Settings** tab, click the **Connections** tab. Information for all registered applications appears (up to 200 rows).

Field	Value
Connection Name	Identifies the back-end connection by name.
Endpoint	The back-end connection URL, or the service document URL.

2. (Optional) Use filtering and sorting options to display a subset of the registration results:
  - In **Filter By**, select **Connection Name** or **Endpoint**, and enter a name to search for a specific entry.
  - Use column sorting and filtering to show only the records that meet your criteria.
3. (Optional) Select a **Connection Name URL** to view more details about a specific back-end connection.

## Defining a Back-end Connection

Define a new back-end connection to a data source or service for a native or hybrid application.

1. From the Management Cockpit, select the **Settings** tab, and select **Connections**.
2. To create a new application connection, click **New**.
3. Enter:

Field	Value
Connection Name	<p>(Appears only when adding a connection under Back-End Connections.) Identifies the back-end connection by name. The connection name:</p> <ul style="list-style-type: none"> <li>• Must be unique.</li> <li>• Must start with an alphabetic character.</li> <li>• Can contain only alphanumeric characters, underscores (_), and periods (.).</li> <li>• Cannot include spaces.</li> </ul>
Endpoint	<p>The URL (back-end connection, or service document) the application uses to access business data on the back-end system or service. The service document URL is the document destination you assigned to the service in Gateway Management Cockpit. Include a trailing slash to avoid triggering a redirection of the URL, and losing important HTTP header details. This is especially important when configuring the application with security, such as SSOToken and Certificates, and when Rewrite URL is enabled. Typical format:</p> <pre>http://host:port/gateway/odata/namespace/Connection_or_ServiceName.../</pre> <p>Examples:</p> <pre>http://testapp:65908/help/abc/app1/opg/sdata/TEST-FLIGHT/</pre> <pre>http://srvc3333.xyz.com:30003/sap/opu/odata/RMTSAMPLE/</pre>
Use System Proxy	<p>(Optional) Whether to use system proxy settings in the SAP Mobile Platform <code>props.ini</code> file to access the back-end system. This setting is typically disabled, because most back-end systems can be accessed on the intranet without a proxy. Enable this setting only in unusual cases, where proxy settings are needed to access a remote back-end system outside of the network. When enabled, this particular connection is routed via the settings in <code>props.ini</code> file.</p>
Rewrite URL	<p>(Optional) Whether to mask the back-end URL with the equivalent SAP Mobile Platform Server URL. Enable this setting to ensure the client makes all requests via SAP Mobile Platform Server, and directly to the back end. Rewriting the URL also ensures that client applications need not do any additional steps to make requests to the back end via SAP Mobile Platform Server. If enabled, the back-end URL is rewritten with the SAP Mobile Platform Server URL. By default, this property is enabled.</p>

Field	Value
Allow Anonymous Access	<p>(Optional) Whether to enable anonymous access, which means the user can access the application without entering a user name and password. However, the back-end system still requires login credentials for data access, whether it is a read-only user, or a back-end user with specific roles.</p> <ul style="list-style-type: none"> <li>If enabled and the back end requires it, enter the login credential values used to access the back-end system: <ul style="list-style-type: none"> <li><b>User name</b> – supply the user name for the back-end system.</li> <li><b>Password</b> – (required if you set a user name) supply the password for the back-end system.</li> </ul> </li> <li>If disabled (the default value) or the back end does not require it, you need not provide these credentials.</li> </ul> <hr/> <p><b>Note:</b> If you use Allow Anonymous Access for a native OData application, do not also assign the No Authentication Challenge security profile to the application; anonymous OData requests are not sent, and Status code: 401 is reported.</p>
Maximum Connections	<p>The number of back-end connections that are available for connection pooling for this application. The larger the pool, the larger the number of possible parallel connections to this specific connection. The default and minimum is 500 connections. Factors to consider when resetting this property:</p> <ul style="list-style-type: none"> <li>The expected number of concurrent users of the application.</li> <li>The load that is acceptable to the back-end system.</li> <li>The load that the underlying hardware and network can handle.</li> </ul> <p>Increase the maximum number of connections only if SAP Mobile Platform Server hardware can support the additional parallel connections, and if the underlying hardware and network infrastructure can handle it.</p>
Certificate Alias	<p>If the back-end system has a mutual SSL authentication requirement, supply the certificate alias name given to the private key and technical user certificate that is used to access the back-end system. The alias is located in <code>smp_keystore</code>. Otherwise, leave the entry blank.</p>

4. Click **Save**.

**See also**

- *Defining Back-end Connections for Native and Hybrid Apps* on page 38
- *HTTPS Back-End Connection Properties* on page 118
- *Administrator Overview for Integration Gateway* on page 3

**Editing a Back-end Connection**

Modify settings for an existing back-end connection.

---

**Note:** To prevent momentary inconsistencies, SAP recommends that you modify back-end connection configurations when few users are active. Users can use the connection without inconsistencies as soon you saves the changes.

---

1. From the Management Cockpit, select **Settings > Connections**, and select the application connection to edit.
2. Select a connection.
3. In the General Information window, edit the connection details as required.
4. Click **Save**.

### **Deleting a Back-end Connection**

Delete a back-end connection. The connection is removed from any application that is configured to use it.

1. From the Management Cockpit, select **Settings > Connections**, and select the back-end connection to delete.
2. Click **Delete**, and **Yes** to confirm.

### **Pinging a Back-end Connection**

Test the back-end connection for a selected application, and the authorization URL status.

1. From Management Cockpit, select **Applications**. Alternately, select **Settings > Connections**.
2. Select the application.
3. Click **Ping** to view the back-end connection status, and the authorization URL status.

**Table 5. Ping Status**

<b>Field</b>	<b>Description</b>
URL	The authorization URL
Profile	The back-end connection profile
Status	The current state of the back-end connection
Status Description	Additional information about the back-end connection

**Table 6. Status Descriptions**

<b>Message</b>	<b>Description</b>
Back-end system reached successfully	The application back-end connection to the datasource is available.
Back-end system cannot be reached	The application back-end connection to the datasource is not available.
Request Failed	An unexpected error has occurred in the server. Check the server logs for more information.

Message	Description
Application inconsistent	You can only ping applications that are consistent, meaning they have valid back-end connections.



4. Click **OK**.

## Reporting Usage Statistics

View aggregated usage statistics for all native and hybrid applications from a single location. The usage information is shown graphically, providing a quick summary of registrations and requests for the application, and the response time for these categories: SAP Mobile Platform Server, authentication, and back-end connection. You can view aggregated usage statistics for a subset of applications, which can be a powerful research and monitoring tool.

1. From the Management Cockpit, select the **Reporting** tab.
2. Under Filter the Results, select the vendor, applications, and time frame to narrow the focus to a subset. Use **Ctrl+Click** to select individual vendors or applications, and **Shift+Click** to select ranges.

Applications that meet the search criteria appear under Filter Results.

3. Select a tab to view these reports based on the filter results:
  - Choose **Registrations** to view active application registrations based on the search criteria.
    - Select **Over time** to view the number of application registrations for the selected time frame. You can choose the **Line Graph**  or **Bar Graph**  icons in the left corner to vary the graphical presentation.
    - Select **By device type** to view the number of application registrations by device types for the time frame selected. You can choose the **Bar Graph** or **Pie Chart** icon in the left corner to vary the graphical presentation.
  - Choose **Requests** to view the active requests for the based on the search criteria. You can choose **Line Graph** or **Bar Graph** icons in the left corner to see a graphical summary.
  - Choose **Response Time** to view the aggregate response time (in milliseconds) per day. Response time is reported for SAP Mobile Platform, authentication time, and back-end response time.
4. Under Details, view the total number of registrations and requests for each application listed.

### See also

- *Enabling/Disabling Client Statistics* on page 121



## Managing Application Logs

Set the verbosity for native and hybrid application logging, and the purge schedule for log files. View application, proxy, and push information, and drill down to view detailed log and trace information.

### Setting and Purging Logs

Use the Log Settings tab to specify log levels and purge logs for native and hybrid applications.

Set the log level in the system at runtime as **Error** (the default) for proxy, push, and onboarding. **All** includes all type of logs, including error and success messages for proxy, push, and onboarding.

1. From the Management Cockpit, select **Logs > Application Log Settings**.
2. Under Log Level, set the error logging verbosity.
  - a) For Log Level, select Error or All.
  - b) Click **Save**.
3. Under Log Purge, set the log purge scheduler and clear the old logs.
  - a) Select the days to automatically purge the log.
  - b) Select the time to purge the log.
  - c) Select the number of days (1 – 30) to retain the error logs.
  - d) Select the number of days (1 – 30) to retain the success logs.
4. Click **Save**.

---

**Note:** To purge the logs immediately, click **Purge Now**. This deletes all logs from the system, keeping only the last success/error days logs as maintained in the setting.

By default, the log purge scheduler runs everyday at 12:00 AM UTC, retaining only error logs from the past seven days.

---

### Viewing Application Logs

View information for all, or a subset, of native and hybrid applications. You can filter by status, type, time frame, and users. You can also view detailed trace log data if available.

1. From the Management Cockpit, select **Logs > Application Logs**. Alternately, view logs for a specific application in the Application tab, under the Logs column, and in the Overview tab, for a selected registered application.
2. Under Search Criteria, enter application log search criteria:
  - For Application, select a specific application or select **All**.
  - For Log Type, select:
    - Status – **Error** or **All**.

- Type – **Application Settings, Deregistration, Registration, Resource Bundles, or All.**
  - For Time Frame, indicate a specific time frame.
  - For User, enter a name in the User Name field.
3. Click **Search**, to view the logs that meet the search criteria:

Column	Description
Status	Log entry status, typically <code>Error</code> or <code>Success</code> .
Registration ID	The unique connection identifier that makes the request to the server.
User Name	The name of the user associated with the application ID.
Created Time	The time and date stamp for the log entry.
Type	The log type, such as application settings, deregistration, and so forth.
Response Code	The response status code for the invocation, such as 403 or 404.
View Trace	The link to the log trace file, for log statements that are associated with the execution request. Optionally, you can download a text file for further analysis.
Application ID	Unique identifier for the application, in reverse domain notation. This is the application or bundled identifier that the application developer assigns or generates during application development. The administrator uses the Application ID to register the application to SAP Mobile Platform Server, and the client application code uses the Application ID while sending requests to the server.
Message	Message text such as <code>Cannot create application connection ...</code> or <code>Application connection is not registered ....</code>

4. (Optional) Click the **View Trace** icon to view trace information for the selected log. The Trace Logs Details window appears with trace information. Click **Download** to download a text file of the trace information to the `Downloads` directory.

### Viewing Proxy Logs

View information for all, or a subset of, native or hybrid applications. Proxy logs record the request-response between client and server. You can filter by status, type, time frame, and user. You can also view detailed log and trace log data, if available.

1. From the Management Cockpit, select **Logs > Proxy**. Alternately, you can view logs for a particular application in the Application tab, under the Logs column; and in the Overview tab for a selected application.

2. Under Search Criteria, enter proxy log specific search criteria:

- For Application, select an application ID or **All**.
- For Log Type, select:
  - Status – Error or All.
  - Type – Delete, Get, Post, Put, or All.
- For Time Frame, select the time frame using **From** and **To**.
- For User, enter the user name in the **User Name** field.

3. Click **Search**, to view the logs that meet the search criteria:

Column	Description
Status	The status of the log entry, typically <code>Error</code> or <code>Success</code> .
Registration ID	The unique registration identifier, which makes the request to the server.
User Name	The name of the user who is associated with the application ID.
Created Time	The time and date stamp for the log entry.
Request Type	The request type of the message, such as Get, Put, Delete, Post, and so forth.
Response Code	The response status code for the invocation, such as 400 or 404.
Log File	The link to the log file (only for error logs).
View Trace	The link to the log trace file. Optionally, you can download a text file for further analysis.
Application ID	Unique identifier for the application, in reverse domain notation. This is the application or bundled identifier that the application developer assigns or generates during application development. The administrator uses the Application ID to register the application to SAP Mobile Platform Server, and the client application code uses the Application ID while sending requests to the server.

4. (Optional) Click the Log Files icon to log file information for the selected log.

The Log Details window appears with log information.

5. (Optional) Click the View Trace icon to view trace information for the selected log.

The Trace Logs Details window appears with trace information. Click **Download** to download a text file to the `Downloads` directory.

**Viewing Push Logs**

View push logs for notifications triggered from the back end, for all, or a subset of, native and hybrid applications. You can filter by application, status, push platform, time frame, or user. You can also view detailed trace log data if available.

**1. From the Management Cockpit, select **Logs > Push**.**

You can also view logs for a particular application in the Application tab, under the Logs column; and in the Overview tab for a selected application.

**2. Under Search Criteria, enter push log search criteria:**

- For Applications, select the Application ID.
- For Log Type, select:
  - Status – Error or All.
  - Push Platform – GCM, APNS, BIS, BES, WNS, MPNS, or All.
- For Time Frame, select the date and time range in From and To.
- For User, enter the user name in the User Name field.

**3. Click **Search**, to view the log that meet the search criteria:**

Column	Description
Status	The status of the log entry, typically <code>Error</code> or <code>Success</code> .
Application ID	Unique identifier for the application, in reverse domain notation. This is the application or bundled identifier that the application developer assigns or generates during application development. The administrator uses the Application ID to register the application to SAP Mobile Platform Server, and the client application code uses the Application ID while sending requests to the server.
User Name	Name of a person using the application registration.
Registration ID	The unique registration identifier that makes the request to the server.
Received Time	The time stamp indicating when the message was received by the server.
Notification Type	Lists the notification types. For example, GCM, APNS, BES, BIS, WNS, or MPNS.
Message	The error message if any. For example <code>Invalid notification target ID</code> or <code>Notification sending failed</code> .
View Trace	The link to the log trace file. Optionally, you can download a text file for further analysis.

**4. (Optional) Click the View Trace icon to view trace information for the selected log.**

The Trace Logs Details window appears with trace information. Click **Download** to download a text file to the Downloads directory.

### **Trace Log Statements**

During execution of application, proxy, and push requests, various points of code are logged in the SAP Mobile Platform database. Each log statement is a row in the database. You can view these execution statements for native and hybrid applications through application, proxy, and push logs, and use the information to troubleshoot problems.

The level of detailed information collected is based on the log level setting. If set to Error, you only collect error messages, if set to All, you collect all messages. The information is kept until you purge it manually, or according to the automated purge schedule you set up.

In a troubleshooting situation, you may want to increase the log level to capture more details. You can use the Trace Logs to find out about request statements (application), client-server interactions (proxy), and push actions (push). Use search criteria and sorting to find the log records and execution statements needed to diagnose a problem.

Once you are finished, it is a good idea to change the log level to collect less information.

## **Configuring Agency Application Logs**

Configure settings for Agency logs. Agency log file settings are configured as part of defining the application.

---

**Note:** Some entries require an SAP Mobile Platform Server start.

---

1. From Management Cockpit, select **Applications > App Specific Settings**.
2. Under Logging, edit log settings.

Definition	Values	Description
Event Log File	Valid file name or path and file name. Default: <code>events.log</code>	The file to which event log items are written. You can change the default to specify a different file name or location. The path can be relative to the server installation folder; or a full path that includes the drive letter. Requires server restart.
Maximum Log Size	Numeric value, assumed to be in bytes. Default: 0	The maximum size of the log file in bytes. A value of 0 indicates no maximum size. Any other value results in the log file rolling over when the file exceeds the value set here.
Log Roll Time	Time of day in 24-hour clock format (HH:MM). Default: 1:00am	The time at which to roll log files. The server must be running for the files to roll over at the specified time. Requires server restart.
Log Rolls At Time	Boolean value of true or false. Default: true	Whether to roll the log files based on the time in Log Roll Time. Requires server restart.

Definition	Values	Description
Message Log File	Valid file name or path and file name. Default: messages.log	The file to which message log items are written. You can change the default to specify a different file name or location. The path can be relative to the server installation folder; or a full path that includes the drive letter. Requires server restart.

- Restart SAP Mobile Platform Server if you changed any setting that requires it.

## Provisioning Applications

---

Provision the application to supported devices so users can install and log in to the application to register with SAP Mobile Platform Server. You can provision applications through native application stores, through enterprise Web site downloads, or through Afaria.

**Note:** You must define and configure the application on the server before users can register.

---

### Provisioning with Afaria

You can use Afaria to provision native and hybrid applications that use the MAF Logon UI component by creating a provisioning file containing a set of parameters.

Afaria uses the provisioning file to deliver application configuration data to the device. The user installs the application through Afaria client. When the application starts, the MAF Logon UI displays connection registration fields based on settings in the provisioning file.

#### Setting Up the Afaria Environment

In order to deliver application configuration data using Afaria, you need to install and configure the Afaria package server and set up automatic enrollment for your devices. Once devices are enrolled you can define the application you are provisioning by setting up application policies. This includes importing the MAF Logon provisioning file, which is used to seed the application with configuration data.

For complete information on installing and configuring Afaria, refer to the following Afaria documentation:

- *Afaria Installation Guide*
- *Afaria Administration Reference*

The tasks listed below are specific to delivering application configuration data. Refer to the Afaria documentation for the details of each task.

**Table 7. Afaria Setup Tasks**

<b>Task</b>	<b>Description</b>	<b>Afaria Documentation</b>
Install and configure the Afaria package server component	Delivers applicaton configura-tion data to devices.	<i>Afaria Installation Guide &gt; Package Server</i>
Set up device enrollment poli-cies	Automates enrolling devices in Afaria management	<i>Afaria Administration Refer-ence &gt; Enrolling Devices in Management</i>  <i>Afaria Administration Refer-ence &gt; Policy &gt; Enrollment Pol-icies</i>
Set up application policies	Defines application packages for devices	<i>Afaria Administration Refer-ence &gt; Application Onboarding &gt; Provisioning Data for iOS and Android Applications</i>  <i>Afaria Administration Refer-ence &gt; Policy &gt; Application Policies</i>

**See also**

- *Creating an MAF Logon Provisioning File* on page 109

**Creating an MAF Logon Provisioning File**

Create a provisioning file for applications that use the MAF Logon UI component. The provisioning file is an ASCII text file. Each setting in the file must be on a separate line, or separated by a semicolon. The settings in the provisioning file provide applications with initial connection registration values that control the fields that display in the MAF Logon UI.

You can create a provisioning file for individual applications or provide settings for all applications in a single file. All settings in the configuration file are optional.

<b>Key</b>	<b>Description</b>
servername=	The host name for the machine that hosts the SAP Mobile Platform Server or reverse proxy, if used.
serverport=	The server port number for SAP Mobile Platform Server. The default is 8080.
ishttps=	Boolean value that determines whether to use HTTPS instead of the default unsecured HTTP.

Key	Description
vaultpolicy=	<p>Controls the use of application passwords in the MAF Logon UI.</p> <p>Possible values:</p> <ul style="list-style-type: none"> <li>• <code>defaulton</code> – password fields are displayed. The user can enable/disable the password,</li> <li>• <code>defaultoff</code> – password fields are not displayed. The user can enter a password by performing additional steps.</li> <li>• <code>alwayson</code> – password fields are displayed. The user must enter a password.</li> <li>• <code>alwaysoff</code> – password fields are not displayed.</li> </ul>
usercreationpolicy=	<p>Controls how user records are created on the server.</p> <p>Possible values:</p> <ul style="list-style-type: none"> <li>• <code>automatic</code> – user records are created automatically in the authentication provider as part of the registration process.</li> <li>• <code>certificate</code> – user records are created automatically based on the client's security certificate.</li> <li>• <code>manual</code> – you must create users records on the server by manually registering applications. The manual user creation policy applies only to applications that connect to a 2.x version of SAP Mobile Platform Server.</li> </ul>
gatewaypingpath=	<p>The Gateway server's ping path. This is used by MAF Logon to identify the SAP NetWeaver Gateway server.</p>
gatewayclient=	<p>Identifies the client on the Gateway server. This is added to registration requests as a URL encoded parameter.</p>



Key	Description
resourcepath=	If you installed a reverse proxy in your landscape, use to specify the URL suffix that the MAF Logon should use during registration requests.
domain=	<p><b>Note:</b> Applies only to applications that connect to a 2.x version of SAP Mobile Platform Server.</p> <p>The server domain that clients will register to.</p>
companyid=	<p><b>Note:</b> Applies only to applications that connect to a 2.x version of SAP Mobile Platform Server.</p> <p>The company or farm ID of the relay server used to connect to the server.</p>
securityconfig=	<p><b>Note:</b> Applies only to applications that connect to a 2.x version of SAP Mobile Platform Server.</p> <p>The security configuration created on SAP Mobile Platform Server.</p>

### Example: provisioning file for automatic user creation

```
servername=smp.server.com;
serverport=8080;
ishttps=false;
vaultpolicy=defaulton;
usercreationpolicy=automatic;
```

### Example: provisioning file for user creation based on certificate

```
servername=smp.server.com;
serverport=8443;
ishttps=true;
vaultpolicy=defaulton;
usercreationpolicy=certificate;
```

### See also

- *Setting Up the Afaria Environment* on page 108



# Server Administration

You can configure server properties to fine tune how the server operates in your environment. You can control which features your server supports. You can adjust log settings and view logs to monitor server health, and integrate with a third party diagnostics tool to gather additional metrics.

## See also

- *Application Administration* on page 31
- *Security Administration* on page 149
- *Application Services (Mobiliser) Administration* on page 261
- *SMS Administration* on page 341

## Starting and Stopping SAP Mobile Platform Server on Windows

---

You can start and stop SAP Mobile Platform Server on Windows in different ways.

**Note:** If you are using a custom database with SAP Mobile Platform, starting and stopping SAP Mobile Platform Server has no effect on the database. Only the Derby database is automatically stopped and started in sync with SAP Mobile Platform Server.

---

### *Starting SAP Mobile Platform Server*

Use any of these methods to start SAP Mobile Platform Server:

- Desktop shortcut – right-click the **Start SAP Mobile Platform Server** icon on the Windows desktop and select **Run as Administrator**.
- Windows Start menu – select **Start > (All) Programs > SAP Mobile Platform 3.0 > Start SAP Mobile Platform Server**.
- Windows Services Control Panel – start the **SAP Mobile Platform Server** service.

Server start-up may take several minutes. The start-up process is complete when you see:

```
The SMP server has initialized and is ready.
```

### *Stopping SAP Mobile Platform Server*

Use any of these methods to stop SAP Mobile Platform Server:

- Desktop shortcut – double-click the **Stop SAP Mobile Platform Server** icon on the Windows desktop.

- Windows Start menu – select **Start > (All) Programs > SAP Mobile Platform 3.0 > Stop SAP Mobile Platform Server**.
- Windows Services Control Panel – stop the **SAP Mobile Platform Server** service.

## Starting and Stopping SAP Mobile Platform Server on Linux

Start and stop SAP Mobile Platform Server on Linux through a terminal window.

---

**Note:** If you are using a custom database with SAP Mobile Platform, starting and stopping SAP Mobile Platform Server has no effect on the database. Only the Derby database is automatically stopped and started in sync with SAP Mobile Platform Server.

---

### *Starting SAP Mobile Platform Server*

You must manually start SAP Mobile Platform Server each time you restart the host system.

1. Open a terminal window.
2. Go to `SMP_HOME/Server/`.
3. Execute `./daemon.sh start`.

Server start-up is complete when you see:

```
The SMP server has initialized and is ready.
```

### *Stopping SAP Mobile Platform Server*

1. Open a terminal window.
2. Go to `SMP_HOME/Server/`.
3. Execute `./daemon.sh stop`.

## Configuring SAP Mobile Platform Server

Configure properties for server communications and performance.

### Configuring HTTP/HTTPS Port Properties

HTTP/HTTPS ports are initially set during installation. You can change the port numbers and other port properties in `SMP_HOME\Server\config_master\org.eclipse.gemini.web.tomcat\default-server.xml`.

The property descriptions below are based on Apache Tomcat 7 documentation at <http://tomcat.apache.org/tomcat-7.0-doc/config>.

---

**Note:** When changing a SAP Mobile Platform Server port number, temporarily stop the server. Restart the server after making the changes.

---

---

**Warning!** Changing any Tomcat properties that are not described in the SAP Mobile Platform Server documentation might have unintended effects on your server.

---

**Table 8. Port Properties in default-server.xml**

Property	Description
port	TCP port number on which this connector creates a server socket and awaits incoming connections
maxThreads	Maximum number of request-processing threads to be created by this connector. The default value is 250.
connectionTimeout	Length of time, in milliseconds, that this connector waits, after accepting a connection, for the request URI line to be presented. The default value is 20000 (20 seconds).
acceptCount	Maximum queue length for incoming connection requests when all possible request-processing threads are in use. Any requests received when the queue is full are refused. The default value is 100.
redirectPort	<p>If this connector supports non-SSL requests, and a request is received for which a matching <code>security-constraint</code> requires SSL transport, Catalina automatically redirects the request to this port number.</p> <p>The default value is 8081. If you change the port number used for oneway SSL authentication, you must also change this port number.</p>

**See also**

- *HTTP/HTTPS Port Number Reference* on page 145

## **Configuring Push Notification Properties**

Configure properties to enable any type of notification to be sent from a back-end system to SAP Mobile Platform Server. For Apple push notifications, you may also need to configure additional properties for sending notifications from SAP Mobile Platform Server to APNS.

Notification properties are set in `SMP_HOME\Server\props.ini`. Create a backup copy of this file before making any changes.

---

**Note:** Reconfigure the SAP Mobile Platform Server Windows service for the changes to take effect.

---

### *Changing the Push URL*

Configure the base URL used to send any type of notification from the back-end system by editing this property in props.ini:

```
-Dcom.sap.mobile.platform.server.notifications.baseurl
```

The default value is `http://SMPHostName:8080/`. Replace `SMPHostName` with the host name of SAP Mobile Platform Server.

---

**Note:** The host name must be accessible from the back-end system that is sending notifications.

---

Change the protocol to `https` if required. Change the port number if required.

### *Configuring APNS Push Notifications*

If you are using APNS and SAP Mobile Platform Server is installed behind a firewall, you can send notifications to APNS using either a SOCKS proxy or a load balancer.

To send notifications through a SOCKS proxy, edit these properties in props.ini:

Property	Description
-DapnsSocksProxyHost	IP address of the SOCKS proxy used to connect to APNS.
-DapnsSocksProxyPort	SOCKS proxy port. The default value is 1080. The <code>apnsSocksProxyHost</code> property must be set in order for this value to be used.

To send notifications through a load balancer, consult your network administrator.

### **See also**

- *Configuring Push for Native and Hybrid* on page 61
- *Apple Push Notifications* on page 62
- *Reconfiguring the Windows Service After Server Configuration Changes* on page 123

## **Configuring Back-end Communications with an HTTP Proxy**

Configure communications between SAP Mobile Platform Server and back-end systems inside or outside the local network. When communicating with a back-end system outside the local network, SAP Mobile Platform Server uses an HTTP proxy. You can set up exceptions to the proxy for hosts that are inside the local network.

1. Create a backup copy of `SMP_HOME\Server\props.ini`.
2. Using a text editor, open the original `SMP_HOME\Server\props.ini`.
3. To configure communication with back-end systems outside the local network, enter your proxy host and port for either HTTP or HTTPS in the following properties:

```
-Dhttp.proxyHost
```

```
-Dhttp.proxyPort
```

```
-Dhttps.proxyHost
```

```
-Dhttps.proxyPort
```

Setting these properties causes some of the components of SAP Mobile Platform Server to use the specified proxy server for HTTP/HTTPS connections. These include:

- the HTTP/HTTPS Authentication provider
  - back-end connections that are configured to use the system proxy
4. To override proxy settings for the HTTP/HTTPS Authentication provider, uncomment the `-Dhttp.nonProxyHosts` property and enter the names of the hosts that are inside the local network.

Separate each host name with a `|` character. You can use a wildcard character (`*`) for pattern matching.

For example:

```
-Dhttp.nonProxyHosts=localhost|*.data.com
```

includes the localhost and every host in the data.com domain.

Every host listed in this property will be accessed directly.

5. Save the file.
6. Reconfigure the SAP Mobile Platform Server Windows service for the changes to take effect.

### See also

- *Defining Back-end Connections for Native and Hybrid Apps* on page 38
- *Reconfiguring the Windows Service After Server Configuration Changes* on page 123

## Changing Database Connection Passwords

If you are using a custom database for SAP Mobile Platform, you select and configure the database during installation. You can change the password after installation. Use caution when changing other database connection properties.

SAP Mobile Platform Server connection properties are located in `SMP_HOME\Server\config_master\connection_data\connection.properties`.

Mobiliser database connection properties are located in `SMP_HOME\Server\config_master\com.sybase365.mobiliser.framework.persistence.jdbc.bonecp.pool\pool.properties`.

To change the password, you first need to encrypt the new password. SAP Mobile Platform Server includes an executable JAR to encrypt configuration values. After encrypting the

password, update the `password` property with the new encrypted password, keeping `{enc}` as the prefix.

Restart the server after making any changes to database connection properties.

### See also

- *Encryption of Configuration Files* on page 230

## Performance Properties

Review and update selected properties to improve server efficiency.

### HTTPS Back-End Connection Properties

Configure back-end connection properties to ensure that adequate resources are allocated for SAP Mobile Platform Server connections to the data source back end.

The number of required connections varies, based on the maximum number of SAP Mobile Platform Server threads in use at any time and the length of time it takes for the back end to respond. If possible, allocate a connection for each thread.

Connection pools allow SAP Mobile Platform Server to reuse established connections to a back-end data source. If the number of connections in the pool is lower than the number of threads, you may experience timeouts. To avoid timeouts, you can adjust the connection timeout value. However, a higher connection timeout value will make it harder for other threads to get a back end connection.

The following connection properties are located in `SMP_HOME\Server\props.ini`.

---

**Note:** Create a backup copy of `props.ini` before making any changes.

---

Property	Description	Default Value
<code>-Dcom.sap.mobile.platform.server.connection.pool.idletimeout</code>	The amount of time, in seconds, that a connection in the pool can be idle before it is removed.	600
<code>-Dcom.sap.mobile.platform.server.maximum.connectionPoolSize</code>	The maximum number of concurrent HTTP connections that the server can make to any URL.	5000
<code>-Dcom.sap.mobile.platform.server.connection.timeout</code>	The amount of time, in milliseconds, before a database connection operation will time out.	60000



Property	Description	Default Value
-Dcom.sap.mobile.platform.server.connection.socketTimeout	The amount of time, in milliseconds, that the socket will wait for data from the back-end data source before disconnecting.	6000
-Dcom.sap.mobile.platform.server.maximum.gzip.PoolSize	The maximum number of concurrent GZIP input streams that can be processed by the server.  Using GZIP compression to compress the HTTP request body reduces network traffic and can improve performance.	5000

---

**Note:** Reconfigure the SAP Mobile Platform Server Windows service for the changes to take effect.

---

### See also

- *Reconfiguring the Windows Service After Server Configuration Changes* on page 123
- *Defining Back-end Connections for Native and Hybrid Apps* on page 38
- *Defining a Back-end Connection* on page 99

### Database Connection Pool Properties

Connection pools allow SAP Mobile Platform Server to reuse established connections to the database. Adjust the maximum number of connections that can be in the connection pool to ensure that adequate resources are allocated. Mobiliser uses its own connection pool. If you have enabled Mobiliser features, you can adjust the minimum and maximum number of connections for each connection pool partition.

#### *Database Connection Pools*

SAP Mobile Platform Server database connection properties are located in `SMP_HOME\Server\config_master\connection_data\connection.properties`.

To change the maximum number of concurrent connections that can be made to the database, add the following line:

```
com.sap.persistence.jdbc.connection.pool.max_active=new value
```

The default value is 8.

---

**Note:** Use caution when changing other properties in this file. Incorrect database connection property values can make the server unusable.

---

Restart the server after making any changes to database connection properties.

#### *Mobiliser Database Connection Pools*

Mobiliser connection pools support partitioning. Depending on the transaction being performed, SAP Mobile Platform Server requires multiple calls to the database to complete

the transaction. A large number of connections are required for high volumes of concurrent transactions. The total number of database connections is the product of the number of partitions and the connections per partition. If there are no operating system or database restrictions, running a higher number of database connections than recommended does not negatively impact performance. The same is also true for the number of partitions.

The Mobiliser database connection properties are located in `SMP_HOME\Server\config_master\com.sybase365.mobiliser.framework.persistence.jdbc.bonecp.pool\pool.properties`.

Setting Name	Default Value	Modified Value
maxConnectionsPerPartition	10	16 * CPU cores
minConnectionsPerPartition	1	16 * CPU cores
partitionCount	2	N/A (default is recommended)

**Note:** Use caution when changing other properties in this file. Incorrect property values can make the server unusable.

Restart the server after making any changes to database connection properties.

### **Database Cache Properties**

A database cache enables you to improve performance by reducing the number of times the database needs to read an object.

SAP Mobile Platform Server database connection properties are located in `SMP_HOME\Server\config_master\connection_data\connection.properties`.

To change the database cache properties, change the following values.

Property	Description	Default Value
eclipselink.cache.size.Application-ConnectionInfo	Number of device registrations which can be cached on the server.	5000
eclipselink.cache.size.Application	Number of applications which can be cached on the server.	50
eclipselink.cache.size.Endpoint-Configuration	Number of back-end connections (endpoints) which can be cached on the server.	50

**Note:** Use caution when changing other properties in this file. Incorrect property values can make the server unusable.

Restart the server after making any changes to database connection properties.

### **Enabling/Disabling Client Statistics**

Client statistics collect data for each native or hybrid application client request to the server. Reviewing these statistics enables you to assess how applications are being used in the end user community.

---

**Note:** By default, client statistics are enabled. Disabling client statistics improves server performance.

---

You can view client/server communications statistics in the Reporting tab of Management Cockpit.

1. Create a backup copy of `SMP_HOME\Server\props.ini`.
2. Using a text editor, open the original `SMP_HOME\Server\props.ini`.
3. For `-Dcom.sap.mobile.platform.server.enable.statistics`, enter `true` to start collecting statistics or `false` to stop collecting statistics.
4. Save the change.
5. Reconfigure the SAP Mobile Platform Server Windows service for the changes to take effect.

### **See also**

- *Reporting Usage Statistics* on page 102
- *Reconfiguring the Windows Service After Server Configuration Changes* on page 123

### **Java Memory Properties**

You can control the amount of memory allocated to SAP Mobile Platform by specifying the heap size for the Java command line process. The heap size represents only part of the memory used by Java.

You can make efficiency gains by setting the minimum amount allocated to the same value as the maximum value allocated. SAP strongly recommends that you do not exceed this value, regardless of the number of available CPU cores.

Java memory properties are set in `SMP_HOME\Server\props.ini`. Create a backup copy of this file before making any changes.

Setting Name	Default Value	Modified Value
-Xms	1024M	512M * CPU cores
-Xmx	2048M	512M * CPU cores
-XX:PermSize	256M	256M
-XX:MaxPermSize	512M	N/A (default is recommended)

---

**Note:** Reconfigure the SAP Mobile Platform Server Windows service for the changes to take effect.

---

### See also

- *Reconfiguring the Windows Service After Server Configuration Changes* on page 123

### **Configuring Server Session Timeout**

To free up memory for the server, you can decrease the duration of server sessions.

The property description below is based on Apache Tomcat 7 documentation at <http://tomcat.apache.org/tomcat-7.0-doc/config/>.

---

**Warning!** Changing any Tomcat properties that are not described in the SAP Mobile Platform Server documentation might have unintended effects on your server.

---

1. Open `SMP_HOME\Server\config_master\org.eclipse.gemini.web.tomcat\web.xml`.
2. For the `session-timeout` property, set the amount of time, in minutes, after creation, that an HTTP session times out.  
The default value is 20 minutes.
3. Save your changes.
4. Restart the server for the changes to take effect.

### **Tomcat Thread Pool Properties**

The Tomcat thread pool settings provide the main worker threads that handle incoming connections before handing operations off to other subsystems. The default maximum thread value of 250 is sufficient for most workloads. Adjust the value to increase the number of threads that can be executed simultaneously or decrease the value to help prevent CPU overload.

The `maxThreads` property is located in `SMP_HOME\Server\config_master\org.eclispe.gemini.web.tomcat\default-server.xml`. Create a backup copy of this file before making any changes.

---

**Note:** You can also add a `minThreads` property to this file.

---

### **Setting the Maximum Number of Client Logs**

(Applies only to hybrid) You can view application registration logs uploaded from clients using Management Cockpit. Uploaded client logs are stored on the server in `SMP_HOME\Server\log\clientlogs\Application ID\Registration ID`. By default, a maximum of ten logs will be stored for an application registration. You can change this setting to a higher or lower number as required. When the maximum number of client logs is reached, logs will be deleted from the server based on the log date (oldest logs deleted first).

1. Create a backup copy of `SMP_HOME\Server\props.ini`.
2. Using a text editor, open the original `SMP_HOME\Server\props.ini`.
3. Enter the maximum number of logs that can be stored on the server for an application connection in:  

```
-Dcom.sap.mobile.platform.server.maximum.clientlogfiles
```
4. Save your changes.
5. Reconfigure the SAP Mobile Platform Server Windows service for the changes to take effect.

## **Reconfiguring the Windows Service After Server Configuration Changes**

If you change the server configuration by updating the `SMP_HOME\Server\props.ini` file, you must reconfigure the SAP Mobile Platform Server service.

### **Prerequisites**

If you do not have permission to start and change Windows services, run these commands as an Administrator. Open an Administrator Command Prompt by right-clicking the **Command Prompt** icon and selecting **Run as administrator**.

### **Task**

---

**Note:** If you are running the service as a user account, you must reassign the user logon account after reconfiguring the service. For more information, see *Setting up a User Account for the SAP Mobile Platform Windows Service* in *Landscape Design and Integration*.

---

1. Stop the SAP Mobile Platform Server:

```
svcutil -stop
```

The script may complete before the server is completely stopped, so make sure the server is stopped before proceeding.

---

**Note:** You can also stop the server using the Windows Services Control Panel.

---

2. Uninstall the server Windows service:

```
svcutil -uninstall
```

The script may complete before the service is completely uninstalled, so make sure the service is uninstalled before proceeding.

3. Generate an installation script based on the new server configuration:

```
svcutil -generate
```

4. Install the server as a service with the new configuration:

```
svcutil -install
```

## Server Administration

This automatically starts the server as a Windows service after it has been installed. To start the server manually after it has been installed, run: `svcutil -install -startup manual`.

5. (Optional) If you specified a manual startup when installing the server, restart the server:

```
svcutil -start
```

The script may complete before the server is completely started, so make sure the server is running before performing any server operations.

---

**Note:** You can also start the server using the Windows Services Control Panel.

---

## Props.ini Reference

Some server configuration tasks require updating `SMP_HOME\Server\props.ini`.

**Warning!** Use caution when changing any `props.ini` properties that are not described in SAP Mobile Platform Server documentation. Changing other properties can have unintended effects on your server.

Property	Description
-DapnsSocksProxyHost	IP address of the SOCKS proxy used to connect to APNS.
-DapnsSocksProxyPort	SOCKS proxy port. The default value is 1080. The <code>apnsSocksProxyHost</code> property must be set in order for this value to be used.
-Dcom.sap.mobile.platform.server.notifications.baseurl	URL used to send any type of notification from the back-end system to SAP Mobile Platform Server. Default value: <code>http://SMPHostName:8080/</code> .
-Dcom.sap.mobile.platform.server.connection.pool.idletimeout	The amount of time, in seconds, that a connection in the pool can be idle before it is removed. Default value: 600
-Dcom.sap.mobile.platform.server.maximum.clientlogfiles	The maximum number of logs that can be stored on the server for an application connection. Default value: 10

Property	Description
-Dcom.sap.mobile.platform.server.maximum.connectionPoolSize	The maximum number of concurrent HTTP connections that can the server can make to any URL. Default value: 5000
-Dcom.sap.mobile.platform.server.connection.timeout	The amount of time, in milliseconds, before a database connection operation will time out. Default value: 60000
-Dcom.sap.mobile.platform.server.connection.sotimeout	The amount of time, in milliseconds, that the socket will wait for data from the back-end data source before disconnecting. Default value: 6000
-Dcom.sap.mobile.platform.server.enable.statistics	Enable or disable client statistics. true = collect statistics false = do not collect statistics
-Dcom.sap.mobile.platform.server.maximum.zipPoolSize	The maximum number of concurrent GZIP stream pool exists in server. This property allows you to increase the performance, so that the number can be set based on maximum number of concurrent requests which uses GZIP data. Default value: 5000
-Dhttp.nonProxyHosts	Host names for back-end communications inside the local network.
-Dhttp.proxyHost	Proxy host for back-end HTTP communications outside the local network.
-Dhttp.proxyPort	Proxy port for back-end HTTP communications outside the local network.
-Dhttps.proxyHost	Proxy host for back-end HTTPS communications outside the local network.
-Dhttps.proxyPort	Proxy port for back-end HTTPS communications outside the local network.

Property	Description
-Xms	Heap size represents the part of the memory used by Java. Default value: 1024M Modified value: 512M * CPU cores
-Xmx	Heap size represents the part of the memory used by Java. Default value: 2048M 512M * CPU cores
-XX:PermSize	Heap size represents the part of the memory used by Java. Default value: 256M Modified value: 256M
-XX:MaxPermSize	Heap size represents the part of the memory used by Java. Default value: 512M Modified value: N/A (default is recommended)

## Server Monitoring and Analysis

Server features that enable you to monitor system health, and analyze problems that can affect server performance or cause a disruption in the production environment.

### Managing Server Logs

Server logs are useful for monitoring SAP Mobile Platform system health, and troubleshooting. You can adjust the number of messages that are captured, set the maximum number of logs that are uploaded and stored, and view summary and detailed messages for various log types.

#### Setting Server Log Levels

You can change the server logging level to capture various levels of system messages for one or more system areas. These system messages can be an indicator of system health.

Logging detailed information consumes more system resources, so SAP recommends that you change the log level only when you suspect a serious problem, or are testing a theory. In most situations, logging errors and warnings is sufficient.



1. From Management Cockpit, select **Logs > Server Log Settings**.
2. Under Server Log Configuration, for each component, select a log level.

**Table 9. System Logging Components**

Component	Description
Foundation	Logs system messages for core SAP Mobile Platform Server functionality.
Security	Logs system messages that are related to SAP Mobile Platform security.
Hybrid Application Management	Logs system messages that relate to managing hybrid apps through the Management Cockpit or API, and client interactions for requesting and downloading updated hybrid apps.
B2C Framework	Logs system messages that are related to the Business-to-Consumer framework used for Mobiliser logs. Mobiliser must be enabled to set log levels.
B2C Application Services	Logs system messages that are related to the Business-to-Consumer services used for Mobiliser logs. Mobiliser must be enabled to set log levels.
Admin	Logs system messages that are related to SAP Mobile Platform administration.
Statistics	Logs system messages that are related to usage statistics.
Integration Gateway	Logs server-side system messages for plug-ins integrated with SAP Mobile Platform Server, such as data sources (SAP and OData), and third-party systems and protocols (such as SOAP, JDBC, HANA Cloud, and NetWeaver PI).
Other	Controls all other loggers not covered here. This is implemented by having a single logger named ROOT in this component, setting the level of the root logger affects any logger that does not fall under one of the loggers have been assigned a level explicitly
Agency user.*	Logs system messages related to Agency application users.

Component	Description
Agentry Server	Logs system messages related to Agentry applications.
Agentry Server.BackEnd.*	Logs system messages related to the Agentry back-end data source connections, such as SQL or Java.
Agentry WebSockets Front End	Logs system messages related to server/client communication using WebSockets connection type.
Agentry WebSockets Front End.connection.*	Logs system messages related to server/client connections.

Log Level	Description
Path	For tracing execution flow. Used, for example, in the context of entering and leaving a method, looping, and branching operations.
Debug	For debugging purposes, includes extensive and low level information.
Info	Informational text, used mostly for echoing what has been performed.
Warn	The application can recover from the anomaly, and fulfill the task, but requires attention from the developer or operator.
Error	The application can recover from the error, but cannot fulfill the task due to the error.
Fatal	The application cannot recover from the error, and the severe situation causes fatal termination.

**3. Select Save.**

**Setting Server Log Size**

Manually set the server log file size, and the number of rollover archive files to maintain in the `serviceability.xml` configuration file.

By default, the server log files grow to 20MB before rolling over, and 12 archived file instances are retained (for a total of 13 files, named `hostname-smp-server.log`, `hostname-smp-server_1.log`, `hostname-smp-server_2.log`, and so forth). Upon reaching the maximum size, the log overwrites the oldest archived file, and a new log is created. Since the log fills up quickly, establish a file size and rollover policy that meets your record-keeping and troubleshooting needs. For example, you might want to keep more and smaller files, and you might want to copy archive files to storage for later analysis before they are overwritten.

You must have sufficient file space to support your decision. To calculate log file system usage, use this formula:

$$(\text{Max} - \text{Min} + 2) * \text{Size} = \text{System Usage}$$

If you have sufficient file space, SAP recommends:

- MinIndex = 1; MaxIndex = 9
- MaxFileSize = 20MB

$$(9 - 1 + 2) * 20\text{MB} = 200\text{MB}$$

1. From File Manager, navigate to `SMP_HOME\Server\log\Server\configuration\`.
2. Open `serviceability.xml` in an editor such as Microsoft WordPad.
3. Set server log file values:

- a) Search for `<appender name="LJS_LOG_FILE"`.

```
<rollingPolicy
class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
  <FileNamePattern>log/hostname-smp-server_%i.log</
FileNamePattern>
  <MinIndex>1</MinIndex>
  <MaxIndex>12</MaxIndex>
</rollingPolicy>
```

```
<triggeringPolicy
class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">
  <MaxFileSize>20MB</MaxFileSize>
</triggeringPolicy>
```

- b) For rolling policy, set the number of archived file instances. For example, change the `<MaxIndex>12</MaxIndex>` value from 12 (the default) to 20 to create more archived files.
- c) For triggering policy, set the server log size. For example, change the `<MaxFileSize>20MB</MaxFileSize>` value from 20MB (the default) to 5MB to create smaller archive files:
- d) Compare the server log system usage requirements:

Default:

$$(12 - 1 + 2) * 20\text{MB} = 260\text{MB}$$

Updated:

$$(20 - 1 + 2) * 5\text{MB} = 105\text{MB}$$

4. Save and close the file.
5. Restart SAP Mobile Platform Server for the changes to take effect.

### **Viewing Server Logs**

View lists of rolling server logs by type, including SAP Mobile Platform Server, and Agency, from a single screen. Select a log file to drill down and view the raw data in the log file.

1. From Management Cockpit, select the **Logs** tab, and select **Server Logs**.
2. Expand SAP Mobile Platform Log to view the list of rolling logs for `hostname-smp-server.log`.  
Select a log file, using the Date Time stamp as a guide, to view log file contents. For example, select `hostname-smp-server1.log` to view its contents.
3. Expand Agentry Log to view the list of Agentry server logs. For example, select `Server-Agentry.log` to view its contents.

### SAP Mobile Platform Server Log File

The SAP Mobile Platform server log file captures server activity according to the log level set.

### *Server Log Format*

The following is an excerpt from a typical server log file. See the list of columns below the example, for a description of output columns. The system inserts the pound sign (#), if information is not included for the column.

```
COLUMNS:Time|TZone|Severity|Logger|ACH|User|Thread|Bundle name|
JPSPACE|JPAPpliance|JPComponent|Text|
SEVERITY_MAP:FINEST|Information|FINER|Information|FINE|Information|
CONFIG|Information|DEBUG|Information|PATH|Information|INFO|
Information|WARNING|Warning|ERROR|Error|SEVERE|Error|FATAL|Error
HEADER_END

2013 07 26
14:39:29#0-700#INFO#com.sap.core.js.mbeanserver.registerer.bundle.A
ctivator##anonymous#Start Level Event Dispatcher###An MBeanServer
has already been registered as an OSGi service. Nothing to register.|
2013 07 26
14:39:29#0-700#INFO#com.sap.core.js.monitoring.sapjvm.SAPJVMMetrics
##anonymous#Component Resolve Thread (Bundle 281)###JVM Vendor is:
SAP AG|
2013 07 26
14:39:29#0-700#INFO#com.sap.core.js.monitoring.MBeanManager##anonym
ous#Component Resolve Thread (Bundle 281)###MBeanServer:
com.sun.jmx.mbeanserver.JmxMBeanServer@25edb870|
2013 07 26
14:39:29#0-700#INFO#com.sap.core.js.monitoring.MBeanManager##anonym
ous#Component Resolve Thread (Bundle 281)###activate() is called
with:
org.eclipse.equinox.internal.ds.impl.ComponentContextImpl@5db2c30d|
2013 07 26
14:39:30#0-700#INFO#com.sap.persistence.dbtech.osgi.Activator##anon
ymous#Start Level Event Dispatcher###----- registered
DataSourceFactory service for MaxDB -----|
2013 07 26
14:39:31#0-700#INFO#com.sap.persistence.hdb.osgi.Activator##anonymo
us#Start Level Event Dispatcher###----- registered DataSourceFactory
service for SAP In-Memory Computing Engine -----|
```

Column	Description
Time	The date and timestamp of the log item, in the format YYYY MM DD HH:MM:SS.
TZone	The time zone, if included.
Severity	The log level, for example, DEBUG, PATH, INFO, WARN, ERROR, and FATAL. This value depends on the log level set for the system logging component.
Logger	The system logging component, such as <code>com.sap.persistence.dbtech.osgi.Activator</code> .
ACH	Not currently used
User	A system user, or anonymous.
Thread	The thread used to communicate.
Bundle Name	The bundle name, such as (Bundle 281).
JPSpace	Not currently used
JAppliance	Not currently used
JPComponent	Not currently used
Text	Message text for the logged item.

### **Configuring Agentry Application Logs**

Configure settings for Agentry logs. Agentry log file settings are configured as part of defining the application.

---

**Note:** Some entries require an SAP Mobile Platform Server start.

---

1. From Management Cockpit, select **Applications > App Specific Settings**.
2. Under Logging, edit log settings.

Definition	Values	Description
Event Log File	Valid file name or path and file name. Default: <code>events.log</code>	The file to which event log items are written. You can change the default to specify a different file name or location. The path can be relative to the server installation folder; or a full path that includes the drive letter. Requires server restart.
Maximum Log Size	Numeric value, assumed to be in bytes. Default: 0	The maximum size of the log file in bytes. A value of 0 indicates no maximum size. Any other value results in the log file rolling over when the file exceeds the value set here.

Definition	Values	Description
Log Roll Time	Time of day in 24-hour clock format (HH:MM). Default: 1:00am	The time at which to roll log files. The server must be running for the files to roll over at the specified time. Requires server restart.
Log Rolls At Time	Boolean value of true or false. Default: true	Whether to roll the log files based on the time in Log Roll Time. Requires server restart.
Message Log File	Valid file name or path and file name. Default: messages.log	The file to which message log items are written. You can change the default to specify a different file name or location. The path can be relative to the server installation folder; or a full path that includes the drive letter. Requires server restart.

3. Restart SAP Mobile Platform Server if you changed any setting that requires it.

### Agentry Logs

Use Management Cockpit to manage Agentry log files. You can enable or disable the various files, specify how log file are generated (based on a processing thread, or more narrow areas of functionality), and set the verbosity level of the message contents.

The log messages generated by the SAP Mobile Platform Server are categorized as follows:

- **Server:** Messages that are generated as a result of Agentry application processing. The contents are an inclusive set of the other categories, with the messages including all those related to any processing performed by the SAP Mobile Platform Server.
- **User:** Messages that are specific to a single user. User logs are inclusive of system connection and client communication logs, with the messages including all those related to any user-specific processing performed by the SAP Mobile Platform Server.
- **System Connection:** Messages that result from processing back-end data synchronization for a specific system connection. The contents are messages that result from processing the various synchronization definitions within the application. Each system connection within an application has its own set of log messages that you can individually enable and configure.
- **Client Communications:** Messages that result from client-server communication. These messages include IP addresses and port numbers, socket opening, data transfer in both directions, and other messages sent between the Agentry client and SAP Mobile Platform Server.

The contents of all log messages, regardless of category, include both error messages and informational or status messages. Each message includes a date and timestamp, precise to the millisecond. The verbosity of the messages generated is configured when the category of log messages are enabled.

### *Log Messages Versus Log Files*

*Log messages* generated by the Agentry application are stored in *log files*. When enabling and configuring the logging behavior of the Agentry application, the contents of the log messages

are configured. In addition, the files into which those messages are written is also configured. By default, when a log message category is enabled, those messages are written to a log file for a processing thread. Each Agentry application processing thread has its own log file containing all messages generated by the Agentry application for processing within that thread, and for the category of log messages that have been enabled.

As an optional additional method for storing log messages, each category of log messages can be written to a category-specific log file. When this is enabled, all log messages generated by the Agentry application for a specific category are written to the same log file, regardless of processing thread. The messages within the category-specific log files are grouped by the processing thread. Therefore, a category log file contains log messages resulting from processing related to that category.

In addition to the thread log files, you can generate log files by category:

- **Server category:** *Server-ServerName.log*, where *ServerName* is the System Name specified in the Agentry application definition. By default, the category is *Agentry*.
- **User category:** *User-UserID.log*, where *UserID* is the user's Agentry client login ID.
- **System connection category:** *BackEnd-SystemConnection.log*, where *SystemConnection* is the name of the system connection section (for example, *SQL-1, Java-2*).
- **Client communications category:** *FrontEnd-FrontEndType.log*, where *FrontEndType* is the connect type (for example, *WebSockets Front End*).

Configuring the log files to generate separately based on the log message category has a negative impact on Agentry application performance, as multiple processing threads are logging messages to the same physical log file. One message at a time is written to such a log; therefore, each thread must wait until the log file is available before it can write its log message. SAP recommends that you configure log files in this manner only in a development environment.

All log files are written to the Agentry application folder in *SMP\_HOME\log\agentry*. Log files are rolled over periodically, during SAP Mobile Platform Server start-up or restart, and, on command, to the subdirectory *rolled*.

### **Viewing Agentry Logs**

View a list of Agentry log files, and rolled-over Agentry server logs from a single screen, and access log file data in individual files.

1. From Management Cockpit, select the **Logs** tab, and select **Server Logs**.
2. Expand Agentry Log to view the list of Agentry server logs.
3. Select a log file to view details. Examples (your list will vary depending on your Agentry implementation):

## Server Administration

- `events.log`
- `messages.log`
- `startup.log`
- `Thread-xxxx.log`
- `rolled` – select to view rolled-over Agentry log files, organized into file folders by date.
- `BackEnd-Java-x.log`
- `BackEnd-SQL-x.log`
- `BackEnd-FileSystem-x.log`
- `BackEnd-HTTP-XML-x.log`

### Agentry messages.log File

The `messages.log` file contains log items that are related to client-server messaging. Each message and response sent between SAP Mobile Platform Server and Agentry client is one log item. The information in each item includes the date and time of the message, the Agentry client's IP address, the type of message or response, and other message information.

The messages log persists until one of the following occurs:

- The log files are rolled by SAP Mobile Platform Server, at which point they are moved to a backup folder named `rolled`, in a subdirectory named for the date and time. By default, this occurs once every 24 hours, provided SAP Mobile Platform Server is running. With a change to the application configuration, this file may also be moved to a backup location based on file size.
- SAP Mobile Platform Server is shut down and restarted. In this case, the `messages.log` file is moved to the backup folder `rolled`, in a subdirectory named for the date and time.

The `messages.log` file cannot be disabled.

### *Messages Log Format*

The following is an excerpt from a typical `messages.log` file.

```
I, 9, 1, 10, 09/12/2013 17:11:18, , 0, 70, , ANGEL: B0A6D984-
C97B-4275-B7E4-AEB8FC9D5210
Q, 9, 1, 10, 09/12/2013 17:11:18, , 0, 70, user1, ANGEL:
B0A6D984-C97B-4275-B7E4-AEB8FC9D5210
A, 9, 1, 10, 09/12/2013 17:11:18, , 0, 70, user1, ANGEL:
B0A6D984-C97B-4275-B7E4-AEB8FC9D5210
S, 9, 1, 10, 09/12/2013 17:11:22, 22, 23, 70, user1, ANGEL:
B0A6D984-C97B-4275-B7E4-AEB8FC9D5210
C, 9, 1, 10, 09/12/2013 17:11:22, , 23, 70, user1, >unknown<
```

- **Message State:** A single character indicating the state of the message in relation to SAP Mobile Platform Server.
- **Message Code:** A numeric value that identifies the type of message. See *Message Codes* for more information.



- **Subsystem Code:** Either 1 or 20. This is a deprecated value that varies only for older SAP products built on Agentry 2.0 code libraries.
- **Message Number:** A number assigned by the Agentry client that increments for each message sent.
- **Date and Time:** The date and time the message was received by the SAP Mobile Platform Server. This is always the date and time of the SAP Mobile Platform Server host system.
- **Response Code:** A numeric code that indicates the response to the message. The message codes “S” and “R” each have a different set of possible response codes. For all other message codes, the response code column is blank.
- **Data Sent/Data Received:** A running total of the amount of data sent and received by the SAP Mobile Platform Server per message, with a value in bytes.
- **User ID:** ID of the user to which the message belongs. It is normal for the user ID to remain blank for messages with a message state of “I”.
- **Client Location:** IP address of the client device. It is normal for the client location to be unknown for messages with a message state of “C”.

**Table 10. Message States—Synchronous Communication**

State	Name	Description
I	Incoming	Message is being received from the client.
Q	Queued	Message is decoded, and placed in an SAP Mobile Platform Server work queue; the user is identified.
A	Active	A previously queued message is being processed by the SAP Mobile Platform Server.
S	Sent Response	SAP Mobile Platform Server sent information or an acknowledgement to the client.
R	Received Response	A client response has been received by SAP Mobile Platform Server.
C	Complete	Message processing is complete.

**Table 11. Push Message States—Asynchronous Communication**

State	Name	Description
O	Outgoing	Message is being sent to the client.
T	Trying	SAP Mobile Platform Server is attempting to connect to a client
L	Linked	SAP Mobile Platform Server has successfully connected to a client.
W	Waiting	SAP Mobile Platform Server failed in an attempt to connect to a client. It will attempt the connection again.

State	Name	Description
R	Received Response	SAP Mobile Platform Server has received a client response.
S	Sent Response	SAP Mobile Platform Server sent information or an acknowledgement to the client.
C	Complete	Message processing is complete.
X	Cancelled	SAP Mobile Platform Server cancelled the message to the client.
F	Failed	Message failed. SAP Mobile Platform Server will not retry.

Message codes indicate the type of message being processed or sent, and are found in the Message Code column of the `messages.log` file.

**Table 12. Log Message Codes**

Code	Description
2	Client logout request
3	Client login request
7	Client user password change request
200	Transaction instance sent to SAP Mobile Platform Server from the client
201	Client has made a fetch processing request
202	Client system information message sent to SAP Mobile Platform Server
203	Client has requested an object be reloaded/refreshed
204	Client has requested the definition of an object
205	Client has requested the definition of a fetch
206	Client has requested the definition of a transaction
207	Client has requested the definition of a screen set
208	Client has requested the definition of an action
209	Client has requested the definition of a rule
210	Client has requested the definition of a report
211	SAP Mobile Platform Server has pushed an object or messages to the client
212	Client has sent an enable push message to SAP Mobile Platform Server for the user

Code	Description
213	Client has requested the definition of a style be sent
622	Client has requested that all complex tables be updated
623	Client has requested that all data tables be updated

### Agentry events.log File

The `events.log` file records various SAP Mobile Platform Server events that occur at run time. Events include startup and shutdown, opening and closing connections to the application's system connections, loading of client-server communication components, memory usage, thread expansion, and used and available storage.

The events log persists until one of the following occurs:

- The log files are rolled by SAP Mobile Platform Server, at which point they are moved to a backup folder named `rolled`, in a subdirectory named for the date and time. By default, this occurs once every 24 hours, provided the SAP Mobile Platform Server is running. With a change to the application configuration, this file may also be moved to a backup location based on file size.
- SAP Mobile Platform Server is shut down and restarted. In this case, the `events.log` file is moved to the backup folder `rolled`, in a subdirectory named for the date and time.

The `events.log` file cannot be disabled.

The following is an excerpt from a typical `events.log` file.

```
07/31/2013 01:45:00, 0, 0, 0, Thr 548, Rollover
time
07/31/2013 02:42:09, 0, 0, 7, Thr 548,
07/31/2013 03:42:10, 0, 0, 7, Thr 548,
07/31/2013 04:42:11, 0, 0, 7, Thr 548,
07/31/2013 05:42:12, 0, 0, 7, Thr 548,
07/31/2013 06:42:13, 0, 0, 7, Thr 548,
07/31/2013 07:42:14, 0, 0, 7, Thr 548,
07/31/2013 08:42:15, 0, 0, 7, Thr 548,
07/31/2013 09:42:16, 0, 0, 7, Thr 548,
07/31/2013 10:42:17, 0, 0, 7, Thr 548,
07/31/2013 11:42:18, 0, 0, 7, Thr 548,
07/31/2013 12:42:19, 0, 0, 7, Thr 548,
07/31/2013 13:42:20, 0, 0, 7, Thr 548,
07/31/2013 14:42:21, 0, 0, 7, Thr 548,
07/31/2013 15:42:22, 0, 0, 7, Thr 548,
07/31/2013 16:42:23, 0, 0, 7, Thr 548,
```

Each log message is written to a single line that begins with a timestamp. The next three columns, separated by commas, are the message type, message group, and the message ID. A message type of 1 indicates an error message, while a message type of 0 indicates an

informational message. The message type is followed by the thread ID, denoted by the text Thr., and finally the message itself.

### Agentry startup.log File

The Agentry startup.log file records various SAP Mobile Platform Server start-up events, for example, when Agentry log files, threads, and configuration files start.

### Agentry startup.log File

This is an excerpt from a typical Agentry startup.log file.

```
14:40:21 07/26/2013: Starting server (64-bit Windows)
14:40:21 07/26/2013: Reading system ini file.
14:40:22 07/26/2013: ID: Agentry, Name: ???, Location: ???
14:40:22 07/26/2013: Starting log file.
14:40:22 07/26/2013: Loading message groups.
14:40:22 07/26/2013: Starting threads.
14:40:22 07/26/2013:      1 initial threads. Threads will auto-scale.
14:40:22 07/26/2013: Starting Server: Agentry v7.0.0.352
14:40:22 07/26/2013: Event: 0, 2, System Startup
14:40:22 07/26/2013: Loading 1 front ends
14:40:22 07/26/2013: Loading front end from angelvine.dll
14:40:22 07/26/2013: ANGEL Front End: loading configuration
14:40:22 07/26/2013: Event: 17, 14, ANGEL Front End v7.0.0.352
14:40:22 07/26/2013: Loading front end from angelvine
14:40:22 07/26/2013: HTTPS Front End: loading configuration
14:40:22 07/26/2013: Event: 17, 14, HTTPS Front End v7.0.0.352
14:40:22 07/26/2013: Event: 0, 2, Loading the Agentry Server's
public/private key for password exchanges.
14:40:22 07/26/2013: Event: 0, 2, Key pair loaded successfully.
14:40:22 07/26/2013: Starting Server Agent.
14:40:22 07/26/2013: Agentry: Starting threads.
14:40:22 07/26/2013:      1 initial threads. Threads will auto-scale.
14:40:22 07/26/2013: Agentry: Adding messages.
14:40:22 07/26/2013: Event: 1, 4, Agentry v7.0.0.352
14:40:22 07/26/2013: Loading 1 agents
14:40:22 07/26/2013: Loading agent from ag3.dll
14:40:22 07/26/2013: Starting Server
14:40:22 07/26/2013: Server: reading ini file
14:40:23 07/26/2013: [System Connections] not found - please use
editor to publish application then restart the server.
14:40:23 07/26/2013: Exception: 14:40:23 07/26/2013 : 25 (General),
Unknown Exception ([System Connections] section not found in
Agentry.ini file (Publish from editor to correct), ), agent
\ChickamingAgent.cpp#2450:ChickamingAgent::startup
14:40:23 07/26/2013: Event: 20, 4, Server v7.0.0.352
14:40:23 07/26/2013: Starting front ends
14:40:23 07/26/2013: ANGEL Front End: Starting threads.
```

- **Time** – the date and time of the log item, including milliseconds, in the format HH:MM:SS MM/DD/YYYY.
- **Message Text** – the message text associated with the start-up event. Some messages indicate a message category, such as Event, Agentry, Server, and Exception.

### Agentry thread-x.log File

The Agentry thread-x.log file records SAP Mobile Platform Server events for the Agentry application.

### Agentry thread-x.log File

This an excerpt from a typical Agentry thread-x.log file.

```

2013/07/26 14:40:23.176: Opening log file
2013/07/26 14:40:23.176: + Thread=4924
2013/07/26 14:40:23.176:   + Server=Agentry
2013/07/26 14:40:23.176:     + Server=Agentry
2013/07/26 14:40:23.176:       Server: [System Connections] section
not found (Publish from editor to correct)
2013/07/26 14:40:24.640: Closing log file

```

- **Time** – the date and time of the log item, including milliseconds, in the format YYYY/MM/DD HH:MM:SS.
- **Message Text** – the thread identifier, and associated messages.

## Workload Analysis with Wily Introscope

Wily Introscope is a third-party diagnostic tool that can be integrated into your system landscape to quickly isolate and resolve performance issues through workload data, basic tracing functionality, and historic diagnostics data. Administrators can use performance data points for several metrics to assess overall performance of the system.

The Enterprise Manager serves as the central repository for all Introscope performance data and metrics collected in an application environment. The SAP Mobile Platform Introscope agent collects key performance metrics, which you can review using the Introscope Enterprise Manager dashboard user interface.

To use Introscope with SAP Mobile Platform Server, install Introscope Enterprise Manager, install SAP Mobile Platform Introscope Agent, and configure the agent.

**Note:** Introscope is not installed by SAP Mobile Platform; you must download and install it separately before you can use this method of monitoring.

---

### Using Introscope Agent for Workload Analysis

Wily Introscope is a third-party diagnostic tool that provides workload data, basic tracing functionality, and historic diagnostics data. To use Introscope with SAP Mobile Platform Server, you need to install the Introscope Enterprise Manager, install the SAP Mobile Platform Introscope Agent, and configure the agent.

### *Installing Introscope Enterprise Manager*

Install Introscope Enterprise Manager.

1. Download Introscope Enterprise Manager from the SAP Service Marketplace Software Download Center. The Introscope Enterprise Manager is located in the SAP Solution Manager Software Downloads section:
  - a. Navigate to the *SAP Service Marketplace Software Download Center* (login required).
  - b. In the left pane, select **Support Packages and Patches > Browse our Download Catalog**, then select **SAP Technology Components > SAP Solution Manager > SAP Solution Manager 7.1 > Entry by Component > Wily Introscope**.
  - c. Select the supported Introscope Enterprise Manager and follow screen prompts for download.
2. Install Introscope Enterprise Manager.

---

**Note:** For information about the supported of the Introscope Enterprise Manager and Introscope Agent, see SAP Product Availability Matrix (PAM) <http://service.sap.com/pam>. Click the **Mobile** link at the top of the page. Scroll to find the appropriate product and version in the product list. In the upper right corner of the SAP Mobile Platform PAM screen under the **Essentials** heading, click **Open in New Window** to open the *Support Matrices for SAP Mobile Platform 3.0* documentation.

---

### *Installing and Configuring the SAP Mobile Platform Introscope Agent*

Install and configure the supported Introscope Agent on SAP Mobile Platform Server to analyze performance metrics using the Introscope Enterprise Manager dashboard user interface.

---

**Note:** Do not configure or enable the agent unless the supported Introscope Enterprise Manager version is already installed in the production environment.

---

1. Navigate to the *SAP Service Marketplace Software Download Center* (login required).
2. In the left pane, select **Support Packages and Patches > Browse our Download Catalog**, then select **SAP Technology Components > SAP Solution Manager > SAP Solution Manager 7.1 > Entry by Component > Agents for Managed Systems > WILY INTRO AGT 9 MIN J5 > OS independent**.
3. Download the Introscope Agent file to *SMP\_HOME*\Introscope\_Agent.
4. Download the file attachments in *SAP Note 1911963* to *SMP\_HOME* \Introscope\_Agent\wily\core\config\.
5. Open the *SMP\_HOME*\server\props.ini file, and add the following lines immediately below the lines that begin with -Dcom:

```
-javaagent:C:\\SAP\\MobilePlatform3\\Introscope_Agent\\wily\\Agent.jar
-Dcom.wily.introscope.agentProfile=C:\\SAP\\MobilePlatform3\\Introsco
p_Agent\\wily\\core\\config\\IntroscopeAgent_SMP.profile
-Dcom.wily.introscope.agent.agentName=SAP Mobile Platform
-XX:-UseSplitVerifier
```
6. Reconfigure the SAP Mobile Platform Server Windows service for the change to take effect.

### *Configuring the IntroscopeAgent\_SMP.profile File*

Edit the `IntroscopeAgent_SMP.profile` file to specify the server name and port number where Enterprise Manager was installed.

1. Open the `SMP_HOME\Introscope_Agent\wily\core\config\IntroscopeAgent_SMP.profile` file as follows:
  - a. Modify `introscope.agent.enterprisemanager.transport.tcp.host.DEFAULT=localhost` to correspond with the installed Enterprise Manager host, instead of `localhost`.
  - b. Modify `introscope.agent.enterprisemanager.transport.tcp.port.DEFAULT=6001` to correspond with the installed Enterprise Manager host's port, which is 6001 by default.
2. Restart SAP Mobile Platform Server.

### *SAP Mobile Platform Metrics Collected By Introscope*

You can view the metrics collected by the SAP Mobile Platform Server in Introscope. Be aware that some metrics are contained and measured as part of other metrics, as identified below, and could be inadvertently be measured twice.

**Table 13. Introscope Metrics for SAP Mobile Platform**

Metric	Description
SMP Dispatcher Authentication	Time to authenticate client.
SMP ODataProxy Backend	Time to send and receive an HTTP request of an OData source, such as the SAP NetWeaver Gateway.
SMP Core Services Registration Create	Create a new entity representing a client registration.
SMP Core Services Registration Retrieve	Retrieve an entity representing a client registration.
SMP Admin Process	Process a request from the Management Cockpit.

### *Introscope Transaction Trace Sessions with SMP*

Introscope includes the ability to trace requests from clients. Introscope refers to this feature as a Transaction Trace Session. The requests traced may be filtered by User ID or by Request Header.

To filter by User ID in the dialog named “New Transaction Trace Session”, select “User ID” in the dropdown list. Then, select “equals”. Finally specify the User ID. The User ID is the same

as the User Name shown in the SAP Mobile Platform Administration display of Application Registrations. The User ID is the same value provided by the client for authentication.

Alternately, if the client specifies an Http header value of either X-SUP-APPCID or X-SMP-APPCID, then transactions may be filtered by a header value. The Http Cookie value with the same name will not work for filtering transactions. In the dialog named “New Transaction Trace Session”, select “Request Header” in the dropdown list. Input either X-SUP-APPCID or X-SMP-APPCID, depending upon which header the client is sending. Then, select “equals”. Finally specify the value sent by the client. The value for the APPCID is shown in the Management Cockpit display of Application Registrations and is labeled Registration ID.

### See also

- *Reconfiguring the Windows Service After Server Configuration Changes* on page 123

## Managing SAP Mobile Platform Server Features

---

Features are groups of individual bundles (also known as Eclipse plugins) and static resources that together form an installable function or set of functions. The SAP Mobile Platform Server installs with a default set of features enabled. To keep the server lightweight and easy to manage, optional features are installed but not enabled. You can enable optional features as required for your environment.

Custom features are assembled by server customization teams and development teams to provide extra server capabilities. To integrate custom features into the SAP Mobile Platform Server, you must identify the repository containing the custom features and then enable the custom features you require.

### *Enabling Optional Mobiliser Features*

Enable the optional Mobiliser features installed with SAP Mobile Platform Server .

---

**Note:** Enabling Mobiliser features can take anywhere between a few seconds and a few minutes, depending on the contents of the feature.

---

**Important:** Before enabling Mobiliser features, you must configure the SAP Mobile Platform Server custom database to support Mobiliser applications. For information on configuring custom databases for Mobiliser, see *Installation: SAP Mobile Platform Server* .

---

1. In the Management Cockpit, select the **SETTINGS** tab, then select **FEATURES**.
2. In the **Available Features** list, select the Mobiliser features that you want to enable.



Feature Name	Enables	Description
com.sap.mobile.platform.server. build.feature.mobiliser.feature.group	Mobiliser services	The services required to run Mobiliser Mobile Web and Mobiliser Web Portal using SAP Mobile Platform Server.  You cannot use the Management Cockpit until the feature is completely enabled. Once the feature is completely enabled, you must close the browser window and restart the Management Cockpit .
com.sap.mobile.platform.server. build.feature.mobiliser.web.mobile.feature.group	Mobiliser Mobile Web	A sample consumer web application. Mobile Web accesses Mobiliser services through the SAP Mobile Platform Server.
com.sap.mobile.platform.server. build.feature.mobiliser.web.portal.feature.group	Mobiliser Web Portal	User interface portals that provide consumer and administrative access to Mobiliser services. The Web Portal accesses Mobiliser services through the SAP Mobile Platform Server.
com.sap.mobile.platform.server.build.feature.mobiliser.odc	On Device Charging (ODC)	The services required to store sensitive data on a smart-phone that can interact with external systems through near-field communication.

After enabling Mobiliser features, perform the necessary Mobiliser setup.

You can customize the Mobiliser Web Portal templates. For more information on customization, see *Application Services (Mobiliser) Portal Templates*.

### *Enabling Custom Development Features*

To enable custom features for SAP Mobile Platform Server, you must first identify the feature repository. A feature repository (also known as an Eclipse update-site) is a Web or file location where features are stored. To integrate custom features with SAP Mobile Platform Server, the custom feature repository must be in a format that is understood by the Eclipse Equinox P2 provisioning process.

---

**Warning!** Enabling custom features can have severe and unintended effects on your server. Before enabling a custom feature, make sure that it is fully supported and tested for your environment.

---

1. In the Management Cockpit, select the **SETTINGS** tab, then select **REPOSITORIES**.
2. Click **New** and enter the URL or file location of the remote or local site where the features are published.
3. Click **Save**.

4. After adding the repository, select **FEATURES**.  
The features in the repository appear in the **Available Features** list.
5. Select the required features, click **Enable**, then click **Yes** on the confirmation dialog.

---

**Note:** Enabling features can take anywhere between a few seconds and a few minutes, depending on the contents of the feature. If it takes longer than 5 seconds, the operation continues in the background, and you must refresh the list to determine when the operation is complete.

---

### *Disabling SAP Mobile Platform Server Features*

You can disable optional or custom features on the SAP Mobile Platform Server. You should disable features if you do not need them anymore. You should also disable all features in a repository before deleting it. If you do not disable features before deleting a repository, they are removed from the list of enabled features when you delete the repository.

---

**Warning!** If you disable a feature that other features depend on, the dependent features will no longer work.

---

1. In Management Cockpit, select the **SETTINGS** tab, then select **FEATURES**.
2. In the **Enabled Features** list, select the feature or features that you want to disable.
3. Click **Disable**.

---

**Note:** Disabling features can take anywhere between a few seconds and a few minutes, depending on the contents of the feature.

---

### *Deleting a Custom Feature Repository*

In the current release, if you need an updated version of custom features, you must delete the custom feature repository, add it again, and then re-enable the required custom features. You may also want to delete a repository if you no longer need it, or if its location has changed.

---

**Note:** Once you delete a repository, you cannot enable or disable any of its features. Disable all features in the repository before you delete it.

---

1. In the Management Cockpit, select the **SETTINGS** tab, then select **REPOSITORIES**.
2. Select the repository to remove and click **Delete**.

### **See also**

- *Mobiliser Setup* on page 263

---

## **Backing Up Server Configuration Data**

---

You should perform regular backups of server configuration data so that it can be restored if there is a host system file corruption.

SAP Mobile Platform Server configuration data is a combination of file system properties, Windows registry settings, and database data.. Back up the following:

- Windows registry settings located at `HKEY_LOCAL_MACHINE\SOFTWARE\SAP\MobilePlatform\3.0`.
- Keystore and certificates located in `SMP_HOME\SAP\MobilePlatform3\Server\configuration\smp_keystore.jks`.
- The server database located in `SMP_HOME\SAP\MobilePlatform3\Server\db`. Back up the database using the vendor's recommended database backup procedures. Perform database backups on a regular basis to ensure the integrity of the backup and limit the size of the database transaction logs.

---

**Note:** Derby is not a supported production database.

---

**Note:** To reduce downtime, you can install a secondary server on a passive host, which can be immediately started upon failure of the primary active host. The practice of running an active/active spare configuration is not recommended in this version of SAP Mobile Platform as the application server in-memory cache is not currently synchronized between nodes.

---

## Port Number Reference

---

Components of SAP Mobile Platform rely on communication ports for inter-process coordination, data transfer, and administrative access.

### See also

- *Managing Firewalls and Intrusion Prevention* on page 215

## HTTP/HTTPS Port Number Reference

---

HTTP and HTTPS ports, default assignments, and protocols.

Type	Default	Protocol	Where Configured
client communications, unsecured	8080	HTTP	File: <code>SMP_HOME\Server\config_master\org.eclipse.gemini.web.tomcat\default-server.xml</code>  Setting: <code>&lt;Connector port="8080" ...</code>

Type	Default	Protocol	Where Configured
client communications with one way SSL authentication	8081 (secure)	HTTP S	File: <i>SMP_HOME</i> \Server\config_master\org.eclipse.gemini.web.tomcat\default-server.xml  Setting: <Connector protocol="com.sap.mobile.platform.coyote.http11.SapHttp11Protocol" port="8081" ...
client communications with mutual SSL authentication	8082 (secure)	HTTP S	File: <i>SMP_HOME</i> \Server\config_master\org.eclipse.gemini.web.tomcat\default-server.xml  Setting: <Connector protocol="com.sap.mobile.platform.coyote.http11.SapHttp11Protocol" port="8082" ...
administration port, Management Cockpit and Kapsel command line interface (CLI) communications with one way SSL authentication	8083 (secure)	HTTP S	File: <i>SMP_HOME</i> \Server\config_master\org.eclipse.gemini.web.tomcat\default-server.xml  Setting: <Connector protocol="com.sap.mobile.platform.coyote.http11.SapHttp11Protocol" port="8083" ...

---

**Note:** Keystore certificates are used for SSL authentication of SAP Mobile Platform Server communication ports.

---

**See also**

- *Starting and Stopping the Management Cockpit on Windows* on page 26

- *Starting and Stopping the Management Cockpit on Linux* on page 27
- *Configuring HTTP/HTTPS Port Properties* on page 114
- *Managing Keystore and Truststore Certificates* on page 223
- *Managing Firewalls and Intrusion Prevention* on page 215

## TCP Port Number Reference

TCP ports, default assignments, and protocols.

Type	Default	Pro- to- col	Where Configured
JMX	1717	TCP	File: <code>SMP_HOME\Server\config_master\com.sap.ljs.commandline.parameters\ljs.properties</code>  Setting: <code>jmxremote.port=1717</code>  File: <code>SMP_HOME\Server\config_master\com.sap.mobile.platform.server.foundation.security.jmx\com.sap.mobile.platform.server.foundation.security.jmx.properties</code>  Settings: <ul style="list-style-type: none"> <li>• <code>jmxPort=1717</code></li> <li>• <code>serviceUrl=service:jmx:rmi://127.0.0.1:1717/jndi/rmi://127.0.0.1:1717/jmxrmi</code></li> </ul>
Derby database (only when Derby is used as the SAP Mobile Platform Server database)	1527  (localhost only)	TCP	File: <code>SMP_HOME\Server\derby.properties</code>  Setting: <code>#derby.drda.portNumber=1527</code>  <b>Note:</b> Line is commented out (defaults internally to 1527). Uncomment the line when changing the port number.
OSGi console	2401  (localhost only)	TCP	File: <code>SMP_HOME\Server\props.ini</code>  Setting: <code>-console localhost:2401</code>

Type	Default	Pro- to- col	Where Configured
Mobiliser TCP	8088	TCP	File: <i>SMP_HOME</i> \Server\config_master\ com.sybase365.mobiliser.frame- work.gateway.tcp/com.sybase365.mo- biliser.framework.gateway.tcp.prop- erties  Setting: port=8088

**See also**

- *Managing Firewalls and Intrusion Prevention* on page 215

# Security Administration

Mobility has rapidly and dramatically changed computing and network environments. Before mobility, enterprise security primarily focused on the firewall and limited access to digital assets to only those users authenticated within the back end. A SAP Mobile Platform deployment introduces a multilayer approach to corporate security designed for mobility.

This approach ensures that Internal and external device users can securely connect to enterprise information systems. Every network link that transfers corporate information and every location that stores enterprise data guarantees confidentiality. Before you can prepare for the scale and scope of administration activities required to secure SAP Mobile Platform, learn which components you can secure, which communication streams you can protect, and how you can control access to mobile digital assets.

## See also

- *Application Administration* on page 31
- *Server Administration* on page 113
- *Application Services (Mobiliser) Administration* on page 261
- *SMS Administration* on page 341

## Planning Your Security Landscape

---

The security support matrices detail how SAP Mobile Platform supports various security configurations for client authentication and when using SSO to your back-end system. To properly plan your security environment in SAP Mobile Platform, understand authentication types, and the corresponding supported authentication providers, client types, and applications.

### *Client authentication*

When administering client authentication in SAP Mobile Platform, ensure you use supported authentication providers for your authentication and client types. Use this matrix to understand SAP Mobile Platform supported authentication scenarios for device (client) to SAP Mobile Platform Server connections, and the corresponding supported application types.

**Table 14. SAP Mobile Platform Client Authentication Matrix**

Authentication Type	Description	Authentication Provider(s)	Native Application Support	Hybrid Application Support	Agency Application Support
Anonymous	<p>No authentication of users, and grants read-only access to application data by assigning the anonymous security profile to the application,</p> <hr/> <p><b>Note:</b> Read-only access is dependent on how the application is configured. If the back-end connection has been configured to connect to a techni-</p>	<ul style="list-style-type: none"> <li>No specific authentication provider is required.</li> </ul> <hr/> <p><b>Note:</b> Do not use the No Authentication Challenge provider if you use back-end SSO.</p> <hr/>	Yes	Yes	Yes



Authentication Type	Description	Authentication Provider(s)	Native Application Support	Hybrid Application Support	Agentry Application Support
	cal user, it is possible for clients to perform write operations.				
Basic authentication	User name and password authentication	<ul style="list-style-type: none"> <li>• HTTP/HTTPS Authentication</li> <li>• Directory Service (LDAP/AD)</li> <li>• System Login (Admin Only)</li> <li>• No Authentication Challenge</li> </ul>	Yes	Yes	Yes

Authentication Type	Description	Authentication Provider(s)	Native Application Support	Hybrid Application Support	Agentry Application Support
External token-based SSO	<p>The application has custom code or logic to obtain a security token from a service external to SAP Mobile Platform. This token is added into the HTTP header and SAP Mobile Platform uses it for authentication.</p> <p>Site Minder is an example of a token-based SSO implementation.</p>	<ul style="list-style-type: none"> <li>• Populate JAAS Subject From Client</li> <li>• HTTP/HTTPS Authentication</li> </ul>	Yes	No	No

Authentication Type	Description	Authentication Provider(s)	Native Application Support	Hybrid Application Support	Agentry Application Support
Network-edge token-based SSO	The user enters credentials (either user name and password or X.509 certificate), and the credentials are checked at the network edge. When the network edge determines the credentials are valid, it may introduce a security token into the proxies client request (typically a cookie), and SAP Mobile Platform validates	<ul style="list-style-type: none"> <li>• Populate JAAS Subject From Client</li> <li>• HTTP/HTTPS Authentication</li> </ul>	Yes	Yes	Yes

Authentication Type	Description	Authentication Provider(s)	Native Application Support	Hybrid Application Support	Agentry Application Support
	the security token rather than the original user credentials.  Site Minder is an example of a token-based SSO implementation.				
X.509 certificate	Mutual certificate authentication	<ul style="list-style-type: none"> <li>X.509 User Certificate</li> </ul>	Yes	Yes	No

*Single sign-on to back-end systems*

When administering SSO to your back-end system with SAP Mobile Platform, make sure you use supported authentication providers for your SSO mechanism and application types. Use this matrix to understand SAP Mobile Platform supported authentication scenarios for SAP Mobile Platform Server to back-end connections, and the corresponding supported application types.

**Table 15. SAP Mobile Platform SSO Authentication Matrix**

<b>SSO Mechanism</b>	<b>Description</b>	<b>Authentication Provider(s)</b>	<b>Native Application Support</b>	<b>Hybrid Application Support</b>	<b>Agency Application Support</b>
Basic authentication	User name and password authentication	<ul style="list-style-type: none"> <li>• HTTP/HTTPS Authentication</li> <li>• No Authentication Challenge</li> </ul>	Yes	Yes	Yes

SSO Mechanism	Description	Authentication Provider(s)	Native Application Support	Hybrid Application Support	Agency Application Support
SSO2 Token	<p>HTTP headers or cookies that have an SSO value integrated with the customer's SSO systems. Use the HTTP/HTTPS Authentication provider to retrieve a MYSAPS-SO2 cookie from a Net Weaver token-issuing service.</p> <hr/> <p><b>Note:</b> Site Minder SSO tokens can be used against Net Weaver to retrieve the MYSAPS-SO2 cookie.</p>	<ul style="list-style-type: none"> <li>• HTTP/HTTPS Authentication</li> </ul>	Yes	No	No

SSO Mechanism	Description	Authentication Provider(s)	Native Application Support	Hybrid Application Support	Agency Application Support
X.509 single sign-on	Mutual certificate authentication	<ul style="list-style-type: none"> <li>• Populate JAAS Subject From Client</li> <li>• X.509 User Certificate</li> </ul> <hr/> <p><b>Note:</b> You must configure the application connection to use the certificate alias in the server keystore that should be used to make the HTTPS connection to the back end.</p> <p>The specified technical user certificate (configured using the certificate alias in the application definition) should be capable of impersonating the end user. The back end should be configured to trust the technical user to have validated the end user certificate. Configure this according to your specific back end in use.</p> <p>The CA certificate that signed the back end server certificate</p>	Yes	Yes	No

SSO Mechanism	Description	Authentication Provider(s)	Native Application Support	Hybrid Application Support	Agency Application Support
		should be imported into the SAP Mobile Platform keystore/truststore.			

**See also**

- *Platform Security Quick Start* on page 159
- *SAP Mobile Platform Authentication Quick Start* on page 159

## Security Postinstallation Checklist

Secure your SAP Mobile Platform Server installation as soon as possible after installation.

Task	Complete?
Secure the server infrastructure to prevent intrusions and allow SAP Mobile Platform Server to run with existing security mechanisms. See <i>Securing the Server Infrastructure</i> on page 215	
Perform your first backup of the server configuration data according to your backup and recovery strategy, and secure the identified backup artifacts. See <i>Backing Up Server Configuration Data</i> on page 144.	
Plan the specific security configurations needed for the rest of your security environment. See <i>Planning Your Security Landscape</i> on page 149.	
Create and map an SAP Mobile Platform Push User logical role for each security configuration that authorizes incoming push notifications. See <i>Notification User</i> on page 173 and <i>Notification Security Profile</i> on page 171.	
Create a Windows user account and assign it to the SAP Mobile Platform Windows service. See <i>Setting up a User Account for the SAP Mobile Platform Windows Service</i> on page 222	
Add a reverse proxy if required in your landscape. See <i>Adding Reverse Proxies</i> on page 209.	



Once these tasks are complete, implement the rest of the specific security configurations you have planned. To get started, see *Configuring Security Fundamentals in SAP Mobile Platform* on page 164.

## Platform Security Quick Start

---

Secure the SAP Mobile Platform Server by performing activities that secure the infrastructure and administration of components, in addition to enabling user authentication and secure communication.

**Table 16. Securing the SAP Mobile Platform Platform**

Task	Task Details
Secure server infrastructure	<i>Securing the Server Infrastructure</i> on page 215
Enable authentication for administrator logins	<i>Enabling Authentication for Administrator Logins</i> on page 217

### See also

- *Planning Your Security Landscape* on page 149
- *Securing the Server Infrastructure* on page 215

## SAP Mobile Platform Authentication Quick Start

---

This quick start task flow identifies security setup activities in an SAP Mobile Platform environment. Activities performed by the SAP Mobile Platform administrator, may also require the collaboration or participation of mobile application developers, or Afaria, security, or database administrators, depending on the role distribution within your organization.

### Prerequisites

Complete the *Platform Security Quick Start* on page 159 tasks before proceeding with authentication quick start tasks.

### Task

SAP Mobile Platform supports the following authentication types:

- **Anonymous authentication:** This provides no authentication of users, and allows access to application data, such as applications that allow users to browse read-only information without logging in by assigning the anonymous security configuration to the application.
- **Basic authentication:** This authentication type uses user name and password only. This is a less secure authentication type.

- **Token-based authentication:** Also called single sign-on (SSO), this authentication type enables access control for multiple, independent systems, affording the user a single login to access multiple secure systems. This is the most common authentication type.
- **Certificate-based authentication:** This authentication type uses X.509, a standard for digital certificates. X.509 provides the strongest form of authentication security in SAP Mobile Platform by managing authentication across platforms and out to the network edge.

**Table 17. Securing Platform Administration Tasks**

<b>Task</b>	<b>Anonymous Authentication</b>	<b>Basic Authentication</b>	<b>Token-based Authentication</b>	<b>Certificate-based Authentication</b>
Configure security profiles	<i>Configuring Security Profiles</i> on page 165	<i>Configuring Security Profiles</i> on page 165	<i>Configuring Security Profiles</i> on page 165	<i>Configuring Security Profiles</i> on page 165
Map a logical role to a physical role	Not applicable	<i>Mapping a Logical Role to a Physical Role</i> on page 174	<i>Mapping a Logical Role to a Physical Role</i> on page 174	<i>Mapping a Logical Role to a Physical Role</i> on page 174
Integrate SAP Mobile Platform with your SSO solution	Not applicable	Not applicable	<i>Integrating SAP Mobile Platform with Your SSO Solution</i> on page 233	Not applicable
Manage keystore and truststore certificates	Not applicable	Not applicable	Not applicable	<i>Managing Keystore and Truststore Certificates</i> on page 223
Deploy applications	<i>Deploy Applications</i> on page 31	<i>Deploy Applications</i> on page 31	<i>Deploy Applications</i> on page 31	<i>Deploy Applications</i> on page 31

**See also**

- *Planning Your Security Landscape* on page 149
- *Single Sign-on Integration Across Client Applications* on page 232

## SAP Mobile Platform Security Overview

---

SAP Mobile Platform provides a multilayer approach to corporate security designed for mobility.

This approach ensures that:

- Internal and external device users can securely connect to back-end systems.
- Every network link that transfers corporate information and every location that stores enterprise data guarantees confidentiality.

### Component Security

SAP Mobile Platform consists of multiple components that are installed on internal networks, primarily on the corporate LAN and the demilitarized zone (DMZ). Each component requires specific administration tasks to secure it.

Platform components use various SAP Mobile Platform security features. Some features are standards of the platform, while others are optional and up to the administrator or developer to implement.

Component	How Secured
Mobile application and local data	<ul style="list-style-type: none"> <li>• Login screens</li> <li>• Encryption of local data</li> <li>• Initial provisioning</li> <li>• Remote administration and security features</li> </ul>
SAP Mobile Platform Server and runtime services	<ul style="list-style-type: none"> <li>• Authentication of users and administrators</li> <li>• Enforcing device registration</li> <li>• Secure communication to subcomponents</li> <li>• Secure administration of server and services</li> </ul>
Database files	<ul style="list-style-type: none"> <li>• Secure database connection during installation</li> </ul>

### Communication Security

Secure SAP Mobile Platform component communications to prevent packet sniffing or data tampering. Different combinations of components communicate with different protocols and different ports.

End-to-end data encryption support is based on Transport Layer Security (TLS) and Secure Sockets Layer (SSL), which secure client/server communication using X.509 certificates.

Communication security spans:

- **Device-to-platform communications:** Devices connect via a reverse proxy. In SAP Mobile Platform deployments, a reverse proxy is the first line of defense to the platform, acting as a proxy for the device, and facilitating interactions with SAP Mobile Platforms installed on the corporate LAN.
- **Server-to-device application communications:** SAP Mobile Platform only supports HTTP/HTTPS connections from the client, and the response is sent back on the same channel. It is recommended that HTTPS protocol be used to secure the data communicated over the connection.
- **Server to Management Cockpit Communications:** Communications between SAP Mobile Platform Server and the Management Cockpit use one-way SSL authentication on port 8083 by default. While SAP Mobile Platform installs a sample certificate to enable the use of one way SSL authentication automatically, you should exchange the certificate with a production-ready one immediately following installation.

## Authentication

SAP Mobile Platform uses standard HTTPS protocol to integrate into the existing security landscape without disruption.

Rather than including an identity management and password and user access rights subsystem, SAP Mobile Platform provides features to integrate with your existing user and permissions stores so you can continue to use your existing user administration tools in conjunction with SAP Mobile Platform. Clients, administrators, and back-end systems can authenticate to SAP Mobile Platform Server using:

### Basic Authentication

Basic authentication is user name and password based.

The authentication mechanism relies on the standard Authorization: basic (base64 encoded username:password) HTTP header. Because the username:password can be decoded from the request, basic authentication should only be used over HTTPS.

Basic authentication in SAP Mobile Platform uses the following authentication providers:

- HTTP/HTTPS Authentication
- System Login (Admin Only)
- Directory Service (LDAP/AD)

### **See also**

- *HTTP/HTTPS Authentication Configuration Properties* on page 187
- *Directory Service (LDAP/AD) Configuration Properties* on page 197
- *System Login (Admin Only) Configuration Properties* on page 180

### **X.509 Certificate Authentication**

X.509 is client certificate authentication requiring an HTTPS connection to SAP Mobile Platform. SAP Mobile Platform can authenticate users based on their personal X.509 certificates.

SAP Mobile Platform supports certificate authentication using X.509, a standard for digital certificates. X.509 provides the strongest form of authentication security in SAP Mobile Platform by managing authentication across platforms and out to the network edge.

This form of authentication always uses the X.509 User Certificate authentication provider. Clients using this form of authentication must always connect to SAP Mobile Platform on the two-way HTTPS port (the default is port 8082). For authentication to succeed, the Certificate Authority (CA) certificate used to sign the user's certificate must be in the SAP Mobile Platform keystore.

#### **See also**

- *X.509 User Certificate Configuration Properties* on page 184
- *Configuring SAP Mobile Platform Server Certificate-based Authentication with a Reverse Proxy* on page 240

### **Single Sign-on and Token-based Authentication**

Single sign-on is token-based authentication where an SSO token, opaque to SAP Mobile Platform, is passed in an HTTP header or cookie

Single sign-on (SSO) enables access control for multiple, independent systems, affording the user a single login to access multiple secure systems. Token-based authentication uses opaque values from HTTP headers or cookies to authenticate the user against the customer's single sign-on systems.

The HTTP/HTTPS Authentication authentication provider is always used for this form of authentication. This authentication provider passes the token to a Web server that is integrated to the SSO system and is able to validate the token and potentially return more information back to SAP Mobile Platform about the user who owns the token. It may also return additional SSO credential material that SAP Mobile Platform can further use with back-end systems to identify the user.

## **Back-End Security**

SAP Mobile Platform secures interactions between SAP Mobile Platform Server and back-end systems. This security component includes device application back-end connections and notification communication.

- **Device application back-end connections** – SAP recommends that you use an encrypted connection whenever a connection to a back end is configured as an endpoint. To encrypt the connection established by an application, configure the security settings in the back-end connection properties.

- **Notification communications** – SAP Mobile Platform supports bi-directional communications between mobile applications and back-end systems. You can subscribe to and retrieve enterprise data, and send business transactions to them. The back end can send notifications to the applications when information they have subscribed to changes.

### Common Security Infrastructure in SAP Mobile Platform

The SAP Mobile Platform Common Security Infrastructure (CSI) provides an extensible model for integrating with existing security infrastructure.

CSI login modules conform with Java JAAS, which enables SAP Mobile Platform to integrate with LDAP, Microsoft Active Directory, and so on.

Security within SAP Mobile Platform applies to the following components:

- **Server security** – SAP Mobile Platform Server provides data services to device clients by interacting with databases. The database is configured along with server components, to the internal corporate LAN. Each runtime service uses its own communication port (secured and unsecured).
- **Application security** – secure interactions with back-end data sources. Application security ensures that device users can access application data by logging in through a configured security profile. If you are using Push Notification, those notifications can be communicated to SAP Mobile Platform Server securely.
- **Device security** – developers and administrators can combine multiple mechanisms to fully secure devices. SAP Mobile Platform security features for devices include data encryption, login screens, and data vaults for storing sensitive data.
- **Agentry security** – the Agentry Server supports client data and password encryption, encrypted client-server communications, and authentication certificates.

### Configuring Security Fundamentals in SAP Mobile Platform

To configure and manage security across SAP Mobile Platform, you need to understand security profiles and how to configure them, logical roles and how they map to your physical roles, and authentication providers and how they are configured in security profiles. You can also configure reverse proxies if needed in your landscape.

### Security Profiles in SAP Mobile Platform

SAP Mobile Platform does not provide proprietary security systems for storing and maintaining users and access control rules, but delegates these functions to the enterprise's existing security solutions.

A security profile determines the scope of user identity, data access, and security by performing authentication and authorization checks. A user must be part of the security repository used by the configured security profiles to access any resources (either a Management Cockpit administration feature or a data set from a back-end data source).

SAP Mobile Platform includes three default security profiles: Admin, Default, and Notification. Administrators can also create new security profiles and assign authentication providers using Management Cockpit.

Security profiles aggregate various security mechanisms for protecting SAP Mobile Platform resources under a specific name, which administrators can then assign. Each security profile consists of:

- Configured security providers: Security provider plug-ins for many common security solutions, such as LDAP, are included with SAP Mobile Platform.
- Role mappings that map SAP Mobile Platform logical roles to back-end physical roles.

A user entry must be stored in the security repository used by the configured authentication provider to access any resources. When a user attempts to access a particular resource, SAP Mobile Platform Server tries to authenticate and authorize the user, by checking the security repository for security access policies on the requested resource and role memberships.

### **See also**

- *Managing Security Profiles* on page 94
- *Defining Application Authentication* on page 58
- *Creating and Configuring Security Profiles with Authentication Providers* on page 165

### **Creating and Configuring Security Profiles with Authentication Providers**

Use security profiles, and corresponding or required authentication providers, to protect SAP Mobile Platform resources, according to the specific security requirements of your environment.

### **See also**

- *Defining Application Authentication* on page 58
- *Security Profiles in SAP Mobile Platform* on page 164

### **Creating and Configuring Security Profiles**

Create and configure security profiles to define parameters that control how the server authenticates the user during onboarding, and request-response interactions with the back end. You can also define additional SAP Mobile Platform administrator users by creating a new security profile and configuring the required authentication provider.

Guidelines:

- Agentry applications have an additional authentication layer that is configured elsewhere; for Agentry applications, a common scenario is to configure No Authentication Challenge as the security provider, so that only Agentry performs authentication.

- You can stack multiple security providers to leverage security features in complex systems. You can order stacked security providers to take advantage of features in the order you chose. The Control Flag must be set for each enabled security provider in the stack.
- You must map logical roles to physical roles, as required by the application.
- When you create a new security profile, a corresponding XML file is created in the `SMP_HOME\configuration\com.sap.mobile.platform.server.security\CSI\` directory. When the security profile is updated, a copy of the XML file is saved to allow you to recover the security profile.

1. From Management Cockpit, select the **Settings** tab, and select **Security Profiles**.
2. Click **New**.
3. Under Security Profile Properties, enter values.

Field	Value
Name	A unique name for the application authentication profile.
Check Impersonation	(Optional) In token-based authentication, whether to allow authentication to succeed when the user name presented cannot be matched against any of the user names validated in the login modules. By default the property is enabled, which prevents the user authentication from succeeding in this scenario.

4. Under Authentication Providers, set up one or more providers for the application.
  - a) Click **New**.
  - b) In the Add Authentication Provider dialog, select a provider from the list, and click **Create**.

Authentication Provider	Description
No Authentication Challenge	Provider that always authenticates the supplied user. The provider offers pass-through security for SAP Mobile Platform Server, and should typically be reserved for development or testing. SAP strongly encourages you to avoid using this provider in production environments—either for administration or device user authentication.
System Login (Admin Only)	Provider that is configured by the installer with the initial administrator credentials only to give platform administrator access to Management Cockpit, so that SAP Mobile Platform Server can be configured for production use. Administrators are expected to replace this authentication provider immediately upon logging in for the first time. SAP encourages you to avoid using this provider in production environments.



Authentication Provider	Description
Populate JAAS Subject From Client	<p>Provider that enables administrators to add client values as named credentials, name principals, and role principals to the authenticated subject. This provider copies values from the client's HTTP request into the JAAS subject as:</p> <ul style="list-style-type: none"> <li>• Principals - identifies the user</li> <li>• Roles - grants access rights to SAP Mobile Platform protected resources</li> <li>• Credentials - provides single-sign-on material to use when connecting to back-end systems</li> </ul> <p>Adding client values as named credentials allows them to be used for single sign-on.</p>
X.509 User Certificate	<p>Provider to use when the user is authenticated by certificates. This provider can be used in conjunction with other authentication providers that support certificate authentication [for example, Directory Service (LDAP/AD)], by configuring X.509 User Certificate before the authentication providers that support certificate authentication. You can only use this provider to validate client certificates when HTTPS listeners are configured to use mutual authentication.</p> <hr/> <p><b>Note:</b> Agentry clients on iOS and Android do not support client/user certificates. Agentry clients on Windows and Windows CE support client-side certificates, but Agentry cannot use these certificates for user identification; Agentry requires separate user name and password authentication as well.</p>
HTTP/HTTPS Authentication	<p>Provider that authenticates the user with given credentials (user name and password, or SSO tokens from your SSO system) against a back end that is integrated to the your management or SSO systems. Optionally this provider may retrieve a cookie that represents additional SSO credentials to use for back-end systems that are also integrated with your SSO system.</p>

Authentication Provider	Description
Directory Service (LDAP/AD)	<p>Provider that integrates with the your Active Directory or other Directory Server identity management system using LDAP. It first connects to your Directory Server using a technical user identity so it can perform an LDAP search to discover the fully qualified distinguished name (DN) of the current user in the directory. It then performs a bind to that DN with the provided password. When the bind succeeds, the user is considered authenticated. The provider then performs an LDAP search to see which groups the user is a member of. These group names are then considered as physical roles in the role mapping definitions that are used later for access controls.</p> <p>This provider is particularly useful in the Admin security profile to grant existing enterprise users usage of the Management Cockpit, and also any custom security profiles used for authenticating enterprise users for SAP Mobile Platform application usage.</p>

- c) Enter values based on the selected authentication provider.
- d) (Optional) Click **New** to add additional security providers on the stack. Use the up and down arrow icons to move the security providers into the desired order.

5. Click **Save**, and confirm.

**Next**

You must configure role mapping (map logical roles to physical roles) as required by the application.

**See also**

- *controlFlag Attribute Values* on page 208
- *Check Impersonation Attribute* on page 244
- *Mapping a Logical Role to a Physical Role* on page 174
- *No Authentication Challenge Configuration Properties* on page 178
- *Default Logical Roles in SAP Mobile Platform* on page 172
- *System Login (Admin Only) Configuration Properties* on page 180
- *Populate JAAS Subject From Client Configuration Properties* on page 182
- *Enabling OCSP* on page 250
- *X.509 User Certificate Configuration Properties* on page 184
- *HTTP/HTTPS Authentication Configuration Properties* on page 187
- *Directory Service (LDAP/AD) Configuration Properties* on page 197
- *Debugging Authentication Errors with CSI Tool* on page 258
- *Single Sign-on Integration Across Client Applications* on page 232

### Stacking Providers and Combining Authentication Results

Implement multiple authentication providers to provide a security solution that meets complex security requirements. SAP recommends provider stacking as a means of eliciting more precise results, especially for production environments that require different authentications schemes for administrators, Push Notification, SSO, and so on.

Stacking is implemented with a `controlFlag` attribute that controls overall behavior when you enable multiple providers. Set the `controlFlag` on a specific provider to refine how results are processed.

For example, say your administrative users (`smpAdmin` in a default installation) are not also users in a back-end system like SAP. However, if they are authenticated with just the default security configuration, they cannot also authenticate to the HTTP/HTTPS Authentication provider used for SSO2Token retrieval. In this case, you would stack a second authentication provider with a `controlFlag=sufficient` authentication provider for your administrative users.

Or, in a custom security profile (recommended), you may also find that you are using a technical user for Push Notification who is also not an SAP user. This technical user does not need SSO because they will not need to access data. However, the technical user still needs to be authenticated by SAP Mobile Platform Server. In this case, you can also stack another authentication provider so this Notification user can login.

1. Use Management Cockpit to create a security profile and add multiple providers as required for authentication.
2. Order multiple providers by selecting an authentication provider, and using the up or down arrows to place the provider correctly in the list.

The order of the list determines the order in which authentication results are evaluated.

3. For each provider:
  - a) Select the provider name.
  - b) Configure the `controlFlag` property with one of the available values: required, requisite, sufficient, optional.  
See *controlFlag Attribute Values* for descriptions of each available value.
  - c) Configure any other common security properties as required.
4. Click **Save**.

For example, if you have sorted these authentication providers in the following order, and used these `controlFlag` values:

- LDAP (required)
- NT Login (sufficient)
- SSO Token (requisite)
- Certificate (optional)

The results are processed as indicated in this table:

Pro- vider	Authentication Status							
	Directo- ry Serv- ice (LDAP/ AD)	pass	pass	pass	pass	fail	fail	fail
NT Log- in	pass	fail	fail	fail	pass	fail	fail	fail
SSO To- ken	*	pass	pass	fail	*	pass	pass	fail
Certifi- cate	*	pass	fail	*	*	pass	fail	*
<b>Overall result</b>	pass	pass	pass	fail	fail	fail	fail	fail

---

**Note:** The \* means the corresponding authentication provider is not called because of the outcomes of previous providers in the list and the controlFlag settings. For example, in the first column of the table, since both LDAP and NT Login succeeded, the controlFlag settings for SSO Token and Certificate are such that they need not be invoked at all. In the 3rd column, the "sufficient" NT Login has failed the SSO Token module is invoked.

---

**See also**

- *Defining Application Authentication* on page 58
- *controlFlag Attribute Values* on page 208

**Predefined Security Profiles**

By default, SAP Mobile Platform Server includes three security profiles, which are used to protect SAP Mobile Platform resources.

The following security profiles are provided in SAP Mobile Platform:

Admin Security Profile

The Admin security profile is used to authenticate and authorize administrative users.

The administrator can configure the Admin security profile to include any combination of security providers as needed for administration. Once the profile is configured, the administrator must edit the `admin-role-mapping.xml` to map the Administrator logical role to the appropriate physical roles.

---

**Note:** Because Gateway on Java Server logical roles map to the SAP Mobile Platform Administrator role by default, pay special attention when mapping roles and using the Admin security profile.

---

### Default Security Profile

The Default security profile is used to authenticate access to SAP Mobile Platform Server if a client application did not specify one explicitly. This profile is also used by any container managed authentication if a Web application is deployed to the SAP Mobile Platform Server.

The administrator can configure the Default security profile to include any combination of security providers as needed. However, there are no specific roles that need to be mapped for normal use of Default security profile.

### Notification Security Profile

The Notification security profile is to enable push notifications using the Notification User role.

The administrator can configure the Notification security profile to include any combination of security providers as needed. Once the profile is configured, the administrator must edit the `Notification-role-mapping.xml` to map the Notification User logical role to the appropriate physical roles.

In SAP Mobile Platform, the back end sends a notification to synchronize data directly to a specific device. When the back end sends a notification to a client application, which has subscribed to it, it connects to the SAP Mobile Platform to forward the notification. For example, for iOS the notification is forwarded via Apple Push Notifications (APNS) and for Android the notification is forwarded via Google Cloud Messaging (GCM).

### Security Profiles for Device Users

To protect SAP Mobile Platform resources, administrators create and name a set of security providers and physical security roles. For device user authentication, create at least one security profile that is not the Admin security profile, which is used exclusively for administrator authentication in Management Cockpit.

## Role Mapping

All users, passwords, and access control information is managed in the your identity management systems using the administrative tools you normally use for these systems.

For administrative users to have access to the SAP Mobile Platform Server in a production environment, you must map the SAP Mobile Platform default logical roles to the corresponding physical roles or groups in the security repository.

### **See also**

- *Adding a Production-Grade Provider* on page 219

- *Mapping a Logical Role to a Physical Role* on page 174

### **Default Logical Roles in SAP Mobile Platform**

SAP Mobile Platform roles are logical roles that are built into the system by default.

To enable role-based access to the administrative interface, configure mapping of the SAP Mobile Platform Administrator, Developer, Helpdesk, and Notification User logical roles to roles that exist in the security repository used for administrative authentication and authorization.

---

**Note:** The SAP Mobile Platform Helpdesk role provides read-only access to the administrative interface.

---

### **See also**

- *System Login (Admin Only) Configuration Properties* on page 180
- *Creating and Configuring Security Profiles* on page 165
- *Mapping a Logical Role to a Physical Role* on page 174

### **Administrator**

Administrators interact with SAP Mobile Platform to perform high-level management.

The administrator can perform all administrative operations in the SAP Mobile Platform Management Cockpit. The logical role for the platform administrator is Administrator. The term "administrator" is used in all documentation to refer to the user with Administrator role.

### **Developer**

Developers interact with SAP Mobile Platform to develop applications.

The developer can perform development operations in the SAP Mobile Platform Management Cockpit, including application development. The logical role for the developer is Developer. The term "developer" is used in all documentation to refer to the user with the Developer role.

---

**Note:** The developer role is not supported in SAP Mobile Platform version 3.0. A user with this role is not granted any access to the Management Cockpit.

---

### **Helpdesk**

Helpdesk operators interact with SAP Mobile Platform to review system information to determine the root cause of reported problems.

Helpdesk operators have read-only access to all administration information in the SAP Mobile Platform administration console. They cannot perform any modification operations on administration console tabs, and cannot save changes made in dialogs or wizards.

The logical role for the helpdesk operator is Helpdesk. The term "helpdesk" is used in all documentation to refer to the user with the Helpdesk role.

---

**Note:** The helpdesk role is not supported in SAP Mobile Platform version 3.0. A user with this role is not granted any access to the Management Cockpit.

---

### Notification User

Notification users interact with SAP Mobile Platform to enable and configure push notifications.

In SAP Mobile Platform, administrators configure the Notification security profile to determine the authentication credentials required to send Push Notifications. The administrator can update the Notification security profile to include any combination of login modules as needed.

Administrators can configure the back end with a user x.509 certificate and connect to SAP Mobile Platform on its HTTPS listener configured to use mutual authentication (port 8082 by default). Once the Notification security profile is configured, you must edit the `Notification-role-mapping.xml` to map the Notification User logical role to the appropriate physical roles:

```
<DefaultMapping>
<LogicalName>Notification User</LogicalName>
<MappedName>user:subjectDN from the certificate</MappedName>
</DefaultMapping>
```

---

**Note:** The logical role for the notification user is Notification User. The term "notification user" is used in all documentation to refer to the user with Notification User role.

---

### Impersonator

The Impersonator role establishes the trust relationship between the reverse proxy and SAP Mobile Platform Server allowing SAP Mobile Platform Server to accept and authenticate the user's public certificate presented in the `SSL_CLIENT_HEADER` over the SSL connection established by the reverse proxy.

---

**Note:** The Impersonator role should be granted to the reverse proxy by mapping the Impersonator role to the subjectDN from the certificate used by the reverse proxy to establish a mutual authentication SSL connection to SAP Mobile Platform Server.

---

### *Copy SubjectDN for Impersonator Role*

Copy the SubjectDN from the server log.

1. Set the Security log level to Debug in the Admin Cockpit.
2. Perform a client request through the reverse proxy.
3. Open the server log, located in `SMP_HOME\Server\log\hostname-smp-server.log`.
4. Copy the SubjectDN, exactly as it is seen by SMP, into the role mapping file using the format:

```
<DefaultMapping>
<LogicalName>Notification User</LogicalName>
```

```
<MappedName>user:subjectDN from the certificate</MappedName>
</DefaultMapping>
```

### Integration Gateway Roles

Integration Gateway works with SAP Mobile Platform, providing development and generation tools for connecting to different datasources (both SAP and non-SAP) to create OData services.

Integration Gateway includes Toolkit for Integration Gateway, which provides an environment to connect to different data sources and to create and deploy artifacts on SAP Mobile Platform Server. In order to generate and deploy content, toolkit users must have the appropriate user role on SAP Mobile Platform Server.

Role	Required For
GenerationAndBuild.generationandbuildcontent	Generate and build operations.
NodeManager.deploycontent	Deploy and undeploy content operations.
IntegrationOperationServer.read	Read only operations.

By default, these roles are mapped to the Administrator logical role. When performing role-mapping, map the Integration Gateway roles to the appropriate physical roles in the Admin security provider.

### **See also**

- *Administrator Overview for Integration Gateway* on page 3

### **Mapping a Logical Role to a Physical Role**

SAP Mobile Platform uses a role-based-access-control (RBAC) security model. SAP Mobile Platform includes predefined logical roles, and uses `HttpServletRequest.isUserInRole(logicalRole)` for its policy enforcement points in the runtime. Physical roles assigned to a user come from the customer's identity management back-end systems.

The most common example is the LDAP groups a user belongs to when using the Directory Service (LDAP/AD) authentication provider. Each LDAP group becomes a physical role attributed to the authenticated user in SAP Mobile Platform.

The CSI then uses the role-mapping configuration to convert the `isUserInRole()` check to see if the user is granted any of the physical roles defined in the role-mapping for the security profile. Role mapping is particularly important for the Admin security configuration where authorized users must be mapped to the Administrator logical role. Additionally, in other security profiles, it is important to map Impersonator and Notification User roles.

The security profiles are persisted in files that are located in `SMP_HOME\Server\configuration\com.sap.mobile.platform.server.security\CSI`. To map a logical role to the appropriate physical role in the underlying security provider in a given



security profile, you must manually edit the corresponding `<Security_Profile_name>-role-mapping.xml` file.

---

**Note:** The Management Cockpit always authenticates against the Admin security profile and requires that the user be granted Administrator role to successfully log into the Management Cockpit.

---

Upon installation, the default authentication provider assigns the `smpAdmin` user to the administrator role. To make your configuration production ready, you must add an authentication provider to the Admin security profile that authenticates against your identity management system (such as LDAP for Active Directory). To do this, you must:

- Determine the physical role names detected by your identity management system (iDMS), for example, the names of LDAP groups to which the user belongs
- Select appropriate logical roles in SAP Mobile Platform

---

**Note:** In order for administrators to access SAP Mobile Platform Server, you must map the default SAP Mobile Platform logical roles to the corresponding physical roles in the Admin security provider. You perform the mapping for the Admin security profile manually by editing the `SMP_HOME\Server\configuration\com.sap.mobile.platform.server.security\CSI\admin-role-mapping.xml` file. To map logical roles to physical roles, perform the following steps which use the Administrator role as an example.

---

1. Navigate to the `admin-role-mapping.xml` file, which by default appears as:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <rm:Mappings xmlns:rm="http://www.sybase.com/csi/3.1/mapping">
- <DefaultMapping>
  <LogicalName>Administrator</LogicalName>
  <MappedName>Administrator</MappedName>
</DefaultMapping>
  <!-- Avatar Deployer Role Mappings -->
- <DefaultMapping>
  <LogicalName>NodeManager.deploycontent</LogicalName>
  <MappedName>Administrator</MappedName>
</DefaultMapping>
- <DefaultMapping>
  <LogicalName>GenerationAndBuild.generationandbuildcontent</
LogicalName>
  <MappedName>Administrator</MappedName>
</DefaultMapping>
- <DefaultMapping>
  <LogicalName>IntegrationOperationServer.read</LogicalName>
  <MappedName>Administrator</MappedName>
</DefaultMapping>
- <DefaultMapping>
  <LogicalName>Developer</LogicalName>
  <MappedName>Developer</MappedName>
```

```
</DefaultMapping>
- <DefaultMapping>
  <LogicalName>Helpdesk</LogicalName>
  <MappedName>Helpdesk</MappedName>
</DefaultMapping>
- <DefaultMapping>
  <LogicalName>Notification User</LogicalName>
  <MappedName>Notification User</MappedName>
</DefaultMapping>
- <DefaultMapping>
  <LogicalName>Impersonator</LogicalName>
  <MappedName>Impersonator</MappedName>
</DefaultMapping>
</rm:Mappings>
```

---

**Note:** Each logical role name is mapped to a physical role of the same name. By default, the Admin security profile assigns the `smpAdmin` user to the Administrator role.

---

2. Edit the file to assign your required physical roles to the corresponding logical roles. For example, if you have a physical role of *Engineering* in an LDAP environment, edit the mapping file to assign the mapped name of *Engineering* to the appropriate logical role:

```
<DefaultMapping>
  <LogicalName>Administrator</LogicalName>
  <MappedName>Administrator</MappedName>
  <MappedName>Engineering</MappedName>
</DefaultMapping>
```

---

**Note:** If there is no physical role or group called Administrator, the mapping should be deleted so that an extra role check can be avoided to speed up the authorization checks.

---

3. Save the file.
4. Restart the server.
5. Login to the Management Cockpit to verify the configuration.

---

**Note:** Be sure to understand your groups of users so you map only the roles intended for the security profile. Mapping large groups risks including more users than necessary in your security profile. To mitigate this risk, consider using the `UserRoleAuthorizer` feature to provide improved security by defining a specific user, and not a group, in LDAP. This technique is required for certificate-based authentication.

---

### See also

- *Creating and Configuring Security Profiles* on page 165
- *No Authentication Challenge Configuration Properties* on page 178
- *Default Logical Roles in SAP Mobile Platform* on page 172
- *System Login (Admin Only) Configuration Properties* on page 180
- *Populate JAAS Subject From Client Configuration Properties* on page 182
- *Enabling OCSP* on page 250

- *X.509 User Certificate Configuration Properties* on page 184
- *HTTP/HTTPS Authentication Configuration Properties* on page 187
- *Directory Service (LDAP/AD) Configuration Properties* on page 197
- *Single Sign-on Integration Across Client Applications* on page 232
- *LDAP Role Computation* on page 194
- *Role Mapping* on page 171
- *UserRoleAuthorizer Provider* on page 207
- *X.509 User Certificate Provider* on page 184

## **Authentication in SAP Mobile Platform**

A security provider verifies the identities of application users and administrators who request access via one or more configured login modules.

Device user authentication and administrator authentication are configured differently:

- Device users are authenticated with custom SAP Mobile Platform Server security profiles created by the platform administrator in Management Cockpit. For SAP back ends, SSO authentication can be configured.
- Administrators are authenticated with the Admin security profile. For first-time logins, administrators are authenticated with the System Login (Admin Only) authentication provider. Once logged in, administrators for production systems should immediately reconfigure the Admin security profile to use the enterprise security back end using Management Cockpit.

### **See also**

- *Defining Application Authentication* on page 58

## **Authentication Providers for User Authentication and Authorization**

A security profiles verifies the identities of application users and administrators who request access via one or more configured authentication provider. SAP Mobile Platform Server supports a variety of built-in authentication providers that authenticate users. Administrators create a security profile and assign one or more providers to it using Management Cockpit.

You can configure a provider of a given type only if that provider is available on the enterprise network.

Administrator authentication and device user authentication are configured differently:

- Administrators are authenticated with the Admin security profile. For first-time logins, administrators are authenticated with the System Login (Admin Only) authentication provider. Once logged in, administrators for production systems should immediately reconfigure the Admin security profile to use the enterprise security back end using Management Cockpit.

- Device users are authenticated with custom SAP Mobile Platform Server security profiles created by the platform administrator in Management Cockpit. For SAP back ends, SSO authentication can be configured.

### No Authentication Challenge Provider

The No Authentication Challenge provider offers pass-through security for SAP Mobile Platform Server, and is intended for use in development environments or for deployments that require no security control. Do not use this provider in production environments, for administration, or for device user authentication.

If you use the No Authentication Challenge provider, all login attempts succeed, no matter what values are used for the user name and password. Additionally, all role and control checks based on attributes also succeed.

---

**Note:** If you configure No Authentication Challenge in a security profile to which you have assigned client applications that you intend to run anonymously, this provider causes your anonymous applications to fail. SAP Mobile Platform authenticates the user even though the user presented no valid credentials. SAP Mobile Platform then proceeds to connect to back-end systems assuming there is an authenticated client, and tries to use SSO credentials for the back end. However, these credentials are absent, and the back-end connection fails.

---

### See also

- *No Authentication Challenge Configuration Properties* on page 178

### *No Authentication Challenge Configuration Properties*

The No Authentication Challenge provider offers pass-through security for SAP Mobile Platform Server, and should typically be reserved for development or testing. SAP strongly encourages you to avoid using this provider in production environments for both administration and device user authentication.

### *Description*

This provider provides authentication services. You need to configure only the authentication properties for the No Authentication Challenge provider.

---

### **Note:**

- If you use Allow Anonymous Access for a native OData application, do not also assign the No Authentication Challenge security profile to the application; anonymous OData requests are not sent, and Status code: 401 is reported.
  - For Agency applications, a common scenario is to configure No Authentication Challenge as the security provider, so that only Agency performs authentication.
-

## Properties

Table 18. Authentication Properties

Property	Default Value	Description
Control Flag	Optional	<p>Indicates how the security provider is used in the login sequence.</p> <ul style="list-style-type: none"> <li>• Optional – the authentication provider is not required, and authentication proceeds down the authentication provider list, regardless of success or failure.</li> <li>• Sufficient – the authentication provider is not required, and subsequent behavior depends on whether authentication succeeds or fails.</li> <li>• Required – the authentication provider is required, and authentication proceeds down the authentication provider list.</li> <li>• Requisite – the authentication provider is required, and subsequent behavior depends on whether authentication succeeds or fails.</li> </ul>
Description	None	<p>(Optional) A meaningful string that describes the providers usage.</p> <p>A description makes it easier to differentiate between multiple instances of the same provider type; for example, when you have multiple authentication providers of the same type stacked in a security profile, and each targets a different repository.</p>

**See also**

- *Creating and Configuring Security Profiles* on page 165
- *Mapping a Logical Role to a Physical Role* on page 174
- *No Authentication Challenge Provider* on page 178

**System Login (Admin Only) Provider**

This provider is intended only to bootstrap the Admin security profile after a new installation. It is not appropriate for production use.

**See also**

- *System Login (Admin Only) Configuration Properties* on page 180

*System Login (Admin Only) Configuration Properties*

The System Login (Admin Only) provider is configured by the installer with the initial administrator credentials only to give platform administrator access to Management Cockpit so that SAP Mobile Platform Server can be configured for production use. Administrators are expected to replace this provider immediately upon logging in for the first time. SAP encourages you to avoid using this provider in production environments.

*Description*

This provider is recommended only to give the platform administrator access to Management Cockpit so it can be configured for production use. Administrators are expected to replace this provider immediately upon logging in for the first time.

The System Login (Admin Only) provider:

- Provides role-based authorization by configuring the provider `com.sap.security.core.RoleCheckAuthorizer` with this authentication provider.
- Authenticates the user by comparing the specified user name and password against the configured user. Upon successful authentication, the configured roles are added as principals to the subject.

*Properties*

**Table 19. System Login (Admin Only) Properties**

Property	Default Value	Description
Control Flag	Optional	<p>Indicates how the security provider is used in the login sequence.</p> <ul style="list-style-type: none"> <li>• Optional – the authentication provider is not required, and authentication proceeds down the authentication provider list, regardless of success or failure.</li> <li>• Sufficient – the authentication provider is not required, and subsequent behavior depends on whether authentication succeeds or fails.</li> <li>• Required – the authentication provider is required, and authentication proceeds down the authentication provider list.</li> <li>• Requisite – the authentication provider is required, and subsequent behavior depends on whether authentication succeeds or fails.</li> </ul>

Property	Default Value	Description
Description	None	<p>(Optional) A meaningful string that describes the providers usage.</p> <p>A description makes it easier to differentiate between multiple instances of the same provider type; for example, when you have multiple authentication providers of the same type stacked in a security profile, and each targets a different repository.</p>
Username	None	<p>A valid user name used to authenticate. Do not use any of these restricted special characters: , = : ' " * ? &amp;.</p>
Password	None	<p>The password for the configured user.</p>
Roles	None	<p>Comma-separated list of roles that are granted to the authenticated user for role-based authorization. SAP Mobile Platform logical roles include Administrator, and Impersonator</p> <ul style="list-style-type: none"> <li>Administrator – role required for using Management Cockpit with administrator privileges. If you assign Administrator to this property, the login ID in the created provider has administrator privileges.</li> <li>Notification – notification users interact with SAP Mobile Platform to enable and configure push notifications.</li> <li>Impersonator – used when configuring a reverse proxy, in the case of client (or mutual) certificate authentication. The reverse proxy needs to be granted the Impersonator role to be able to impersonate the end user (for example, to propagate the end-user certificate via SSL_CLIENT_HEADER).</li> </ul> <p>If multiple roles are defined for this property, the role with more privileges (Administrator) is used to authorize users.</p> <hr/> <p><b>Note:</b> If you use other values, ensure you map SAP Mobile Platform roles to the one you define here.</p>

### See also

- *Default Logical Roles in SAP Mobile Platform* on page 172
- *Creating and Configuring Security Profiles* on page 165

- *Mapping a Logical Role to a Physical Role* on page 174
- *System Login (Admin Only) Provider* on page 179
- *Basic Authentication* on page 162

### Populate JAAS Subject From Client Provider

This provider is used to copy HTTP header and cookie values from the client request into the authenticated JAAS subject as user Principals (which identifies who the client user is), roles (which determines the permissions the user may have), and credentials (which provide single sign-on material for back-end systems).

This provider will always fail authentication because it does not validate user credentials at all. It simply copies request values to where they can be used downstream in the authentication process. If you add this provider to a security profile, be sure to set its controlFlag to "optional" so the overall authentication can succeed based on the other providers in the profile.

### **See also**

- *Populate JAAS Subject From Client Configuration Properties* on page 182

### *Populate JAAS Subject From Client Configuration Properties*

The Populate JAAS Subject From Client provider enables administrators to add client values as named credentials, name principals, and role principals to the authenticated subject.

### *Description*

This provider adds the configured values from the shared-context client HTTP map as the specified NamedCredentials to the authenticated subject. Adding client values as named credentials allows them to be used for single sign-on. When authenticating the user using a token from the client session, if the corresponding authentication provider is unable to retrieve the user name from the token and add it as a principal for use in impersonation checking, the administrator can configure this provider to add the appropriate header value from the client session as a principal to the authenticated subject.

---

**Note:** Rogue applications could intentionally insert HTTP headers with arbitrary values to obtain principals, roles, or credentials that they otherwise would not receive using the other login modules. Use this provider in an environment where you know the network edge behavior and have ensured that applications cannot bypass or override that environment.

---

This provider does not authenticate the subject but adds the NamedCredential if the user is successfully authenticated by other providers. It always returns “false” from the login method and should always be configured with the control flag set to “optional” to avoid affecting the outcome of authentication process.



## Properties

Table 20. Populate JAAS Subject From Client Properties

Configuration Option	Default Value	Description
Control Flag	Optional	<p>Indicates how the security provider is used in the login sequence.</p> <ul style="list-style-type: none"> <li>• Optional – the authentication provider is not required, and authentication proceeds down the authentication provider list, regardless of success or failure.</li> <li>• Sufficient – the authentication provider is not required, and subsequent behavior depends on whether authentication succeeds or fails.</li> <li>• Required – the authentication provider is required, and authentication proceeds down the authentication provider list.</li> <li>• Requisite – the authentication provider is required, and subsequent behavior depends on whether authentication succeeds or fails.</li> </ul>
Description	None	<p>(Optional) A meaningful string that describes the providers usage.</p> <p>A description makes it easier to differentiate between multiple instances of the same provider type; for example, when you have multiple authentication providers of the same type stacked in a security profile, and each targets a different repository.</p>
Client HTTP Values As Named Credentials	None	<p>Comma-separated list of mappings that specify the names of attributes (headers and cookies) from the client HTTP communication channel that should be added as credentials after successful authentication and the corresponding names to be associated with the credentials. For example:</p> <pre>httpHeaderName:credentialName1 httpCookieName:credentialName2</pre>
Client HTTP Values As Name Principals	None	<p>Comma-separated list of values of attributes (headers and cookies) from the client HTTP communication channel that should be added as name principals after successful authentication. For example:</p> <pre>clientPropertyName2, clientPropertyName10</pre>

Configuration Option	Default Value	Description
Client HTTP Values As Role Principals	None	Comma-separated list of values of attributes (headers and cookies) from the client HTTP communication channel that should be added as role principals after successful authentication. For example: <code>clientPropertyName2, clientPropertyName10</code>

**See also**

- *Creating and Configuring Security Profiles* on page 165
- *Mapping a Logical Role to a Physical Role* on page 174
- *Populate JAAS Subject From Client Provider* on page 182
- *Propagate Single Sign-on Using Populate JAAS Subject From Client* on page 244

**X.509 User Certificate Provider**

Use X.509 User Certificate when the client has authenticated using HTTPS and X.509 certificates for mutual authentication.

The client has already authenticated at the HTTPS protocol layer before this provider is called. This module then validates that the user's certificate is valid:

- signed by a trusted certificate authority
- not expired,
- not revoked via certificate revocation lists or OCSP

If the certificate validates, then authentication is successful. The client request must have been received at SAP Mobile Platform via HTTPS with the mutual authentication listener in order to succeed. This provider may create a Subject Principal where the principal name is the fully qualified SubjectDN in the user's certificate. That subject principal name may then be used in conjunction with the UserRoleAuthorizer to grant roles to this user.

**See also**

- *X.509 User Certificate Configuration Properties* on page 184
- *Enabling OCSP* on page 250
- *Mapping a Logical Role to a Physical Role* on page 174

**X.509 User Certificate Configuration Properties**

The X.509 User Certificate provider enables mutual authentication. This provider should be used when certificates are authenticated by the container.

*Description*

X.509 User Certificate can be used with other providers that support certificate authentication (for example, Directory Service (LDAP/AD)) by configuring X.509 User Certificate before

the providers that support certificate authentication. Use this provider to validate client certificates only when HTTPS listeners are configured to use mutual authentication. Add and configure provider properties for X.509 User Certificate, or accept the default settings.

### Properties

**Table 21. X.509 User Certificate Properties**

Property	Default Value	Description
Control Flag	Optional	<p>Indicates how the security provider is used in the login sequence.</p> <ul style="list-style-type: none"> <li>• Optional – the authentication provider is not required, and authentication proceeds down the authentication provider list, regardless of success or failure.</li> <li>• Sufficient – the authentication provider is not required, and subsequent behavior depends on whether authentication succeeds or fails.</li> <li>• Required – the authentication provider is required, and authentication proceeds down the authentication provider list.</li> <li>• Requisite – the authentication provider is required, and subsequent behavior depends on whether authentication succeeds or fails.</li> </ul>
Description	None	<p>(Optional) A meaningful string that describes the providers usage.</p> <p>A description makes it easier to differentiate between multiple instances of the same provider type; for example, when you have multiple authentication providers of the same type stacked in a security profile, and each targets a different repository.</p>
Validated Certificate Is Identity	False	<p>(Optional) Whether the certificate should set the authenticated subject as the user ID. If the X.509 User Certificate is used with other providers that establish user identity based on the validated certificate, set this value to false.</p>

Property	Default Value	Description
Validate Cert Path	True (see description)	If true, performs certificate chain validation, starting with the certificate being validated. Verifies that the issuer of that certificate is valid, and that the certificate has been issued by a trusted certificate authority (CA). If it is not, this property instructs the provider to look up the issuer of that certificate in turn, and verify it is valid and is issued by a trusted CA (building up the path to a CA that is in the trusted certificate store). If the trusted store does not contain any of the issuers in the certificate chain, path validation fails.
Enable Revocation Checking	False	(Optional) Enables online certificate status protocol (OCSP) certificate checking for user authentication. If you enable this option, you must also enable OCSP in SAP Mobile Platform Server. This provider uses the OCSP configuration properties that are defined in <code>SMP_HOME\sapjvm_7\jre\lib\security\java.security</code> . Revoked certificates result in authentication failure when: <ul style="list-style-type: none"> <li>• Revocation checking is enabled, and</li> <li>• OCSP properties are configured correctly.</li> </ul>
key:value Pair	None	Attributes identified using an arbitrary name, where the key is the name, and the value is the content.  Custom properties can be used to specify CRL URLs. The custom properties have names such as <code>cr1.1</code> , <code>cr1.2</code> , <code>cr1.3</code> , and so on. The values of each property have a URL which returns expected content to <code>java.security.cert.CertificateFactory.generateCRLs</code> .  <b>Note:</b> SAP Mobile Platform also supports <code>ldap://URL</code> .  For more information, see <a href="http://docs.oracle.com/javase/7/docs/api/java/security/cert/CertificateFactory.html">http://docs.oracle.com/javase/7/docs/api/java/security/cert/CertificateFactory.html</a> .

**See also**

- *Creating and Configuring Security Profiles* on page 165
- *Mapping a Logical Role to a Physical Role* on page 174
- *Enabling OCSP* on page 250
- *X.509 User Certificate Provider* on page 184
- *X.509 Certificate Authentication* on page 163

- *Single Sign-on for SAP* on page 242
- *Single Sign-on Authentication* on page 242
- *Preparing Your SAP Environment for Single Sign-on* on page 243
- *Configuring SAP Mobile Platform Server Certificate-based Authentication with a Reverse Proxy* on page 240

### *HTTP/HTTPS Authentication Provider*

Use the HTTP/HTTPS Authentication provider to authenticate the user with a call to a separate Web server to validate the user's credentials. It is useful to validate user name and password basic credentials, or to validate a single sign-on token from the client (in lieu of passwords).

The HTTP/HTTPS Authentication authentication provider validates standard user name and password style credentials by passing them to a Web server. Configure the URL property to point to a Web server that challenges for basic authentication.

This provider is enhanced to authenticate the user by validating a token specified by the client by sending the configured client values to the HTTP back end in the specified format (header/cookie). Any parameter value, for example HTTP header, or HTTP cookie can be specified in the ClientHttpValuesToSend property so that the provider can retrieve the value of the configured parameter(s) and pass them to the Web server in the format required by the SendClientHttpValuesAs configuration property.

For example, to extract the cookie "MyCookie" from the client session to SAP Mobile Platform Server and pass it along to the Web server as the cookie "testSSOCookie", set the properties ClientHttpValuesToSend to "MyCookie" and set SendClientHttpValuesAs to cookie:testSSOCookie.

---

### **Note:**

---

Best practice guidelines include:

- Using an HTTPS URL to avoid exposing credentials.
- If the Web server's certificate is not signed by a well known CA, import the CA certificate used to sign the Web server's certificate into the SAP Mobile Platform Server keystore. The keystore is pre-populated with CA certificates from reputable CAs.
- If this Web server returns a cookie as part of successful authentication, set the SSO Cookie Name configuration property to the name of this cookie. Upon successful authentication, SSO cookie value is made available as single sign-on material for back-end connections later.

### **See also**

- *HTTP/HTTPS Authentication Configuration Properties* on page 187

### *HTTP/HTTPS Authentication Configuration Properties*

The HTTP/HTTPS Authentication provider authenticates the user with given credentials (user name and password) against the secured back end that requires basic authentication. To

facilitate single sign-on (SSO), you can configure it to retrieve a cookie with the configured name and add it to the JAAS subject.

### *Description*

Configure this provider to authenticate the user by:

- Using only the specified user name and password.
- Using only the specified client value or values.
- Attempting token authentication. If that fails, revert to basic authentication using the supplied user name and password. You may find this helpful when using the same security configuration for authenticating users with a token, such as device users hitting the network edge, and when Push Notification requests from within a firewall present a user name and password but no token.

---

**Note:** The HTTP/HTTPS Authentication provider allows token validation by connecting to an HTTP server capable of validating the token specified in the HTTP header and cookie set in the session.

---

**Note:** This provider can either be used for SSO tokens or HTTP basic without SSO. The sole condition being that the back-end support HTTP basic authentication.

---

**Note:** Note that if "ClientHttpValuesToSend" property is configured, the provider only attempts to authenticate the user using those values. It does not set the user name and password credentials in the http session to the Web server. If the specified client values are not found in the client session to SAP Mobile Platform or if the Web server fails to validate the specified token, then this provider fails the authentication unless the property "TryBasicAuthIfTokenAuthFails" is set to `true` to enable it to revert to passing the user name and password credentials to respond to the basic authentication

---

## Properties

Table 22. HTTP/HTTPS Authentication Properties

Configuration Option	Default Value	Description
Control Flag	Optional	<p>Indicates how the security provider is used in the login sequence.</p> <ul style="list-style-type: none"> <li>• Optional – the authentication provider is not required, and authentication proceeds down the authentication provider list, regardless of success or failure.</li> <li>• Sufficient – the authentication provider is not required, and subsequent behavior depends on whether authentication succeeds or fails.</li> <li>• Required – the authentication provider is required, and authentication proceeds down the authentication provider list.</li> <li>• Requisite – the authentication provider is required, and subsequent behavior depends on whether authentication succeeds or fails.</li> </ul>
Description	None	<p>(Optional) A meaningful string that describes the providers usage.</p> <p>A description makes it easier to differentiate between multiple instances of the same provider type; for example, when you have multiple authentication providers of the same type stacked in a security profile, and each targets a different repository.</p>
URL	None	<p>Required. The HTTP or HTTPS URL that authenticates the user. For SSO, this is the server URL from which SAP Mobile Platform Server acquires the SSO cookie/token.</p>
Disable Server Certificate Validation	False	<p>(Optional) If true, this property disables certificate validation when establishing an HTTPS connection to the secured Web server (SWS) using the configured URL. Set to true only for configuration debugging.</p>

Configuration Option	Default Value	Description
HTTP Connection Timeout Interval	60000	The value, in seconds, after which an HTTP connection request to the Web-based authentication service times out. If the HTTP connection made in this provider (for either user authentication or configuration validation) does not have a timeout set, and attempts to connect to a Web-based authentication service that is unresponsive, the connection also becomes unresponsive, which might in turn cause SAP Mobile Platform Server to become unresponsive. Set the timeout interval to ensure that authentication failure is reported without waiting indefinitely for the server to respond.
Send Password As Cookie	None	(Deprecated) Use only for backward compatibility. New configurations should use <code>SendClientHttpValuesAs</code> and <code>ClientHttpValuesToSend</code> to configure token authentication.  Sends the password to the URL as a cookie with this name. If not specified, the password is not sent in a cookie. This property is normally used when there is a cookie-based SSO mechanism in use (for example, SiteMinder), and the client has included an SSO token into the password.
Client HTTP Values To Send	None	A comma-separated list of client HTTP values to be sent to the HTTP server. For example:  <code>ClientHttpValuesToSend=client_personalization_key, client_cookie_name</code>  Set this property if you are using token authentication.  Setting this property triggers token authentication. Only token authentication is attempted, unless <code>TryBasicAuthIfTokenAuthFails</code> is configured to true in conjunction with <code>ClientHttpValuesToSend</code> .  This property does not apply if the user is to be authenticated using only the supplied user name and password.
Send Client HTTP Values As	None	Comma-separated list of strings that indicate how to send <code>ClientHttpValuesToSend</code> to the HTTP server. For example:  <code>SendClientHttpValuesAs=header:header_name, cookie:cookie_name</code>  This property does not apply if the user is to be authenticated using only the supplied user name and password.



Configuration Option	Default Value	Description
Try Basic Auth if Token Auth Fails	False	<p>Whether the provider should attempt basic authentication using the specified user name and password credentials if token authentication is configured and fails. This property is applicable only if token authentication is enabled.</p> <p>This property does not apply if the user is to be authenticated using only the supplied user name and password.</p>
Successful Connection Status Code	200	HTTP status code that is interpreted as successful when a connection is established to the secured Web server.
SSO Cookie Name	None	<p>(Optional) Name of the cookie that is set in the session between the authentication provider and the secured Web server, which holds the SSO token for single sign-on. The provider looks for this cookie in the connection to the Secured Web Server. If the cookie is found, it is added to the authenticated subject as a named credential.</p> <p>The authentication provider ignores the status code when an SSO cookie is found in the session; authentication succeeds regardless of the return status code.</p>
Credential Name	None	Name to set in the authentication credential that contains the token returned in SSOCookieName. If this property is not configured, the SSOCookieName property value is set as the name of the token credential
Username HTTP Header	None	<p>HTTP response header name returned by the HTTP server with the user name retrieved from the token. Upon successful authentication, the retrieved user name is added as a SecName-Principal.</p> <p>This property does not apply if the user is to be authenticated using only the supplied user name and password.</p>

Configuration Option	Default Value	Description
Regex For User Name Match	None	<p>Regular expression used to match the supplied user name with the user name that is returned by the HTTP server in UsernameHttpHeader. The string "{username}" in the regex is replaced with the specified user name before it is used . If specified, it compares the user name retrieved from the Username Http Header with the user name specified in the callback handler. If the user names do not match, authentication fails. If the user names match, both the specified user name and the retrieved user name are added as SecNamePrincipals to the authenticated subject.</p> <p>This property does not apply if the user is to be authenticated using only the supplied user name and password.</p>
Roles HTTP header	None	<p>(Optional) Name of an HTTP header that the server may return. The header value contains a comma-separated list of roles to be granted.</p>
Token Expiration Time HTTP Header	None	<p>HTTP response header name that is returned by the HTTP server with the validity period of the token, in milliseconds, since the start of January 1, 1970. If the header is returned in the HTTP response from the secured Web server, the token is cached for the duration it remains valid unless TokenExpirationInterval is also configured. If this response header is not returned with the token, it might result in unintended use of the token attached to the authenticated context even after it has expired.</p> <p>This property does not apply if the user is to be authenticated using only the supplied user name and password.</p>
Token Expiration Interval	0	<p>Interval, in milliseconds, to be deducted from the actual expiration time returned in TokenExpirationTimeHttpHeader. This ensures that the token credential that is retrieved from the authenticated session remains valid until it is passed on to the secure Web server for SSO. This property does not apply if the user is to be authenticated using only the supplied user name and password. If the configured TokenExpirationInterval value exceeds the amount of time the token is valid, authentication by HTTP/HTTPS Authentication fails even if the token is validated successfully by the secured Web server.</p>

**See also**

- *Creating and Configuring Security Profiles* on page 165

- *Mapping a Logical Role to a Physical Role* on page 174
- *HTTP/HTTPS Authentication Provider* on page 187
- *Basic Authentication* on page 162

### Directory Service (LDAP/AD) Provider

Add an LDAP provider to a security profile to authenticate administrator logins (on the Admin security profile) or device user logins (any custom security profile for that purpose).

The Directory Service (LDAP/AD) module provides authentication services. You can also implement some LDAP providers with providers of other types. If you are stacking multiple LDAP providers, be aware of and understand the configuration implications.

### **See also**

- *Directory Service (LDAP/AD) Configuration Properties* on page 197

### *Configuration Best Practices for Multiple LDAP Trees*

Use the SAP Mobile Platform administration perspective to configure LDAP authentication providers, which are used to locate LDAP user information when organizational user groups exist within multiple LDAP trees.

To accommodate an LDAP tree structure that cannot be directly accessed using one search base:

- Create an LDAP authentication module for each level in the hierarchy – during the authentication process, SAP Mobile Platform tries to authenticate against every authentication provider in the ordered list until authentication succeeds or until it reaches the end of the list. Depending on the number of authentication providers you configure, this approach may have some performance issues.
- Use different AuthenticationScopes for performing user searches – specify the root node of a particular LDAP tree, by entering `AuthenticationSearchBase="dc=sap, dc=com"` and set `Scope=subtree`. SAP Mobile Platform performs an LDAP query against the entire subtree for authentication information. Depending on the number of AuthenticationScope within the LDAP tree structure, this approach can have performance implications.
- If multiple servers are clustered together to form a large logical directory tree, configure the Directory Service (LDAP/AD) by setting the `Referral` property to `follow`.
- If a user has been made a member of too many LDAP groups and appears in too many rows, performance may be impacted. If the security profile does not require any role mapping, the role lookup becomes unnecessary and can be avoided. Set the **SkipRoleLookup** property to `true` to eliminate the need to search all the roles defined in the role search base. This mainly applies to security profiles for applications, but not the Admin security profile.

### *LDAP Role Computation*

Role checks are the primary means of performing access control when using LDAP authentication. Authentication utilizes role computation techniques to enumerate roles that authenticated users have.

There are three distinct types of role constructs supported by LDAP providers; each may be used independently, or all three may be configured to be used at the same time.

- User-level role attributes, specified by the `UserRoleMembershipAttributes` configuration property, are the most efficient role definition format. A user's roles are enumerated by a read-only directory server-managed attribute on the user's LDAP record. The advantage to this technique is the efficiency with which role memberships can be queried, and the ease of management using the native LDAP server's management tools. These constructs are supported directly by ActiveDirectory, and use these configuration options:
  - `UserRoleMembershipAttributes` – the multivalued attribute on the user's LDAP record that lists the role DNs that the user is a member of. An example value for this property is "memberOf" on ActiveDirectory.
  - `RoleSearchBase` – the search base under which all user roles are found, for example, "ou=Roles,dc=sap,dc=com". This value may also be the root search base of the directory server.
  - `RoleFilter` – the search filter that, coupled with the search base, retrieves all roles on the server.
  - (Optional) `RoleScope` – enables role retrieval from subcontexts under the search base.
  - (Optional) `RoleNameAttribute` – choose an attribute other than "cn" to define the name of roles.

These properties are set to default values based on the configured server type. However, these properties can be explicitly set to desired values if the server type is not configured or set to overwrite the default values defined for a server type.

- LDAP servers allow groups to be members of other groups, including nested groups. The LDAP provider does not compute the group membership information recursively. Instead, nested group membership information is taken into consideration for role computation only if the LDAP server provides a user attribute that contains the complete list of group memberships, including static, dynamic, and nested group memberships.
- Freeform role definitions are unique in that the role itself does not have an actual entry in the LDAP database. A freeform role starts with the definition of one or more user-level attributes. When roles are calculated for a user, the collective values of the attributes (each of which may be multivalued) are added as roles to which the user belongs. This technique may be useful when the administration of managing roles becomes complex. For example, assign a freeform role definition that is equivalent to the department number of the user. A role check performed on a specific department number is satisfied only by users who have the appropriate department number attribute value. The only property that

is required or used for this role mapping technique is the comma-delimited `UserFreeformRoleMembershipAttributes` property.

---

**Note:** The physical roles looked up from the LDAP repository may need to be mapped to the logical roles defined in SAP Mobile Platform Server and the applications (if the name of the role defined in the LDAP repository is different from the role name defined in SAP Mobile Platform). Role mapping needs to be done manually for SAP Mobile Platform.

---

### See also

- *Mapping a Logical Role to a Physical Role* on page 174

### Gathering Provider Group Information

Production environments rely on a production-grade security provider (commonly an LDAP directory) to authenticate administrators. To map the SAP Mobile Platform default logical roles to the corresponding physical roles in the security provider, you must understand how the provider organizes users.

### Prerequisites

SAP Mobile Platform cannot query all supported enterprise security servers directly; for successful authentication, you must know the physical roles that your back-end systems require.

### Task

You can map a logical role to one or more physical roles. You can also map multiple logical roles to the same physical role. If a role does not exist, you can also add or delete names as needed.

Consider which users need to be in the Administrator, Developer, Helpdesk, Impersonator, and Notification User roles, then identify or create groups in your provider that correspond to these roles.

#### 1. Evaluate existing groups.

If there are existing groups that seem to already contain the right subjects that correspond to Administrator, Developer, Helpdesk, Impersonator, and Notification User, you can use those groups. The names need not be exact, as you can map them manually to address any differences.

#### 2. If required, have your LDAP administrator:

- Create new LDAP groups that will correspond to and be mapped to SAP Mobile Platform logical roles.
- Use LDAP tools to add users to those LDAP groups so that when the LDAP users authenticate to SAP Mobile Platform (with the Directory Service (LDAP/AD) provider configured) the corresponding physical roles are attributed to them and the logical role mapping can grant appropriate access.

3. Add subjects to these groups to assign SAP Mobile Platform corresponding permissions.
4. Determine what values are needed for the authentication provider properties in SAP Mobile Platform.  
For example, for an LDAP authentication provider you need values for the providerURL, serverType, bind user, bind password, search base and so on.

### *Nested Groups and Roles in LDAP*

The LDAP provider computes the roles granted to an authenticated user using the role and group membership information from the LDAP repository. To support nested roles and groups, LDAP servers allow roles and groups to be members of other roles and groups respectively.

The LDAP provider retrieves role membership from the user attribute specified by `UserRoleMembershipAttributes` configuration property, and does not compute the role membership information recursively. Therefore any nested and dynamic roles are taken into consideration only if the LDAP server provides a user attribute that contains the complete list of role memberships, including static, dynamic, and nested role memberships. For example, in SunOne server, the `UserRoleMembershipAttributes` property for the LDAP provider should be set to "nsRole" instead of the default value "nsRoleDN" to enable it to retrieve the nested roles information.

Similarly LDAP group memberships are stored and checked on a group-by-group basis. Each defined group, typically of objectclass `groupofnames` or `groupofuniqueNames`, has an attribute listing all of the members of the group. The LDAP provider does not support nested or dynamic groups (groups that are populated with objects found by doing an LDAP search rather than static members). For example, it does not recursively compute all the groups to which the user has membership. Therefore any nested and dynamic groups are taken into consideration only if the LDAP server provides a user attribute that contains the complete list of group memberships, including static, dynamic, and nested group memberships. For example, in Active Directory server, the `UserRoleMembershipAttributes` property for the LDAP provider should be set to "tokenGroups" to enable it to retrieve the nested group membership information.

### *Skipping LDAP Role Lookups (SkipRoleLookup)*

When configuring an LDAP provider, use the `SkipRoleLookup` configuration option to grant the user all the roles retrieved using the `UserRoleMembershipAttributes` property from the LDAP user entry without looking up all the roles defined in the role search base.

Setting **SkipRoleLookup** to true grants all the roles retrieved using the **UserRoleMembershipAttributes** property from the LDAP user entry. The user roles are not cross-referenced with the roles retrieved from the role search base using the role search filter.

This eliminates the need to look up all the roles defined in the role search base and match the role filter as roles are retrieved. If the list of roles granted to the authenticated user is to be restricted to the roles defined in the role search base, set **SkipRoleLookup** to false.

### *Configuring an LDAP Provider to Use SSL*

If your LDAP server uses a secure connection, and its SSL certificate is signed by a nonstandard certificate authority, for example it is self-signed, import the certificate into the key store, then use Management Cockpit to configure an LDAP security profile.

1. Use the keytool utility to import the certificate into the keystore.
2. Restart SAP Mobile Platform services.
3. Log in to Management Cockpit for SAP Mobile Platform.
4. Navigate to **Settings**, then select **Security Profiles**. Select the desired security configuration in which to add the LDAP provider.
5. Add a Directory Service (LDAP/AD) authentication provider to an existing security profile or create a new profile if needed.
6. In **Authentication Provider Settings**, configure the ProviderURL, Security Protocol, ServerType, Bind DN, Bind Password, Search Base, and other properties as determined by you and your LDAP administrator.
7. Choose **one** of the two methods below to secure a connection to the LDAP server:
  - a) Use `ldaps://` instead of `ldap://` in the **ProviderURL**.
  - b) Use `ssl` in the **Security Protocol**.
8. Click **Save**.

### *Directory Service (LDAP/AD) Configuration Properties*

Use these properties to configure the LDAP provider used to authenticate Management Cockpit administration logins or to configure the LDAP provider used to authenticate device application logins.

#### *Description*

If you are configuring an LDAP provider for device application logins in the Management Cockpit, then SAP Mobile Platform administrators use Management Cockpit. These properties are saved to the `SMP_HOME\Server\configuration\com.sap.mobile.platform.server.security\CSI` directory.

The Java LDAP provider consists of three provider modules. The Directory Service (LDAP/AD) provides authentication services. Through appropriate configuration, you can enable certificate authentication in Directory Service (LDAP/AD).

---

**Note:** Role-mapping is a required manual step to log in to Management Cockpit with an LDAP user. In role mapping, add the Directory Service (LDAP/AD) authentication provider to the Admin security profile.

---

Use this table to help you configure properties for one or more of the supported LDAP providers. When configuring providers or general server properties in Management Cockpit, note that properties and values can vary, depending on which provider or server type you configure.

---

**Note:** The following characters have special meaning when they appear in a name in LDAP: , (comma), = (equals), + (plus), < (less than), > (greater than), # (number or hash sign), ; (semicolon), \ (backslash), / (forward slash), LF (line feed), CR (carriage return), " (double quotation mark), ' (single quotation mark), \* (asterisk), ? (question mark), & (ampersand), and a space at the beginning or end of a string. LDAP providers do not handle these special characters in any of the names or DN's, in any of the configuration properties. Additionally, some of the properties, as identified below, cannot use these special characters in common names.

---

*Properties*

**Table 23. Directory Service (LDAP/AD) Properties**

Property	Default Value	Description
Control Flag	Optional	<p>Indicates how the security provider is used in the login sequence.</p> <ul style="list-style-type: none"> <li>• Optional – the authentication provider is not required, and authentication proceeds down the authentication provider list, regardless of success or failure.</li> <li>• Sufficient – the authentication provider is not required, and subsequent behavior depends on whether authentication succeeds or fails.</li> <li>• Required – the authentication provider is required, and authentication proceeds down the authentication provider list.</li> <li>• Requisite – the authentication provider is required, and subsequent behavior depends on whether authentication succeeds or fails.</li> </ul>
Description	None	<p>(Optional) A meaningful string that describes the providers usage.</p> <p>A description makes it easier to differentiate between multiple instances of the same provider type; for example, when you have multiple authentication providers of the same type stacked in a security profile, and each targets a different repository.</p>



Property	Default Value	Description
Server Type	None	<p>Optional. Type of LDAP server to which you are connecting:</p> <ul style="list-style-type: none"> <li>• sunone5 -- SunOne 5.x OR iPlanet 5.x</li> <li>• msad2k -- Microsoft Active Directory, Windows 2000</li> <li>• nsds4 -- Netscape Directory Server 4.x</li> <li>• openldap -- OpenLDAP Directory Server 2.x</li> </ul> <p>The value you choose establishes default values for these other authentication properties:</p> <ul style="list-style-type: none"> <li>• Role Filter</li> <li>• User Role Membership Attributes</li> <li>• Role Member Attributes</li> <li>• Authentication Filter</li> <li>• Digest MD5 Authentication</li> <li>• Use User Account Control</li> </ul>
Provider URL	ldap://localhost:389	<p>The URL used to connect to the LDAP server. Without this URL configured, SAP Mobile Platform Server cannot contact your server. Use the default value if the server is:</p> <ul style="list-style-type: none"> <li>• Located on the same machine as your product that is enabled with the common security infrastructure.</li> <li>• Configured to use the default port (389).</li> </ul> <p>Otherwise, use this syntax for setting the value:</p> <pre>ldap://&lt;hostname&gt;:&lt;port&gt;</pre>
Initial Context Factory	com.sun.jndi.ldap.LdapCtxFactory	<p>The LDAP provider relies on an available JNDI LDAP provider, and this argument determines which JNDI provider will be used.</p>
Security Protocol	None	<p>The protocol to be used when connecting to the LDAP server. The specified value overrides the environment property <code>java.naming.security.protocol</code>.</p> <p>To use an encrypted protocol, use SSL instead of ldaps in the URL.</p>

Property	Default Value	Description
Bind DN	None	<p>The user DN to bind against when building the initial LDAP connection.</p> <p>In many cases, this user may need read permissions on all user records. If you do not set a value, anonymous binding is used. Anonymous binding works on most servers without additional configuration.</p>
Bind Password	None	<p>The password for Bind DN, which is used to authenticate any user. Bind DN and Bind Password separate the LDAP connection into units.</p> <p>The Authentication Method property determines the bind method used for this initial connection.</p>
Enable LDAP Connection Trace	Disabled	Enables LDAP connection tracing. The output is logged to a file in the <code>temp</code> directory. The location of the file is logged to the server log.
Referral	Ignore	The behavior when a referral is encountered. Valid values are dictated by <code>LdapContext</code> , for example, <code>follow</code> , <code>ignore</code> , <code>throw</code> .
Authentication Method	Simple	<p>The authentication method to use for all authentication requests into LDAP. Legal values are generally the same as those of the <code>java.naming.security.authentication</code> JNDI property. Choose one of:</p> <ul style="list-style-type: none"> <li>• <code>simple</code> – for clear-text password authentication.</li> <li>• <code>DIGEST-MD5</code> – for more secure hashed password authentication. This method requires that the server use plain text password storage and only works with JRE 1.4 or later.</li> </ul>
Digest MD5 Authentication Format	DN For OpenLDAP: User name	The <code>DIGEST-MD5</code> bind authentication identity format.

Property	Default Value	Description
Default Search Base	None	<p>The LDAP search base that is used if no other search base is specified for authentication, roles, attribution and self registration:</p> <ol style="list-style-type: none"> <li>1. <code>dc=&lt;domainname&gt;,dc=&lt;tld&gt;</code> For example, a machine in <code>sap.com</code> domain would have a search base of <code>dc=sap,dc=com</code>.</li> <li>2. <code>o=&lt;company name&gt;,c=&lt;country code&gt;</code> For example, this might be <code>o=SAP,c=us</code> for a machine within the SAP organization.</li> </ol> <hr/> <p><b>Note:</b> When you configure this property in the Admin security profile used to authenticate the administrator in Management Cockpit, the property value should not contain any special characters, as listed above, in any of the common names or distinguished names.</p>
Authentication Filter	<p>For most LDAP servers: (<code>&amp;</code>; (uid={uid}) (objectclass=person))</p> <p>or</p> <p>For Active Directory e-mail lookups: (<code>&amp;</code>; (userPrincipalName={uid}) (objectclass=user)) [ActiveDirectory]</p> <p>For Active Directory Windows user name lookups: (<code>&amp;</code>; (sAMAccountName={uid}) (objectclass=user))</p>	<p>The filter to use when looking up the user.</p> <p>When performing a user name based lookup, this filter is used to determine the LDAP entry that matches the supplied user name.</p> <p>The string "{uid}" in the filter is replaced with the supplied user name.</p> <hr/> <p><b>Note:</b> When you use this property to authenticate a user in Management Cockpit:</p> <ul style="list-style-type: none"> <li>• The property value should not contain any special characters, as listed above, in any of the common names or distinguished names.</li> <li>• Do not use Chinese or Japanese characters in user names or passwords of this property.</li> </ul>

Property	Default Value	Description
Authentication Scope	onelevel	<p>Determines whether the search for a user should be limited to the search base or the subtree rooted at the search base. The supported values for this are:</p> <ul style="list-style-type: none"> <li>• onelevel</li> <li>• subtree</li> </ul> <p>If you do not specify a value or if you specify an invalid value, the default value is used.</p>
Authentication Search Base	None	<p>The search base used to authenticate users. If this property is not configured, the value for Default Search Base is used.</p> <hr/> <p><b>Note:</b> When you configure this property in the Admin security profile used to authenticate the administrator in Management Cockpit, the property value should not contain any special characters, as listed above, in any of the common names or distinguished names.</p> <hr/>
Use User Account Control Attribute	For Active Directory: true	<p>When this property is set to true, the User Account Control attribute is used to detect if a user account is disabled, if the account has expired, if the password associated with the account has expired, and so on. Active Directory uses this attribute to store this information.</p>
Skip Role Lookup	False	<p>Set this property to true to grant the roles looked up using the attributes specified by the property User Role Membership Attributes without cross-referencing them with the roles looked up using the Role Search Base and Role Filter.</p> <p>LDAP configuration validation succeeds even when an error is encountered when listing all the available roles. The error is logged to the server log during validation but not reported in Management Cockpit, allowing the configuration to be saved. This has an impact when listing the physical roles for role mapping as well as in Management Cockpit. To successfully authenticate the user, set the Skip Role Lookup property to true.</p> <hr/> <p><b>Note:</b> Currently, only manual configuration validation is supported.</p> <hr/>

Property	Default Value	Description
Role Search Base	None	<p>The search base used to retrieve lists of roles. If this property is not configured, the value for Default Search Base is used.</p> <p>Setting the Role Search Base to the root in Active Directory (for example "DC=example,DC=com") may result in a PartialResultsException error when validating the configuration or authenticating a user. If users encounter the PartialResultsException, they should confirm they can reach example.com:389. The DNS lookup may successfully resolve example.com to an IP address, but port 389 may not be open with an Active Directory server listening on that port. In this case, add an entry to the hosts file (for example, systemroot\system32\drivers\etc\hosts or /etc/hosts) on the machine where SAP Mobile Platform is installed to resolve any communication error.</p> <hr/> <p><b>Note:</b></p> <ul style="list-style-type: none"> <li>• When you configure this property in the Admin security profile used to authenticate the administrator in Management Cockpit, the property value should not contain any special characters, as listed above, in any of the common names or distinguished names.</li> <li>• Currently, only manual configuration validation is supported.</li> </ul>
Role Scope	onelevel	<p>Determines whether the search for the roles should be limited to the search base or the subtree rooted at the search base. Supported values include:</p> <ul style="list-style-type: none"> <li>• onellevel</li> <li>• subtree</li> </ul> <p>If you do not specify a value or if you specify an invalid value, the default value is used.</p>

Property	Default Value	Description
<p>Role Filter</p>	<p>For SunONE/iPlanet: (&amp; (object-class=ldapsubentry) (object-class=nsroledefinition))</p> <p>For Netscape Directory Server: (  (object-class=groupofnames) (object-class=groupofuniquenames))</p> <p>For ActiveDirectory: (  (object-class=groupofnames) (object-class=group))</p>	<p>The role search filter. This filter should, when combined with the role search base and role scope, return a complete list of roles within the LDAP server. There are several default values, depending on the chosen server type. If the server type is not chosen and this property is not initialized, no roles are available.</p> <hr/> <p><b>Note:</b> When you use this property to authenticate a user in Management Cockpit:</p> <ul style="list-style-type: none"> <li>• The property value should not contain any special characters, as listed above, in any of the common names or distinguished names.</li> <li>• Do not use Chinese or Japanese characters in user names or passwords of this property.</li> </ul>
<p>Role Member Attributes</p>	<p>For Netscape Directory Server and OpenLDAP Server: member,unique-member</p>	<p>A comma-separated list of role attributes from which LDAP derives the DN's of users who have this role.</p> <p>These values are cross-referenced with the active user to determine the user's role list. One example of the use of this property is when using LDAP groups as placeholders for roles. This property has a default value only when the Netscape server type is chosen.</p>
<p>Role Name Attribute</p>	<p>cn</p>	<p>The attribute of the role entry used as the role name in SAP Mobile Platform. This is the role name displayed in the role list or granted to the authenticated user.</p>

Property	Default Value	Description
User Role Membership Attributes	For iPlanet/SunONE: nsRoleDN  For Active Directory: memberOf  For all others: none	<p>Defines a user attribute that contains the DNs of all of the roles a user is a member of.</p> <p>These comma-delimited values are cross-referenced with the roles retrieved in the role search base and search filter to generate a list of user's roles.</p> <p>If Skip Role Search property is set to true, these comma-delimited values are not cross-referenced with the roles retrieved in the role search base and role search filter.</p> <hr/> <p><b>Note:</b> If you use nested groups with Active Directory, you must set this property to tokenGroups.</p>
User Freeform Role Membership Attributes	None	The freeform role membership attribute list. Users who have attributes in this comma-delimited list are automatically granted access to roles whose names are equal to the attribute value. For example, if the value of this property is department and user's LDAP record has the following values for the department attribute, { sales, consulting }, then the user will be granted roles whose names are sales and consulting.
LDAP Pool Max Active	8	<p>Caps the number of concurrent LDAP connections to the LDAP server. A non-positive value indicates no limit. If this option is set for multiple LDAP providers, the value set by the first LDAP provider loaded takes precedence over all the others. When LDAP Pool Max Active is reached, any further attempts by the LDAP provider classes to borrow LDAP connections from the pool are blocked indefinitely until a new or idle object becomes available in the pool.</p> <p>Connection pooling improves the LDAP provider's performance and resource utilization by managing the number of TCP connections established with configured LDAP servers. A separate pool is associated with different SAP Mobile Platform security profiles, ensuring that the LDAP connections in the connection pool for a particular security profile are isolated from any changes occurring outside this security configuration. A separate pool also ties the connection pool life cycle to that of the security profile.</p>

Property	Default Value	Description
Connect Timeout	0	Specifies the timeout, in milliseconds, when connecting to the LDAP server. The property value sets the JNDI <code>com.sun.jndi.ldap.connect.timeout</code> property, when attempting to establish a connection to a configured LDAP server. If the LDAP provider cannot establish a connection within the configured interval, it aborts the connection attempt. An integer less than or equal to zero results in the use of the network protocol's timeout value.
Read Timeout	0	Controls the length of time, in milliseconds, the client waits for the server to respond to a read attempt after the initial connection to the server has been established. The property value sets the JNDI <code>com.sun.jndi.ldap.read.timeout</code> property, when attempting to establish a connection to a configured LDAP server. If the LDAP provider does not receive an LDAP response within the configured interval, it aborts the read attempt. The read timeout applies to the LDAP response from the server after the initial connection is established with the server. An integer less than or equal to zero indicates no read timeout is specified.
Enable Certificate Authentication	Disabled	Whether to enable certificate authentication when this provider is configured with X.509 User Certificate.
Certificate Authentication Filter	None	The filter to use when authenticating the user with a certificate. The filter determines the LDAP entry that matches the supplied certificate encoded form.
Certificate Attributes	None	Comma-separated list of attributes in the certificate to be used for authenticating the user, instead of the certificate binary.
LDAP Attributes	None	Comma-separated list of attributes that map to the certificate attributes, to be used to select the LDAP entry that matches the values in the certificate.
Unmapped Attribute Prefix	LDAP	Prefix assigned to unmapped LDAP attributes when moving them into the CSI namespace. A period is added to the prefix, followed by the LDAP attribute name. For example, <code>employeeNumber</code> is converted to <code>LDAP.employeeNumber</code> .



Property	Default Value	Description
Serialization Key	None	<p>Specifies a unique configuration serialization key. Within a CSI configuration file, each LDAP configuration block must have a unique value. The default value is computed automatically based upon the LDAP URL. This is sufficient for most situations.</p> <p>However, if multiple LDAP login providers are configured against the same LDAP URL, then this property must be set to a unique value for each to identify which configurations are active when serializing sessions. By default, the value of Provider URL configuration option is used.</p>
key:value Pair	None	<p>Attributes identified using an arbitrary name, where the key is the name, and the value is the content. Because SAP Mobile Platform does not make use of user attributes retrieved from LDAP, there is no need to set any custom properties.</p>

### See also

- *Creating and Configuring Security Profiles* on page 165
- *Mapping a Logical Role to a Physical Role* on page 174
- *Debugging Authentication Errors with CSI Tool* on page 258
- *Directory Service (LDAP/AD) Provider* on page 193
- *Basic Authentication* on page 162

### UserRoleAuthorizer Provider

The UserRoleAuthorizer provider grants logical roles to specific users when the user's roles cannot be retrieved by the configured authentication provider from the back end. You cannot manually configure this provider.

This provider is part of all security configurations that are created or updated in Management Cockpit. UserRoleAuthorizer simply implements the checkRole method to compare the physical role name passed in to the current user name.

This authorizer allows the role check for the role "user:"+userName to succeed. For example, with this authorization module enabled, an administrator can map Notification User to "user:jsmith". The user who authenticates as jsmith is then added in the physical role user:jsmith and is granted the logical Notification User and can perform Notification Push.

---

**Note:** When the user is authenticated using the X.509 User Certificate provider, this authorizer allows the role check for the role "user:"+<subjectDN from the certificate used to authenticate the user> to succeed.

---

UserRoleAuthorizer features enable you to map the DN from a client certificate to a role.

**See also**

- *Mapping a Logical Role to a Physical Role* on page 174

controlFlag Attribute Values

The SAP implementation uses the same control flag (controlFlag) attribute values and definitions as those defined in the JAAS specification.

If you stack multiple providers, you must set the control flag attribute for each enabled provider.

Control Flag Value	Description
Required	The authentication provider is required. Authentication proceeds down the authentication provider list.
Requisite	The authentication provider is required. Subsequent behavior depends on the authentication result: <ul style="list-style-type: none"> <li>• If authentication succeeds, authentication continues down the authentication provider list.</li> <li>• If authentication fails, control returns immediately to the application (authentication does not proceed down the authentication provider list).</li> </ul>
Sufficient	The authentication provider is not required. Subsequent behavior depends on the authentication result: <ul style="list-style-type: none"> <li>• If authentication succeeds, control returns immediately to the application (authentication does not proceed down the authentication provider list).</li> <li>• If authentication fails, authentication continues down the authentication provider list.</li> </ul>
Optional (default)	The authentication provider is not required to successfully authenticate the user. Regardless of success or failure, authentication proceeds down the authentication provider list.

**Example**

Providers are listed in this order and with these controlFlag settings:

1. X.509 User Certificate (sufficient)
2. Directory Service (LDAP/AD) (optional)
3. HTTP/HTTPS Authentication (sufficient)

A client performing certificate authentication (for example, X.509 SSO to SAP) can authenticate immediately. Subsequent providers are not called, because they are not required. Regular user name and password credentials, if they exist, go to LDAP, which may

authenticate them, and set them up with roles from the LDAP groups they belong to. Then NativeOS is invoked, and if that succeeds, SAP Mobile Platform picks up roles based on the Windows groups they belong to.

### See also

- *Creating and Configuring Security Profiles* on page 165
- *Stacking Providers and Combining Authentication Results* on page 169

## Adding Reverse Proxies

Once the installation of SAP Mobile Platform Server is complete, install and configure a reverse proxy if you need it.

For third-party reverse proxy solutions, SAP supports:

- Apache reverse proxy for Native and Hybrid applications
- nginx for Agentry applications

When adding a reverse proxy, determine the application types you need to support.

Application Type	Reverse Proxy
Native	Apache
Hybrid	Apache
Agentry	Nginx

### Common Requirements of Reverse Proxies

Reverse proxies that are used with SAP Mobile Platform must comply with specific requirements for content encoding, HTTP headers, timeouts, and the URL that is passed to SAP Mobile Platform.

A reverse proxy that is used with SAP Mobile Platform must be a straight passthrough proxy server. Ensure that any reverse proxy used:

- Does not change the content encoding of requests or responses. Chunked transfer encoding is the required data transfer mechanism. Content-length encoding is not supported.
- Does not remove any HTTP headers.
- Sets a timeout period, if used, that is greater than the timeout used by the clients.
- Passes the resulting URL to SAP Mobile Platform Server in the form `http://Host_Name:Port_No.`

### **Using Apache Reverse Proxy for HTTP Clients**

For organizations that have HTTP clients that are designed to consume SAP Mobile Platform Server services, you may implement an Apache reverse proxy in your production environment.

For an example of how to implement an Apache Reverse Proxy, see:

1. *Installing and Configuring Apache Reverse Proxy*

Edit the `httpd.conf` file to load the modules that are required to prepare the Reverse Proxy for SAP Mobile Platform use.

2. *Decrypting Certificates for HTTPS Connections*

The Apache 2.2 Windows version does not support encrypted certificate key files. If the key file is encrypted, you must first decrypt it.

### **Installing and Configuring Apache Reverse Proxy**

Edit the `httpd.conf` file to load the modules that are required to prepare the Reverse Proxy for SAP Mobile Platform use.

In SAP Mobile Platform, communication can be either:

- Unencrypted to port 8080
- One-way authenticated and encrypted to port 8081
- Two-way authenticated to port 8082

---

**Note:** If clients are authenticating to the reverse proxy with X.509 certificates, only the port 8082 path can be used to propagate the client certificates on the `SSL_CLIENT_CERT` header in a trusted fashion.

---

Apache is installed as one or more modules, in addition to `mod_proxy`:

`mod_proxy_http`, `mod_proxy_ftp`, `mod_proxy_ajp`, `mod_proxy_balancer`, and `mod_proxy_connect`. Thus, to use one or more of the particular proxy functions, load `mod_proxy` and the appropriate module into the server. For the reverse proxy, load these modules:

- `headers_module`
- `ssl_module`
- `proxy_module`
- `proxy_connect_module`
- `proxy_http_module`

For information about running a reverse proxy in Apache, see <http://www.apachetutor.org/admin/reverseproxies>. For information about SSL and proxy modules, see [http://httpd.apache.org/docs/2.2/mod/mod\\_ssl.html](http://httpd.apache.org/docs/2.2/mod/mod_ssl.html) and [http://httpd.apache.org/docs/2.2/mod/mod\\_proxy.html](http://httpd.apache.org/docs/2.2/mod/mod_proxy.html).

1. Download Apache 2.2 from a reliable source, and install the proxy according to package instructions.
2. In a text editor, open `Apache2.2\conf\httpd.conf`.
3. Uncomment these lines to load headers, and required SSL and proxy modules:

```
LoadModule headers_module modules/mod_headers.so
LoadModule ssl_module modules/mod_ssl.so
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
LoadModule proxy_http_module modules/mod_proxy_http.so
```

The three `proxy_*` modules are required by three proxy modes: HTTP, one-way HTTPS, and two-way HTTPS. The `ssl_module` is required by both HTTPS proxy modes. The `headers_module` is required by two-way HTTPS proxy mode.

4. Add these lines to enable port 8080 as an HTTP proxy:

```
#####
Listen 8080
    <VirtualHost *:8080>
        ServerName proxy-server
            ErrorLog "C:/Apache2.2/logs/error.log"
            TransferLog "C:/Apache2.2/logs/access.log"
        <Location />
            ProxyPass http://sup-server:8080/
            ProxyPassReverse http://sup-server:8080/
        </Location>
    </VirtualHost>
#####
```

5. Add these lines to enable port 8081 as a one-way HTTPS proxy:

```
#####
Listen 8081
    <VirtualHost *:8081>
        ServerName proxy-server
            ErrorLog "C:/Apache2.2/logs/error.log"
            TransferLog "C:/Apache2.2/logs/access.log"
            # activate HTTPS on the reverse proxy
            SSLEngine on
            SSLCertificateFile "C:/Apache2.2/conf/proxy-
server.crt"
            SSLCertificateKeyFile "C:/Apache2.2/conf/proxy-
server.key"
            SSLCertificateChainFile "C:/Apache2.2/conf/proxy-
server-ca.crt"
            SSLProxyEngine On
            SSLProxyCACertificateFile C:/Apache2.2/conf/sup-
server-ca.crt
        <Location />
            ProxyPass https://sup-server:8081/
            ProxyPassReverse https://sup-server:8081/
        </Location>
    </VirtualHost>
```

6. Add these lines to enable port 8082 as a two-way HTTPS proxy:

```
#####
Listen 8082
<VirtualHost *:8082>
    ServerName proxy-server
    ErrorLog "C:/Apache2.2/logs/error.log"
    TransferLog "C:/Apache2.2/logs/access.log"
    # activate HTTPS on the reverse proxy
    SSLEngine on
    SSLCertificateFile "C:/Apache2.2/conf/proxy-
server.crt"
    SSLCertificateKeyFile "C:/Apache2.2/conf/proxy-
server.key"
    SSLCertificateChainFile "C:/Apache2.2/conf/proxy-
server-ca.crt"
    # activate the client certificate
    authentication
    SSLCACertificateFile "C:/Apache2.2/conf/trusted-
client-ca.crt"
    SSLVerifyClient require
    SSLVerifyDepth 10
    SSLProxyEngine On
    SSLProxyCACertificateFile C:/Apache2.2/conf/sup-
server-ca.crt
    SSLProxyMachineCertificateFile C:/Apache2.2/conf/
proxy-client.pem
    # initialize the special headers to a blank value to
avoid http header forgeries
    RequestHeader set SSL_CLIENT_CERT ""
    <Location />
    # add SSL_CLIENT_CERT header to forward real client
certificate
    RequestHeader set SSL_CLIENT_CERT "%
{SSL_CLIENT_CERT}s"
    ProxyPass https://sup-server:8082/
    ProxyPassReverse https://sup-server:8082/
    </Location>
</VirtualHost>
#####
```

7. Save the file.

8. Validate the configuration by opening a browser and testing these URLs:

- <https://proxy-server:8080/debug/app1>
- <https://proxy-server:8081/debug/app1>
- <https://proxy-server:8082/debug/app1>

### Decrypting Certificates for HTTPS Connections

The Apache 2.2 Windows version does not support encrypted certificate key files. If the key file is encrypted, you must first decrypt it.

1. To decrypt an encrypted file, from a command prompt, run:

```
openssl rsa -in encrypted.key -out decrypted.key
```

2. Use the decrypted key in the `httpd.conf` file.

## **Using Nginx Reverse Proxy for Agentry Clients**

For organizations that have Agentry clients that are designed to consume SAP Mobile Platform Server services, you may implement an Nginx reverse proxy in your production environment.

For an example of how to implement an Nginx reverse proxy, see:

### **1. *Installing Nginx Web Server***

Download and install the Nginx Web server.

### **2. *Configuring Nginx as a Reverse Proxy***

Edit the `nginx.conf` file to configure Nginx as a reverse proxy for SAP Mobile Platform and enable SSL.

## **See also**

- *Agentry Security* on page 255

### **Installing Nginx Web Server**

Download and install the Nginx Web server.

The steps below are for installing Nginx on Windows. If you are installing Nginx on Linux, refer to documentation on the Nginx Web site: <http://nginx.org/en/docs/>.

1. Download the `.zip` file for the mainline Windows version of Nginx from the Nginx Web site (<http://nginx.org>) to a temporary location.
2. Install Nginx by extracting the contents of the downloaded `.zip` file to a directory on the host system.

The top-level directory in the `.zip` file identifies the Nginx version. For example, if you have downloaded version 1.5.9 and you extract the contents of the `.zip` file into `C:\Program Files`, the Nginx home directory under `C:\Program Files` is named `nginx-1.5.9`.

The rest of these instructions refer to this Nginx home directory as `<Nginx_Home>`.

3. In a text editor, open `<Nginx_Home>\conf\nginx.conf`.
4. Make the changes indicated below and save the file.

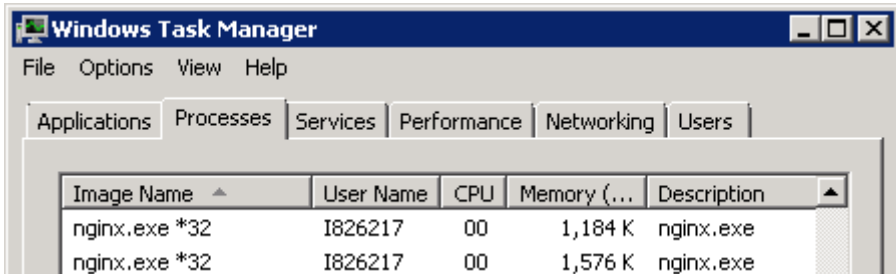
```
...
server {
    listen      <Nginx_server_name>:8000;
    server_name <Nginx_server_name>;
...

```

5. In a command prompt window, navigate to the `<Nginx_Home>` directory and run `nginx.exe`.

You should see no error messages displayed in the command prompt window.

- Close the command prompt window, open Task Manager, and look for two `nginx.exe` processes:



- As a final test, enter the URL for the Nginx Web server in a browser window. The URL is `http://<server_name>:8000`. The Nginx Web server should display this Welcome page:



Configuring Nginx as a Reverse Proxy

Edit the `nginx.conf` file to configure Nginx as a reverse proxy for SAP Mobile Platform and enable SSL.

The `nginx.conf` that you just modified to test that the Nginx Web server could be started should still be open in a text editor.

To use Nginx reverse proxy with SSL, make the changes indicated below in the `server{ }` section of `<Nginx_Home>\conf\nginx.conf`.

```
...
server {
    listen      <Nginx_server_name>:8000;
    server_name <Nginx_server_name>;

    ## edit the rest of the server{} section to look like this ##
    ssl_certificate C:/nginx-1.4.4/cert/NginxServer.crt;
    ssl_certificate_key C:/nginx-1.4.4/cert/NginxServer.key;

    ssl_ciphers HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;
}
```



```

ssl_session_timeout 5m;

access_log C:/nginx-1.4.4/logs/access_8001.log;
error_log C:/nginx-1.4.4/logs/error_8001.log;

root html;
index index.html index.htm;

location / {
    access_log off;
    proxy_pass https://<SMP_server_name>:8081/<application_name>;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
}
...

```

For information on setting up certificates, see:

- For Nginx (NginxServer.crt in the nginx.conf file listing above) – the Nginx Web site: <http://nginx.org/en/docs/> and [http://nginx.org/en/docs/http/configuring\\_https\\_servers.html](http://nginx.org/en/docs/http/configuring_https_servers.html).
- For SAP Mobile Platform Server – *Managing Keystore and Truststore Certificates in Administrator*.

## Server Security

---

SAP Mobile Platform Server provides data services to device clients by interacting with databases. The database is configured along with server components, to the internal corporate LAN.

Secure the server runtime by performing tasks that secure the infrastructure and administration of components, as well as enabling user authentication and secure communication.

### Securing the Server Infrastructure

Before you can secure platform administration, you must first secure the underlying infrastructure, preventing files from being tampered with on the host and allowing the SAP Mobile Platform Server to run with existing security mechanisms.

#### See also

- *Platform Security Quick Start* on page 159

### Managing Firewalls and Intrusion Prevention

A personal firewall, or intrusion detection and prevention software (IPS or IDPS), can cause SAP Mobile Platform components to malfunction or not function at all. SAP Mobile Platform

uses regular IP communication between components on the primary network interface of a computer, even when all components are installed on the same host.

If the local network interface is secured by intrusion detection and prevention software (for example, McAfee Host Intrusion Prevention software or equivalent), you must configure the security software to allow all network communication between SAP Mobile Platform components.

Try one of these options to work around the limitations imposed by the host intrusion prevention software and policy settings, without violating any security policy, until the settings of your security software are adjusted to the needs of SAP Mobile Platform.

- Remove the host machine from the network – this option ensures that all interconnections between SAP Mobile Platform components are treated as local traffic and should not be flagged as incoming connections from external sources, thereby causing connection failures due to security policy setting. This option is suitable when you use your laptop in a network other than your corporate network, and want to demonstrate a mobile solution using a simulator or emulator with all components running on the same machine. To use this option:
  1. Stop SAP Mobile Platform services in the correct order.
  2. Disconnect the host from all networks.
  3. Restart SAP Mobile Platform services in the correct order.
  4. Change the Management Cockpit URL link to use "localhost" or *<yourhostname>* as the host name, instead of the original fully qualified host name of the machine that included the domain name. Accept any security warnings to connect to Management Cockpit.
- Connect the host to the corporate network – this option ensures that all interconnections among SAP Mobile Platform components are internal to your corporate network and validated against the corporate network security policy. The option of connecting to corporate network through VPN is especially suitable when you use your laptop in a network other than your corporate network, and want to demonstrate a mobile solution using your physical devices, and need outgoing connections to a back end.
  1. Stop the SAP Mobile Platform services.
  2. Reconnect the host to your corporate network directly or through corporate VPN, to ensure that the corporate network security policy applies.
  3. Restart SAP Mobile Platform services.
  4. Change the Management Cockpit URL link to use "localhost" or *<yourhostname>* as the host name, instead of the original fully qualified host name of the machine that included the domain name (for example: `https://localhost:8083/Admin`, or `https://yourhostname:8083/Admin`). Accept any security warnings to connect to Management Cockpit.
- To ensure required internal component communication ports are not blocked, configure the firewall software to allow connections to the ports SAP Mobile Platform uses.

**See also**

- *Port Number Reference* on page 145
- *HTTP/HTTPS Port Number Reference* on page 145
- *TCP Port Number Reference* on page 147

**Setting File System Permissions**

After installation, you can restrict access by removing most users and groups from the SAP Mobile Platform installation directory.

The SAP Mobile Platform Windows installer configures SAP Mobile Platform to run as a Local System account and no credentials are needed.

---

**Note:** For Linux systems, use the **chmod -R** command to set file permissions under `$SMP_HOME`.

---

1. Open Windows File Explorer.
2. Right-click `SMP_HOME`, and click **Properties**.
3. On the **Security** tab, click **Advanced**.
4. In the **Permissions** tab, select **Change Permissions** and configure permissions accordingly
5. Click **OK**.

**Securing Platform Administration**

Use the Web-based Management Cockpit to remotely and securely administer SAP Mobile Platform.

**Enabling Authentication for Administrator Logins**

Authentication for administrators is always performed by SAP Mobile Platform Server. Management Cockpit automatically delegates administrator authentication to the providers that are configured for the Admin security profile.

Initially, the Admin security profile has just one authentication provider, System Login (Admin Only), populated with a user name and password that was prompted during installation. This user is meant to be temporary to enable initial use of the Management Cockpit. To make the Admin security profile production-ready, you must initially log in using the administrator credentials defined with the installer, and replace the System Login (Admin Only) module with production-ready providers.

The System Login (Admin Only) provider does not enforce password strength or change policies that are typically in place for a production environment. Therefore, substitute the System Login (Admin Only) module with a authentication provider that is suitable for a production environment. Subsequent logins are then performed with user credentials assigned to the Administrator role.

The Admin security profile authenticates and authorizes administrative users. In your production system, remove this user as soon as the Admin security profile is configured to use your identity management system.

SAP recommends that you restrict the use of the Admin security profile to administration authentication only.

### 1. *Logging in to Management Cockpit with an Installer-Defined Password*

The platform administrator logs in to Management Cockpit for the first time after installation.

### 2. *Making Admin Security Profile Production-Ready*

Replace the default security provider, System Login (Admin Only), with new production-ready providers.

### 3. *Resetting the smpAdmin Password*

You can manually reset the current platform administration password.

### *Logging in to Management Cockpit with an Installer-Defined Password*

The platform administrator logs in to Management Cockpit for the first time after installation.

During installation, the person installing SAP Mobile Platform defines a password for the administrator user. This password is used to configure the System Login (Admin Only) authentication provider that performs the administrator authentication.

---

**Note:** This installer-defined password is not intended to be a permanent administrator credential. Also, you must replace the System Login (Admin Only) authentication provider with a production-grade authentication module, typically LDAP.

---

1. Launch Management Cockpit.

2. Enter the administrator user name password selected at the time of installation. The user name is case-sensitive.

3. Click **Login**.

### *Making Admin Security Profile Production-Ready*

Replace the default security provider, System Login (Admin Only), with new production-ready providers.

### **See also**

- *Resetting the smpAdmin Password* on page 221

### *Adding a Production-Grade Provider*

Modify the Admin security profile to add a production-grade provider, typically Directory Service (LDAP/AD). Most companies use an LDAP directory to maintain internal user accounts. This module integrates with most LDAP servers, including Active Directory.

### **Prerequisites**

Determine what values are needed for the security provider properties in SAP Mobile Platform by gathering this information from the security provider you plan to use. For example, for a Directory Service (LDAP/AD) module, you need values for the providerURL, serverType, bind user, bind password, search base, and so on.

### **Task**

Configure the Admin security profile to authenticate only administrator user(s). SAP recommends that you create custom security profiles for SAP Mobile Platform application users. The steps here include examples for adding an LDAP provider.

1. In the Management Cockpit, select **Settings**.
2. In Security Profiles, click **admin (Cannot be deleted)** to select the installer-defined administrator user.
3. Under Authentication Providers, click **New**.
4. Select a provider from the list, for example Directory Service (LDAP/AD) then click **Create**.
5. Configure the values as determined by you and your provider administrator.
6. Click **Save**.
7. Click **OK**.
8. Click **Save**.

### **Next**

You must map the SAP Mobile Platform logical Administrator role to physical roles in your back end to enable your intended administrator users to be authorized to use the Management Cockpit. You must restart the server to recognize the role mapping changes before you test if your new administrator users are able to login.

### **See also**

- *Role Mapping* on page 171

### *Validating the Production Admin Security Profile*

Once you add a provider to the Admin security profile for SAP Mobile Platform Server, for example LDAP, you can test the login before removing the System Login (Admin Only) authentication provider from both components' configurations.

### **Prerequisites**

To ease the troubleshooting process in the event validation problems arise, temporarily increase your log levels for the security component to capture more detailed events.

### **Task**

1. Log in to Management Cockpit using the login values of a user mapped to the Administrator role, for example an LDAP user that is in an LDAP group mapped to the Administrator logical role.
2. If the login succeeds:
  - a) Remove System Login (Admin Only) authentication provider from SAP Mobile Platform Server and Management Cockpit setup locations.
  - b) Reduce the logging levels for both SAP Mobile Platform Server and Management Cockpit to Info or Warn.
3. If the login fails, refer to the server log in `SMP_HOME\Server\log\hostname-smp-server.log`.

### *Changing Security Log Levels*

You can increase log levels to capture more detailed events for troubleshooting. Then reduce log levels to a value more appropriate for normal security operations

### **Increasing Security Log Levels**

1. Increase security log levels to troubleshoot problems.
  - a) In Management Cockpit, select **Logs** and click **Server Log Settings**.
  - b) Increase the security log level in the **Security Log Configuration** list.
  - c) Click **Save**.
2. Following troubleshooting, reduce security log levels for normal security operations.
  - a) In Management Cockpit, select **Logs** and click **Server Log Settings**.
  - b) Reduce the security log level in the **Security Log Configuration** list.
  - c) Click **Save**.

### **See also**

- *Security Monitoring and Validation* on page 258

Resetting the smpAdmin Password

You can manually reset the current platform administration password.

---

**Note:** You must contact your IT department or administrator before changing or resetting the password. Only the person who has the right permission to change these files should perform this procedure.

---

1. Open the SAP Mobile Platform Server `admin.xml` file, located in `SMP_HOME\Server\configuration\com.sap.mobile.platform.server.security\CSI`, and modify this line:
 

```
<options encrypted="false" name="password" value="{TXT:}s3pAdmin" />
```

In this example, password encryption is set to false and the new password is set to `s3pAdmin`. You can replace this password with any password you choose.

---

**Note:** Do not remove the `{TXT:}` prefix.

---

2. Save the file.
3. Restart SAP Mobile Platform Server and Management Cockpit.
4. Log in to Management Cockpit using `s3pAdmin` as the new password.
 

When login succeeds, Management Cockpit opens the management view on the local SAP Mobile Platform Server.
5. In Management Cockpit, click **Settings**:
  - a) In Security Profiles, click **admin**.
  - b) Select System Login (Admin Only).
  - c) Enter the new password.
  - d) Click **Save**.
  - e) When you see the Authentication Provider Changes message, click **OK**.
  - f) Click **Save**.

The file is overwritten using the correct syntax.

6. Log in again to Management Cockpit using the new password.
7. Go to `SMP_HOME\Server\configuration\com.sap.mobile.platform.server.security\CSI`, and verify that the `admin.xml` encryption is configured correctly, and that the password is no longer recorded in clear text.

```
<authenticationProvider controlFlag="optional" name="com.sap.security.core.PreConfiguredUserLoginModule">
<options name="username" value="smpAdmin"/>
<options name="roles" value="Administrator,Notification User"/>
<options encrypted="true" name="password" value="1-AAAAEgQQWd8NguXX5nswpWF1vUFpTcJhjmoiSYUzEAAiY3vWkZ+Y/33cWAoUD+EV/D80Yo4vie/
```

```
XIyZVoBZbTT9ijxHDe7wbIBsagzS0DdAvS51TRvRRNVp83+pTjQ3mmMNt5FmxrGvU  
V5fVQ2JI1YaTPbd+Tw==" />
```

### See also

- *Making Admin Security Profile Production-Ready* on page 218

### **Setting Up a User Account for the SAP Mobile Platform Windows Service**

Use a user account to run the SAP Mobile Platform Windows service, rather than a local administrator account. This limits the amount of damage an attacker can cause to your entire system.

By default, the SAP Mobile Platform Windows service runs under a local administrator account. Local administrators have many Windows permissions that are not required for SAP Mobile Platform. For example, local administrators can change files under `Program Files` and `System32` directories, and can modify registry entries. If SAP Mobile Platform Server is compromised by an attacker, the attacker will be able to exploit these additional administrator permissions to damage your system. By running SAP Mobile Platform under a user account, which has fewer permissions, you limit the damage an attacker can cause.

The Windows administrator can change the SAP Mobile Platform Server service account to a user account after SAP Mobile Platform Server is installed.

---

**Note:** The Windows administrator must also change the SAP Mobile Platform service account to a user account after reconfiguring the Windows service as a result of server configuration changes.

---

1. Create a user named `smpServiceUser` using the command line or Windows Control Panel.

---

**Note:** You can also use an existing account for the SAP Mobile Platform service.

---

2. Give `smpServiceUser` a strong password and set the password to never expire.
3. Give `smpServiceUser` permission for full control of the server folder `SMP_HOME \Server` using the command line or Windows Explorer.

---

**Note:** Reduce the permissions for other users, based on their need to access SAP Mobile Platform Server files.

---

4. Click the Windows **Start** button, right-click the `<your computer name>` shortcut and select **Manage**.
5. Expand **Services and Applications** and select **Services**.
6. Right-click **SAP Mobile Platform Server** and select **Properties**.
7. Select the **Log On** tab.
8. Click **This Account**.
9. Enter `smpServiceUser` and the password.



10. Click **OK**.

### See also

- *Reconfiguring the Windows Service After Server Configuration Changes* on page 123

## Managing Keystore and Truststore Certificates

SAP Mobile Platform uses a shared keystore and truststore to store private and public keys.

### Keystore

The keystore contains private keys and their associated certificates for use by SAP Mobile Platform to identify itself to clients (using the server certificate) or to back-end systems (using the technical user certificates). The keystore also contains public certificates of trusted entities, typically the CA signing certificates of the back-end systems it will connect to, and the certificate used to sign client user certificates.

Administrators can make changes to the keystore using the **keytool** utility located in `SMP_HOME\sapjvm_7\bin`.

---

**Note:** While the keytool utility enables you to have different passwords for each private key entry, SAP Mobile Platform requires all the private key passwords to be the same as the keystore password or it will not work.

---

### Truststore

The truststore contains certificates from external parties, or from certificate authorities trusted to identify other parties.

### See also

- *Defining Back-end Connections for Native and Hybrid Apps* on page 38
- *HTTP/HTTPS Port Number Reference* on page 145
- *Keystore/Truststore Properties* on page 229
- *Keytool Utility* on page 224
- *Enabling a Direct HTTPS Connection to SAP Mobile Platform Server* on page 249
- *Using SSL Between a Client and SAP Mobile Platform Server* on page 247
- *Using SSL Between SAP Mobile Platform Server and the Back End* on page 248

### Certificate Alias

A certificate alias is the name given to a CA certificate located in the keystore.

Each entry in the keystore has an alias to help identify it. You may need to use the **-alias** command-line qualifier when using the **keytool** utility to work on a specific entry in the keystore.

The certificate alias identifies the alias of a specific certificate in the system keystore that must be used when making an HTTPS connection to the specified URL. This certificate alias is

required when the back end requires mutual SSL connectivity. Use the alias of a certificate stored in the SAP Mobile Platform Server keystore.

The "smp\_crt" alias identifies the SAP Mobile Platform Server certificate it uses with its HTTPS listeners. As per HTTP conventions, the CN of the generated X.509 certificate should be the fully qualified hostname.domain of the server where SAP Mobile Platform is installed.

Other certificates in the keystore can be user certificates that SAP Mobile Platform uses when other back-end systems require mutual certificate authentication.

### See also

- *Defining Back-end Connections for Native and Hybrid Apps* on page 38
- *Keytool Utility* on page 224
- *Keystore/Truststore Properties* on page 229
- *Using SSL Between a Client and SAP Mobile Platform Server* on page 247
- *Using SSL Between SAP Mobile Platform Server and the Back End* on page 248
- *Changing Installed Certificates Used for HTTPS Listeners* on page 227
- *Generating Certificates and Keys* on page 224

### Generating Certificates and Keys

Use a PKI system and a trusted CA to generate production-ready certificates and keys that encrypt communication among different SAP Mobile Platform components. You can then use the **keytool** utility to import and export certificate to the keystore.

---

**Note:** Any changes to the keystore require the server to be restarted.

---

### See also

- *Certificate Alias* on page 223
- *Keytool Utility* on page 224
- *Keystore/Truststore Properties* on page 229
- *Single Sign-on Integration Across Client Applications* on page 232

### Keytool Utility

**keytool** is a JDK utility that manages a keystore (database) of private keys and associated certificates, as well as certificates from trusted entities.

SAP Mobile Platform uses a single keystore file, located at `SMP_HOME\Server\configuration\smp_keystore.jks`. This is the file to configure and protect.

**keytool** is in `SMP_HOME\sapjvm_7\bin`. **keytool** lets users create and manage their own public and private key pairs and associated certificates for use in self-authentication, or data integrity and authentication services, using digital signatures. It also lets users cache the public keys (in the form of certificates) of their communicating peers.

---

**Note:** After importing a certificate into the keystore, you must restart SAP Mobile Platform Server before logging into the Management Cockpit.

---

## **Syntax**

```
keytool -list | -printcert | -import | -export | -delete | -selfcert |  
-certreq | -genkey [options]
```

## **Parameters**

- **-list** – displays the contents of a keystore or keystore entry.
- **-printcert** – displays the contents of a certificate stored in a file. Check this information before importing a certificate as a trusted certificate. Make sure the certificate prints as expected.
- **-import** – imports:
  - a certificate or certificate chain to the list of trusted certificates, or,
  - a certificate reply received from a certificate authority (CA) as the result of submitting a certificate signing request (CSR).

The value of the `-alias` option indicates the type of import you are performing. If the alias exists in the database, then it is assumed you want to import a certificate reply.

**keytool** checks whether the public key in the certificate reply matches the public key stored with the alias, and exits if they do not match. If the alias identifies the other type of keystore entry, the certificate is not imported. If the alias does not exist, it is created, and associated with the imported certificate.

- **-export** – exports a certificate to a file.
- **-delete** – deletes a certificate from the list of trusted certificates.
- **-selfcert** – generates a self-signed certificate. The generated certificate is stored as a single-element certificate chain in the keystore entry identified by the specified alias, where it replaces the existing certificate chain.
- **-certreq** – generates a certificate signing request (CSR), using the PKCS #10 format. A CSR is intended to be sent to a CA, which authenticates the certificate requestor and returns a certificate or certificate chain that replaces the existing certificate chain in the keystore.
- **-genkey** – generates a key pair (a public key and associated private key). Wraps the public key into an X.509 v1 self-signed certificate, which is stored as a single-element certificate chain. This certificate chain and the private key are stored in a new keystore entry identified by `<alias>`.

<b>-genkey Option</b>	<b>Description</b>
<code>-keystore &lt;key-storeLocation&gt;</code>	Name and location of the persistent keystore file for the keystore managed by <b>keytool</b> . If you specify a keystore that does not exist, a keystore is created. If you do not specify a <code>-keystore</code> option, the default keystore is a file named <code>.key-store</code> in your home directory. If that file does not exist, it is created.
<code>-storepass &lt;password&gt;</code>	The password that protects keystore integrity. The password must be at least 6 characters long and provided to all commands that access the keystore contents. If a <code>-storepass</code> option is not provided at the command line, the user is prompted for it.
<code>-file &lt;certificateFile&gt;</code>	The certificate file location.
<code>-noprompt</code>	During import, removes interaction with the user.
<code>-trustcacerts</code>	When importing a certificate reply, the certificate reply is validated using trusted certificates from the keystore and the certificates configured in the <code>cacerts</code> keystore file. <code>cacerts</code> resides in the JDK security properties directory, <code>java.home\lib\security</code> , where <code>java.home</code> is the runtime environment's directory. The <code>cacerts</code> file represents a system-wide keystore with CA certificates. System administrators can configure and manage that file using <b>keytool</b> , specifying "jks" as the keystore type.
<code>-alias &lt;alias&gt;</code>	The logical name for the certificate you are using.
<code>-keypass &lt;password&gt;</code>	The password that protects the private key of the key pair. Press Enter at the prompt to set the key password to the password associated with the keystore. <code>keypass</code> must be at least 6 characters long.

**Examples**

- **Example 1: Display the contents of the keystore** – `keytool -list -keystore <filePath>\configuration\smp_keystore.jks -storepass <storepass>`
- **Example 2: Import a certificate reply from a CA** – `keytool -import -file <certificate file> -keystore <filePath>\configuration\smp_keystore.jks -keypass <storepass> -storepass <storepass> -noprompt -trustcacerts -alias <alias>`

---

**Note:** Use `-keypass <storepass>` because alias passwords must match the storepass to work properly in SAP Mobile Platform.

---

- **Example 3: Delete a certificate** – `keytool -delete -alias <alias> -keystore <filePath>\configuration\smp_keystore.jks -storepass <storepass>`
- **Example 4: Generate a key pair** – `keytool -genkey -alias -keystore <filePath>\configuration\smp_keystore.jks`

The certificate request must be signed by a CA or self-signed by using the `-selfcert` **keytool** option.

---

**Note:** Use the `-alias` flag to give the keys an explicit name you can remember. Then, use a `-certreq` command (using the same `-alias`) to generate a certificate signing request based on that keypair. Once you get back the signed certificate, use the `-import` command with the same `-alias` to get a private certificate where you can use it. The `-alias` for the default server certificate used on HTTPS listeners is "smp\_cert".

---

- **Example 5: Use the `-sigalg SHA1withRSA` parameter** – The jdk1.7 keytool uses the SHA256withRSA algorithm by default; however, some certificate authorities (CAs) do not support this algorithm and use the `-sigalg SHA1withRSA` parameter where the CSR is generated.

```
keytool.exe -certreq -keyalg RSA -alias smp_cert -sigalg
SHA1withRSA-file request.csr -keystore smp_keystore.jks -
storepass keystorepassword
```

- **Example 6: Use `-import` to import a trusted CA certificate**

```
keytool -import -file <certificate file> -keystore <filePath>
\configuration\smp_keystore.jks -storepass <storepass> -noprompt
-trustcacerts
-alias <alias>
```

This is identical to Example 2 except since CA certs do not have private keys, the `-alias` does not need a password so you may remove the `-keypass <storepass>` argument.

### See also

- *Certificate Alias* on page 223
- *Using SSL Between a Client and SAP Mobile Platform Server* on page 247
- *Using SSL Between SAP Mobile Platform Server and the Back End* on page 248
- *Changing Installed Certificates Used for HTTPS Listeners* on page 227
- *Generating Certificates and Keys* on page 224
- *Keytool Utility* on page 224
- *Keystore/Truststore Properties* on page 229
- *Managing Keystore and Truststore Certificates* on page 223
- *Enabling a Direct HTTPS Connection to SAP Mobile Platform Server* on page 249

### Changing Installed Certificates Used for HTTPS Listeners

SAP Mobile Platform Server includes a default self-signed certificate. You must change this default certificate with a production-ready certificate after you install SAP Mobile Platform.

For shared certificates, SAP recommends that you maintain the existing certificate alias when importing the new certificates. Then, when you replace the default certificate with the new production certificate, you are updating change the certificate for all listeners simultaneously.

SAP Mobile Platform Server and Management Cockpit share the same keystore (that is, `SMP_HOME\Server\configuration\smp_keystore.jks`).

### 1. Generate new production-ready certificates:

Use your PKI system to generate SAP Mobile Platform Server certificates and key pairs, and have them signed with the Certificate Authority (CA) certificate used in your organization. Ensure that you:

- Use the existing alias (`smp_crt`).
- Set the CN of the certificate to `*.MyDomain`.

SAP Mobile Platform is compliant with certificates and key pairs generated from most well-known PKI systems.

### 2. Import the production-ready certificates and keys using the keytool.

See information on the keytool utility.

### See also

- *Certificate Alias* on page 223
- *Keytool Utility* on page 224
- *Keystore/Truststore Properties* on page 229

### Changing Keystore and Truststore Passwords

The SAP Mobile Platform (used by both SAP Mobile Platform Server and Management Cockpit to manage certificates and keys) keystore and truststore locations are protected by a password. In production environments, the initial keystore password is set during installation. The keystore password must be the same as all the private key passwords associated with the aliases in the store.

### Prerequisites

Before you begin, back up the contents of `SMP_HOME\Server\configuration\smp_keystore.jks`. This is the combined keystore and truststore for the server.

### Task

In production environments, use the keytool utility to change the passwords for the keystore and truststore.

1. Use **keytool -storepass** and **-keypass commands** repeatedly to change the password of the keystore itself, and each of the passwords for all private keys in the store. Passwords for both must be the same.

## 2. Configure the SAP Mobile Platform configuration to recognize the new password.

a) Encrypt the new password by obtaining the secret key from the `-DsecretKey` property in `SMP_HOME\Server\props.ini`.

b) Run the following the command:

```
java -jar tools\cipher\CLIEncrypter.jar <secretKey>
<newPassword>
```

where `<secretKey>` is the secret key obtained from `props.ini` and

`<newPassword>` is the new password for the keystore and truststore.

c) Open `SMP_HOME\Server\config_master\com.sap.mobile.platform.server.foundation.config.encrypted\com.sap.mobile.platform.server.foundation.config.encrypted.properties` and update `privateKeystorePass` to replace the existing password with the new encrypted password, keeping `{enc}` as the prefix.

d) Save the changes.

e) Restart restart the server for the changes to take effect.

### **Keystore/Truststore Properties**

The `SMP_HOME\Server\configuration\smp_keystore.jks` file contains properties for the combined SAP Mobile Platform truststore and keystore.

Change the default properties for:

Property	Default	Description
keyStore/trustStore	<code>SMP_HOME\Server\configuration\smp_keystore.jks</code>	The default location of the keystore/truststore used by Management Cockpit.
keyStorePassword/trustStorePassword	None. This password is set during SAP Mobile Platform installation.	The password that unlocks the keystore/truststore.

### **See also**

- *Certificate Alias* on page 223
- *Using SSL Between a Client and SAP Mobile Platform Server* on page 247
- *Using SSL Between SAP Mobile Platform Server and the Back End* on page 248
- *Changing Installed Certificates Used for HTTPS Listeners* on page 227
- *Generating Certificates and Keys* on page 224
- *Keytool Utility* on page 224
- *Managing Keystore and Truststore Certificates* on page 223
- *Enabling a Direct HTTPS Connection to SAP Mobile Platform Server* on page 249

## **Strong Encryption for JVM Security**

Enable strong encryption in your environment to ensure adequate security by modifying the default encryption on your Java virtual machine (JVM).

### **Enabling Strong Encryption in Java Development Kit**

By default, a Java Development Kit (JDK) installation supports only advanced encryption standard (AES) with 128-bit key length, which may be considered insecure.

The SAP Mobile Platform VM included in installation is approved for international distribution. However, you may choose to enable strong encryption for added security.

1. To enable strong cryptography on your Java Virtual Machine (JVM), download `UnlimitedJCEPolicyJDK7.zip` to replace the following existing files provided in SAP Mobile Platform at `SMP_HOME\sapjvm_7\jre\lib\security`
  - `local_policy.jar`
  - `US_export_policy.jar`
2. Go to <http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html>:
  - a) Accept the license agreement.
  - b) Download `UnlimitedJCEPolicyJDK7.zip`.
3. Stop SAP Mobile Platform Server.
4. Extract the zip file contents to overwrite the two existing `.jar` files.
5. Restart the server to use the strong security.

### **Encryption of Configuration Files**

SAP Mobile Platform Server configuration files in `SMP_HOME\Server\config_master\XXX` folder support encrypted configuration values.

Within each configuration directory under `config_master`, there is a file called `metaconfig.json`. Within this file, the "apply strategy" should be set to "config admin". The master key for encrypting the encrypted configuration values is stored in the `SMP_HOME\Server\props.ini` file:

- `-DsecretKey=<PASSWORD>`
- `-DsecretKeylength=<128|256>`

The 256-bit key length works only if you replaced the JVM's encryption policy files; the decryption of these values happens transparently to the application. This also means that inside the SAP Mobile Platform Server, encrypted values are in clear text. To indicate that a value is encrypted, it must be prefixed with `{enc}`. An entry must look like:

```
- <KEY>={enc}<ENCRYPTED-VALUE>
```



## CLIEncrypter

The SAP Mobile Platform Server includes an executable JAR in `SMP_HOME\Server\tools\cipher` folder to encrypt configuration values according to this specification.

Simply run:

```
\tools\cipher>Java -jar CLIEncrypter.jar <KEY> <TEXT> [<KEYLENGTH>]
```

The `<KEY>` must match the configured key from `SMP_HOME\Server\props.ini`, `<KEYLENGTH>` is optional and defaults to 128 bits - 256 will only work if you have updated your Java encryption policy file.

## Alternative Method Using OSGi

The AES/CBC/PKCS5Padding encryption is used. The encrypted value is expected to be Base64-encoded, and the first 16 bytes are interpreted as the initialization vector (IV). The encryption key is derived from the password using PBKDF2HmacWithSHA1 hashing with the static salt {97,101,105,111,117,85,79,73,69} and 65536 iterations.

---

**Note:** The OSGi commands are executed through the OSGi shell console which is a server-based service. This is an alternative method for encrypting configuration values.

---

SAP Mobile Platform Server includes a command that encrypts configuration values according to this specification:

```
- >osgi>cm_security enc <TEXT>
```

### Examples:

```
osgi>cm_security enc password Encoded
Text:QWIFsLnfJeE78tkRDx8ES3xDkNHbrbWdvNsEzfdv6IA=
osgi>cm_security dec
QWIFsLnfJeE78tkRDx8ES3xDkNHbrbWdvNsEzfdv6IA=Decoded Text:password
```

### See also

- *Changing Database Connection Passwords* on page 117

## Application Security

---

Application security ensures that device users can access application data by logging in through a configured security profile.

Applications govern the use of resources assigned to them, but not which resources are assigned. Using application security, applications then must determine the use of these resources. Application security measures prevent security policy exceptions in either the application itself or in the underlying system. Use the Management Cockpit to configure and manage applications and their security components.

## Authentication for User Logins

Enable authentication for device user logins by creating a new security profile (that is, one that is distinct from the Admin security profile), and mapping roles, then assigning it.

### Credential Types for Supported Authentication Providers

Different security providers allow users to supply different user credentials. If your security policy mandates that a specific credential type or strength be used, review the providers that are available to you.

**Table 24. Credentials Types and Providers**

Credential	Providers Available
User name and password	Directory Service (LDAP/AD), HTTP/HTTPS Authentication
E-mail address and password. E-mail addresses must follow certain requirements for SAP Mobile Platform to recognize them correctly, especially when they are used to define a security configuration.	Directory Service (LDAP/AD), HTTP/HTTPS Authentication
X.509 certificates	X.509 User Certificate
Tokens	HTTP/HTTPS Authentication
Basic authentication	No Authentication Challenge, System Login (Admin Only)

### Single Sign-on Integration Across Client Applications

Administrators can use their single sign-on system (SSO) of choice with SAP Mobile Platform to achieve end-to-end integration across client applications and back-end resources.

In addition to supporting X.509 certificate security, SAP Mobile Platform expands single sign-on support to third-party and standard single sign-on mechanisms. With expanded single sign-on support, SAP Mobile Platform enables the authentication framework to accept HTTP headers and cookies propagated by the client or a proxy server and then authenticate and propagate the user to the back end.

#### See also

- *SAP Mobile Platform Authentication Quick Start* on page 159
- *Creating and Configuring Security Profiles* on page 165
- *Mapping a Logical Role to a Physical Role* on page 174
- *Generating Certificates and Keys* on page 224

### *Integrating SAP Mobile Platform with Your Single Sign-on Solution*

SAP Mobile Platform integrates with SSO systems, including SiteMinder.

In SAP Mobile Platform, when the single sign-on service uses cookies (opaque to SAP Mobile Platform but understandable to other services that are integrated with the your SSO system), there are three authentication scenarios SAP Mobile Platform supports: network-edge authentication, token-based authentication, and basic authentication. In any of these three authentication scenarios, the net result is an authenticated user with SSO credentials that can be used for back-end system interactions.

In SAP Mobile Platform, the only SSO mechanism that can be used with back-end systems is the proprietary SAP MYSAPSSO2 cookie.

#### **See also**

- *Configuring SAP Mobile Platform SiteMinder Integration with the SAP SSO2 Mechanism* on page 237
- *Configuring Unprotected Network-Edge Authentication with a SiteMinder-Protected Back End* on page 238

#### *Network-Edge Authentication*

In network-edge authentication, the SSO system intercepts an unauthenticated client request headed to SAP Mobile Platform, challenges the client to authenticate, and adds an SSO cookie to the request before forwarding to SAP Mobile Platform.

SAP Mobile Platform supports network-edge authentication by allowing the administrator to configure which client values set in the connection to SAP Mobile Platform using network-edge authentication are to be used for authentication into SAP Mobile Platform server.

Client applications can connect to reverse proxy servers or agents at the network edge. These agents perform authentication, and return authenticated tokens delivered as HTTP cookies or HTTP headers. An example of an HTTP-based SSO provider is SiteMinder running inside the enterprise and its SiteMinder agent running at the network edge inside an Apache reverse proxy server.

SAP Mobile Platform uses the HTTP/HTTPS Authentication provider to reach out to a Web server integrated to the SSO system to validate the SSO cookie and derive information about the user identified by that cookie, how long the cookie is valid for, and any security roles the user has.

---

**Note:** When the network edge is forcing basic authentication, typically the authorization header the client uses to respond to the challenge is forwarded to SAP Mobile Platform. So even though SAP Mobile Platform may not be actively processing that header for login purposes, the user name and password of the user are leaked, which is not desirable from a security attack surface perspective, where all of the systems may be compromised if an attacker is able to exploit a vulnerability elsewhere.

---

Network-edge authentication is the most common SAP Mobile Platform SSO scenario.

---

**Note:** The Check Impersonation option in the security profile settings in Management Cockpit ensures that SAP Mobile Platform knows who the user is after successful SSO-based login. In network-edge authentication, the user identity (Principal) may be added as an additional header at the network edge.

---

### See also

- *Check Impersonation Attribute* on page 244

### *Token-based Authentication*

With token-based authentication in SSO, the customized client application obtains the SSO token from the SSO system using whatever means you designate.

The SSO token is injected into the cookie jar of the SAP Mobile Platform client application and is automatically forwarded to SAP Mobile Platform on any request. Login processing on SAP Mobile Platform then proceeds the same as in network-edge authentication; however, with the added benefit that SAP Mobile Platform never has access to the user's password, and therefore cannot leak it if compromised.

Token-based authentication is the most secure SAP Mobile Platform SSO scenario.

---

**Note:** The Check Impersonation option in the security profile settings in Management Cockpit ensures that SAP Mobile Platform knows who the user is after successful SSO-based login. In token-based authentication, the user identity (Principal) may be returned to the HTTP/HTTPS Authentication as an HTTP header.

---

### See also

- *Check Impersonation Attribute* on page 244

### *Basic Authentication Against an SSO-Integrated Service*

With basic authentication to an SSO-integrated back end, the user name and password of the basic credentials are sent to SAP Mobile Platform where the security profile uses the HTTP/HTTPS Authentication provider to pass the credentials to the SSO-enabled Web server for validation.

In response, the security profile receives the SSO cookie used by other SSO-enabled back ends.

Basic authentication against an SSO-integrated service is the least secure and the least common SAP Mobile Platform SSO scenario.

The Check Impersonation option in the security profile settings in Management Cockpit ensures that SAP Mobile Platform knows who the user is after successful SSO-based login. When using the basic authentication scenario, SAP Mobile Platform already has the user name, but additional principals can be returned as well.

---

**Note:** Impersonation checking can be disabled; however, logging, notifications, auditing and more will not work in SAP Mobile Platform, and administrators have less information about who their users are and what they are doing. Additionally, disabling impersonation checking makes it possible for an attacker to steal an SSO cookie and use it without the user's knowledge while masquerading as someone else.

---

### See also

- *Check Impersonation Attribute* on page 244

### Single Sign-on Integration with SiteMinder

Configure SAP Mobile Platform for SSO integration with your SiteMinder environment.

CA SiteMinder enables policy-based authentication and single sign-on with SAP Mobile Platform. You can configure SiteMinder and SAP Mobile Platform integration in a number of ways, depending on your environment.

### See also

- *Configuring SAP Mobile Platform SiteMinder Integration with the SAP SSO2 Mechanism* on page 237
- *Configuring Unprotected Network-Edge Authentication with a SiteMinder-Protected Back End* on page 238

### *SiteMinder Client Authentication*

SiteMinder provides various client authentication options for SAP Mobile Platform.

In SiteMinder client authentication

- SAP Mobile Platform uses the SSO cookie name SMSESSION.
- When network edge authentication is used, SiteMinder adds an SM\_USER header to the client's request along with the SMSESSION cookie. The Populate JAAS Subject From Client provider should set SM\_USER as a Subject Principal so that check Impersonation can be enabled.

---

**Note:** SAP Mobile Platform does not support using SMSESSION as SSO credentials to any back-end systems.

---

SiteMinder client authentication includes:

- Network edge – when a reverse proxy in the DMZ is protected by SiteMinder, the SAP Mobile Platform client is challenged for basic authentication credentials. If the credentials are valid, an SMSESSION cookie is issued and the client is allowed through to the SAP Mobile Platform server. The client begins a session by sending an HTTPS request to the reverse proxy. The reverse proxy detects the unauthenticated request, and challenges using

basic authentication. After the 401 challenge, the client may already have network credentials configured, or executes a callback to prompt for credentials.

- Unprotected-network edge – the network edge (reverse proxy) is not protected. The client's request is allowed to flow to SAP Mobile Platform, where an authentication provider presents the basic credentials to a SiteMinder-protected Web server on behalf of the client. SAP Mobile Platform server retains the SMSESSION cookie and credentials for the client.
- External tokens – the SAP Mobile Platform client application obtains an SMSESSION cookie that is external to the SAP Mobile Platform libraries using custom application processing. This SMSESSION token passes into the SAP Mobile Platform libraries as a cookie. SAP Mobile Platform libraries add the cookie to subsequent HTTP requests to SAP Mobile Platform server. The cookie may or may not be checked at the network edge.

### *Security Profile for a SiteMinder-Protected Back End*

With SAP Mobile Platform, SiteMinder authentication is used in protected and unprotected network-edge configurations.

### **Network-edge and Token-based Authentication**

With network-edge and token-based authentication, a security profile that integrates with applications that use a SiteMinder-protected back end, must use a Populate JAAS Subject From Client provider. The Populate JAAS Subject From Client assigns `sm_user` as a principal, and the SiteMinder agent adds an `sm_user` header to client requests. Use that header in the Populate JAAS Subject From Client provider to set a user Principal.

---

**Note:** If the SiteMinder agent does not add an `sm_user` header, then disable impersonation checking.

---

You should also have an HTTP/HTTPS Authentication provider configured for a SiteMinder-protected URL where SAP Mobile Platform can verify the validity of the user's SMSESSION cookie.

SAP Mobile Platform must send the SMSESSION cookie to the URL. If the URL is a SiteMinder Agent for an SAP-protected back end, then the SSOCookie value should be MYSAPSSO2, the SSO token used against other back-end SAP systems.

When integrating with a back-end system that is not SAP protected, SAP Mobile Platform simply requires a 200 status in the response to indicate the SMSESSION was valid.

### **Basic Authentication**

With basic authentication, the SSOCookie is set to SMSESSION, which is returned upon successful authentication. SAP Mobile Platform has no further use of the SSOCookie; therefore, this is not a commonly used scenario.

### Configuring SAP Mobile Platform SiteMinder Integration with the SAP SSO2 Mechanism

To create a security profile for your single sign-on (SSO) applications, use Management Cockpit.

**1.** Create a security profile for network-edge authentication in Management Cockpit:

- a) In Management Cockpit, select **Settings** and click **New**.
- b) Enter a name in the Security Profile Properties **Name** field, for example `ne_auth`.
- c) Under Authentication Providers, click **New**.
- d) Select Populate JAAS Subject From Client from the list, then click **Create**.
- e) Enter these values:

Field	Value
Client HTTP Values As Name Principals	SM_USER

- f) Under Authentication Providers, click **New**.
- g) Select HTTP/HTTPS Authentication from the list, then click **Create**.
- h) Enter these values:

Field	Value
URL	Point to a SiteMinder-protected reverse proxy that can proxy to a SiteMinder Agent for SAP that can issue the MYSAPSSO2 cookie with the SSO2 ticket in it
SSO Cookie Name	MYSAPSSO2
Client HTTP Values To Send	SMSESSION
SendClientHttpValuesAs	cookie:SMSESSION

---

**Note:** The reverse proxy should add the WASUSERNAME header to the proxy request, where the header value contains the user ID for which you need to generate the SSO2 ticket.

---

- i) Click **Save**.
  - j) Click **OK**.
  - k) Click **Save**.
- 2.** Create an application in Management Cockpit:
- a) In Management Cockpit, select **Applications**, and click **New**.
  - b) Enter these values:

Field	Value
ID	ID name
Name	ID name
Vendor	Vendor name
Type	Application type
Description	Description information

- c) Click **Save**.
- 3. Create the application end point URL:
  - a) Click **Back End**.
  - b) In the **Endpoint** field, enter the URL of the Web service that expects the MYSAPSSO2 token to authenticate the user.
  - c) Click **Authentication**, select **Existing Profile**.
  - d) From the Name list, select the name configured in Step 1b, for example ne\_auth.
  - e) Click **Save**.
  - f) Click **Yes**.

**See also**

- *Integrating SAP Mobile Platform with Your Single Sign-on Solution* on page 233
- *Single Sign-on Integration with SiteMinder* on page 235

*Configuring Unprotected Network-Edge Authentication with a SiteMinder-Protected Back End*

Configure the SMSESSION cookie for unprotected network-edge authentication to a SiteMinder-protected back end.

Unprotected network-edge authentication requires you to change the Web service endpoint in Management Cockpit to use the SMSESSION cookie for single sign-on. By default, SAP Mobile Platform is configured with the server name set to SAP Mobile Platform Server or a reverse proxy, depending on your configuration.

- 1. Create a security profile for network-edge authentication in Management Cockpit:
  - a) In Management Cockpit, select **Settings** and click **New**.
  - b) Enter nne\_auth in the Security Profile Properties field.
  - c) Under Authentication Providers, click **New**.
  - d) Select HTTP/HTTPS Authentication from the list, then click **Create**.
  - e) Enter these values:



Field	Value
URL	Point to the reverse proxy set up in the network edge that authenticates the user before forwarding the request to SAP Mobile Platform Server
SSO Cookie Name	MYSAPSSO2

- f) Click **Save**.
  - g) Click **OK**.
  - h) Click **Save**.
2. Create an application in Management Cockpit:
- a) In Management Cockpit, select **Applications** and click **New**.
  - b) Enter these values:

Field	Value
ID	nne_test
Name	nne_test
Vendor	SAP
Type	Hybrid
Description	Unprotected network-edge testing of Site-Minder

- c) Click **Save**.
3. Create the application end point URL:
- a) Click **Back End**.
  - b) In the **Endpoint** field, enter the URL of the Web service that expects the MYSAPSSO2 token to authenticate the user.
  - c) Click **Authentication**, select **Existing Profile**.
  - d) Select nen\_auth in the Name list.
  - e) Click **Save**.
  - f) Click **Yes**.

### See also

- *Integrating SAP Mobile Platform with Your Single Sign-on Solution* on page 233
- *Single Sign-on Integration with SiteMinder* on page 235

### *SiteMinder Web Agent Configuration for SAP Mobile Platform*

When integrating with SAP Mobile Platform, SiteMinder uses default settings for the Web agent to stop cross-site scripting (XSS) attacks. The SiteMinder default settings do not allow use of special characters and can lead to integration issues with SAP Mobile Platform.

By default, the Web agent does not allow certain characters that are often seen in XSS attacks to be included in the URLs it processes. The Web agent allows only characters that are legal according to the defined HTTP standard.

Native HTTP OData applications typically use, and sometimes require, URLs that contain characters within open and close parentheses and within single quotes. The left and right parenthesis and single-quotes characters are prohibited.

The SiteMinder administrator must modify the Web agent configuration in the policy server to either disable XSS filtering, or change the default forbidden characters.

### *X.509 Authentication at the Network Edge with a Reverse Proxy*

Enable HTTP with mutual certificate authentication at the network edge with a third-party intermediary reverse proxy server.

### *Preparing Certificates*

The client, reverse proxy, and SAP Mobile Platform Server each use their own certificate; you can create or sign these certificates from one root certificate.

By default, SAP Mobile Platform Server includes one default self-signed certificate in:

```
SMP_HOME\Server\configuration\smp_keystore.jks.
```

The certificate (the alias name is "smp crt") is used for HTTPS communications.

The Windows version of Apache2 does not support an encrypted certificate key file. You can use OpenSSL to decrypt the file:

```
openssl rsa -in encrypted.key -out decrypted.key
```

The `SSLCertificateKeyFile` and the private key in `SSLProxyMachineCertificateFile` must be unencrypted.

The `SSLProxyMachineCertificateFile` must be a public key merged with an unencrypted private key.

### *Configuring SAP Mobile Platform Server Certificate-based Authentication with a Reverse Proxy*

Configure SAP Mobile Platform to allow certificate-based authentication when there is a reverse proxy handling client requests at the network edge, and the SSL is terminated before reaching SAP Mobile Platform Server.

The user's certificate arrives at SAP Mobile Platform Server in a `SSL_CLIENT_CERT` HTTP header, and you must configure SAP Mobile Platform to trust the header during

authentication. Trust is established by requiring a mutual certificate authentication between the reverse proxy and SAP Mobile Platform, where the reverse proxy has a technical user certificate signed by a CA in the SAP Mobile Platform truststore.

You must then ensure that the technical user is in the Impersonator role. Once these requirements are met, SAP Mobile Platform processes the `SSL_CLIENT_CERT` header and trust that certificate.

1. Add X.509 User Certificate security provider to validate the user certificate presented over the HTTPS connection.
  - a) In the Management Cockpit, select **Settings**.
  - b) In Security Profiles, select the security profile to be used by the application to authenticate the user.
  - c) Under Authentication Providers, click **New**.
  - d) Select X.509 User Certificate security provider from the list, then click **Create**.
  - e) Configure the values, and click **Save**.
  - f) Click **OK**, then click **Save**.
2. Update the corresponding role mapping file following the process and map the Impersonator logical role to the subjectDN from the certificate the reverse proxy at the network edge is configured with. This is a required step so that the reverse proxy can be trusted to have validated the end-user certificate presented to it over the mutual authentication connection that the client establishes to the network edge.

```
<DefaultMapping>
  <LogicalName>Impersonator</LogicalName>
  <MappedName>Impersonator</MappedName>
  <MappedName>user:EMAILADDRESS=john.doe@sap.com,
  CN=reverse_proxy_user,OU=SMP, O=SAPAG, ST=CA, C=US</MappedName>
</DefaultMapping>
```

3. Obtain a valid signed server certificate for your SAP Mobile Platform Server.
4. Import the certificate into the keystore using the "smp\_crt" alias.
  - a) Import the CA signing certificate used to sign client certificates into the `smp_keystore.jks` as a trusted CA certificate so that SAP Mobile Platform is able to validate client certificates later.
5. Restart SAP Mobile Platform Server.

---

**Note:** The `<MappedName>` value in the role mapping file must exactly match what SAP Mobile Platform extracts from the reverse proxy technical user certificate, including upper and lowercase letters and any spacing. The easiest way to ensure an exact match is to set the security logging level to `DEBUG` from the Management Cockpit, then attempt a client connection through the reverse proxy. Go into the server log where you can find the DN from the reverse proxy certificate printed out exactly as SAP Mobile Platform sees it. Cut and paste from the log file into the role-mapping file and restart the server. If the role mapping is not an exact match, the Impersonator role is not granted and SAP Mobile

Platform does not trust the `SSL_CLIENT_CERT` header and refuses to execute the request in the context of the mobile user.

---

### See also

- *X.509 Certificate Authentication* on page 163
- *Single Sign-on for SAP* on page 242
- *Single Sign-on Authentication* on page 242
- *Preparing Your SAP Environment for Single Sign-on* on page 243
- *X.509 User Certificate Configuration Properties* on page 184

### Single Sign-on for SAP

SAP Mobile Platform supports single sign-on (SSO) authentication for mobile clients that access data from an SAP back end by forwarding credentials from the end user.

The credential types that may be used (in preferential order within SAP Mobile Platform are:

- X.509 certificate
- SSO2 token
- User name and password

An SAP back-end connection may be checked for "Allow Anonymous", in which case SSO is performed only if one of the above three credential types is available. Otherwise, technical user credentials are used for the back-end connection, which is not SSO.

### See also

- *X.509 User Certificate Configuration Properties* on page 184
- *Configuring SAP Mobile Platform Server Certificate-based Authentication with a Reverse Proxy* on page 240

### Single Sign-on Authentication

Understand the role of user credentials and X.509 certificates in single sign-on authentication.

Single sign-on authentication comprises three main areas:

- SAP Mobile Platform to a back end
- Client to SAP Mobile Platform
- Back-end user mapping

Configuring SAP Mobile Platform to perform single sign-on to the back end requires the applications created in SAP Mobile Platform Server to be configured to use the HTTPS protocol with mutual certificate authentication to communicate with the back end. Use Management Cockpit to navigate to the application, and set the property "Certificate Alias" — that is, give the name of a certificate alias in `SMP_HOME\Server\configuration\smp_keystore.jks`.

During mutual certificate authentication between the client and SAP Mobile Platform, the client presents a certificate to SAP Mobile Platform Server. For authentication to succeed, the

client's certificate, or more typically the certificate authority (CA) that signed the client certificate must be present in the SAP Mobile Platform Server keystore.

---

**Note:** When administrators import a certificate to the keystore, they must use the same password for the key alias entry as the keystore password, and thus the same value for the Certificate Alias.

---

### See also

- *X.509 User Certificate Configuration Properties* on page 184
- *Configuring SAP Mobile Platform Server Certificate-based Authentication with a Reverse Proxy* on page 240

### *Preparing Your SAP Environment for Single Sign-on*

Verify that the SAP back end is configured correctly to accept SSO connections from SAP Mobile Platform Server.

1. Set all parameters for the type of credentials accepted by the server:
  - SSO2 token – verify everything is set properly with the SSO2 transaction.
  - X.509 certificate – set up, import, and verify certificates using the Trust Manager (transaction STRUST).
2. Use the ICM configuration utility to enable the ICM HTTPS port.
3. Set the type of authentication to enable communication over HTTPS.
  - Server authentication only – the server expects the client to authenticate itself using basic authentication, not SSL
  - Client authentication only – the server requires the client to send authentication information using SSL certificates. The ABAP stack supports both options. Configure the server to use SSL with client authentication by setting the ICM/HTTPS/verify\_client parameter:
    - 0 – do not use certificates.
    - 1 – allow certificates (default).
    - 2 – require certificates.
4. Use the Trust Manager (transaction STRUST) for each PSE (SSL server PSE and SSL client PSE) to make the server's digitally signed public-key certificates available. Use a public key-infrastructure (PKI) to get the certificates signed and into the SAP system. For information, see *SAP Trust Manager* at [http://help.sap.com/saphelp\\_ain710/helpdata/en/0e/fb993af7700577e10000000a11402f/frameset.htm](http://help.sap.com/saphelp_ain710/helpdata/en/0e/fb993af7700577e10000000a11402f/frameset.htm).
5. To enable secure communication, SAP Mobile Platform Server and the SAP server that it communicates with must exchange valid CA X.509 certificates. Deploy these certificates, which are used during the SSL handshake with the SAP server, into the SAP Mobile Platform Server keystore.
6. The user identification (distinguished name), specified in the certificate must map to a valid user ID in the AS ABAP, which is maintained by the transaction SM30 using table view (VUSREXTID).

See *Configuring the AS ABAP for Supporting SSL* at [http://help.sap.com/saphelp\\_ain710/helpdata/en/49/23501ebf5a1902e1000000a42189c/frameset.htm](http://help.sap.com/saphelp_ain710/helpdata/en/49/23501ebf5a1902e1000000a42189c/frameset.htm)

### See also

- *X.509 User Certificate Configuration Properties* on page 184
- *Configuring SAP Mobile Platform Server Certificate-based Authentication with a Reverse Proxy* on page 240

### Propagate Single Sign-on Using Populate JAAS Subject From Client

Applications use HTTP headers and cookies to pass data that should be used for single sign-on into the back end. The Populate JAAS Subject From Client enables administrators to add named credentials, name principals, and role principals to the authenticated subject.

Adding client values as named credentials allows them to be used for single sign-on. When authenticating the user using a token from the client, if the corresponding authentication provider cannot retrieve the user name from the token and add it as a principal for use in impersonation checking, the administrator can configure this provider to add the appropriate header value from the client session as a principal to the authenticated subject.

---

**Note:** Rogue applications can intentionally insert HTTP headers with arbitrary values to obtain principals, roles, or credentials that they otherwise would not receive using the other authentication providers. Use this authentication provider in an environment where you know what the network edge behavior and have ensured that applications cannot bypass or override that environment.

---

To avoid a client setting an HTTP header/cookie value to work around the impersonation check, use this configuration only when the SSO framework requires it, and when the deployed applications ensure that the client cannot manipulate the headers set into the session. HTTP headers set by the network edge take precedence.

This authentication provider does not authenticate the subject but adds the NamedCredential if the user is successfully authenticated by other authentication providers. It always returns false from the login method and should always be configured with the controlFlag set to “optional” to avoid affecting the outcome of authentication process.

### See also

- *Populate JAAS Subject From Client Configuration Properties* on page 182

### Check Impersonation Attribute

The Check Impersonation option in the security profile settings in Management Cockpit ensures that SAP Mobile Platform verifies who the user is after successful SSO-based login. Disabling Check Impersonation in Management Cockpit allows authentication to proceed without verifying that the presented token is associated with the user.

The Check Impersonation attribute allows authentication to succeed when, in token-based authentication, the presented user name cannot be matched against any of the user names

validated in the authentication providers. In token-based authentication, even though a valid token may be presented to SAP Mobile Platform, the token may not be associated with the user indicated by the user name. To prevent the user authentication from succeeding in this scenario, the Check Impersonation attribute is enabled by default. When an unauthenticated request is received by SAP Mobile Platform (for example, from a device or Push Notification request), it may contain a token (in an HTTP header or cookie) that should be validated to authenticate the user. In some cases, a user name can be extracted from the token. In SAP Mobile Platform, the specified user name is matched to the name of at least one of the public principals added by the authentication providers. If the user name cannot be extracted from the token as part of the validation, then the specified user name is not added as a principal.

In certain situations, it may not be possible for the token validation server to return the user name embedded in the token. If no such custom authentication provider is available, then the administrator can allow authentication to succeed even when the user name presented cannot be matched against any of the user names validated by the configured authentication providers. In these situations, a custom authentication provider that maps the token to a user name and adds a principal with that user name may be used. To allow this authentication, uncheck the Check Impersonation check box in the Management Cockpit for the associated security profile.

---

**Note:** When SAP Mobile Platform recognizes user IDs, it can perform auditing, logging, registration, usage reporting, tracking user transactions during troubleshooting, notifications, application life cycle management, and more. Disabling the Check Impersonation attribute causes these features to no longer work in SAP Mobile Platform. With Check Impersonation disabled, administrators have less information about who their users are and what they are doing. Additionally, disabling Check Impersonation makes it possible for an attacker to steal an SSO cookie and use it without the user's knowledge while masquerading as someone else.

---

**Table 25. Check Impersonation Attribute**

Attribute	Default	Description
Check Impersonation	Enabled	(Optional) Determines whether to allow SSO authentication to succeed when the user name cannot be matched against any of the user names validated in the authentication providers.

**See also**

- *Creating and Configuring Security Profiles* on page 165
- *Network-Edge Authentication* on page 233
- *Token-based Authentication* on page 234
- *Basic Authentication Against an SSO-Integrated Service* on page 234

**Using SSL to Secure HTTPS Channel Communications in SAP Mobile Platform**

Configure Secure Sockets Layer (SSL) and mutual SSL encryption for communication ports in SAP Mobile Platform.

SAP Mobile Platform uses HTTPS to create a secure channel between:

- The client (device) and SAP Mobile Platform Server
- SAP Mobile Platform Server and the back end

In both of these scenarios, you can have one-way or mutual SSL.

SAP Mobile Platform uses the following HTTP and HTTPS ports, default assignments, and protocols.

Type	Default	Protocol	Where Configured
Client communications, unsecured	8080	HTTP	File: <i>SMP_HOME</i> \Server\config_master\org.eclipse.gemini.web.tomcat\default-server.xml  Setting: <Connector port="8080" ...
Client communications with one-way SSL authentication	8081 (secure)	HTTPS	File: <i>SMP_HOME</i> \Server\config_master\org.eclipse.gemini.web.tomcat\default-server.xml  Setting: <Connector protocol="com.sap.mobile.platform.coyote.http11.SapHttp11Protocol" port="8081" ...



Type	Default	Protocol	Where Configured
Client communications with mutual SSL authentication	8082 (secure)	HTTPS	File: <i>SMP_HOME</i> \Server\config_master\org.eclipse.gemini.web.tomcat\default-server.xml  Setting: <Connector protocol="com.sap.mobile.platform.coyote.http11.SapHttp11Protocol" port="8082" ...
Administration port, Management Cockpit and Kapsel command line interface (CLI) communications with one-way SSL authentication	8083 (secure)	HTTPS	File: <i>SMP_HOME</i> \Server\config_master\org.eclipse.gemini.web.tomcat\default-server.xml  Setting: <Connector protocol="com.sap.mobile.platform.coyote.http11.SapHttp11Protocol" port="8083" ...

---

**Note:** If you change these default ports, you must restart SAP Mobile Platform Server for the changes to take effect.

---

### Using SSL Between a Client and SAP Mobile Platform Server

Use SSL to secure HTTPS channel communication between a client (a device) and SAP Mobile Platform Server.

The self-signed certificate `smp_cert` is created during installation and contains the fully qualified domain name of the system as its CN. By default, the same certificate is configured for all secure connections in SAP Mobile Platform Server. Because this certificate is self-signed, it does not have any CA for validation. Use a PKI system and a trusted CA to generate production-ready certificates and keys that encrypt communication between the client and the server.

---

**Note:** Ensure the certificate you use is contained in the server keystore. You can use the `keytool` utility to import and export certificates to the keystore. Any changes to the keystore require the server to be restarted.

---

1. Use Management Cockpit to create an application with an HTTP/HTTPS back end.
2. Place the CA of the server certificate into the client keystore.
3. Connect to the server. When you connect to the server using HTTPS, the server sends back its certificate.
  - a) For one-way SSL, connect using `https://<servername>:8081/`.
  - b) For mutual SSL, connect using `https://<servername>:8082/`.
4. Validate the server certificate from the application. If the server certificate is valid, both will exchange cipher; which is used for encoding further communication.
5. Upon successful certificate validation, SAP Mobile Platform establishes a client-to-server connection, and further request responses occur using the secure channel until the session expires.

### See also

- *Certificate Alias* on page 223
- *Keytool Utility* on page 224
- *Keystore/Truststore Properties* on page 229
- *Managing Keystore and Truststore Certificates* on page 223

### Using SSL Between SAP Mobile Platform Server and the Back End

Use SSL to secure HTTPS channel communication between SAP Mobile Platform Server and the back end.

SAP Mobile Platform secures interactions between SAP Mobile Platform Server and back-end systems. This security component includes device application back-end connections and notification communication.

1. Use Management Cockpit to create an application with an HTTP/HTTPS back end.
  - a) For one-way SSL, do not provide a certificate alias.
  - b) For mutual SSL, provide a certificate alias.
2. Add the back-end certificate into the SAP Mobile Platform Server keystore.

---

**Note:** Ensure the certificate you use is contained in the SAP Mobile Platform Server keystore. You can use the keytool utility to import and export certificates to the keystore. Any changes to the keystore require the server to be restarted.

---

3. Upon connection between SAP Mobile Platform Server and the back end, the back end sends its certificate. SAP Mobile Platform Server validates against the back end certificate stored in the keystore.
4. Upon successful certificate validation, SAP Mobile Platform establishes a server to back-end connection, and further request responses occur using the secure channel until the session expires.

**See also**

- *Certificate Alias* on page 223
- *Keytool Utility* on page 224
- *Keystore/Truststore Properties* on page 229
- *Managing Keystore and Truststore Certificates* on page 223

**Anonymous Access Applications**

Applications that do not require tight security can use anonymous access. Anonymous access applications can be run without a specific combination of user name and authorization code or a combination of code and password.

When anonymous connections are enabled in Management Cockpit, the application user can access the application without entering a user name and password or a combination of authorization code and password. However, the back-end system still requires log on credentials to access data, whether it is a read-only user, or a back-end user with specific roles.

---

**Note:** If you configure the No Authentication Challenge authentication provider in a security profile to which you have assigned client applications that you intend to run anonymously, this provider causes your anonymous applications to fail. SAP Mobile Platform authenticates the user even though the user presented no valid credentials. SAP Mobile Platform then proceeds to connect to back-end systems assuming there is an authenticated client, and tries to use SSO credentials for the back end. However, these credentials are absent, and the back-end connection fails.

---

SAP also supports an "anonymous optional" scenario, where an anonymous application may provide a limited set of functionality to anonymous users. A user who chooses to authenticate may have more functionality exposed (for example, real user credentials which are propagated via SSO to the back end and allows more access).

---

**Note:** When an endpoint is configured with the "Allow Anonymous" attribute, and technical user credentials are provided, then clients can use that endpoint anonymously. Even though the business content SAP Mobile Platform accesses through the endpoint may have been deemed non-sensitive and not require a high degree of security, often the technical user may have access to other parts of the back-end system that are sensitive. Because of this, the technical user's credentials need to be protected. Always use an HTTPS connection to the back-end system in order to protect the technical user credentials from being compromised as they are passed over the network.

---

**Enabling a Direct HTTPS Connection to SAP Mobile Platform Server**

SAP Mobile Platform includes two HTTPS listeners that clients can use to directly communicate with the SAP Mobile Platform Server HTTPS port. There is a one-way HTTPS listener where the server certificate goes to the client, and there is a two-way HTTPS listener where the client must also send its certificate to the server for mutual authentication.

Both listeners use the server certificate identified by the "smp\_crt" alias in the keystore. The SAP Mobile Platform installation process creates this self-signed certificate. Most clients or servers do not trust a self-signed certificate, so SAP recommends that customers use a trusted CA to sign a replacement certificate for the server. The signed certificate should be imported to the keystore using the same "smp\_crt" alias.

The summary steps for enabling a direct HTTPS connect to SAP Mobile Platform Server includes:

1. Obtain a valid signed server certificate for your SAP Mobile Platform Server.
2. Import the certificate into the keystore using the "smp\_crt" alias. Import the CA signing certificate used to sign client certificates into the `smp_keystore.jks` as a trusted CA certificate so that SAP Mobile Platform is able to validate client certificates later.
3. Add the X.509 User Certificate provider to your security profile assigned to your application.
4. Restart the server to pick up the new certificate.

### See also

- *Managing Keystore and Truststore Certificates* on page 223
- *Keystore/Truststore Properties* on page 229
- *Keytool Utility* on page 224

## Enabling OCSP

(Optional) Enable Online Certificate Status Protocol (OCSP) to check if a certificate has been revoked.

A CA may issue a certificate to a user that would remain valid for a months or perhaps even years. If the certificate becomes compromised (for example, due to a lost device or unauthorized access to the private key) the CA system can be notified to revoke that certificate. Unless SAP Mobile Platform explicitly checks for revocation, the certificate appears valid.

In the X.509 User Certificate of your security profile, set the property to enable revocation checking in addition to setting up OCSP in the `java.security` file.

Enable OCSP.

1. Edit the `SMP_HOME\sapjvm_7\jre\lib\security\java.security` file.

```
#
# Properties to configure OCSP for certificate revocation checking
#
# Enable OCSP
#
# By default, OCSP is not used for certificate revocation
checking.
# This property enables the use of OCSP when set to the value
"true".
```

```

#
# Note: SocketPermission is required to connect to an OCSP
responder.
#
# Example,
#   ocspl.enable=true

#
# Location of the OCSP responder
#
# By default, the location of the OCSP responder is determined
implicitly
# from the certificate being validated. This property explicitly
specifies
# the location of the OCSP responder. The property is used when the
# Authority Information Access extension (defined in RFC 3280) is
absent
# from the certificate or when it requires overriding.
#

```

2. Uncomment and configure your required OCSP properties. For more information, see Java documentation at [http://docs.oracle.com/cd/B28196\\_01/idmanage.1014/b28168/toc.htm](http://docs.oracle.com/cd/B28196_01/idmanage.1014/b28168/toc.htm)

### See also

- *Creating and Configuring Security Profiles* on page 165
- *Mapping a Logical Role to a Physical Role* on page 174
- *X.509 User Certificate Configuration Properties* on page 184
- *X.509 User Certificate Provider* on page 184

## Device Security

---

To fully secure devices, developers and administrators can combine multiple mechanisms. In addition to using the built-in security features of both the device and SAP Mobile Platform, SAP recommends that you also use Afaria so you can remotely initiate security features as required.

Application authentication is defined by the developer, managed by the administrator in the Management Cockpit, and processed by the core CSI in SAP Mobile Platform Server.

Device security in SAP Mobile Platform follows this process:

1. The client sends the application ID and user credentials (including user name and password, certificate, or token) to SAP Mobile Platform.
2. SAP Mobile Platform uses the application ID to find the security profile that should authenticate the user credentials, and invokes the authentication providers in that profile to perform the authentication.

3. When authentication succeeds, the user credentials or additional credentials derived during the authentication process are made available as SSO material towards the back-end systems.

In SAP Mobile Platform Server, the client always provides the credentials defined in their security profile, and not the back-end system. If you are configuring multiple back ends, then following options are possible:

- Use SAP SSO2 Token when connecting to an SAP back-end system
  - User provides credentials for the SAP Mobile Platform Server authentication, which in turn provides a MYSAPSSO2 token.
  - That same token can be used to connect to all back-end systems.
- Use X.509 certificate when connecting to an SAP back-end system
  - A trusted certificate can be used with all back-end systems.
- Use basic authentication when connecting to any back-end system
  - The SAP Mobile Platform Server authentication and all back-end systems should have same user name and password.

Developers define SAP Mobile Platform security features for devices, including data encryption, login screens, and data vaults for storing sensitive data. Developers use the Client Hub, integrated with Logon Manager, which simplifies user onboarding and configuration to enable easier and faster enterprise-wide deployments. The Client Hub reduces the effort required by the end user to manage multiple passwords for mobile applications and improves the user experience.

### **Limiting Application Access**

Application access to SAP Mobile Platform runtime is tightly controlled: before a user can access a mobile application, he or she must provide a passcode; before the application can access the runtime, the application must be registered and provisioned with required connections and security profiles.

Applications that do not require tight security can use anonymous access. Anonymous access applications can be run without a specific combination of user name and authorization code or a combination of code and password.

### **Device Data Security**

Securing all data on the device client requires multiple techniques employed by both the developer and the administrator.

Some SAP Mobile Platform components do not support encryption. Review this table to see which components can enable this security feature.

---

**Note:** The encryption key for all client databases is an AES-128 sized key, and the data vault does not generate the encryption key.

---

Component	Implementation Notes
Device data	If the application has a local database, SAP recommends the database be encrypted.
Device client database	Set by the developer. The encryption key is stored in the data vault.
Data vault	Set by the developer. The DataVault API provides a secure way to persist and encrypt data on the device.
Client password policy	Set by the administrator. Applies to passwords used to unlock the data vault (when implemented). Administered through Management Cockpit.

### See also

- *Defining Client Password Policy* on page 59

### **Application, Device, and User Registration**

Before any application can access the runtime, the user, device, and application must be identified by first registering with SAP Mobile Platform Server and pairing them with a device and user entry. Subscriptions can be made only when all three entities are known.

In Management Cockpit, administrators set up application registration.

### **Application Recognition**

Applications are recognized by SAP Mobile Platform Server by the properties that define them. Administrators define applications with a unique application ID and other key application properties, such as security configuration.

An application cannot register a connection unless a definition has been created for it. If your development team has not yet set these application properties, administrators must do so before the application connection can be registered.

## **Securing Sensitive Data On-Device with DataVault**

To securely store data on the device, developers should use a data vault with device applications. Administrators then define the password policy using Management Cockpit

The data vault provides encrypted storage of occasionally used, small pieces of data. The data vault holds sensitive artifacts securely, because all data or artifacts in the data vault are encrypted with an AES-256 bit key. Content can include encryption keys, user and application login credentials, synchronization profile settings, and certificates.

---

**Note:** For iOS, the contents of the data vault are strongly encrypted using AES-128.

---

The data vault requires a password to unlock and access the data from the application. Therefore, a device application must prompt the user to enter this password when the

application is opened. Once unlocked, the application can retrieve any other secrets from the vault as needed, all without prompting the user.

Administrators can define a password policy using Management Cockpit that defines the requirements for an acceptable password. The client password policy is stored in the server-side settings database and the client gets those settings when it connects to SAP Mobile Platform Server as part of the settings exchange protocol.

When the client receives the password policy settings, it can populate the settings objects to the data vault. The data vault stores the settings. The client uses the DataVault API to create a vault with a default password, set the password policy, and change the password to one that is compatible with the policy. If you do not change the password after setting a password policy, the application throws an exception if you attempt to access the application or unlock the vault with an incompatible password.

Administrators should discuss the data vault strategy with developers before it is implemented, especially regarding:

- **Failed logins** – developers can set the number of failed login attempts allowed before the data vault is deleted. Once the vault is deleted, the encrypted databases are not usable. The application needs to be reinstalled or reinitialized, including deleting the database files to recover.
- **Timeouts** – developers can also set a timeout value so that the data vault locks itself when it is not in use. The user must reenter the vault password to resume using the application.

### See also

- *Defining Client Password Policy* on page 59

### **Client Password Policy for Data Vault Logins**

Administrators can define a client password policy for native or hybrid device application logins in a new or existing application. A password policy ensures that user-defined passwords conform to corporate security requirements.

A policy cannot be enforced unless developers add enforcement code to the data vault for an application.

### See also

- *Defining Client Password Policy* on page 59

### **Login Screens for Data Vaults**

An application that implements a login screen is considered secure. Mobile application developers are responsible for creating login screens for the applications they create. A login screen allows the device user to enter a password to unlock the data vault after an administrator has set a client password policy in Management Cockpit.

A secure application that uses a login screen:



- Prompts the user to enter the data vault password to open the application and gain access to the local client database. If the wrong password is used, the application is rendered useless: the key that encrypts and decrypts data in the vault cannot be used to access data until this code is accurately entered.
- Can self-destruct after a configured number of incorrect password attempts.

The Lock Timeout limits the length of time (seconds) the data vault can be left unlocked within the application so the user can continue to use it. If the data vault is not accessed within this timeout, it locks itself and the user has to go back to the login screen to re-enter their password in order to continue using the application.

To implement a login screen, developers must create the login and define the password. The screen and the password unlock the data vault. Unlocking the vault enables access to application data offline or online.

Administrators can allow users to change this password, when defining the client password policy in Management Cockpit.

The SAP Mobile Platform data vault password policy does not include any password change frequency requirements. The user can change their password whenever desired. When a user does change their password, any changes the administrator may have made to the policy would be applied to the new user password only. This is dependent upon the developer has included logic to:

- Allow the user to change the password
- Pick up password policy changes and apply them to the vault when they occur

## Agentry Security

---

There are numerous security features available to Agentry applications. The Agentry Server supports client data and password encryption, encrypted client-server communications, and authentication certificates.

In general, Agentry security features are organized into two categories:

- Those that are built into the platform, which may require configuration during implementation, and
- Those that are part of the application deployed on Agentry, which are therefore a part of the application definitions and components.

To implement security features in Agentry, understand data and password encryption methods, security protocols, client-server communications, and client-server certificate authentication within Agentry.

### *Client-Side Data Encryption*

When defining an Agentry Client application, you can specify whether to encrypt data stored locally in the Application Definition using the Agentry Editor. An encrypted client encrypts

all production data and application data stored on the client device. This functionality provides a layer of security for all data stored on the client device by the Agentry Client. See the specifications for details on the encryption strength and protocols used.

### *WebSockets*

WebSockets is a standard for allowing bidirectional real-time communication between clients and servers that is encapsulated within another transport protocol, such as HTTPS. Agentry on the SAP Mobile Platform uses WebSockets to route its binary communication protocol (known as ANGEL in previous Agentry releases) through HTTPS.

WebSockets enable Agentry components to work consistently across the enterprise, in tandem with other SAP Mobile Platform components, and allow Agentry to leverage WebSockets-aware HTTP reverse proxies.

The Agentry Editor automatically converts the transmit configurations of older Agentry applications into WebSockets transmit configurations.

---

**Note:** Older transmit configurations that contained non-default host name or port configurations will lose that aspect of their configuration, as Agentry does not currently support having WebSockets transmit configurations that use other URLs besides what the client used in its initial transmit.

---

### *RSA Key Pairs*

In addition to SSL certificates, there is an RSA key pair that Agentry uses within its communications. This key pair is generated by the server at installation time, and serves two purposes:

- To encrypt user passwords that are sent from the client to the server (which is a bit redundant given that we're using SSL; it was originally there for use with non-encrypted transports that Agentry no longer supports).
- Clients use the server's public key to encrypt the key that is used for database encryption, so that the database encryption key can be decrypted by the server during a user change on the client side.

The strength of this key pair is controlled by a server setting in Management Cockpit. It is the "publicKeyLength" setting in the "Server" section, and defaults to 4096 bits in SAP Mobile Platform.

### *Client Password Encryption*

The passwords entered by users during login to the Agentry Clients are encrypted based on an encryption key received from the Agentry Server. This key is the public key portion of a public-private key pairing generated by the server, therefore clients are tied to that server after an initial transmit. If clients need to connect to more than one server, as in clustered environments, you can export a server's encryption key and import it to additional servers.

This encryption protects user passwords that are entered on clients. The password value is stored and transmitted in encrypted form. It is decrypted by the server when a client connects,

and when it is read in by the client during user login. In both cases, the decrypted value is used only for validation of the user; it is not permanently stored.

### *Trusted Certificates in Agentry*

Agentry uses HTTPS for server authentication, which requires a server certificate. That certificate is managed by the Web server that services all of SAP Mobile Platform. Agentry Clients do not support client certificate authentication. With integration into SAP Mobile Platform, Agentry now uses platform-wide security components to define how Agentry Clients trust the server's certificate. Key differences from legacy Agentry configurations include:

- The Agentry Server no longer uses the `AgentryServer.pfx` file that held the server-side SSL certificate in previous Agentry releases. Agentry now uses the same server-side SSL certificate that is used by the rest of SAP Mobile Platform, because it uses the same Web server.
- Agentry Clients on Windows no longer use `AgentryTrustedCertificates.sst` to hold the CA certificate that validates the Agentry Server's SSL certificate. Instead, Agentry Clients on all platforms validate the server's SSL certificate by using the trusted certificates that are held by the native operating system. If an in-house CA certificate is needed to validate the server's SSL certificate, then this CA certificate needs to be added to the operating system's trusted CA list using the standard means for the operating system.
- Agentry no longer comes with a default self-signed SSL certificate accepted by clients regardless of the server's name. Clients now only accept SSL certificates that properly contain the DNS name of the Agentry Server. As such, it is not possible to connect an Agentry Client to an Agentry Server "out-of-the-box" without either obtaining a valid server certificate that has been signed by a global CA, or by adding the SAP Mobile Platform's generated self-signed certificate to the trusted certificate store on the client device.

### **See also**

- *Using Nginx Reverse Proxy for Agentry Clients* on page 213

## **Agentry Security Specifications Reference**

Various points at which data is encrypted within SAP Mobile Platform, and the related default algorithms and cipher strengths.

### *Encryption Specifications*

Client-side data encryption specifications are the same for all supported devices.

Data Encryption	Key Exchange Algorithm & Strength	Encryption Algorithm & Default Strength
<b>Client Password (over network)</b>	RSA, strength is configured by the server (the default is 4096).	RSA
<b>Client-Side Data Encryption</b>	PBKDF2-SHA1 128 bit	AES-128

## Security Monitoring and Validation

---

SAP recommends that administrators use multiple diagnostic sources, such as security logs and debugging tools. Monitoring data can effectively supplement obscure and abbreviated debug/syslog output.

### See also

- *Changing Security Log Levels* on page 220

### Checking the Security Log

If you experience problems with security profiles or the authentication, check the SAP Mobile Platform Server log for issues.

Troubleshoot security profile or authentication errors by analyzing the server log. See information about setting the levels of the security log.

### Debugging Authentication Errors with CSI Tool

Use the CSI tool to debug security profile errors that are encountered during user authentication.

Use the CSI tool to debug authentication failures and validate your security configuration outside the SAP Mobile Platform environment.

1. Copy these files to a separate, temporary directory:
  - `csi-core.jar`, found in `SMP_HOME\Server\tools\csi`.
  - the security profile XML file along with the corresponding role mapping file found in `SMP_HOME\Server\configuration\com.sap.mobile.platform.server.security\CSI`.
2. Execute the CSI tool command from the separate temporary directory specifying the following options:
  - `com.sap.security.BootstrapConfigurationFile` - this is required when the configuration file contains encrypted properties. The same directory that contains the `csibootstrap.properties` file must contain the keystore referenced in the bootstrap file, as a relative path is used to resolve the reference to it.
  - `classpath` - should include the `csi-tool.jar` in the temporary directory as well as `SMP_HOME/Server/plugins` directory content.

```
ex: java -Dcom.sap.security.BootstrapConfigurationFile="C:\SAP
\MobilePlatform3\Server\configuration
\com.sap.mobile.platform.server.security\csibootstrap.properties"
-cp csi-tool.jar;C:\SAP\MobilePlatform3\Server\plugins\*
-Djava.util.logging.config.file=logging.properties com.sap.
security.tools.CSILauncher csi.diag.authenticate --USERNAME
"smpAdmin" --PASSWORD
D "s3pAdmin" --CONFIG_FILE C:\SAP\MobilePlatform3\Server
\configuration\com.sap.mobile.platform.server.security\CSI
\admin.xml
```

3. Review the log output to troubleshoot the authentication failure.

CSI uses Java logging API. The following example shows how to configure logging.properties to obtain FINEST level log messages from the classes in the `com.sap.security.ldap` package while setting the log level for rest of the CSI classes to INFO. Use this configuration to debug authentication failures with LDAP providers. You can also use this configuration to debug errors encountered when looking up user roles from the LDAP repository. The value `debug.log` for the property `java.util.logging.FileHandler.pattern` should be the path to the log file.

```
java -Djava.util.logging.config.file=logging.properties -jar csi-
tool.jar csi.diag.authenticate --USERNAME "test_username" --PASSWORD
"test_password"
--CONFIG_FILE "absolute_path_of_the_configuration_xml_file"
```

where `logging.properties` contains:

```
handlers=java.util.logging.ConsoleHandler,
java.util.logging.FileHandler.level=INFO
com.sap.security.ldap.level=FINEST
java.util.logging.FileHandler.formatter=java.util.logging.SimpleFor
matter
java.util.logging.FileHandler.level=FINEST
java.util.logging.FileHandler.pattern=debug.log
```

**See also**

- *Creating and Configuring Security Profiles* on page 165
- *Directory Service (LDAP/AD) Configuration Properties* on page 197

## **Migrating Security Components**

When you migrate from Sybase Mobile Platform and SAP Mobile Platform 2.x to the latest version of SAP Mobile Platform, some adjustments are required to ensure security components and applications runs correctly.

## **Migrating Authentication for Native OData HTTP Clients**

The X.509 User Certificate (CertificateAuthenticationLoginModule) has been removed from the `csi-core.jar` library, and is no longer available for use in security profiles.

**Developer tasks**

Migrate existing Sybase Mobile Platform and SAP Mobile Platform 2.x OData native HTTP clients to use mutual SSL authentication instead of certificate blob.

**Administrator tasks**

In Management Cockpit, add the X.509 User Certificate authentication provider to your designated security profile.

# Application Services (Mobiliser) Administration

SAP® Mobiliser is a fully integrated system with a back-end database, a set of application services as a middle-tier, and a front-end Web application for configuration, management and utilization. It is based on the Open Services Gateway initiative (OSGi) modular architecture.

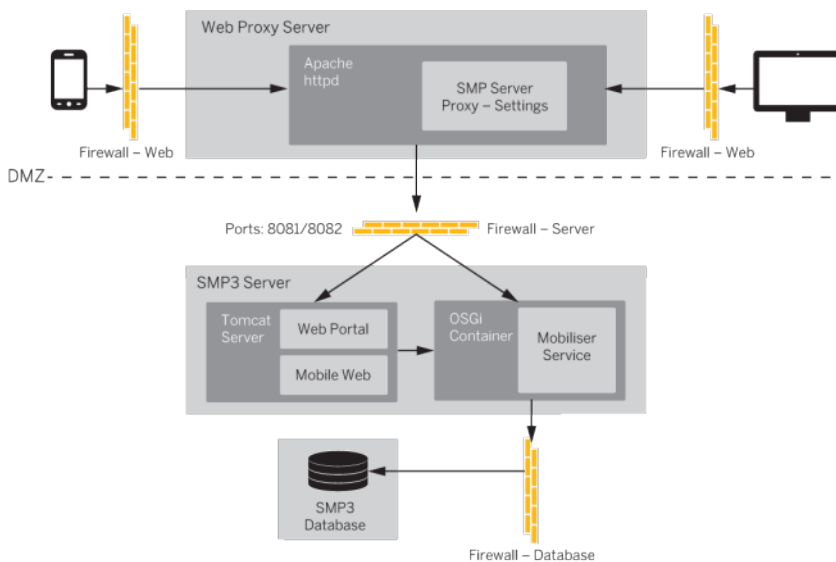
## See also

- *Application Administration* on page 31
- *Server Administration* on page 113
- *Security Administration* on page 149
- *SMS Administration* on page 341

## Basic Deployment Model

Each Mobiliser host must meet the requirements for operating system and available disk space. In a development or test environment, you can install the system on a single physical host or virtual machine. In a production environment, deploy the system in a tiered manner to aid in administration, maintenance, and security.

A standard tiered architecture contains a Web layer, an application layer, and a database layer.



## Standard Reverse Proxy Setup

---

Any reverse proxy, such as Apache HTTPD, can accept incoming requests from the Internet in the DMZ and forward them to the Mobiliser core running on the application server tier.

This example is for an Apache HTTPD server with proxy modules installed:

```
<Proxy *>
    Order deny,allow
    Allow from all
</Proxy>
ProxyPass /mobiliser/smartphone
http://localhost:8080/mobiliser/smartphone

ProxyPassReverse /mobiliser/smartphone
http://localhost:8080/mobiliser/smartphone

ProxyPass /mobiliser/rest/smartphone
http://localhost:8080/mobiliser/rest/smartphone

ProxyPassReverse /mobiliser/rest/smartphone
http://localhost:8080/mobiliser/rest/smartphone

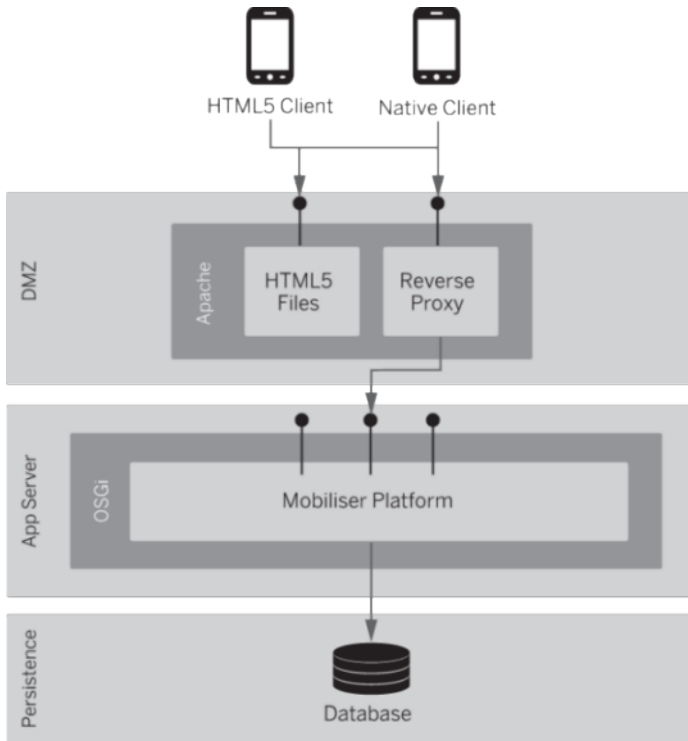
ProxyPass /mobiliser/binary
http://localhost:8080/mobiliser/binary

ProxyPassReverse /mobiliser/binary
http://localhost:8080/mobiliser/binary

ProxyPass /mobiliser/rest/binary
http://localhost:8080/mobiliser/rest/binary

ProxyPassReverse /mobiliser/rest/binary
http://localhost:8080/mobiliser/rest/binary
```





In addition to shielding direct access to the core, the Apache server can provide access to the HTML5 version used by a smartphone, or any other HTML5 application. The same origin policy requires that the HTML files and the Ajax services be provided by the same server (host name + port). See [http://en.wikipedia.org/wiki/Same\\_origin\\_policy](http://en.wikipedia.org/wiki/Same_origin_policy).

The reverse proxy can also be used for SSL termination. When you deploy Mobiliser in the standard reverse proxy model, you must also make changes to the configuration on the Apache Web UI server (located on the DMZ layer) and one of the database preferences (located in the persistence layer).

## Mobiliser Setup

In SAP Mobile Platform, you can enable optional Mobiliser server features. Once you enable these features, you must perform a few other tasks to ensure that Mobiliser functions properly.

### 1. *Creating or Updating Hashed Password for the Universal User*

There are several preconfigured values that you must change, for security reasons, after a fresh installation. You must set a new password for the universal user (customer ID 100) and configure the password in the configuration accordingly.

### 2. *Creating or Updating the Encrypted Password for Preferences*

The encrypted password hash for preferences is used by the portal, and that encrypted value is placed in the MOB\_PREFERENCES table in the database. The first hash is made from any plain text password, while the second (encrypted) hash is built from the chosen value for the first hash. Both hashes have specific places in the database.

### 3. *Keystore Configuration*

Keys are used in Mobiliser to secure communication between hosts (HTTPS) and to encrypt sensitive information, for example, credit card data. By default, Mobiliser does not contain any keys; you must create them as part of the overall installation process.

### 4. *Creating the Keystore for Data Encryption*

For credit card payments, the default Mobiliser configuration uses asymmetric encryption to secure credit card and bank account information in the front end, and a dummy payment handler implementation in the back end to decrypt credit card payments.

### 5. *Provision Secure Element Keys for DIRECT Mode*

Each new secure element that is issued by the SAP Mobile Platform operator can be identified by a unique ID, and requires a specific keyset. The secure element unique ID is stored in a structure called Card Production Life Cycle (CPLC) data, which uniquely identifies each secure element and is stored into each secure element prior to configuration.

### 6. *Generate Private Keys Used by On-Device Charging*

By default, encrypting communications between the MER and the point of sale, on device charging requires two root keys—Mer Private chargeKey (MPcK) and Mer Private readKey (MPrK)—that are installed into each MER, and generate a specific and separate keyset for each merchant. The keys, which are 192 bits in size, are used by 3-DES algorithms (DESede/CBC/PKCS5Padding).

### 7. *Performance Considerations*

A standalone SAP Mobile Platform Server includes a default configuration that is appropriate for workloads that do not require high transaction rates. A server that is running with the default settings is generally limited to development and proof-of-concept scenarios.

### 8. *Business Logic Configuration*

All configuration options, such as Preferences and ConfigAdmin, handle different scenarios for Mobiliser components.

### 9. *Hashing Customer Credentials*

Any customer (consumer, merchant, agent, or system user) credentials are stored, in a hashed format, in MOB\_CUSTOMER\_CREDENTIALS. SAP Mobile Platform supports different hashing algorithms. The STR\_CREDENTIAL is always prefixed with the hashing algorithm in curly brackets, for example, {<HASH-ALGORITHM>}<HASHVALUE>.

### 10. *Virus Protection*

Antivirus software is one of the most important tools for safeguarding vital information and personal data from the daily onslaught of viruses and worms.

**See also**

- *Managing SAP Mobile Platform Server Features* on page 142

## Mobiliser Universal User

---

For Mobiliser to function properly, you must configure the universal user, using the required hashed credential, in the MOB\_CUSTOMERS-CREDENTIALS table of the database. The Web portal uses the universal user to authenticate the server. Therefore, you must set the password in hashed format in the database and in encrypted format in the portal configuration.

## Creating or Updating Hashed Password for the Universal User

---

There are several preconfigured values that you must change, for security reasons, after a fresh installation. You must set a new password for the universal user (customer ID 100) and configure the password in the configuration accordingly.

**Prerequisites**

To enable strong cryptography, download the Java Cryptography Extension (JCE) unlimited strength jurisdiction policy file from your JDK vendor and update the security policy JAR files within the JVM that is running SAP Mobile Platform Server.

**Task**

---

**Note:** The SQL command in step 6 is for an Adaptive Server<sup>®</sup> Enterprise database. For DB2 and Oracle database, substitute the appropriate SQL syntax.

---

**1. Navigate to:**

```
SMP_HOME\Server\tools\mobiliser
\com.sybase365.mobiliser.vanilla.cli-tools-5.1.3.RELEASE-
CLIPasswordEncoderClient.jar
```

**2. From a command line, execute:**

```
java -jar com.sybase365.mobiliser.vanilla.cli-
tools-5.1.3.RELEASE-CLIPasswordEncoderClient.jar
```

**3. Select the hashing method.****4. Enter the plain password.****5. May not be required, depending on the hashing method you selected, enter the salt.****6. In the database, update the hash value by running:**

```
UPDATE "MOB_CUSTOMERS_CREDENTIALS" SET STR_CREDENTIAL =
'<Hash Value>' WHERE ID_CUSTOMER = 100
```

- Update the creation date for both the universal user (customer ID 100) and sysmgr user (customer ID 106) with:

Database	States
ASE	UPDATE MOB_CUSTOMERS_CREDENTIALS SET DAT_CREATION = GETDATE() WHERE ID_CUSTOMER IN (100,106)
DB2	UPDATE MOB_CUSTOMERS_CREDENTIALS SET DAT_CREATION = CURRENT_TIMESTAMP WHERE ID_CUSTOMER IN (100,106)
Oracle	UPDATE MOB_CUSTOMERS_CREDENTIALS SET DAT_CREATION = SYSDATE WHERE ID_CUSTOMER IN (100,106)

## Creating or Updating the Encrypted Password for Preferences

The encrypted password hash for preferences is used by the portal, and that encrypted value is placed in the MOB\_PREFERENCES table in the database. The first hash is made from any plain text password, while the second (encrypted) hash is built from the chosen value for the first hash. Both hashes have specific places in the database.

### Prerequisites

To enable strong cryptography, download the Java Cryptography Extension (JCE) unlimited strength jurisdiction policy file from your JDK vendor and update the security policy JAR files within the JVM that is running SAP Mobile Platform Server.

---

**Note:** Disabling or enabling portal through the console overrides the any value that is set previously.

---

### Task

- Navigate to:

```
SMP_HOME\Server\tools\mobiliser
\com.sybase365.mobiliser.vanilla.cli-tools-5.1.3.RELEASE-
CLIEncrypterClient.jar
```

- From a command line, execute:

```
java -jar com.sybase365.mobiliser.vanilla.cli-tools-
<version>-CLIEncrypterClient.jar <key> <value>
```

- Configure **<key>** in *SMP\_HOME*\webapps\portal\META-INF\context.xml as the value for the environment element with name prefs/secret.

```
<Environment description="The prefs secret key"
  name="prefs/secret" type="java.lang.String"
  value="paybox" />
```

```
<Environment description="The directory that hosts this file"
```

```
name="logging/baseDir" type="java.lang.String"
value="./webapps/portal/META-INF" />
```

where **<key>** represents the decryption key that is used by the Web portals to decrypt data coming from the Preferences service.

---

**Warning!** Do not choose a key at random. The key you enter must be identical to the one used by the Web portals to decrypt the data from Preferences; otherwise, the portals cannot connect to Mobiliser.

---

**<value>** represents the clear text password used when creating the hashed password for the universal user.

4. After you have successfully created the encrypted value, update the database with the new preferences:

```
UPDATE MOB_PREFERENCES SET STR_VALUE = '{AES-128-
PBKDF2}<Hash Value>' WHERE STR_NAME = 'mobiliser.password'
AND ( STR_PATH = '/presentationlayer/system/com/sybase365/
mobiliser/web/util/DynamicServiceConfiguration/' OR
STR_PATH = '/presentationlayer/system/com/sybase365/
mobiliser/util/tools/wicketutils/services/Configuration/')
```

## Encryption and Keystore Configuration

---

Encryption protects data within Mobiliser from unauthorized access using methods and a key or keys to encode plain text into a unreadable ciphertext. A key is required to decrypt the encrypted information and make it human-readable again. Keystores securely store and manage encryption keys.

### Encryption in MOB\_PREFERENCES

Preference configuration values are stored encrypted in the MOB\_PREFERENCES table.

Prefix encrypted preferences values with the encryption algorithm, such as:

- {AES-128-PBKDF2}<ENCRYPTED-VALUE>
- {AES-256-PBKDF2}<ENCRYPTED-VALUE >

Decryption is transparent to an application; however, the developer who is using a particular preferences node must explicitly enable encryption-support for that node explicitly.

For Mobiliser, the encryption/decryption key is configured in:

```
SMP_HOME\Server\config_master
\com.sybase365.mobiliser.util.prefs.encryption.aes
\com.sybase365.mobiliser.util.prefs.encryption.aes.properties
```

For applications using remote access to preferences, configure the encryption/decryption key using one of these methods:

- **System property** – –  
Dcom.sybase365.mobiliser.money.prefs.secret=<KEY>
- **JNDI entry** – <Environment description="Preferences key"  
name="prefs/secret" type="java.lang.String" value="<KEY>" /  
>  
  
The JNDI entry is usually configured in <TOMCAT\_HOME>/conf/server.xml.
- **Property file on class path** – sybase-preferences.properties  
**with line:** encryption-secret=<KEY>

The AES/CBC/PKCS5Padding encryption is automatically used. The encrypted value must be Base64-encoded, and the first 16 bytes are interpreted as the initialization vector (IV). The encryption key is derived from the password using PBKDF2HmacWithSHA1 hashing with the static salt {97,101,105,111,117,85,79,73,69} and 65536 iterations. SAP Mobile Platform Server includes an executable JAR in the ./tools folder that encrypts configuration values according to this specification.

Run:

```
./tools> java -jar SMP_HOME\Server\tolls\mobiliser  
\com.sybase365.mobiliser.vanilla.cli-tools-5.1.0.RELEASE-  
CLIEncrypterClient.jar  
<KEY> <TEXT> [<KEYLENGTH>]
```

---

**Note:** In SAP Mobile Platform installations, the encryption tool requires the installation of X-Windows in the system environment to execute properly; however, for Mobiliser Platform 5.1 SP01 and later installations, you can run the encryption tool without X-Windows capability. The <KEY> must match the configured key from one of the configuration places listed above. <KEYLENGTH> is optional and defaults to 128 bits; 256 works only if you have updated your Java encryption policy file.

Alternatively, once your system is up and running you can also log in to the Operations Dashboard to change the preferences through the portal. Use the same encryption key there as well.

---

### See also

- *Creating the Keystore for Data Encryption* on page 271

## Keystore Configuration

Keys are used in Mobiliser to secure communication between hosts (HTTPS) and to encrypt sensitive information, for example, credit card data. By default, Mobiliser does not contain any keys; you must create them as part of the overall installation process.

### *SecurityConfigProvider*

- **Preferences node:** – /businesslayer/com/sap/odc/core/security/manager/SecurityConfigProvider/
- **ConfigAdmin PID:** –  
com.sap.odc.core.security.manager.SecurityConfigProvider

You can set these configuration options:

Key	Description
key.se.password	Defines the password for the private key that was created with the mobiliser_odc_se_ks alias.
key.signing.password	Defines the password for the private key that was created with the mobiliser_odc_signing alias.
key.store.password	Defines the password for the keystore mobiliser.jks file.

### *RsaPublicKeyLogicImpl*

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/money/businesslogic/system/impl/RsaPublicKeyLogicImpl/
- **ConfigAdmin PID:** –  
com.sybase365.mobiliser.money.businesslogic.system.impl.RsaPublicKeyLogicImpl

You can set this configuration option:

Key	Description
key.store.password	Defines the password for the keystore mobiliser.jks file.

### *RsaPrivateKeyLogicImpl*

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/money/businesslogic/system/impl/RsaPrivateKeyLogicImpl/

- **ConfigAdmin PID:** –  
`com.sybase365.mobiliser.money.businesslogic.system.impl.Rs  
 aPrivateKeyLogicImpl`

You can set these configuration options:

Key	Description
key.store.pass- word	Defines the password for the keystore mobiliser.jks file.
key.mobilis- er_card.password	Defines the password for the private key that was created with the mobiliser_card alias.
key.mobilis- er_bank.password	Defines the password for the private key that was created with the mobiliser_bank alias.

### *DummyCardPaymentHandler*

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/  
 money/businesslogic/payment/handlers/card/impl/  
 DummyCardPaymentHandler/
- **ConfigAdmin PID:** –  
`com.sybase365.mobiliser.money.businesslogic.payment.handle  
 rs.card.impl.DummyCardPaymentHandler`

You can set these configuration options:

Key	Description
key.store.pass- word	The password for the keystore mobiliser.jks file.
key.password	The password for the private key that was created with the mobiliser_card alias.

### *AbstractCardPaymentHandler*

Sensitive payment instrument data, such as credit card and bank account numbers, are stored encrypted in the database. The AbstractCardPaymentHandler provides a function to decrypt the credit card number with a key that is provided in a key store. All relevant configurations are stored in preferences.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/  
 money/businesslogic/payment/handlers/configuration/  
 SecurityConfiguration/AbstractCardPaymentHandler
- **ConfigAdmin PID:** –  
`com.sybase365.mobiliser.money.businesslogic.payment.handle  
 rs.configuration.SecurityConfiguration.AbstractCardPayment  
 Handler`



You can set these configuration options:

Key	Default	Description
key.store		Indicates the full path and name to the keystore (either on the file system or Java classpath).
key.store.type	JCEKS	Defines the type of keystore.
key.store.password		Indicates the password of the keystore – usually identical to the key.password.
key.alias		Indicates the name of the key in the keystore to use for the decryption.
key.password		Indicates the password of the key in the keystore - usually identical to the key.store.password.

### See also

- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320
- *Encrypting Preferences Using Operations Dashboard* on page 273
- *Provision Secure Element Keys for DIRECT Mode* on page 274
- *Creating the Keystore for Data Encryption* on page 271

## Creating the Keystore for Data Encryption

For credit card payments, the default Mobiliser configuration uses asymmetric encryption to secure credit card and bank account information in the front end, and a dummy payment handler implementation in the back end to decrypt credit card payments.

Execute all operations as the **sybase** user and be sure to note all passwords for later reference.

1. Log in to SAP Mobile Platform Server.

2. Create a new directory:

```
mkdir -p SMP_HOME\Server\configuration
\com.sap.mobile.platform.server.mobiliser.core\keys
```

3. Change to the new directory:

```
cd SMP_HOME\Server\configuration
\com.sap.mobile.platform.server.mobiliser.core\keys
```

4. Generate a new keystore and key and modify the **dname** parameters as required:

```
keytool -genkey -validity 7305 -keystore mobiliser.jks -
alias mobiliser_card -keysize 2048 -storepass changeit -
keypass changeit -keyalg RSA -dname "CN=Mobiliser Platform,
OU=System, O=Sybase, L=Raunheim, S=Hessen, C=DE"
```

---

**Note:** Use the same passwords for both the keystore and the key itself.

---

5. Export the mobiliser\_card public key using the keystore password entered in step 4:  
`keytool -export -alias mobiliser_card -file mobiliser_card.crt -keystore mobiliser.jks`
6. Import the mobiliser\_card certificate in the new keystore and change the keystore password:  
`keytool -import -alias mobiliser_card -file mobiliser_card.crt -keystore mobiliser_pub.jks -storepass changeit`

---

**Note:** Use a different password than step 4.

---

7. Generate a new key in the same keystore generated in step 4:  
`keytool -genkey -validity 7305 -keystore mobiliser.jks -alias mobiliser_bank -keysize 2048 -storepass changeit -keypass changeit -keyalg RSA -dname "CN=Mobiliser Platform, OU=System, O=Sybase, L=Raunheim, S=Hessen, C=DE"`

---

**Note:** Use the same keystore password, but select a different key password.

---

8. Export the mobiliser\_bank public key using the keystore password entered in step 4.  
`keytool -export -alias mobiliser_bank -file mobiliser_bank.crt -keystore mobiliser.jks`
9. Import the mobiliser\_bank certificate into the keystore using the same keystore password entered in step 6:  
`keytool -import -alias mobiliser_bank -file mobiliser_bank.crt -keystore mobiliser_pub.jks -storepass changeit`
10. Generate a new key into the same keystore entered in step 4:  
`keytool -genkey -validity 7305 -keystore mobiliser.jks -alias mobiliser_odc_se_ks -keysize 2048 -storepass changeit -keypass changeit -keyalg RSA -dname "CN=Mobiliser Platform, OU=System, O=Sybase, L=Raunheim, S=Hessen, C=DE"`

---

**Note:** Use the same keystore password, but select a different key password.

---

11. Export the mobiliser\_odc\_se\_ks public key using the password entered in step 4:  
`keytool -export -alias mobiliser_odc_se_ks -file mobiliser_odc.crt -keystore mobiliser.jks`
12. Import the mobiliser\_odc\_se\_ks certificate into the keystore using the same keystore password entered in step 6:  
`keytool -import -alias mobiliser_odc_se_ks -file mobiliser_odc.crt -keystore mobiliser_pub.jks -storepass changeit`
13. Generate another key into the same keystore created in step 4:

```
keytool -genkey -validity 7305 -keystore mobiliser.jks -
alias mobiliser_odc_signing -keysize 1024 -storepass
changeit -keypass changeit -keyalg RSA -dname "CN=Mobiliser
Platform, OU=System, O=Sybase, L=Raunheim, S=Hessen, C=DE"
```

---

**Note:** Use the same keystore password, but select a different key password.

---

- 14.** Export the mobiliser\_odc\_signing public key using the password entered in step 4:

```
keytool -export -alias mobiliser_odc_signing -file
mobiliser_odc.crt -keystore mobiliser.jks
```

- 15.** Import the mobiliser\_odc\_signing certificate into the keystore using the same keystore password entered in step 6:

```
keytool -import -alias mobiliser_odc_signing -file
mobiliser_odc.crt -keystore mobiliser_pub.jks -storepass
changeit
```

- 16.** Change the access privileges for the keystore that contains the keys:

```
chmod 0600 mobiliser.jks
```

### Next

The public keys are loaded from the Web portals via a Web service call. Therefore, you must use the Operations Dashboard to configure passwords.

### See also

- *Encryption in MOB\_PREFERENCES* on page 267
- *Provision Secure Element Keys for DIRECT Mode* on page 274
- *Keystore Configuration* on page 269

## Encrypting Preferences Using Operations Dashboard

After installation, you might need to change some application configurations, such as entering an encrypted value for passwords.

1. Log in to the Operations Dashboard at <https://localhost:8081/portal> as the opsmgr user.
2. In the left pane, select **Preferences**.
3. Navigate to the node.
4. Do one of the following:
  - If a preference does not exist, click **Add a Preference** and enter all required information.
  - For an existing preference, click **Edit** in the Actions column.
5. Select **AES-128-PBKDF2** from the Save With Encryption list.
6. Enter a passphrase.

The screenshot shows the 'Edit a Preference' form in the SAP Mobile Platform Administration console. The form is titled 'Preferences » Node » Edit a Preference'. It contains the following fields:

- Application: businesslayer
- Node Full Path: /com/sybase365/mobiliser/money/businesslogic/system/impl/RsaPrivateKeyLogicImpl
- Key: key.mobiliser\_bank\_password
- Value: {AES-128-PBKDF2}3RfpcvPUI9!
- Type: java.lang.String
- Description: password for the keystore
- Save With Encryption: AES-128-PBKDF2
- Using Passphrase: secret

At the bottom of the form, there are two buttons: 'Save' (orange) and 'Cancel' (blue).

7. Click **Save**.

8. Define the passphrase as the default in these files:

- **businesslayer** – `SMP_HOME\Server\config_master\com.sybase365.mobiliser.util.prefs.encryption.aes\com.sybase365.mobiliser.util.prefs.encryption.aes.properties`
- **presentationlayer** – `SMP_HOME\Server\webapps\portal\META-INF\context.xml`

### See also

- *Adding a Preference Node* on page 320
- *Node and System Preferences* on page 319
- *Keystore Configuration* on page 269

## On-Device Charging Installation and Configuration

On-device charging provides the capability to store sensitive data, such as stored-value account (SVA) balances, on a smartphone, which can interact with external systems through near-field communication.

### Provision Secure Element Keys for DIRECT Mode

Each new secure element that is issued by the SAP Mobile Platform operator can be identified by a unique ID, and requires a specific keyset. The secure element unique ID is stored in a structure called Card Production Life Cycle (CPLC) data, which uniquely identifies each secure element and is stored into each secure element prior to configuration.

The association between the secure element and the unique keyset is usually provided by the secure element manufacturer, and is generally required by the secure element issuer, to maximize security. The card issuer would be taking a large risk if all secure element accesses were based on a single keyset.

To deploy the MER on secure element in DIRECT mode, on device charging needs the CPLC/keyset pair for each secure element. Each pair is stored in the on device charging table ODC\_DIRECT\_SE\_INFO.

The secure element manufacturer generally includes an additional CSV file that contains all secure element information (CPLC/keyset pair) with each secure element batch. On device charging registers the CSV files using:

```
### CPLC_data; keyVersionNumber; keyIdentifier; key1_type;
key1_value ; key2_type; key2_value; key3_type; key3_value ###
CPLC_data: a string containing hexadecimal numbers.
keyVersionNumber: integer, a technical number provided by the party
who performs the keys installation at personalization phase.
keyIdentifier: integer, a technical number provided by the party who
performs the keys installation at personalization phase.
keyX_type: one of the following strings ("senc", "smac", "dek").
keyX_value: a string containing hexadecimal numbers.
```

Example of line:

```
2A4790502116716320431790159B5EE10C664647926242167362571674627200000
00000000000000000000,42,0,senc,
25C69649A518622044BF1915BF65F7AB7737CF11B26EA506F5DE6163B08CB876E1E
D5DE0D3BF457F6418D321BAD62AEDEA6220423A7FC87C08C0F71748CFF2CA20EDC2
9AECC6879C951D26861305F37FE218288DA46A11D28B28603A8B0A6263686DDD19E
4B894F31E2758E8EA53EBC0EE7703A8565F87A5C90DE6B8201471AB3287B5A0477A
9326A66CA390182BF794D6877D49283C168CC9EB2D44D63A8D5328D418F9BAADA7F
5AA88E04665990927411AAFE13A84290783DC2C21BDED9751BB512592014C42C4E0
CFFDD552D41E1493E013D1F7C7DA03BA70E799D80A8CF9A4AA13DF1A31173330B4F
20FC8BCBB5D311FD5A9B7C9F418B0E81A591DD37229,smac,
62ED9BAC8A66493F89B6BB6C5079A2ED1F69646B98FCC49AECCFF76119C323AAD20
0C1BFAD210E7E0EBFD4D0BCC5E86BB26F5092942AD3E2004AA87632776D27DF3AC7
1E5C944E423982C5B8541E0B1E62EDC94D391E8FE1D31F226A450F9556F806DDDDC
6FEE4E2B9EE6DD9F96A822348F537417451C9B235D3B8E369D81B01C8DEDEBCB9638
0603B5E976468878B6EFF1C33D67E5323578B592ABEACBFC30E956E1233ABCA608B
6A0EF09861D2BB6943DA615ECCEFC95CCE2F423706C9F5FBFAFA206377C589D57F2D
01C66CBE6D1065365A14170BC25B56E4C45D930D623D82AFFB57F20B7A7CEDA0E47
318829382C492949FFB1454A12060F2AB467B0F349D,dek,
55E2A101A100F498ECB7C72563FE712CDD00682F174CA0C0BD4214FBAE9AA949A1F
DF807AD78CBF6F887C61320729FC7FF3CBEC696E32BBAE28B43830B9883E108D04
DFFE359D8152151D38E94CECA041C14C1C91D5C77850CD18EC753BF49C326ACAAAF0
24D34D5F9979CF8BC37D1EE3BDBD4EAE66C9BF8022A8CD78C5E73B2BF2C04764C86
203B989E90BCB7C1C8CD2F78F8B6580C3C669B4D544D4367C6D465AC0D456F10665
4942E75BF216746284ACC3A14C6F60ABB721814AA6DBA9B4F2BCC772379EB02EEB8
3DADC899182EC825CC8F0FA1932E235AAA3979D54C0721DF9A3837B7AB0DFAB6AA1
A4C864F1FF566758F22CEA42F2B6FF94016F21AF2B3
```

To guarantee the security, all the key values will have to be ciphered by the secure element manufacturer with the public key stored in the mobiliser\_odc.crt file.

The ciphering algorithm that must be used by the secure element manufacturer is RSA/ECB/PKCS1Padding.

To install this file into the ODC\_DIRECT\_SE\_INFO table:

1. Uncompress the `SMP_HOME\Server\tools\mobiliser\com.sap.odc.tool.security.odckeytool-1.0.0.RC7-dist.zip` file.
2. Change directory to `com.sap.odc.tool.security.odckeytool-1.0.0.RC7\com.sap.odc.tool.security.odckeytool-1.0.0.RC7.jar`.
3. Execute:

```
java -jar com.sap.odc.tool.security.odckeytool-1.0.0.RELEASE.jar
populate_se_info -url < smp_server_url> -login
< mobiliser_user_login> -passwd <passwd> -csv <csvFilePath>]
```

where `< mobiliser_user_login>` and `<passwd>` authenticates the user against Mobiliser after having successfully passed the SAP Mobile Platform HTTP gateway.

---

**Note:** This command can be executed each time a new secure element batch file is registered. At last, this command manages the doubloons.

---

### See also

- *Keystore Configuration* on page 269
- *Creating the Keystore for Data Encryption* on page 271

## Generate Private Keys Used by On-Device Charging

By default, encrypting communications between the MER and the point of sale, on device charging requires two root keys—Mer Private chargeKey (MPcK) and Mer Private readKey (MPrK)—that are installed into each MER, and generate a specific and separate keyset for each merchant. The keys, which are 192 bits in size, are used by 3-DES algorithms (DESede/CBC/PKCS5Padding).

In addition, on device charging requires an addition MPsK key for signing the transactions and producing an eToken (a signed transaction). By default, the encryption algorithm used by on device charging/MER for signing the generated transactions is RSA/ECB/PKCS1Padding.

However, the user can alternatively switch on a 3-DES algorithm (DESede/CBC/PKCS5Padding) to generate a smaller signature size and, thus, increase the number of eTokens that can be stored into the secure element. You must install the Bouncycastle package on the server side for verifying the eTokens.

To generate the required keys:

1. Uncompress the `SMP_HOME\Server\tools\mobiliser\com.sap.odc.tool.security.odckeytool-1.0.0.RC7-dist.zip` file.
2. Change directory to `com.sap.odc.tool.security.odckeytool-1.0.0.RC7\com.sap.odc.tool.security.odckeytool-1.0.0.RC7.jar` .

**3. Execute:**

```
java -jar com.sap.odc.tool.security.odckeytool-1.0.0.RELEASE.jar
gen_odc_keys -url < smp_server_url> -login < mobiliser_user_login>
-passwd <passwd> [-desSigning]
```

where *< mobiliser\_user\_login>* and *<passwd>* authenticates the user against SAP Mobile Platform after having successfully passed the SAP Mobile Platform HTTP gateway.

---

**Note:** If you specify the **-desSigning** option, the eToken signing process uses 3-DES instead of the default RSA signing algorithm.

---

This command:

- Automatically generates all the keys used by 3-DES algorithms,
- Reads the private RSA signing key from the SAP Mobile Platform key store,
- Encrypts all those private keys by using the *mobiliser\_odc\_se\_ks* private key, and
- Stores them into the security keyset database.

You can generate keys only once, during installation. Attempting to generate on device charging private keys multiple times prevents the deployed MER from communicating with the existing registered merchant point of sale, and also prevents existing customers from using on device charging with new merchants.

## Performance Considerations

---

A standalone SAP Mobile Platform Server includes a default configuration that is appropriate for workloads that do not require high transaction rates. A server that is running with the default settings is generally limited to development and proof-of-concept scenarios.

By making a few minor adjustments, you can scale the server to your available hardware resources, providing high-transactional throughput with low-latency tolerances. Use the performance properties to configure the most significant impact performance improvement. The recommended values are provided in units as a factor of the number of available CPU cores on the host running the server. Most scenarios require the same number of CPU cores on the host running the database server.

## Event Handling Threads

---

Event generation and processing is an integral part of the underlying Mobiliser framework.

### *Thread Pool*

The default values for the number of event-handling threads can create a situation where the events generated during a transaction are processed after the transaction. Using the default values can reduce database load to a minor degree, but if a server restart is necessary, might also require regenerative event processing. You can largely avoid this effect by using the built-in autodetect logic when sizing the event-handling thread pools. In a high-throughput scenario, using this logic generally leads to more consistent load patterns.

## Application Services (Mobiliser) Administration

Preferences path: `SMP_HOME\Server\config_master  
\com.sybase365.mobiliser.framework.event.core.properties`

Requires server restart to take effect: Yes

You can set these configuration options:

Variable	Default Value	Modified Value
<code>regeneration.batch.size</code>	10	1024 * CPU cores
<code>delayedq.capacity</code>	1000	65536 * CPU cores
<code>processq.capacity</code>	1000	65536 * CPU cores
<code>catchupq.capacity</code>	1000	65536 * CPU cores

### *Handler*

Each event handler has two private variables that you can set as needed.

Configuration location: Multiple locations

Requires server restart to take effect: No

You can set these configuration options in preferences:

Variable	Default Value	Modified Value
<code>event.handler.maxActive</code>	Varies	-1 (infinity)
<code>event.handler.maxIdle</code>	Varies	-1 (infinity)

## **Business Logic Configuration**

---

All configuration options, such as Preferences and ConfigAdmin, handle different scenarios for Mobiliser components.

The ConfigAdmin persistent identifier (PID) relates to the configuration file of the same name in: `SMP_HOME\Server\config_master\`.

## **Framework**

To configure the Mobiliser Framework, use the ConfigAdmin option. Individual configuration options are described in the following sections.

### **Gateway**

The gateway provides common interfaces to define services such as Java Management Extension (JMX) authentication, security filters, exception mapping, Open Data Protocol (OData), and Transmission Control Protocol (TCP).



Java Management Extension Authentication

Java Management Extension (JMX) authentication configures an external listener for JMX connections into the framework.

- **Preferences node:** – /businesslayer/com/sap/mobile/platform/server/foundation/security/jmx
- **ConfigAdmin PID:** – com.sap.mobile.platform.server.foundation.security.jmx

The JMX authentication has the following configuration options:

Key	Default	Description
jmxPort	1717	Defines the port to listen on for external JMX connections.
objectName	connector:name= rmi	Defines the object name under which this connector is registered as an MBean with the server.
serviceUrl	service:jmx:rmi:// 127.0.0.1:1717/jndi/ rmi:// 127.0.0.1:1717/jmxrmi	Defines the URL that clients use to connect. Ensure that this port matches the JMX port.  <b>Note:</b> The parameters are set in two locations that check which service URL to use.
initialContextFactory	com.sun.jndi.rmi. registry. RegistryContext Factory	Indicates which context factory to use.
exportInitialContext- Factory	true	Defines whether to export the used initial context factory to the OSGi registry.
requiredRole	JMX_ACCESS	Defines the privilege required for access.
ssl	false	Defines whether to use an SslRmiServerSocketFactory. If this is true, you must also configure the keystore and truststore through the standard system properties: javax.net.ssl.

**See also**

- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320

Standard Security Filters

Configures the standard security filters for Mobiliser.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/framework/gateway/security/filters/standard
- **ConfigAdmin PID:** – com.sybase365.mobiliser.framework.gateway.security.filters.standard

Key	Default	Description
ehCache-BasedUserCache.location	<code>SMP_HOME\Server\config_master\com.sybase365.mobiliser.framework.gateway.security.filters.standard\com.sybase365.mobiliser.framework.gateway.security.filters.standard.properties</code>	Ehcache configuration location for the user details cache.
osgiProviderManager.eraseCredentialsAfterAuthentication	context	Defines whether Spring security removes the credentials from the authentication object after successful authentication. To upgrade password hashing algorithms, set this key to false, since the password is needed to update the hash.
matcher-Mode	standard	Sets the HTTP path expressions. When matching HTTP paths for security expressions, Spring security normally uses the request path built by <code>request.getServletPath() + request.getPathInfo()</code> . For some environments, build the path with <code>request.getContextPath() + request.getPathInfo()</code> . If the request path has been built using the context path, set this key to "context".

Key	Default	Description
baseUrl	/mobiliser	<p>Sets the base URL for the security configurations picked up from the OSGi registry, which can be relative or absolute. Relative configurations do not begin with a slash (/); therefore, the base URL configured as relative is prepended to the configuration before the HTTP path expression is configured. This should match the servlet name configured in the PID:</p> <pre>com.sybase365.mobiliser.framework.gateway.httpservice</pre>
realm-Name	MOBILISER	<p>Defines the realm name for the unauthorized response header. If the server receives a request for an access-protected object, and the request is denied, the server responds with a 401 response code and a "WWW-Authenticate" header.</p>
channel	any	<p>Defines the channel of the default security configuration for the servlet:</p> <ul style="list-style-type: none"> <li>• any</li> <li>• https</li> <li>• http</li> </ul> <p>You can override the default by providing specific configurations elsewhere within the container.</p>
roles	MOBILISER_ACCESS	<p>Defines the roles of the default security configuration for the servlet, which is a comma-separated list and uses an OR expression. You can override the default by providing specific configurations elsewhere within the container.</p>

Key	Default	Description
port_mapping_xxx		Defines the mapping between secure and insecure ports. If a channel is set to something other than “any,” whether with the default or other specific configuration, Spring security must know the mapping between secure and insecure ports to properly send the client a 302 response with a “Location” header. You may have any number of these configurations to specify the mappings between these ports. If you are using nonstandard ports in jetty.xml for your connectors, configure the ports here.

SAP Mobile Platform uses a `UserDetailsCache` class to obtain the user details that are used during authentication and authorization, which is, in a standard setup, located in:

```
SMP_HOME\Server\configuration
\com.sap.mobile.platform.server.mobiliser.core\userdetails-ehcache.xml
```

The default EhCache configuration for the `UserDetailsCache` looks like:

```
<?xml version="1.0" encoding="UTF-8"?>
<ehcache xsi:noNamespaceSchemaLocation="http://ehcache.org/ehcache.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<defaultCache
timeToLiveSeconds="0" timeToIdleSeconds="600"
memoryStoreEvictionPolicy="LRU"
overflowToDisk="false" eternal="false" maxElementsInMemory="50"/>
<cache timeToLiveSeconds="5" timeToIdleSeconds="5"
memoryStoreEvictionPolicy="LRU"
overflowToDisk="false" eternal="false" maxElementsInMemory="100"
name="userDetailsCache"/> </ehcache>
```

### See also

- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320

### Exception Mapping

Exceptions thrown by endpoints are automatically caught and mapped to error codes, which are set in the `MobiliserResponse` object. The base `MobiliserServiceException` has an implicit error code field that is used for this mapping. Other exceptions are mapped to a constant value, or are configured in this file to those values. To avoid mapping an exception (allowing it to pass), add a mapping to the configuration file that maps it to nothing.

ConfigAdmin PID: `com.sybase365.mobiliser.framework.service.api`

You can set these configuration options, which can be updated at runtime:

```

1
2# Simple mapping from exception class to error code
3# you can list parent classes here and subclasses
4# will be mapped to the error code of the parent
5
6# you can also add entries and map no number which will
7# cause the exception not to be mapped, but propagated
8
9 org.springframework.dao.DataAccessException=9935
10 org.springframework.dao.DuplicateKeyException=9930
11 org.springframework.dao.DataIntegrityViolationException=9931
12 org.springframework.jdbc.BadSqlGrammarException=9932
13 org.springframework.orm.ObjectRetrievalFailureException=9933
14 org.springframework.transaction.TransactionException=9935
15
16 org.springframework.security.access.AccessDeniedException=

```

Open Data Protocol Interface

This configuration defines the open data protocol (OData) interface details.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/framework/gateway/odate
- **ConfigAdmin PID:** – com.sybase365.mobiliser.framework.gateway.odata

You can set these configuration options:

Key	Default	Description
mode	standard	Sets the OData interface mode either as standard or proxy. Proxy is used only when deploying into the validating proxy server.
ignore_context_regex	^\/?(prefs  management)\$	Indicates the regular expression to be ignored. Any context matching this regular expression is not added to the OData interface.
path_match_ctx		Creates a new OData context from any configuration option with the prefix path_match_xxx. The value assigned is a regular expression, which is matched against contexts in the server. If they match, this endpoint is sorted into this OData context.  Example: path_match_custom=^(bank club)\$  Any context matching "bank" or "club" is sorted into the OData context called "custom". You can have any number of these mappings.

Key	Default	Description
path_privileges_ctx		Indicates a comma-separated list of privileges, which are required to access this OData context for any endpoints sorted into the context <code>ctx</code> .  Example: <code>path_privileges_ctx=MY_SPECIAL_PRIVILEGE</code>
path_ports_ctx		Indicates a comma-separated list of ports, which must be used to access this OData context for any endpoints sorted into the context <code>ctx</code> .  Example: <code>path_privileges_ctx=7070,8080</code>
default_context	odata	Sorts the endpoint into the default OData context if no matching <code>path_match_xxx</code> configuration is found.

**See also**

- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320

**Transmission Control Protocol Interface**

This configuration defines the transmission control protocol (TCP) interface to Mobiliser.

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/framework/gateway/tcp`
- **ConfigAdmin PID:** – `com.sybase365.mobiliser.framework.gateway.tcp`

You can set these configuration options:

Key	Default	Description
port	8088	Defines the port to use for the TCP socket.
replyTimeout	10000	Defines the time, in milliseconds, for which the gateway waits for a reply.
useNio	false	Indicates whether or not the connection uses New I/O (NIO).
applySequence	false	Sequences messages when using NIO. If attribute is true, <code>correlationId</code> and <code>sequenceNumber</code> headers are added to messages received.

Key	Default	Description
soTimeout	0	Defaults to 0 (infinity), except for server connection factories with single-use="true," in which case, it defaults to the default reply timeout.
soSendBufferSize	0	See java.net.Socket.setSendBufferSize(), located at: <a href="http://docs.oracle.com/javase/6/docs/api/java/net/Socket.html#setSendBufferSize%28int%29">http://docs.oracle.com/javase/6/docs/api/java/net/Socket.html#setSendBufferSize%28int%29</a>
soReceiveBufferSize	0	See java.net.Socket.setReceiveBufferSize(), located at: <a href="http://docs.oracle.com/javase/6/docs/api/java/net/Socket.html#setReceiveBufferSize%28int%29">http://docs.oracle.com/javase/6/docs/api/java/net/Socket.html#setReceiveBufferSize%28int%29</a>
soTcpNoDelay	false	See java.net.Socket.setTcpNoDelay(), located at: <a href="http://docs.oracle.com/javase/6/docs/api/java/net/Socket.html#setTcpNoDelay%28boolean%29">http://docs.oracle.com/javase/6/docs/api/java/net/Socket.html#setTcpNoDelay%28boolean%29</a>
lookupHost	false	Specifies whether reverse lookups are performed on IP addresses to convert to host names for use in message headers. If false, the IP address is used instead of the host name.
poolSize	false	This attribute is deprecated. For backward compatibility, users should specify the connection backlog in server factories.
singleUse	false	Specifies whether a connection can be used for multiple messages. If true, a new connection is used for each message.
localAddress		Specifies an IP address for the interface to which the socket is bound on a multi-homed system.
taskSchedulerPoolSize	10	Defines the size of the task scheduler used by Spring Integration.
maxMessageSize	10240	Defines the maximum size allowed for a message; bridges TCP and Spring Web Services.
endpointPathFilter	/smartphone, /spmmbanking	Indicates a comma-separated list of context paths that are not considered for use with the TCP interface.

Key	Default	Description
ip_XXX		Specifies any number of options with the prefix ip_ followed by an IP address matcher. Map this key to a valid user in the system. The call is authenticated to this user. Access to the port from a particular host must be handled outside the TCP interface since it does no other checking that considering a host as being preauthenticated with a particular user, such as: <code>ip_ 10.0.0.0/8=mobiliser</code>

The configuration options map directly to the IP endpoint configuration in Spring integration:

<http://static.springsource.org/spring-integration/reference/htmlsingle/#ip-endpoint-reference>

### See also

- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320

### **Hibernate**

Configuration defines the properties for the Hibernate Session Factory. Runtime updates to these values are ignored.

ConfigAdmin PID:

```
com.sybase365.mobiliser.framework.persistence.hibernate.sessionfactory
```

The options for Hibernate configuration come directly from the Hibernate configuration documentation: <http://docs.jboss.org/hibernate/orm/3.3/reference/en/html/session-configuration.html>

You can set this configuration option:

Key	Default	Description
ehCacheConfigura- tion	<code>./configuration/ com.sap.mo- bile.platform.serv- er.mobiliser.core/ mob-ehcache.xml</code>	Defines the location of the cache configura- tion file.



**Java Database Connectivity - BoneCP**

Currently, the BoneCP bundle has been configured.

ConfigAdmin PID:

```
com.sybase365.mobiliser.framework.persistence.jdbc.bonecp.pool
```

The options for this file come directly from the BoneCP configuration documentation:

<http://jolbox.com/bonecp/downloads/site/apidocs/index.html?com/jolbox/bonecp/BoneCPConfig.html>

**Messaging**

Manage all message configuration data through the preferences option in the Operations Dashboard.

**Engine**

Use the preferences option in the Operations Dashboard to manage engine configuration data.

*QueueBasedFutureFactory*

The QueueBasedFutureFactory manages responses in synchronous channel implementations.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/engine/impl/QueueBasedFutureFactory
- **ConfigAdmin PID:** – com.sybase365.mobiliser.util.messaging.channelmanager.engine.impl.QueueBasedFutureFactory

You can set these configuration options:

Key	Default	Description
synchronousPollTimeout	60000	Sets the amount of time, in milliseconds, the queued thread waits for an answer before returning null. You can update this value at runtime.
synchronousOfferTimeout	60000	Sets the amount of time, in milliseconds, the response thread waits while trying to place the response on the queue before giving up. You can update this value at runtime.

*PickupJmsMessages*

The PickupJmsMessages picks up messages from configured JMS queues and forwards them to ChannelManager to be sent through a channel implementation. This links the JMS feature with the OSGi ChannelManager.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/engine/jmspickup/PickupJmsMessages
- **ConfigAdmin PID:** – com.sybase365.mobiliser.util.messaging.channelmanager.engine.jmspickup.PickupJmsMessages

You can set these configuration options:

Key	Default	Description
sessionCache-Size	1	Sets the number of sessions to cache. Runtime updates are ignored. The pickup class makes use of a CachingConnectionFactory. See the Javadoc at: <a href="http://static.springframework.org/spring/docs/3.0.x/javadoc-api/org/springframework/jms/connection/CachingConnectionFactory.html">http://static.springframework.org/spring/docs/3.0.x/javadoc-api/org/springframework/jms/connection/CachingConnectionFactory.html</a> .
maxConcurrentConsumers	1	Sets the number of consumers to create. Runtime updates affect only newly registered queues after the update. Listener containers that are already created are not destroyed and re-created. The pickup class makes use of a DefaultMessageListenerContainer. See the Javadoc at: <a href="http://static.springframework.org/spring/docs/3.0.x/javadoc-api/org/springframework/jms/listener/DefaultMessageListenerContainer.html">http://static.springframework.org/spring/docs/3.0.x/javadoc-api/org/springframework/jms/listener/DefaultMessageListenerContainer.html</a> .
mode	standard	Defines the mode for the picking up messages. For backward compatibility with older SMS Builder, such as Mobiler wizard, instances should set this value to <b>legacy</b> .

### JmsReceiveCallback

The JmsReceiveCallback forwards incoming messages to a JMS queues with the name of the destination supplied by the receiving channel. This links the JMS feature with the OSGi ChannelManager.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/engine/jmsreceiver/JmsReceiveCallback
- **ConfigAdmin PID:** – com.sybase365.mobiliser.util.messaging.channelmanager.engine.jmsreceiver.JmsReceiveCallback

You can set these configuration options:

Key	Default	Description
sessionCache-Size	1	Sets the number of sessions to cache. Runtime updates are ignored. The pickup class makes use of a CachingConnectionFactory. See the Javadoc at: <a href="http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/jms/connection/CachingConnectionFactory.html">http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/jms/connection/CachingConnectionFactory.html</a> .
mode	standard	Defines the mode for the picking up messages. For backward compatibility with older SMS Builder, such as Mobiler wizard, instances should set this value to <b>legacy</b> .

### *ActiveMQConnectionConfiguration*

The ActiveMQConnectionConfiguration configures a JMS connection factory that is used by some OSGi ChannelManager components.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/engine/jmsconnection/ActiveMQConnectionConfiguration
- **ConfigAdmin PID:** – com.sybase365.mobiliser.util.messaging.channelmanager.engine.jmsconnection.ActiveMQConnectionConfiguration

You can set these configuration options:

Key	Default	Description
brokerURL	tcp://localhost:61616	Defines the URL of the JMS broker to use for the connection.
startBroker	false	Indicates whether to start the broker. If true, the broker starts automatically.

### **See also**

- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320

### **Encryption**

The encryption classes allow for easy and common access to the Java Crypto API. All encryption values can be updated at runtime.

#### *AesEncryptionStrategy*

The AesEncryptionStrategy lets you encrypt mail jobs using the AES-256 algorithm.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/util/messaging/encryption/aes/AesEncryptionStrategy

- **ConfigAdmin PID:** –  
`com.sybase365.mobiliser.util.messaging.encryption.aes.AesEncryptionStrategy`

You can set this configuration option:

Key	Default	Description
secret		Indicates the passphrase to use with the AES-256 algorithm for encrypting mail jobs. Alternatively, you can update this value at runtime.

### *TripleDesEncryptionStrategy*

The TripleDesEncryptionStrategy lets you encrypt mail jobs using the 3-DES algorithm.

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/encryption/tripledes/TripleDesEncryptionStrategy`
- **ConfigAdmin PID:** –  
`com.sybase365.mobiliser.util.messaging.channelmanager.encryption.tripledes.TripleDesEncryptionStrategy`

You can set this configuration option:

Key	Default	Description
secret		Indicates the passphrase to use with the 3-DES algorithm for encrypting mail jobs. Alternatively, you can update this value at runtime.

## Logic

Use the preferences option in the Operations Dashboard to manage logic configuration data.

### *MessageLogicImpl*

The MessagingLogicImpl is the core business logic class in the message gateway.

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/util/messaging/logic/impl/MessagingLogicImpl`
- **ConfigAdmin PID:** –  
`com.sybase365.mobiliser.util.messaging.logic.impl.MessagingLogicImpl`

You can set these configuration options:

Key	Default	Description
defaultChannel	defaultQ	Allows clients of the message gateway to specify the name of the channel to use when sending a message. If no value is specified, the default channel ID is used. You can update this value at runtime.
disableEncryption	false	Overrides message encryption at the global level. Encrypted mail jobs are decrypted, but new jobs are written to the database in plain text. This can be useful during testing.

### *MessageDispatcher*

The MessageDispatcher polls the mail jobs table and dispatches messages to ChannelManager for sending.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/util/messaging/logic/dispatcher/MessageDispatcher
- **ConfigAdmin PID:** – com.sybase365.mobiliser.util.messaging.logic.dispatcher.MessageDispatcher

You can set these configuration options:

Key	Default	Description
interval	3000	Interval between runs of the dispatcher, in milliseconds. You can update this value at runtime.
batchSize	50	Maximum number of mail jobs to select for processing during a run. You can update this value at runtime.
maxInterval	60000	Maximum interval if no jobs are found to dispatch. You can update this value at runtime.
prefsBully-Name	GW-MES-SAGE	Name of the bully algorithm name used to prevent duplicate processing of mail jobs. You can override this value using this system property:  com.sybase365.mobiliser.util.messaging.logic.dispatcher.MessageDispatcher.bullyService  Runtime updates of the preference value or the system property are ignored.

**Template**

Use templates, which can be stored in the database, to define messages in a standard way and allow clients to simply provide placeholders to fill in required data. All template values can be updated at runtime.

*CharsetUtilsImpl*

The `CharsetUtilsImpl` is a utility class used by the template framework for choosing the best character set.

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/util/messaging/template/implCharsets/CharsetUtilsImpl`
- **ConfigAdmin PID:** – `com.sybase365.mobiliser.util.messaging.template.implCharsets.CharsetUtilsImpl`

You can set this configuration option:

Key	Default	Description
charsetList	us-ascii, iso8859-1, iso8859-15, utf-8	Indicates a list of character set to be checked in ascending order (simplest to most complex). When choosing a character set, the engine bases its selection on this list. This value can be updated at runtime.

*BinaryContentTypeHandler*

The `BinaryContentTypeHandler` processes binary attachments in templates.

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/util/messaging/template/implhandlers/BinaryContentTypeHandler`
- **ConfigAdmin PID:** – `com.sybase365.mobiliser.util.messaging.template.implhandlers.BinaryContentTypeHandler`

You can set this configuration option:

Key	Default	Description
defaultBinary-Types	application/octet-stream, application/pdf, application/zip, application/x-gzip, image/gif, image/jpeg, image/png, image/tiff	Indicates a list of mime-types for which this content type handler is responsible. Runtime updates are ignored.

**See also**

- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320

**Channels**

Use the preferences option in the Operations Dashboard to manage channel configuration data. You can set the values at runtime.

**EmailChannel**

The EmailChannel sends e-mail messages through SMTP.

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/engine/impl/ChannelInstantiator/emailchannel1`
- **ConfigAdmin PID:** – `com.sybase365.mobiliser.util.messaging.channelmanager.engine.impl.ChannelInstantiator.emailchannel1`

You can set these configuration options in preferences:

Key	Default	Description
channelId	defaultQ	Indicates the channel's ID.
mail.host	localhost	Defines the SMTP server host name.
mail.port	25	Defines the port to use when connecting to the SMTP server.
mail.username		Indicates whether the server authentication requires the user name.
mail.password		Indicates whether the server authentication requires the password.
mail.protocol	smtp	Defines the protocol used between the mail program and the server. Options are SMTP or Simple Mail Transfer Protocol Security (SMTPS).
mail.sign	false	Indicates whether outgoing e-mail messages are pretty good privacy (PGP) signed. All other configuration values are relevant only if this value is set to true.
sign.keyId		Indicates the ID of the secret key to use for signing.

Key	Default	Description
sign.hashAlgorithm	-1	Defines the hash algorithm to use when signing. If this value is not set, the default from the key itself is used. Otherwise, use this option to force a hash algorithm. For details, refer to the Javadoc located at: <a href="http://www.bouncycastle.org/docs/pgdocs1.50n/org/bouncycastle/bcpg/HashAlgorithm-Tags.html">http://www.bouncycastle.org/docs/pgdocs1.50n/org/bouncycastle/bcpg/HashAlgorithm-Tags.html</a>
sign.passphrase		Indicates the passphrase for the secret key.
sign.secretRingResource		Indicates the resource for the secret keyring in Spring notation. For example, file:/home/mobiliser/.gnupg/secring.gpg.

**Note:** The default channel for all outgoing messages is `htmlchannel1`. Once an outgoing message is sent by Mobiliser, the contents of the messages can be found at the following URL: <https://localhost:8081/mobiliser/channelmgr/html>, where you are asked for the Mobiliser mobiliser user before you can view the outgoing messages. To use the `emailchannel1` as the method for sending outgoing messages, you must first deactivate `htmlchannel1`. To deactivate it, set the "\_active" preference entry to false via the Operations Dashboard in: `/com/sybase365/mobiliser/util/messaging/channelmanager/engine/impl/ChannelInstantiator/htmlchannel1`.

### See also

- *HtmlChannel* on page 294

### HtmlChannel

The `HtmlChannel` is a test channel that shows outgoing messages in an HTML table under a specific URL.

- **Preferences node:** `- /businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/engine/impl/ChannelInstantiator/htmlchannel1`
- **ConfigAdmin PID:** `- com.sybase365.mobiliser.util.messaging.channelmanager.engine.impl.ChannelInstantiator.htmlchannel1`

You can set these configuration options:

Key	Default	Description
channelId	defaultQ	Indicates the channel's ID.
maxSize	100	Defines the number of messages held in memory.



Key	Default	Description
urlSupplement	html	Appends the supplement to the channel manager URL, making the HTML channel accessible.

### See also

- *EmailChannel* on page 293

### HttpChannelEnd

The `HttpChannelEnd` is a test synchronous channel to which you can submit messages through HTTP.

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/engine/impl/ChannelInstantiator/httpchannell`
- **ConfigAdmin PID:** – `com.sybase365.mobiliser.util.messaging.channelmanager.engine.impl.ChannelInstantiator.httpchannell`

You can set these configuration options:

Key	Default	Description
channelId	defaultQ	Indicates the channel's ID.
urlSupplement	http	Appends the supplement to the channel manager URL, making the HTTP channel accessible. It is a suffix on the URL for internal processing.
incomingChannelId		Indicates the destination channel ID that is used when this channel submits a message to <code>ChannelManager</code> .

### JabberChannel

The `JabberChannel` is a test channel through which you can send messages to a customer using Extensible Messaging and Presence Protocol (XMPP).

To register a mapping between a Jabber ID and a mobile phone number, the customer must send a message consisting of a backslash plus his or her phone number (`\+642222222`) to the SAP Mobile Platform customer. The customer specifies the short code to which the message should be sent by prefixing his or her text with the short code in brackets. Responses from the back end have the short code in brackets also preceding the text.

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/engine/impl/ChannelInstantiator/jabberchannell`

- **ConfigAdmin PID:** –  
`com.sybase365.mobiliser.util.messaging.channelmanager.engine.impl.ChannelInstantiator.jabberchannel1`

You can set these configuration options:

Key	Default	Description
channelId	defaultQ	Indicates the channel's ID.
urlSupplement	http	Appends the supplement to the channel manager URL, making the HTTP channel accessible. It is a suffix on the URL for internal processing.
incoming-ChannelId		Indicates the destination channel ID that is used when this channel submits a message to ChannelManager.
xmppHost		Indicates the host name of the XMPP server.
xmppPort		Indicates the port to use for the XMPP server.
xmppUsername		Indicates the Jabber user name, for example, mobiliser.
xmppPassword		Indicates the password of the Jabber user.
xmppService-Name		Indicates the host part of the Jabber user, for example, jabber.org.
welcomeText	Welcome! Please enter your MSISDN with a leading \\ (you can change it at any time)	Indicates the welcome message sent to a new Jabber ID.
instruction-Text	Please prefix your message with the short code in brackets, e.g. [234872] balance	Indicates the message instructing the Jabber user on how to communicate with the Jabber channel.

### SmppChannel

The SmppChannel sends or receives short message service (SMS) via short message peer-to-peer (SMPP).

Many configuration values require knowledge of SMPP. See [http://www.nowsms.com/discus/messages/1/SMPP\\_v3\\_4\\_Issue1\\_2-24857.pdf](http://www.nowsms.com/discus/messages/1/SMPP_v3_4_Issue1_2-24857.pdf).

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/engine/impl/ChannelInstantiator/httpchannel1`

- **ConfigAdmin PID:** –  
`com.sybase365.mobiliser.util.messaging.channelmanager.engine.impl.ChannelInstantiator.httpchannel1`

You can set these configuration options:

Key	Default	Description
channelId	defaultQ	Indicates the channel's ID.
urlSupplement	http	Appends the supplement to the channel manager URL, making the HTTP channel accessible. It is a suffix on the URL for internal processing.
incomingChannelId	brandInQueue	Indicates the destination channel ID that is used when this channel submits a message to ChannelManager.
sendAsDataSm	false	Sends SMS as a SUBMIT_SM. If set to true, then it sends SMS as DATA_SM.
registeredDelivery	0	Defines the value of the registered_delivery to use for outgoing SMS. See 5.2.17 of the SMPP Protocol Specification v3.4.
validityOffset		Defines the message validity period, in seconds, for an outgoing SMS. There is not default value, which delegates the validity offset responsibility to the defaulted defined in the Short Message Service Center (SMSC).
smscTimeZone		Defines the time zone for SMSC. If undefined, the SmpChannel assumes the same time zone as the JVM. When using validityOffset, the validity string must be computed using the time zone of the SMSC.
ussdManagerNoResponseTimeout	300000	Defines the amount of time, in milliseconds, before a Unstructured Supplementary Service Data (USSD) session is regarded as timed out.
sourceNumberFormatter	const	Defines the source number format for outgoing messages. Options include: <ul style="list-style-type: none"> <li>• international</li> <li>• national</li> <li>• const</li> <li>• numeric-international</li> <li>• short-international</li> </ul>

Key	Default	Description
destinationNumber-Formatter	const	Defines the destination number format for outgoing messages. Options include: <ul style="list-style-type: none"> <li>international</li> <li>national</li> <li>const</li> <li>numeric-international</li> <li>short-international</li> </ul>
srcTon	0	Defines the type of number (TON) value to use for the source mobile phone number. This should match the configured formatter. See 5.2.5 of the SMPP Protocol Specification v3.4.
destTon	0	Defines the TON value to use for the destination mobile phone number. This should match the configured formatter. See 5.2.5 of the SMPP Protocol Specification v3.4.
srcNpi	0	Defines the numbering plan identification (NPI) value to use for the source mobile phone number. This should match the configured formatter. See 5.2.6 of the SMPP Protocol Specification v3.4.
destNpi	0	Defines the NPI value to use for the destination mobile phone number. This should match the configured formatter. See 4.1 of the SMPP Protocol Specification v3.4.
bindType	transceiver	Defines the bind type to use when connecting to the SMSC. Options include: <ul style="list-style-type: none"> <li>transceiver</li> <li>transmitter</li> <li>receiver</li> <li>transmitterAndReceiver</li> </ul> <p>See 5.2.6 of the SMPP Protocol Specification v3.4. This setting affects all of the remaining options, listed with a leading underscore ( _ ). Therefore, replace the underscore with transceiver, transmitter, or receiver. If you are using transmitterAndReceiver, configure the following twice, once with each prefix.</p>
_.host		Defines the host name of the SMPP server.

Key	Default	Description
_.password		Defines the password of the user connecting to the SMPP server.
_.port		Defines the port number of the SMPP server.
_.systemId		Defines the system ID to use when binding to the SMPP server.
_.systemType		Defines the system type to use when binding to the SMPP server.
_.enquireLinkTimer	60000	Defines the session timer, in milliseconds. If no activity is recorded, the underlying library sends an enquire link request to the server.
_.transactionTimer	10000	Defines the time, in milliseconds, to wait for the response to a sent request. For example, the length of time to wait for the SUBMIT_SM_RESP after sending a SUBMIT_SM.
_.initialReconnectDelay	1000	Defines the time, in milliseconds, to wait after starting the SmpChannel before attempting to connect to the server.
_.reconnectDelay	1000	Defines the time, in milliseconds, to wait before attempting to reconnect to the server after being disconnected.
_.usingSSL		Indicates whether to use a SSL connection when connecting to the SMPP server.
_.disableGsmAlphabet		Encodes the text in the global system for mobile (GSM) communication alphabet, if possible, and falls back to a multibyte encoding. If you are using a language that normally cannot be encoded using the GSM alphabet, set this key to true, which disables this check and always uses a multibyte encoding.

**Note:** The default channel for all outgoing messages is `htmlchannel1`. Once an outgoing message is sent by Mobiliser, the contents of the messages can be found at the following URL: <https://localhost:8081/mobiliser/channelmgr/html>, where you are asked for the Mobiliser mobiliser user before you can view the outgoing messages. To use the `emailchannel1` as the method for sending outgoing messages, you must first deactivate `htmlchannel1`. To deactivate it, set the "`_active`" preference entry to `false` via the Operations Dashboard in: `/com/sybase365/mobiliser/util/messaging/channelmanager/engine/impl/ChannelInstantiator/htmlchannel1`.

### GCMChannel

The GCMChannel uses the Google Cloud Messaging (GCM) service to push key-value pairs to an Android application. You can use the gcmchannel only via sendPush messaging method.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/util/messaging/channelmanager/engine/impl/ChannelInstantiator/gcmchannel1
- **ConfigAdmin PID:** –  
com.sybase365.mobiliser.util.messaging.channelmanager.engine.impl.ChannelInstantiator.gcmchannel1

You can set these configuration options:

Key	Default	Description
channelId		Indicates the channel's ID.
urlSupplement	gcm	Appends the supplement to the channel manager URL, making the HTTP channel accessible. It is a suffix on the URL for internal processing.
gcmUrl	GCM Service 1	Defines the URL of the GCM service. When Mobiliser must use a proxy server, configure these environment variables: <ul style="list-style-type: none"> <li>• http(s).proxyHost</li> <li>• http(s).proxyPort</li> </ul>

### Audit

Auditing occurs at the message dispatch level that wraps the implemented endpoint. Each audit manager is called asynchronously with the results of the service call.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/framework/service/audit
- **ConfigAdmin PID:** –  
com.sybase365.mobiliser.framework.service.audit

ConfigAdmin PID:

You can set these configuration options:

Key	Default	Description
disabled	false	Disables the complete auditing subsystem.
allowCoreThreadReadTimeOut	true	Defines the configuration of the internal executor service.

Key	Default	Description
corePoolSize	3	Defines the configuration of the internal executor service.
keepAliveSeconds	60	Defines the configuration of the internal executor service.
maxPoolSize	2147483647	Defines the configuration of the internal executor service.
queueCapacity	0	Defines the configuration of the internal executor service.

Set your defaults to use a synchronous queue and an unlimited number of threads, such as `queueCapacity=0` and `maxPoolSize`. Alternatively, you can use a fixed number of threads and a large queue capacity so that tasks are placed on a queue, then processed by the fixed number of threads:

```
allowCoreThreadTimeOut=true
corePoolSize=10
keepAliveSeconds=60
maxPoolSize=10
queueCapacity=2147483647
```

**Note:** If you have a large queue, do not set the `corePoolSize` to 1; the executor service waits for the queue to fill before spawning new threads, the end result of which is concurrency. If the queue size is too small and no threads are available, the audit subsystem starts dropping audits.

### See also

- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320

### **Database Audit Manager and Audit Dispatcher**

The database audit manager records audit information to a persistent database table, allowing reports to be written in SQL to get a snapshot of incoming requests. The audit dispatcher dispatches a worker to process queued audits into the database.

#### *Database Audit Manager*

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/money/auditmgr/DatabaseAuditManager`
- **ConfigAdmin PID:** – `com.sybase365.mobiliser.money.auditmgr.DatabaseAuditManager`

You can set this configuration option:

Key	Default	Description
disabled	false	Disables audit manager.

### *Audit Dispatcher*

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/money/auditmgr/AuditDispatcher
- **ConfigAdmin PID:** – com.sybase365.mobiliser.money.auditmgr.AuditDispatcher

Preferences node:

You can set these configuration options:

Key	Default	Description
interval	3000	Defines the interval, in milliseconds, for running the dispatch worker to write entries to the database.
batchSize	50	Defines the maximum number of entries to write into the database as part of a batch.
interval	60000	Defines the maximum value for the interval between runs. If no entries are found, the interval between runs is increased up to this maximum value.
maxConcurrent	3	Defines the maximum concurrent workers to process the load. If the worker finds entries to write in the database and there are more than the batch size left after the run, the worker submits additional workers.
shutdownWait	5000	Defines the shutdown wait time, in milliseconds, for the executor dispatching the workers. If set to 0, the executor shuts down immediately losing any queued audits. Otherwise, the executor stops accepting new workers and waits for processing to complete or until the defined amount of time elapses before forcing a shutdown.

### **JavaScript Object Notification Audit Manager**

You can use the JavaScript Object Notification (JSON) audit manager to safely dump incoming request information into a log file.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/framework/service/jsonaudit/JsonAuditManager/



- **ConfigAdmin PID:** –  
com.sybase365.mobiliser.framework.service.jsonaudit.JsonAuditManager

You can set these configuration options:

Value	Key Description
regex	Regex values that mask fields that should not be logged in clear text.
bean	Beans that should not be logged at all. You can use this option to skip certain request or response types.
field	Any number of keys with value field. Any field with this name is masked before logging.
path	Any number of keys with the value path. These are path expressions that mask values from being logged in clear text.

This example describes how to configure the audit manager:

```
# mask any fields name pin or password
pin=field
password=field

# mask any request bean named GetPreferencesRequest
com.sybase365.mobiliser.util.contract.v5_0.prefs.GetPreferencesRequest=bean
# mask any fields matching this regex
com\.sybase365\.mobiliser\.money\.contract\.v5_0\.customer
\.security\. (Login
CustomerRequestType|SetCredentialRequest)/credential=regex

com\.sybase365\.mobiliser\.money\.contract\.v5_0\.transaction
\.Authentication
Continue/Credential/ (payer|payee)=regex
com\.sybase365\.mobiliser\.util\.contract\.v5_0\.messaging\.
(CreateAttachment
Request|FindAttachmentsResponse|GetAllAttachmentsResponse)/
attachment(s)?/
content=regex

# mask any fields matching this path
com.sybase365.mobiliser.util.contract.v5_0.messaging.TemplateMessageRequest
Type/message/parameters/value=path
```

## Event Handler

All event handlers share a certain configuration. The preferences path is usually equivalent to the event or task implementation class name.

You can set these configuration options:

Key	Default	Description
event.retry.delay	60	Defines the retry delay time, in seconds.
event.retry.maxRetries	3	Defines the maximum number of retries.
event.expiry	0	Defines the time after which an event is regarded as expired and can no longer be handled.
event.handler.maxActive	4	Controls the maximum number of objects that can be allocated, such as checked out to clients or idle awaiting check-out by the pool at any given time. A negative value indicates that there is no limit to the number of objects that can be managed by the pool at one time. When maxActive is reached, the pool is said to be exhausted.
event.handler.maxIdle	2	Controls the maximum number of objects that can sit idle in the pool at any time. A negative value indicates that there is no limit to the number of objects that can be idle at one time.
internal.dao.callerId	-1	Defines the caller ID of the internal user. Used as the customer ID when creating or updating database records (ID_CUSTOMER_CREATION, ID_CUSTOMER_LAST_UPDATE).
internal.service.userName		Defines the user name that authenticates the service call in the event that a task, job, or event handler must make a call to a Mobiliser service.
internal.service.password		Defines the password for the user that authenticates the service call.

## Tasks

All tasks share a certain configuration. The preferences path is usually equivalent to the task implementation class name.

You can set these configuration options:

Key	Default	Description
task.cronpattern	0 0/5 * ? * *	Defines the cron pattern on which the task execution is scheduled.
internal.dao.callerId	-1	Defines the caller ID of the internal user. Used as the customer ID when creating or updating database records (ID_CUSTOMER_CREATION, ID_CUSTOMER_LAST_UPDATE).

Key	Default	Description
internal.service.user-name		Defines the user name that authenticates the service call in the event that a task, job, or event handler must make a call to a Mobiliser service.
internal.service.password		Defines the password for the user that authenticates the service call.

## **Jobs**

All Mobiliser jobs share a certain configuration. The preferences path is usually equivalent to the job implementation class name.

You can set these configuration options:

Key	Default	Description
internal.dao.callerId	-1	Defines the caller ID of the internal user. Used as the customer ID when creating or updating database records (ID_CUSTOMER_CREATION, ID_CUSTOMER_LAST_UPDATE).
internal.service.user-name		Defines the user name that authenticates the service call in the event that a task, job, or event handler must make a call to a Mobiliser service.
internal.service.password		Defines the password for the user that authenticates the service call.

The other job implementation class name configuration is read from the MOB\_JOBS table, which controls the execution schedule, and an additional parameter string that is handed over to the job upon execution.

## **cron Job Task Handler**

A cron job execution task regularly retrieves jobs from database tables, based on the job handler name and cron job schedule. The cron job task must have a cron job schedule and job handler name, which are configured in the preferences node.

Tasks are used for multiple purposes, such as housekeeping or monitoring. The task ensures that each job does not run in parallel multiple times, cancels jobs that are unresponsive, and synchronizes job execution across JVMs through a database semaphore.

Job executions are managed centrally in the MOB\_JOBS table, but you can manage jobs in the Operations Dashboard portal. Jobs can also execute only on a single server.

Preferences node: /businesslayer/com/sybase365/mobiliser/money/jobs/task/cronjob/handler/CronjobTaskHandler

You can set these configuration options:

Key	Default	Description
task.cronpatter	0 0/1***?	Defines the cron job schedule in cron expression format. The default expression triggers a job every minute.
task.jobHandlerNames	MOBILIS- ER	<p>Defines the job handler names that correspond to the values in MOB_JOBS.STR_HANDLER_NAME database table.</p> <p>When you create new handler names, use commas to separate each name, for example, MOBILISER1, MOBILISER2, MOBILISER3.</p> <p>New handler names create new records in the MOB_BULLY table. Verify that there is one MOB_BULLY table entry configured for each task.jobHandlerName. For example, the default handler name MOBILISER should have an entry with the STR_SERVICE="MOBILISER". If not, insert a new one, using these values:</p> <ul style="list-style-type: none"> <li>• STR_SERVICE = MOBILISER</li> <li>• BOL_IS_ACTIVE=Y</li> <li>• INT_LEASE_TERM=300</li> </ul> <p>Configure the name of the system property as the value for the task.jobHandlerNames key, for example \${JOB_HANDLER_NAME}. This value is replaced by the system property with the same name. You can set the system property individually for each server by adding a -D parameter to the SMP_HOME\Server\props.ini file.</p> <p>For example, -DJOB_HANDLER_NAMES=MOBILISER,MOBILISER2.</p>

**See also**

- *cron Expression Reference* on page 338
- *Jobs* on page 321
- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320

**MOB\_JOBS Table**

cron jobs are configured in the MOB\_JOBS database table.

Field	Re-quired	Description
ID_JOB	Yes	Indicates the unique integer to identify a job.
STR_HANDLER_NAME	Yes	Defines the handler name. <hr/> <b>Note:</b> This handler name defines the cron job task that handles this job. It is different than the SAP Mobile Platform task handler name that is described in the events system. <hr/>
STR_SCHEDULE	Yes	Defines job schedule in cron expression format. See cron Expression Reference.
ID_IMPLEMENTATION_TYPE	Yes	Defines the meaning of the URL_IMPLEMENTATION string. Valid values are: <ul style="list-style-type: none"> <li>• using the full class name.</li> <li>• using the job bean name as the filter string.</li> </ul> The value 2 means using full class name and the value 3 means using job bean name as filter string.
URL_IMPLEMENTATION	Yes	Defines the service filter to apply to find the proper implementation of IMobiliserCronJob in the OSGi service registry. If ID_IMPLEMENTATION_TYPE=2, use the full class name with the package name where the job is implemented. If ID_IMPLEMENTATION_TYPE=3, use the job bean name.
BOL_IS_ACTIVE	Yes	Indicates whether job is active or not. Y if active; otherwise, N.
DAT_LAST_EXECUTION	No	Indicates the date the job was last executed. Set to NULL initially.
INT_MAX_DELAY	No	Sets the maximum amount of time, in minutes, a job can still execute after the scheduled time (for example, in case of system restart).
INT_MAX_DURATION	No	Sets the maximum length of time, in minutes, the job can continue. After this length of time, the job is handled as failed.
STR_JOB	No	Defines the description of the job.
STR_PARAMETER	No	Indicates the parameters that the job handler requires for job processing.

Field	Re-quired	Description
DAT_CREA-TION	No	Indicates the date the job is created.
ID_CUSTOM-ER_CREATION	No	Indicates the customer ID of who created the job.
DAT_LAST_UP-DATE	No	Indicates the date the job was last updated.
ID_CUSTOM-ER_LAST_UP-DATE	No	Indicates the customer ID of who last updated the job

**See also**

- *cron Expression Reference* on page 338
- *Jobs* on page 321

**Miscellaneous Configuration**

Various configuration items that are configured via Preferences.

**Security Endpoint**

The SecurityEndpoint sends out system-generated passwords. You can use preferences to specify the channel name for sending notifications containing generated credentials.

Preferences node: /businesslayer/com/sybase365/mobiliser/money/services/umgr/SecurityEndpoint

You can set these configuration options:

Key	Default	Description
messageChan-nel.email		Specifies the message channel to use for e-mail messages.
messageChan-nel.sms		Specifies the message channel to use for SMS.

**See also**

- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320

### **SMS Available on Cell Authentication**

The `SmsAocAuthenticationHandler` implements the `IAuthenticationHandler` interface and can be used for consumer authentication in line with transaction processing. The handlers leverage a preconfigured “application” in SMS Builder to start an interactive SMS session with the customer.

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/money/businesslogic/authentication/handlers/smsaoc/SmsAocAuthenticationHandler`
- **ConfigAdmin PID:** – `com.sybase365.mobiliser.money.businesslogic.authentication.handlers.smsaoc.SmsAocAuthenticationHandler`

You can set these configuration options:

Key	Default	Description
<code>regex.yes</code>	<code>(ja) (yes) (oui) (si)</code>	Defines the regular expression to use to validate the positive user response.
<code>brand.base</code>	<code>http://localhost:8081</code>	Indicates the configured service URL of SMS Builder.
<code>brand.client</code>		Indicates the configured client context for SMS Builder service.
<code>brand.shortcode</code>		Indicates the configured short code to use in SMS Builder.
<code>brand.keyword.payer</code>	<code>payeraoc</code>	Specifies the configured keyword in SMS Builder to launch the application implementing the payer Advice of Charge authentication process.

### **One-Time Password Generation**

The one-time password (OTP) business logic includes a service that sends and validates non-persistent OTPs. The handling is different for standard (persistent) OTPs, and the required configuration is done via preferences.

- **Preferences node:** – `/businesslayer/com/sybase365/mobiliser/money/businesslogic/customer/configuration/CustomerOtpConfiguration`
- **ConfigAdmin PID:** – `com.sybase365.mobiliser.money.businesslogic.customer.configuration.CustomerOtpConfiguration`

You can set these configuration options:

Key	Default	Description
channel		Indicates the channel in channel manager to use to send out the message. This provides a fallback in case channel.email or channel.sms is not set.
channel.email		Indicates the channel in channel manager to use to send out the OTP via an e-mail message.
channel.sms		Indicates the channel in channel manager to use to send out the OTP via SMS.
tokenLength	6	Sets the length of the OTP (token) that is to be generated.
otpTypeAuthToken	100	Defines the authorization token for the OTP type in use.
tokenTimeToleranceMinutes	2	Sets the time tolerance (+/-), in minutes, when verifying whether the token is valid (timestamp is part of a generated token before hashing).
smsTokenTemplate		Defines the name of the template to use when sending out OTP tokens (nonpersisted).

### See also

- *Node and System Preferences* on page 319
- *Adding a Preference Node* on page 320

### Transaction Configuration

Most transaction-related configuration takes place in specific database tables. Preferences configuration is performed only in very few places of the transaction-related business logic.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/money/businesslogic/transaction/TransactionConfiguration
- **ConfigAdmin PID:** – com.sybase365.mobiliser.money.businesslogic.transaction.TransactionConfiguration

You can set these configuration options:



Key	Default	Description
disableOpenTransactionsCheck	false	Disables the check for open transactions for the payer. Depending on the system and use case configuration, there must be only one "open" transaction (error code = 0 &&, status code = 0) for the payer of a transaction (mostly to match an incoming, asynchronous authentication request that must be matched to the transaction).
alwaysCreate AuthorisationCode	false	Creates a transaction authorization code, even if the authorization fails.
defaultExpirationMinutes	20,160 (14 days)	Sets the default authorization expiry date when use case does not have a specific configuration.

### **Demand for Payment**

Demand for payment is sent by customers to request money. An invoice is created for this purpose and assigned to the customer being requested to send money. There is some configuration required to create a new invoice type (demand-for-payment). and a default invoice form. This is done in preferences.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/money/businesslogic/transaction/demandforpayment/impl/DemandForPaymentImpl
- **ConfigAdmin PID:** – com.sybase365.mobiliser.money.businesslogic.transaction.demandforpayment.impl.DemandForPaymentImpl

You can set these configuration options:

Key	Default	Description
demandForPaymentUseCaseId	202	Specifies the use case ID that is used when creating the invoice configuration and also when selecting the applicable payment instruments.
invoiceTypeHandlerTypeId	3	Specifies the ID of the bill payment handler that executes demand-for-payment invoices.
invoiceTypeGroupId	3	Specifies the default invoice type group ID that is used when creating new invoice types or when searching for existing invoice types.

## Security Fundamentals

---

Use the information about each of the various mechanisms to learn about the required configuration that is necessary to harden your application.

### Prevent Unauthorized Access

All application level user accounts are managed by Mobiliser in its central database.

The relevant tables are:

- **MOB\_CUSTOMERS** – stores information about the type of user, and status information.
- **MOB\_CUSTOMERS\_IDENTIFICATIONS** – stores unique identifications for the users, such as user names.
- **MOB\_CUSTOMERS\_CREDENTIALS** – stores hashed user passwords.

Other than for the initial setup, there is usually no direct interaction with the tables required.

Based on various criteria that is specific to the individual customer or customer type, users are automatically equipped with privileges that allow them to make certain service calls into Mobiliser. Each service call is normally protected by an individual privilege. In addition, the service is bundled into a context. The context may require another privilege and defines the URL under which a service is exported. In some situations, technical users are used, for example, by the Web portal or by other internal components that invoke services.

### Web Portal Access to Mobiliser

The Web portals provide two different system level users to gain access to Mobiliser.

The first user is defined along with the URL to the preferences service as a Java Naming and Directory Interface (JNDI) string resources (usually in conf/context.xml).

The configuration URL consists of multiple parts:

```
<scheme>://<user>:<password>@<host:port>/<path>?pollInterval=
<interval>&clientType=<clientType>&applicationIdentifier=
<applicationIdentifier>
```

**Note:** When you store the URL in an XML file, make sure the URL is XML encoded, for example, use `&amp;` to display the ampersand.

Fragment	Description
scheme	Defines the scheme values: <code>prefs</code> and <code>prefss</code> . If <code>prefss</code> is set, the connection is done via HTTPS; otherwise, HTTP is used.

Fragment	Description
user	Defines the user who accesses the preferences services. The default user is <code>prefsread</code> , which has the minimum set of roles configured to access the services.
password	This password must match that of the configured user in the platform.
host:port	Defines the host name and the port at which the platform is accessible.
path	Defines the URL path, usually <code>mobiliser/rest/prefs</code> , to the service.
interval	Sets the time interval, in milliseconds, between checks for new configuration with the server.
clientType	Indicates the protocol to use when calling the service. Default is "json".
applicationIdentifier	<p>Identifies the configuration set being used. The preferences option allows multiple configuration trees; each of which is identified by an applicationIdentifier. The standard configuration generally uses both of these configuration sets:</p> <ul style="list-style-type: none"> <li>• <b>businesslayer</b> – is for the back end (Mobiliser).</li> <li>• <b>presentationlayer</b> – is used by the Web portals.</li> </ul> <p>You can grant read and write access to the applicationIdentifiers individually.</p>

The Web portal instance uses the environment user (`prefs2/config`) to access the preferences, which run as part of the Mobiliser feature. The user has access only to the services that allow read access to preference information.

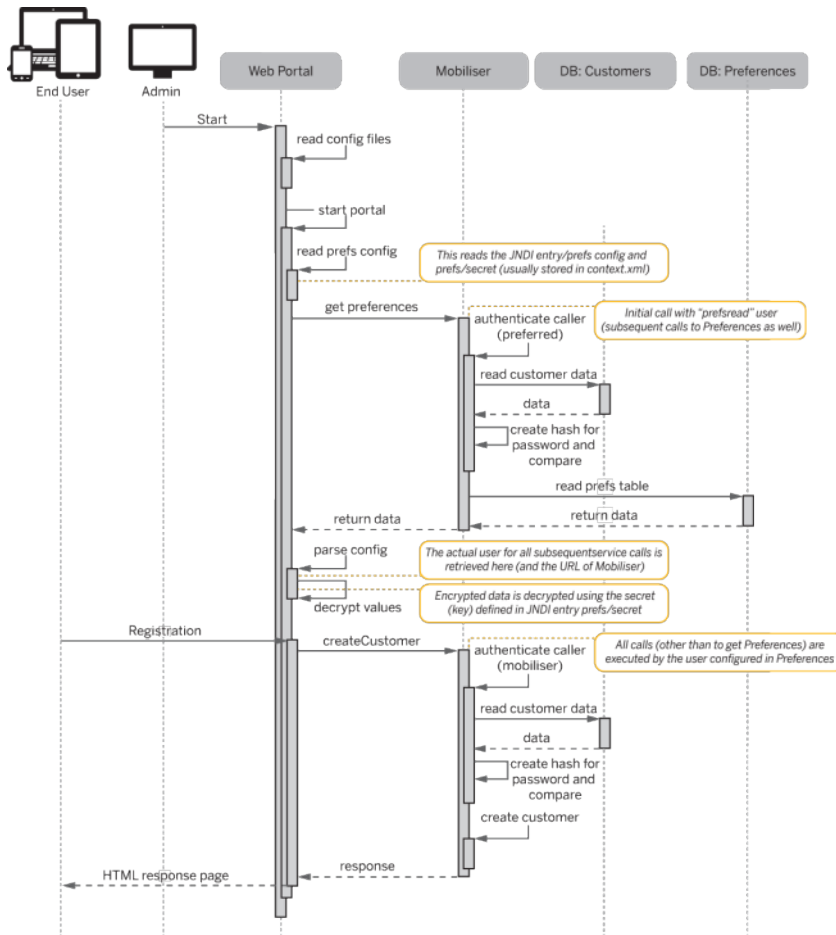
The configuration data from preferences contains all other application-level configuration, including the user that is used for subsequent service calls from the portal to the platform.

Some data, such as passwords, are stored in the preferences in an encrypted format. The key for decrypting the data is also stored as a JNDI string resource, with the name "prefs/secret". If you change this value, reencrypt all the encrypted data stored in preferences (for the given applicationIdentifier) using the command line tool and SQL, or the Operations Dashboard preferences functionality.

By default, the Web portal uses the "mobiliser" user to make service calls to the back end; configure the user name and password in preferences. The password, which is stored encrypted in preferences, must match the hashed password that is stored for the corresponding user (mobiliser) in the database (`MOB_CUSTOMERS_CREDENTIALS`).

For security reasons, the password is not set in the standard configuration. Set it manually during the platform installation and setup.

## Application Services (Mobiliser) Administration



## Hashing Customer Credentials

Any customer (consumer, merchant, agent, or system user) credentials are stored, in a hashed format, in MOB\_CUSTOMER\_CREDENTIALS. SAP Mobile Platform supports different hashing algorithms. The STR\_CREDENTIAL is always prefixed with the hashing algorithm in curly brackets, for example, {<HASH-ALGORITHM>}<HASHVALUE>.

Configuration of hashing algorithms is controlled through preferences.

- **Preferences node:** - /businesslayer/com/sybase365/mobiliser/money/businesslogic/umgr/impl/SmartPasswordEncoder
- **ConfigAdmin PID:** - com.sybase365.mobiliser.money.businesslogic.umgr.impl.SmartPasswordEncoder

You can set these configuration options:

Key	Value	Description
algorithms	SHA,SHA-256,SHA-512,SHA-512:1,SHA-512:10000,PBKDF2WithHmacSHA1:10000,BCRYPT:10,SSHA-512:10000,SPBKDF2WithHmacSHA1:10000	Indicates a comma-separated list of supported hashing algorithms.
encodeAlgorithm	SSHA-512:10000	Defines the algorithm to use for storing and encoding new credentials.
defaultAlgorithm	SHA	Defines the algorithm to use for credential validation if the algorithm is not specified with the stored credential.

You can change the default configurations within certain boundaries. You can add new hashing algorithms only when they are provided through JCE—that is, the hashing algorithms come with your JDK. However, you can change the number of iterations, which is the numeric value after the colon, to either increase or decrease performance or security if required.

### *Security Logic*

Each time a customer's credential is checked, Mobiliser validates whether the hashing algorithm is configured to be updated with the 'encodeAlgorithm'.

- **Preferences node:** – /businesslayer/com/sybase365/mobiliser/money/businesslogic/umgr/impl/SecurityLogic
- **ConfigAdmin PID:** – com.sybase365.mobiliser.money.businesslogic.umgr.impl.SecurityLogic

You can set this configuration option:

Key	Description
hashUpdatePattern	<p>Identifies the hash upgrade pattern, which is a Java regular expression (regex) pattern class. If set to &lt;null&gt;, no password upgrade is performed; otherwise, any hashed password that matches this pattern is rehashed using the current 'encodeAlgorithm'. By default, this value is not configured. SAP suggests that if you use this key, write a negated regex. For example, to transition all hashes to BCrypt:10, use:</p> <pre>^(?!\\{BCRYPT:10\\}) .+\$</pre> <p><b>Note:</b> BCrypt tremendously decreases performance, so use it only if it is a strong security requirement.</p>

### Spring Security

You must reconfigure Spring Security to allow access to the plain text password for rehashing. Update this node:

```
com.sybase365.mobiliser.framework.gateway.security.filters.standard
```

Set the `osgiProviderManager.eraseCredentialsAfterAuthentication` key to `FALSE`, which means credentials are not cleared from the authentication object returned from Spring Security, allowing the plain text password to be rehashed.

The actual value stored in `STR_CREDENTIAL` depends on the hashing algorithm. All hash values are Base64-encoded. For all algorithms that do not use random salt values, the customer ID is used as the salt value. Random salts are always 16 byte.

```
SHA: BASE64 (HASH (<SALT>|<HASH>))
SSHA: BASE64 (<SALT>HASH (<SALT><HASH>))
PBKDF2: BASE64 (HASH (<SALT>, <HASH>))
SPEKDF2: BASE64 (<SALT>HASH (<SALT>, <HASH>))
BCrypt: $2a$<ROUNDS#>$BASE64 (<SALT><HASH>)
```

SAP Mobile Platform comes with a Java executable for computing hash values:

```
./tools> java-jar com.sybase365.mobiliser.vanilla.cli-tools-5.1.3.RELEASE-CLIPasswordEncoderClient.jar
```

## Default Mobiliser Web Portal Accounts

When you enable Mobiliser in SAP Mobile Platform, Web portals are deployed for both administrative and consumer access.

The Web portal (<https://localhost:8081/portal>) connects to the endpoint that allows administrative and consumer users to log in. Administrative users can create and manage user accounts, notifications and alerts, and merchants.

The administrative users are:

User Name	Password	Access
cstfull	secret	Full customer support privileges to the administration portal
usermgr	secret	Manage agent accounts
notifmgr	secret	Manage notifications and alerts
headquarter	secret	Create and manage merchants
opsmgr	secret	View and manage system configuration
sysmgr	secret	Monitor all functions of the server

## Changing the Mobiliser Web Portal Passwords

System administrators can change the password of the default Web UI accounts.

1. Log in to the internal Web UI portal using any default user account.
2. Click **SELFCARE**.
3. Select **Change Password**.
4. Enter the old password for the logged-in user.
5. Enter the new password.
6. Confirm the new password.
7. Click **Update**.

## Operations Dashboard Reference

The Operations Dashboard provides system administrators with a high-level operational view of the enterprise mobility management system. The dashboard aggregates the information to give administrators an overview of the system, which aids in operational support.

Administrators can track individual statistics for servers to monitor performance and general operational efficiency.

## Logging in to the Operations Dashboard

Log in to the Operations Dashboard to view and manage system configuration.

1. Navigate to the Operations Dashboard at:  
`https://localhost:8081/portal`
2. Enter:
  - User name – `opsmgr`
  - Password – `secret`

3. Click **Login**.

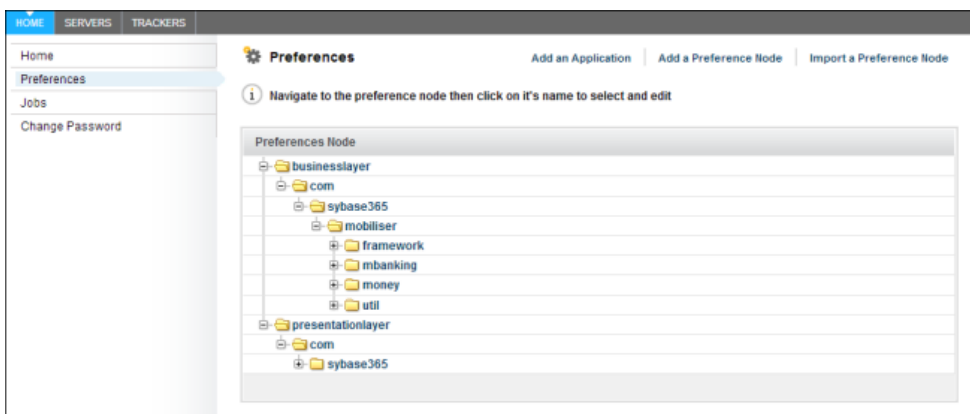
## Preferences

Preferences are the standard mechanism for configuring applications. Use the Preferences option to manage operation-level configuration data such as timeouts, retries for communicating with other systems, and thread pool sizes.

The standard installation comes with two applications:

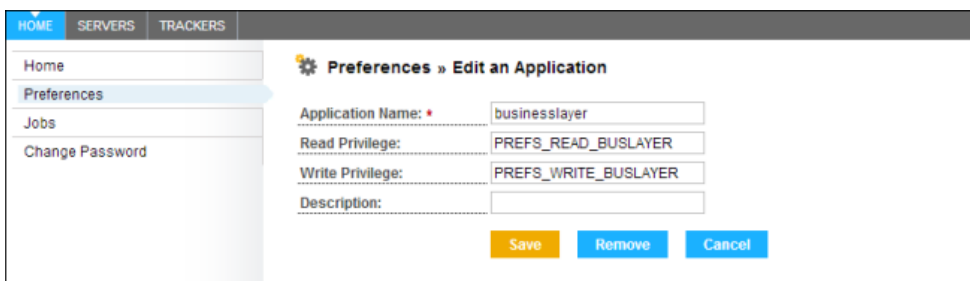
- businesslayer
- presentationlayer

You can add applications and preference nodes. You can import the node information from an XML file, which contains the application name, path of the node, and preference keys and values. When you import data, it is added to the node path that is defined in the XML file.



## Applications

You can define preferences for multiple applications, each with a unique name and access rights. An application must have a unique name, and may optionally have a description. You can define read and write privileges for an application. If you do not define read or write privileges, any user who invokes the application services can retrieve or set preference values. You can edit or remove applications.

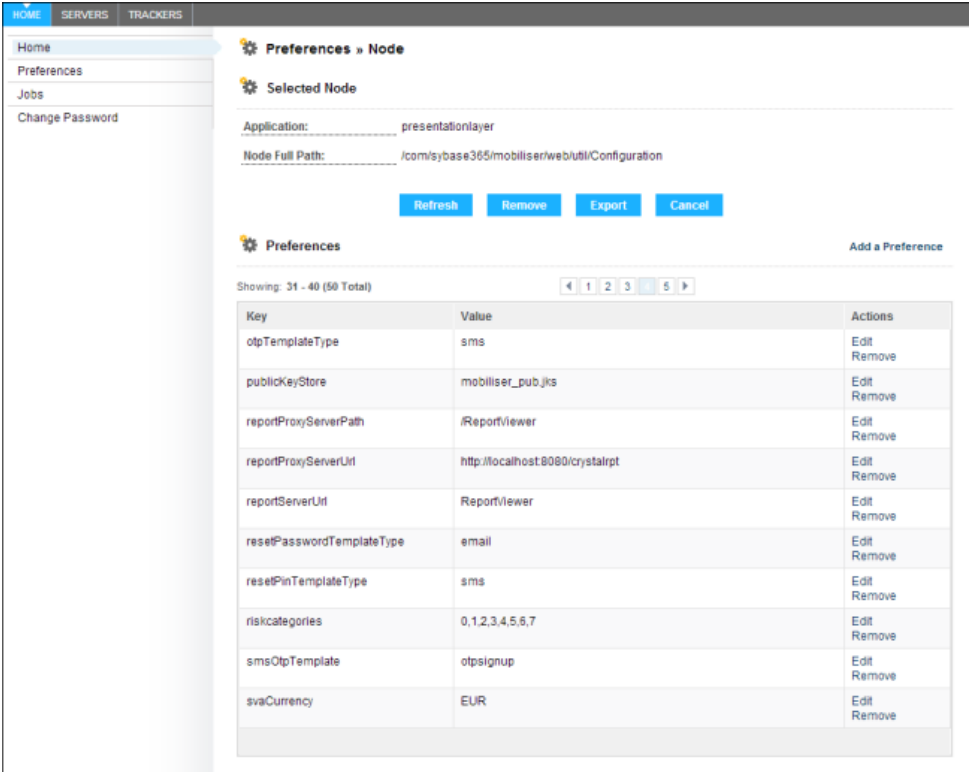




## Node and System Preferences

Preference nodes store system preferences and configuration data. Each system preference contains a key-value pair that is associated with a preference path. You can add or remove a preference node, but you cannot edit them. You can export node information to an XML file, which includes the application name, path of the node, and preference keys and values.

When you click a node, you see a new page that lists all the Preferences for that node. Each page shows 10 entries; you might need to navigate through the pages to find the entry you are looking for.



The screenshot shows the 'Preferences' page for a selected node. The page includes a navigation menu on the left with options like 'Home', 'Preferences', 'Jobs', and 'Change Password'. The main content area is titled 'Preferences » Node' and shows the 'Selected Node' information, including the Application name ('presentationlayer') and the Node Full Path ('/com/sybase365/mobiliser/web/utill/Configuration'). Below this, there are buttons for 'Refresh', 'Remove', 'Export', and 'Cancel'. The 'Preferences' section shows a table of 10 entries, with a total of 50 preferences available. The table has columns for 'Key', 'Value', and 'Actions'.

Key	Value	Actions
otpTemplateType	sms	Edit Remove
publicKeyStore	mobiliser_pub.jks	Edit Remove
reportProxyServerPath	/Report/viewer	Edit Remove
reportProxyServerUrl	http://localhost:8080/crystalrpt	Edit Remove
reportServerUrl	Report/viewer	Edit Remove
resetPasswordTemplateType	email	Edit Remove
resetPinTemplateType	sms	Edit Remove
riskcategories	0,1,2,3,4,5,6,7	Edit Remove
smsOtpTemplate	otpsignup	Edit Remove
svaCurrency	EUR	Edit Remove

### See also

- *Keystore Configuration* on page 269
- *Adding a Preference Node* on page 320
- *Encrypting Preferences Using Operations Dashboard* on page 273
- *Java Management Extension Authentication* on page 279
- *Standard Security Filters* on page 280
- *Open Data Protocol Interface* on page 283
- *Transmission Control Protocol Interface* on page 284

- *Engine* on page 287
- *Template* on page 292
- *Audit* on page 300
- *cron Job Task Handler* on page 305
- *Security Endpoint* on page 308
- *One-Time Password Generation* on page 309

### **Adding a Preference Node**

You can add a preference node, which requires the application name and full node path. If the node does not exist, you must add it using the full node path and then add the configuration data. You can remove preference nodes, but you cannot edit them.

1. Log in to the Operations Dashboard as the `opsmgr` user.
2. In the left pane, select **Preferences**.
3. Click **Add a Preference Node**.
4. In the Application field, select:
  - **businesslayer**
  - **presentationlayer**
5. In the Full Node Path field, enter the path of the node.  
For example: `/com/sybase365/mobiliser/util/messaging/channelmanager/engine/impl/ChannelInstantiator/`
6. Click **Save**.
7. Navigate to the new preference node.
8. Click **Add a Preference** and enter the configuration options.
9. Click **Save** after each entry.
10. Click **Refresh** to ensure that preference changes are committed.

### **See also**

- *Encrypting Preferences Using Operations Dashboard* on page 273
- *Node and System Preferences* on page 319
- *Keystore Configuration* on page 269
- *Java Management Extension Authentication* on page 279
- *Standard Security Filters* on page 280
- *Open Data Protocol Interface* on page 283
- *Transmission Control Protocol Interface* on page 284
- *Engine* on page 287
- *Template* on page 292
- *Audit* on page 300
- *cron Job Task Handler* on page 305

- *Security Endpoint* on page 308
- *One-Time Password Generation* on page 309

## Jobs

The Jobs option lets you schedule background jobs to run at certain times using the **cron** utility. For example, you can schedule a job to run at midnight to transfer commissions to the individual partners. You can also schedule a job to run every five minutes to generate new invoices. A **cron** daemon can retrieve jobs regularly from a database using the job handler name. The task makes sure that each job does not run in parallel multiple times, cancels jobs that are not responding, and synchronizes job execution across Java Virtual Machines (JVMs).

The screenshot shows the 'Edit a Job' form with the following fields and values:

- Handler: MOBILISER
- Serviced By: Class
- Implementation: com.sybase365.mobiliser.mon
- Schedule: 0 0-59/15 \* ? \* \* \*
- Parameters: clearingfrequency=DD,cutoffime
- Job Name: starts the clearing
- Active: Yes
- Max Delay: 5
- Max Duration: 20

Field	Description
Handler	Defines the <b>cron</b> daemon that handles the task. The handler corresponds to a defined value in the database. The default handler is MOBILISER. You can enter multiple handlers separated by commas (MOBILISER, MOBILISER1, and so on).
Serviced By	Service filter for the job: Class or Bean.
Implementation	Defines the service filter. <ul style="list-style-type: none"> <li>• If the service filter is set to Class, use the fully qualified class name.</li> <li>• If the service filter is set to Bean, use the job bean name.</li> </ul>
Schedule	Defines the job schedule in a <b>cron</b> format. For example, 0 0/5 * ? * * * defines a <b>cron</b> daemon that runs every 5 minutes. The default expression is 0 0/1 * * * ? , which runs every minute.

Field	Description
Parameters	Parameters that the job handler requires for job processing. Parameters can be any string that depends on the expected result of the job.
Job Name	Job description.
Active	Defines whether the job is active or inactive.
Max Delay	Maximum number of minutes after the scheduled time that starting the job can be delayed. If the maximum delay is exceeded, the job is not started.
Max Duration	Maximum duration in minutes that the job runs. If the maximum duration time is exceeded, the job is cancelled and marked "failed."

**See also**

- *cron Expression Reference* on page 338

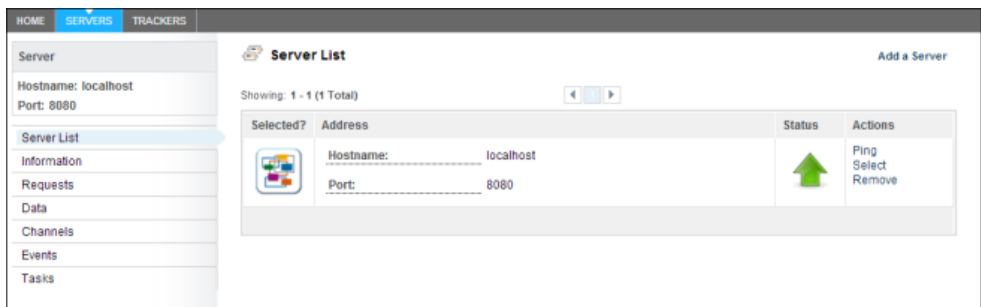
**Servers**

The Servers option displays a list of available servers. You can select a server that is online to view information, requests, data, channels, events, or tasks.

**Server List**

The Server List displays the online and offline servers in the environment. You can select a server that is online to view its information or ping its host. When you select a server, its host name and port number display in the left pane. If a server is offline, the visual indicator in the status column shows an orange circle with an exclamation mark. You cannot select or ping a server that is offline.

You can add other servers to the list. To add a server, click **Add a Server**, and enter the host name or IP address, and the port number. You can also remove servers from the list.



**Information**

The Information option summarizes the basic system environment information for the selected server, such as the number of processors and the operating system. You can also view

the amount of total and free physical memory, committed virtual memory, swap space, class paths, and length of time the server has been available.

The screenshot displays the 'SERVERS' section of the Mobiliser Administration interface. The left sidebar contains navigation options: Server, Hostname: localhost, Port: 8080, Server List, Information (selected), Requests, Data, Channels, Events, and Tasks. The main content area is divided into several sections:

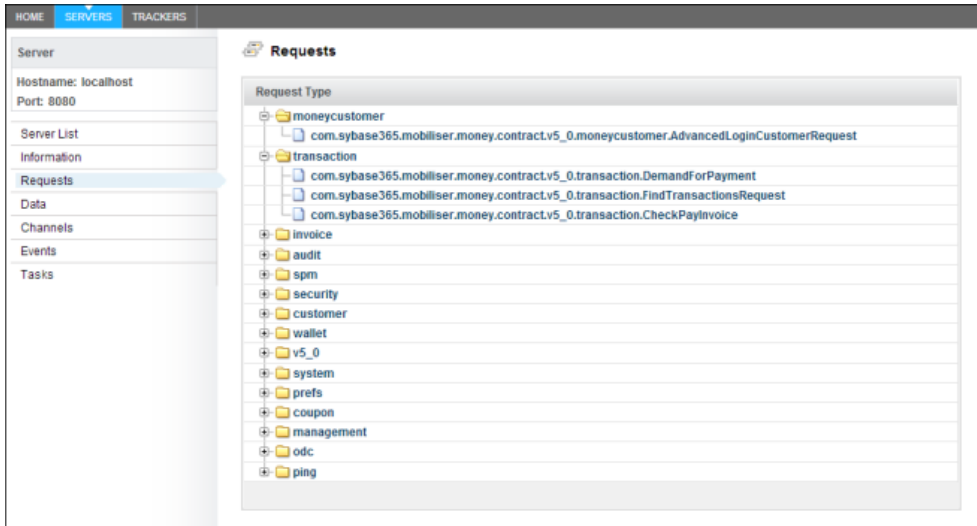
- Information:** A header section with a server icon.
- System Environment:** A table of system details:
 

Operating System:	Linux 2.6.32-358.2.1.el6.x86_64	Total Physical Memory:	8,061 MB
Architecture:	amd64	Free Physical Memory:	158 MB
Number of Processors:	2	Total Swap Space:	1,048 MB
Committed Virt. Memory:	2,922 MB	Free Swap Space:	1,048 MB
- VM Environment:** A table of VM details:
 

Process:	862@demo03.resdev.lab	Up Time:	6d 01:29:09.085
Name/Version:	Java HotSpot(TM) 64-Bit Server VM 23.21-b01	Start Time:	4/19/13 4:18:22 PM
Vendor:	Oracle Corporation	JIT Compiler:	HotSpot 64-Bit Tiered Compilers
- Paths:** A section listing various paths:
  - Class Path:** /opt/sybase/mobiliser/bundles/com.sybase365.mobiliser.vanilla.scripts-5.2.0-SNAPSHOT.jar, /opt/sybase/mobiliser/bundles/org.apache.felix.main-4.0.3.jar
  - Boot Class Path:** /usr/java/jdk1.7.0\_21/jre/lib/resources.jar, /usr/java/jdk1.7.0\_21/jre/librt.jar, /usr/java/jdk1.7.0\_21/jre/lib/sunrsasign.jar, /usr/java/jdk1.7.0\_21/jre/lib/jsse.jar, /usr/java/jdk1.7.0\_21/jre/lib/ce.jar, /usr/java/jdk1.7.0\_21/jre/lib/charsets.jar, /usr/java/jdk1.7.0\_21/jre/lib/jfr.jar, /usr/java/jdk1.7.0\_21/jre/classes
  - Library Path:** /usr/java/packages/lib/amd64, /usr/lib64, /lib64, /lib, /usr/lib

## Requests

The Requests option displays all requests made to the server, for example, transaction requests. You can drill down to see statistics for each request, such as the total number of requests, the success and failure count, and the average response time.



## Data

The Data option displays performance monitoring statistics for the selected server, such as EhCache statistics, connection, caching, and entity access. You can enable or disable performance monitoring.

The screenshot shows the 'Data' section of the Application Services Administration interface. The left sidebar contains a navigation menu with options: Server, Server List, Information, Requests, Data (selected), Channels, Events, and Tasks. The main content area is titled 'Hibernate Statistics' and has four tabs: 'EhCache Statistics', 'Connection', 'Caching', and 'Entity Access'. The 'EhCache Statistics' tab is active, showing an 'Enable' button. Below the tabs are three sections: 'General Settings', 'Counts', and 'Timing'. The 'General Settings' section shows 'Hibernate STAT Supported' set to 'true' and 'Region Caches Enabled' set to 'true'. The 'Counts' section displays various performance metrics, all with a value of 0: PrepareStatement Count, Query Execution Count, Close Statement Count, Session Open Count, Session Close Count, Transaction Count, Successful TXN Count, Flush Count, and Optimistic Failure Count. The 'Timing' section shows 'Max Request Duration (ms)' and 'Min Request Duration (ms)', both with a value of 0. The 'Query Execution Rate' is shown as 0.0.

## Channels

The Channels option displays the number of messages received, sent, and failed to send. The list shows the last 100 messages that were generated. You can select a message to view the

details, such as the date and time stamp. You can also refresh the list to show the most recently sent or received messages.

The screenshot displays the Mobiliser Administration web interface. At the top, there are navigation tabs for HOME, SERVERS, and TRACKERS. The left sidebar contains a menu with options: Server, Server List, Information, Requests, Data, Channels (highlighted), Events, and Tasks. The main content area is divided into two sections: Channels and Messages. The Channels section shows statistics: Available Channels [default], Messages Received 0, Messages Sent 7, and Messages Failed To Send 0. The Messages section, with a Reload button, lists several messages with details like Recipient, Sender, and Subject.

Channels	
Available Channels	[default]
Messages Received	0
Messages Sent	7
Messages Failed To Send	0

Messages	
MIME EMAIL MESSAGE	Recipient: johnnymerchant@live.com Sender: mobiliser@sybase.com Subject: Your new password Body: XXX
SMS Message	Recipient: +9705551234 Sender: 625477
SMS Message	Recipient: +9705551234 Sender: 625477
SMS Message	Recipient: 3075551234 Sender: 625477
SMS Message	Recipient: 9705551234 Sender: 625477
SMS Message	Recipient: 3075556789 Sender: 625477
SMS Message	Recipient: 13036216898 Sender: 625477

### Events

The Events option displays statistics generated by the event system. The event summary displays the total number of events that the event handlers have generated and processed. An event handler is a procedure that is called when a corresponding event occurs. For an event to be processed by a handler, there has to be an event handler registered for the event name and an available thread from the process pool as determined by the event handler. A single event handler instance is associated with a single event name only.



The screenshot shows the 'Events' page in the Administration interface. On the left is a sidebar with a menu: HOME, SERVERS, TRACKERS. Under 'SERVERS', there's a 'Server' section with details: Hostname: localhost, Port: 8080. Below that are links for Server List, Information, Requests, Data, Channels, Events (highlighted), and Tasks. The main content area is titled 'Events' and contains a 'Summary' section with four statistics: No. of Regular Events (684), No. of Transient Events (26507), No. of Regenerated Events (0), and No. of Scheduled Events (0). Below the summary are two tables. The first is 'Event Queues - Physical' with columns 'Queue Name', 'Current Size', and 'Maximum Size'. It lists DelayedQ (0/2), CatchupQ (0/0), and ProcessQ (0/3). The second is 'Event Queues - Virtual' with the same columns, listing InvoiceStatusAdviceTask (0/1) and UnlockLockedTransactions (0/1).

Event Queues

Event queues display lists of physical and virtual queues, and real-time counts of events in the queues.

**Table 26. Event Queues**

Queue	Description
Physical	Number of events that are in the physical queue at that instance in time. If the queue is empty, no events are pending for processing.
Virtual	Number of events in the virtual queue for each event, one virtual queue per event. If no virtual queues are shown, no events have been created.

Scheduled Events

The Scheduled Events option displays the internal scheduler-system view of all events that are scheduled. An empty list indicates there are no scheduled events.

Field Label	Description
Scheduled Event Id	Internal ID of the scheduled event.
Time Zone	An optional time zone that defines when the <b>cron</b> expression runs.
Cron Expression	An expression that conforms to UNIX <b>cron</b> standards for defining scheduled events.
End Time	Time after which no trigger fires; empty if not set.

Field Label	Description
Start Time	Time the first trigger fired.
Next Fire Time	Time the next trigger fires.
Last Fire Time	Last time the trigger fired.
Trigger	<p>A set of criteria that triggers an event:</p> <ul style="list-style-type: none"> <li>• Simple One-off – triggers the event once.</li> <li>• Cron Repeating – triggers the event at repeated intervals.</li> </ul>

### Event Handlers

The Event Handlers option displays a list of handlers that are registered with the event system.

Field Label	Description
Status	<p>Current status:</p> <ul style="list-style-type: none"> <li>• Listening – active and waiting to be notified when an event occurs.</li> <li>• Catchup – events are still processing.</li> </ul>
Event Name	Name of the event with which the handler is registered.
Current Active Thread	Number of threads that are currently running.
Current Idle Thread	Number of inactive threads that are allocated to this handler's thread pool.
Max Active Thread	Maximum size of the event handler's thread pool.
Max Idle Thread	Maximum number of inactive threads in the pool.
Total Number of Runs	<p>Number of times the task handler was invoked.</p> <hr/> <p><b>Note:</b> This number may be different from Total Events Processed. If the handler cannot get a processing status lock on an event, or if an event is expired, the handler cannot process the event.</p> <hr/>
Last Run At	Date and time of last run.
Total Events Processed	Number of events the handler processed.
Average Process Time (ms)	Average number of milliseconds the handler spent processing an event.

Field Label	Description
Total Events Success	Number of events the handler processed successfully; the <code>process</code> method returned true.
Total Events Fail	Number of events that failed; the <code>process</code> method returned false, or threw an exception.
Last Fail At	Date and time of last failed processing event.
Events Marked Expired	Number of events that expired before processing.
Events Marked Catch Up	Number of events that are still processing regenerated events.

### **Tasks**

The Tasks option displays statistics generated by the event system for tasks and task handlers. Tasks are internal date and time actions scheduled for execution at known repeated intervals. Tasks call task handlers. An empty task list indicates there are no scheduled events. A task is not directly related to an event, because tasks are not stored in the event system and do not require a task handler to generate historical events. However, the event system initiates and processes task actions.

The screenshot displays the 'Tasks' configuration page in the Mobiliser administration tool. On the left is a navigation menu with options like 'Server', 'Information', 'Requests', 'Data', 'Channels', 'Events', and 'Tasks'. The main content area is divided into two sections: 'Tasks' and 'Tasks Handlers'.

**Tasks**

Tasks Name	Cron Expression
UnlockLockedTransactions	0 0/5 * ? * *
InvoiceStatusAdviceTask	0 0/5 * ? * *
CancelExpiredTransactionTask	0 0/5 * ? * *
TermsConditionsCheckTask	00 30 23 ? * *
FileExportTask	0 0/5 * ? * *
CancelInitialTransactionsTask	0 0/5 * ? * *
CancelExpiredVouchersTask	0 0/5 * ? * *
SidTask	0 15 0/2 ? * *
InvoiceUpdateCreateTask	0 0/5 * ? * *
CronjobTask	0 0/1 * * * ?
CancelAuthWaitingTransactionsTask	0 0/5 * ? * *

**Tasks Handlers**

Handler Name	Status	Event
...m.sybase365.mobiliser.money.jobs.tasks.sid.SidTask	LISTENING	SidTask
...jobs.tasks.invoice.update.invoiceUpdateCreateTask	LISTENING	InvoiceUpdateCreateTask
...liser.money.jobs.tasks.cleanup.LockedTransactions	LISTENING	UnlockLockedTransactions
...jobs.tasks.invoice.status.invoiceStatusAdviceTask	LISTENING	InvoiceStatusAdviceTask
...money.jobs.task.cronjob.handler.CronjobTaskHandler	LISTENING	CronjobTask
...liser.money.jobs.tasks.cleanup.initialTransactions	LISTENING	CancelInitialTransactionsTask
...tasks.cancelexpired.CancelExpiredTransactionsTask	LISTENING	CancelExpiredTransactionTask
...365.mobiliser.money.ams.export.task.FileExportTask	LISTENING	FileExportTask
...bs.tasks.terms.conditions.TermsConditionsCheckTask	LISTENING	TermsConditionsCheckTask
...bs.tasks.expired.voucher.CancelExpiredVouchersTask	LISTENING	CancelExpiredVouchersTask
...r.money.jobs.tasks.cleanup.AuthWaitingTransactions	LISTENING	CancelAuthWaitingTransactionsTask

Task Details

Task Details display task information and statistics, such as task start times, next and last fire times, and triggers.

Field Label	Description
Scheduled Event Id	Internal ID of the scheduled event.
Time Zone	An optional time zone that defines when the <b>cron</b> expression runs.
Cron Expression	An expression that conforms to UNIX <b>cron</b> standards for defining scheduled events.
End Time	Time after which no trigger fires; empty if not set.
Start Time	Time the first trigger fired.

Field Label	Description
Next Fire Time	Time the next trigger fires.
Last Fire Time	Last time the trigger fired.
Trigger	A set of criteria that triggers an event: <ul style="list-style-type: none"> <li>• Simple One-off – triggers the event once.</li> <li>• Cron Repeating – triggers the event at repeated intervals.</li> </ul>

### Task Handlers

The Task Handlers option displays a list of existing task handlers, which are called by tasks. A task handler coordinates the activities of a task.

Field Label	Description
Status	Current status: <ul style="list-style-type: none"> <li>• Listening – active and waiting to be notified when an event occurs.</li> <li>• Catchup – events are still processing.</li> </ul>
Event Name	Name of the event with which the handler is registered.
Current Active Thread	Number of threads that are currently running.
Current Idle Thread	Number of inactive threads that are allocated to this handler's thread pool.
Max Active Thread	Maximum size of the event handler's thread pool.
Max Idle Thread	Maximum number of inactive threads in the pool.
Total Number of Runs	Number of times the task handler was invoked. <hr/> <b>Note:</b> This number may be different from Total Events Processed. If the handler cannot get a processing status lock on an event, or if an event is expired, the handler cannot process the event. <hr/>
Last Run At	Date and time of last run.
Total Events Processed	Number of events the handler processed.
Average Process Time (ms)	Average number of milliseconds the handler spent processing an event.
Total Events Success	Number of events the handler processed successfully; the <code>process</code> method returned true.

Field Label	Description
Total Events Fail	Number of events that failed; the process method returned false, or threw an exception.
Last Fail At	Date and time of last failed processing event.
Events Marked Expired	Number of events that expired before processing.
Events Marked Catch Up	Number of events that are still processing regenerated events.

## Trackers

The Trackers option lets you visually monitor system statistics through a series of charts, such as line, bar, and gauge. For example, memory usage is presented as a bar chart, and preauthorization of a transaction request is presented as a gauge chart.

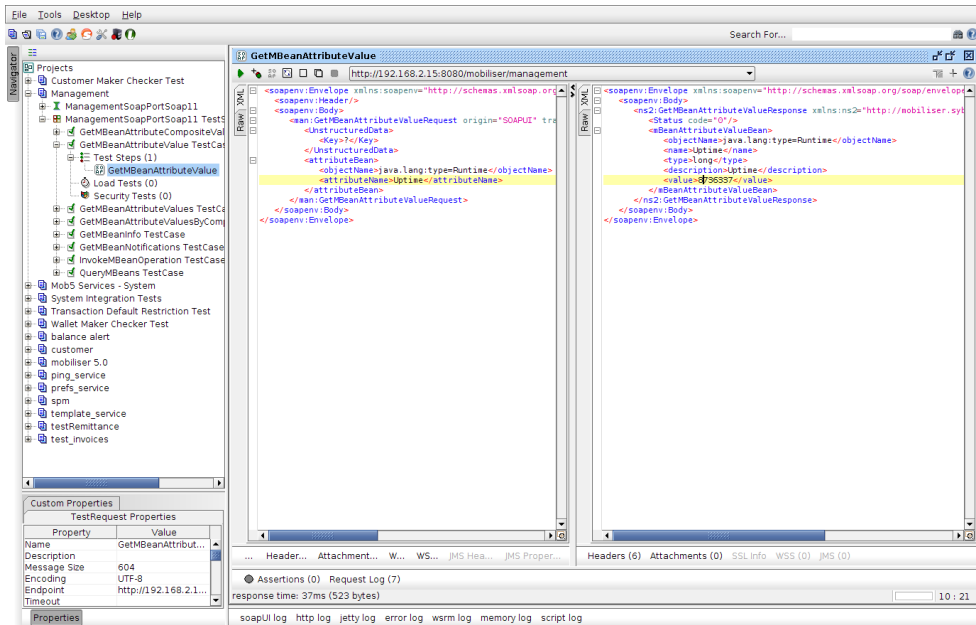
You can monitor and track status changes from a particular server information point. You can view trackers either as a summary on the All Trackers page, or individually on the View Tracker page.

**Note:** Dashboard developers can add trackers by configuring them in XML.



## SOAP/REST Interface Management

The Operations Dashboard presents Java Management Extensions (JMX) information, translates Simple Object Access Protocol (SOAP) requests into requests for local JMX platform objects and attributes, and sends them back as SOAP responses. The JMX interface can also be accessed via Representational State Transfer (REST), which returns either XML or JavaScript Object Notation (JSON) data.



## Data Archiving, Retention and Deletion

Currently, SAP Mobile Platform does not support data archiving, data retention, or deletion policies. It is therefore important that you use default database technology to implement any desired procedures and policies.

### Data Archiving

You can safely move transactional data from the online transactional database. However, SAP recommends that you do not move customer data, since this information is required in the transactional online database to ensure referential integrity. Customer data should be small compared to the amount of transactional data in the system. When archiving data from the online database, the data is no longer visible through the user interfaces.

**Note:** When moving transactional data, be aware for foreign-key constraints.

Choose a repository and move all information associated to a transaction, including the audit, history, and traceable request data:

- MOB\_TXNS
- MOB-SUB\_TXNS.ID\_TXN->MOB\_TXNS.ID\_TXN
- MOB\_TXN\_ATTRIBUTES.ID\_TXN->MOB\_TXNS.ID\_TXN
- MOB\_FEES.ID\_SUB\_TXN->MOB\_SUB\_TXNS.ID\_SUB\_TXN
- MOB\_HISTORY (tracks changes to individual columns in the database)
- MOB\_AUDIT\_LOGS (tracks each remote service call)
- MOB\_TRACEABLE\_REQUESTS (stores 24 hours of data for nonrepudiation and response dehydration)

If the transaction is an invoice payment, you must also move the invoice information:

- MOB\_INVOICES
- MOB\_INV\_TXNS.ID\_TXN->MOB\_TXNS.ID\_TXN
- MOB\_INV\_TXNS.ID\_INVOICE->MOB\_INVOICES.ID\_INVOICE
- MOB\_INV\_ATTRIBUTES.ID\_INVOICE->MOB\_INVOICES.ID\_INVOICE

### **Data Retention and Deletion**

SAP Mobile Platform does not include automated procedures to implement data retention and deletion policies. Set up cron jobs or manually perform tasks that delete data after the retention period has expired.

Since the database holds a number of referential integrity constraints that bind a customer record to transactions and other entities, you must encrypt the customer data instead of physically deleting it. That is, to delete the customer record from the system, overwrite any personally identifiable information (PII) with random text, for example, DELETED.

Customer data is stored in these tables:

- MOB\_CUSTOMERS
- MOB\_CUSTOMERS\_IDENTIFICATIONS.ID\_CUSTOMER->MOB\_CUSTOMER.ID\_CUSTOMER
- MOB\_CUSTOMERS\_CREDENTIALS.ID\_CUSTOMER->MOB\_CUSTOMER.ID\_CUSTOMER
- MOB\_CUSTOMERS\_IDENTITIES.ID\_CUSTOMER->MOB\_CUSTOMER.ID\_CUSTOMER
- MOB\_CUSTOMERS\_ATTRIBUTES.ID\_CUSTOMER->MOB\_CUSTOMER.ID\_CUSTOMER
- MOB\_ADDRESSES.ID\_CUSTOMER->MOB\_CUSTOMER.ID\_CUSTOMER
- MOB\_NOTES.ID\_CUSTOMER->MOB\_CUSTOMER.ID\_CUSTOMER
- MOB\_PIS.ID\_CUSTOMER->MOB\_CUSTOMER.ID\_CUSTOMER
- MOB\_PIS.ID\_PI->MOB\_WALLET->MOB\_MOB\_CUSTOMER.ID\_CUSTOMER



- MOB\_SVA.ID\_PI->MOB\_PIS.ID\_PI
- MOB\_CREDIT\_CARDS.ID\_PI->MOB\_PIS.ID\_PI
- MOB\_BANK\_ACCOUNTS.ID\_PI->MOB\_PIS.ID\_PI
- MOB\_EXTERNAL\_ACCOUNTS.ID\_PI->MOB\_PIS.ID\_PI

---

**Note:** Customized projects may introduce more tables holding PII.

---

## Deletion Script

Use the deletion script to remove and deactivate customer-related information, such as mobile phone numbers, passwords, PINs, and payment instruments. The deletion script also removes and marks customers as inactive in the system. Attachments, notes, and change history is also removed. Further processing of financial transactions is impossible.

```
-- delete information about bank accounts
UPDATE MOB_BANK_ACCOUNTS SET STR_NAME = '###', STR_NAME_BANK = '###',
STR_CITY_BANK = '###', STR_INSTITUTION_CODE = '###',
STR_BRANCH_CODE = '###', STR_ACCOUNT_NUMBER = '###',
STR_DISPLAY_NUMBER = '###' WHERE
ID_PI in (SELECT ID_PI FROM MOB_PIS WHERE ID_CUSTOMER = ?);
-- delete information about "other" financial accounts
UPDATE MOB_EXTERNAL_ACCOUNTS SET STR_ID1 = '###', STR_ID2 = '###',
STR_ID3 = '###', STR_ID4 = '###',
STR_ID8 = '###', STR_ID7 = '###', STR_ID6 = '###', STR_ID5 = '###'
WHERE
ID_PI in (SELECT ID_PI FROM MOB_PIS WHERE ID_CUSTOMER = ?);
-- delete credit card information
UPDATE MOB_CREDIT_CARDS SET STR_CARD_NUMBER = '###',
STR_CARD_HOLDER_NAME = '###', STR_DISPLAY_NUMBER = '###' WHERE
ID_PI in (SELECT ID_PI FROM MOB_PIS WHERE ID_CUSTOMER = ?);
-- delete names of accounts
UPDATE MOB_WALLET SET STR_ALIAS = '###' WHERE ID_CUSTOMER = ?;
-- mark all accounts as inactive
UPDATE MOB_PIS SET BOL_IS_ACTIVE = 'N' WHERE ID_CUSTOMER = ?;
-- delete all identifications, such as mobile phone number and make
them inactive
UPDATE MOB_CUSTOMERS_IDENTIFICATIONS SET STR_IDENTIFICATION = '###',
BOL_IS_ACTIVE = 'N' WHERE ID_CUSTOMER = ?;
-- delete all passwords and PINs and make them inactive
UPDATE MOB_CUSTOMERS_CREDENTIALS SET STR_CREDENTIAL = '###',
BOL_IS_ACTIVE = 'N' WHERE ID_CUSTOMER = ?;
-- delete all identity (e.g. passport) information
UPDATE MOB_CUSTOMERS_IDENTITIES SET STR_IDENTITY = '###',
STR_ISSUE_PLACE = '###', STR_ISSUER = '###', BOL_IS_ACTIVE = 'N'
WHERE ID_CUSTOMER = ?;
-- delete all general purpose attributes
UPDATE MOB_CUSTOMERS_ATTRIBUTES SET STR_VALUE = '###' WHERE
ID_CUSTOMER = ?;
-- delete all binary attachments
UPDATE MOB_ATTACHMENTS SET STR_NAME = '###', BIN_CONTENT = null WHERE
ID_CUSTOMER = ?;
-- delete all notes (system generated or manually entered)
UPDATE MOB_NOTES SET STR_SUBJECT = '###', STR_TEXT = '###' WHERE
ID_CUSTOMER = ?;
```

```
-- mark customer as inactive
UPDATE MOB_CUSTOMERS SET STR_DISPLAY_NAME = '###',
STR_SECURITY_QUESTION = '###', STR_SECURITY_ANSWER = '###',
STR_REFERRAL_CODE = '###', BOL_IS_ACTIVE = 'N', DAT_DATE_OF_BIRTH =
null WHERE ID_CUSTOMER = ?;
-- delete all address information
UPDATE MOB_ADDRESSES SET STR_FIRST_NAME = '###', STR_MIDDLE_NAME =
'###', STR_LAST_NAME = '###', STR_TITLE = '###', STR_COMPANY1 =
'###', STR_COMPANY2 = '###',
STR_COMPANY_SHORTNAME = '###', STR_POSITION = '###', STR_STREET1 =
'###', STR_STREET2 = '###', STR_HOUSE_NUMBER = '###', STR_ZIP =
'###', STR_CITY = '###',
STR_STATE = '###', STR_PHONE1 = '###', STR_PHONE2 = '###', STR_FAX =
'###', STR_EMAIL = '###', STR_URL = '###', STR_NAME_ADDRESS = '###'
WHERE ID_CUSTOMER = ?;
-- delete all information regarding change history
UPDATE MOB_HISTORY SET STR_OLD_VALUE = '###', STR_NEW_VALUE = '###'
WHERE ID_OBJECT = ?;
UPDATE MOB_HISTORY SET STR_OLD_VALUE = '###', STR_NEW_VALUE = '###'
WHERE ID_OBJECT in (SELECT ID_PI FROM MOB_PIS WHERE ID_CUSTOMER = ?);
UPDATE MOB_HISTORY SET STR_OLD_VALUE = '###', STR_NEW_VALUE = '###'
WHERE ID_OBJECT in (SELECT ID_ADDRESS FROM MOB_ADDRESSES WHERE
ID_CUSTOMER = ?);
UPDATE MOB_HISTORY SET STR_OLD_VALUE = '###', STR_NEW_VALUE = '###'
WHERE ID_OBJECT in (SELECT ID_IDENTITY FROM MOB_CUSTOMERS_IDENTITIES
WHERE ID_CUSTOMER = ?);
UPDATE MOB_HISTORY SET STR_OLD_VALUE = '###', STR_NEW_VALUE = '###'
WHERE ID_OBJECT in (SELECT ID_CUSTOMER_IDENTIFICATION FROM
MOB_CUSTOMERS_IDENTIFICATIONS WHERE ID_CUSTOMER = ?);
UPDATE MOB_HISTORY SET STR_OLD_VALUE = '###', STR_NEW_VALUE = '###'
WHERE ID_OBJECT in (SELECT ID_CUSTOMER_CREDENTIAL FROM
MOB_CUSTOMERS_CREDENTIALS WHERE ID_CUSTOMER = ?);
UPDATE MOB_HISTORY SET STR_OLD_VALUE = '###', STR_NEW_VALUE = '###'
WHERE ID_OBJECT in (SELECT ID_NOTE FROM MOB_NOTES WHERE ID_CUSTOMER
= ?);
-- commit all data
commit;
```

## Virus Protection

---

Antivirus software is one of the most important tools for safeguarding vital information and personal data from the daily onslaught of viruses and worms.

## Configuring the Virus Scan Adapter for SAP NetWeaver

---

Virus Scan Adapter for SAP NetWeaver<sup>®</sup>, which scans all files uploaded via Web services, is introduced with Mobiliser. The adapter uses a plug-in to connect to various virus scan engines that scan binary data.

For more details, see *Setting Up Virus Scan Providers* located at:

[http://help.sap.com/saphelp\\_nw04/helpdata/EN/ca/7cb340be761b07e10000000a155106/frameset.htm](http://help.sap.com/saphelp_nw04/helpdata/EN/ca/7cb340be761b07e10000000a155106/frameset.htm)

1. Install/copy the virus scan adapter for your virus scanner, which is provided by your virus scan vendor.

The NW-VSI integration bundle comes with a graphical configuration and test GUI, which is part of the VSI bundle:

```
SMP_HOME\Server\plugins\com.sap.security.nw.vsi_1.9.2.jar
```

2. Start the GUI:

```
java -jar com.sap.security.nw.vsi_1.9.2.jar
```

Test the connection with the European Institute for Computer Antivirus Research (EICAR) test pattern and mark the provider as the default. The SAP Mobile Platform engine uses only the default provider.

3. Open the configuration file:

```
SMP_HOME\Server\config_master\com.sybase365.mobiliser.framework.vsi
\com.sybase365.mobiliser.framework.vsi.properties
```

4. Restart the server and examine the `SMP_HOME\Server\log\hostname-smp-server.log` file.

## Configuring ClamAV on a Linux Server

ClamAV is a widely used virus scan engine for UNIX, which utilizes specialized packages to recognize viruses and other threats. You can use the ClamSAP library to connect the virus scan adapter to the ClamAV engine.

### Prerequisites

The SAP Mobile Platform virus scan adapter with ClamAV requires:

- **ClamAV** – is the virus scan engine and development package usually available from Linux distributors.
- **libclamsap** – is required to recognize potential threats and is used when SAP Mobile Platform can access a local ClamAv engine. The libclamsap is available from <http://sourceforge.net/projects/clamsap/files/>.

### Task

1. Enable the default adapter, then edit the adapter path to point to the libclamsap shared library:

```
SMP_HOME/Server/config_master/
com.sybase365.mobiliser.framework.vsi/
com.sybase365.mobiliser.framework.vsi.properties
(...)
vsi.provider.Virus_Scan_Adapter.Adapters.VSA_DEFAULT=VSA_DEFAULT
```

```
vsi.provider.Virus_Scan_Adapter.Adapters.VSA_DEFAULT.Active=true
vsi.provider.Virus_Scan_Adapter.Adapters.VSA_DEFAULT.AdapterPath=
/home/sybase/libclamsap.so
vsi.provider.Virus_Scan_Adapter.Adapters.VSA_DEFAULT.Description=
DEFAULT PROVIDER
vsi.provider.Virus_Scan_Adapter.Adapters.VSA_DEFAULT.Group=DEFAULT
vsi.provider.Virus_Scan_Adapter.Adapters.VSA_DEFAULT.PoolInstance
Timeout=3600
vsi.provider.Virus_Scan_Adapter.Adapters.VSA_DEFAULT.PoolMaxInsta
nces=50
vsi.provider.Virus_Scan_Adapter.Adapters.VSA_DEFAULT.ReInitTime=0

(...)
```

**2. Restart SAP Mobile Platform and examine the server log.**

The adapter is loaded successfully when you see these log lines in the *SMP\_HOME/Server/log/hostname-smp-server.log* file:

```
(...)
2012-09-06 08:43:48,747 [aims-init-15] DEBUG
com.sybase365.mobiliser.framework.vscan.scanner.impl.VScanImpl -
VSI Virus Scan Service initialization was successful
(...)
```

The virus scan installation fails if the adapter does not successfully load. Analyze the server log file to troubleshoot the issue.

## **cron Expression Reference**

A **cron** expression is a string comprised of six or seven fields separated by white space. Fields can contain any allowed values, including special characters. Expressions can be as simple as \* \* \* \* ? \* or as complex as 0/5 14,18,3-39,52 \* ? JAN,MAR,SEP MON-FRI 2002-2010.

**Table 27. cron Expression Format**

Field Name	Allowed Values	Allowed Special Characters
Seconds	0-59	, - * /
Minutes	0-59	, - * /
Hours	0-23	, - * /
Day-of-Month	1-31	, - * ? / L W
Month	1-12 or JAN-DEC	, - * /

Field Name	Allowed Values	Allowed Special Characters
Day-of-Week	1-7 or SUN-SAT	, - * ? / L #
Year (Optional)	empty, 1970-2199	, - * /

**Table 28. Special Characters**

Character	Description
*	Asterisks match all values. For example, * in the Minutes field means every minute.
?	Question marks mean "no specific value," and are allowed in both Day-of-Month and Day-of-Week fields. To indicate any day of the month or week, use a question mark instead of an asterisk.
-	Use hyphens to define a range. For example, 10-12 in the Hours field means the hours of 10, 11, and 12.
,	Commas separate items in a list. For example, "MON,WED,FRI" in the Day-of-Week field means the days Monday, Wednesday, and Friday.
/	Forward slashes indicate increments. For example, 0/15 in the Seconds field means every 15 seconds starting at 0. 1/3 in the Day-of-Month field means every 3 days starting on the first day of the month.
L	Abbreviation for last, L is allowed in both the Day-of-Month and Day-of-Week fields, with different meanings: <ul style="list-style-type: none"> <li>Day-of-Month – the last day of the month.</li> <li>Day-of-Week – either Saturday (7th day) or, if preceded by a digit (1–7), means the last of the month; for example, 6L, is the last Friday of the month.</li> </ul>
W	Abbreviation for weekday, W is allowed in the Day-of-Month field only. W represents the weekday nearest the given day. For example, 15W means the nearest weekday to the 15th of the month. Therefore, if the 15th is a Saturday, the job runs on Friday, the 14th. You can use both L and W characters in the Day-of-Month field. For example, LW means the last weekday of the month.
#	Hash marks specify constructs. For example, 6#3 in the Day-of-Week field means the third Friday of the month.

**Table 29. cron Expression Examples**

<b>Expression</b>	<b>Description</b>
0 0 12 * * ?	Runs at 12:00 p.m. (noon) every day
0 15 10 ? * *	Runs at 10:15 a.m. every day
0 15 10 * * ?	Runs at 10:15 a.m. every day
0 15 10 * * ? *	Runs at 10:15 a.m. every day
0 15 10 * * ? 2005	Runs at 10:15 a.m. every day during the year 2005
0 * 14 * * ?	Runs every minute starting at 2:00 p.m. and ending at 2:59 p.m., every day
0 0/5 14 * * ?	Runs every 5 minutes starting at 2:00 p.m. and ending at 2:55 p.m., every day
0 0/5 14,18 * * ?	Runs every 5 minutes starting at 2:00 p.m. and ending at 2:55 p.m.; and runs every 5 minutes starting at 6:00 p.m. and ending at 6:55 p.m., every day
0 0-5 14 * * ?	Runs every minute starting at 2:00 p.m. and ending at 2:05 p.m., every day
0 10,44 14 ? 3 WED	Runs at 2:10 p.m. and at 2:44 p.m. every Wednesday in the month of March
0 15 10 ? * MON-FRI	Runs at 10:15 a.m. every Monday, Tuesday, Wednesday, Thursday and Friday
0 15 10 15 * ?	Runs at 10:15 a.m. on the 15th day of every month
0 15 10 L * ?	Runs at 10:15 a.m. on the last day of every month
0 15 10 L-2 * ?	Runs at 10:15 a.m. on the 2nd-to-last last day of every month
0 15 10 ? * 6L	Runs at 10:15 a.m. on the last Friday of every month
0 15 10 ? * 6L 2002-2005	Runs at 10:15 a.m. on every last Friday of every month during the years 2002, 2003, 2004 and 2005
0 15 10 ? * 6#3	Runs at 10:15 a.m. on the third Friday of every month
0 0 12 1/5 * ?	Runs at 12:00 p.m. (noon) every 5 days every month, starting on the first day of the month
0 11 11 11 11 ?	Runs every November 11 at 11:11 a.m.

# SMS Administration

SMS administrators configure SMS Builder components and ensure that the production system works efficiently as a result of that configuration.

---

**Note:** If you install the SMS Builder SDK, the SMS administration features are also installed.

---

SMS Builder defines three types of administrators: system administrators, platform administrators, and workspace administrators.

Typically, system administrators:

- Are IT professionals
- Install software
- Secure production systems
- Start and stop the server
- Configure server properties
- Monitor and tune the system

A platform administrator:

- Is created in the production database when you run the database scripts
- Has the SUPER\_ADMIN role; user name is admin
- Is the first user to log in to the SMS Builder Web UI
- Creates workspaces and default menus
- Adds and configures users
- Sets up communication channels

Workspace administrators:

- Are created by the platform administrator: one for each workspace
- Have the ADMIN role
- In their workspaces, create users, manage default menus and categories, and generate reports

## See also

- *Application Administration* on page 31
- *Server Administration* on page 113
- *Security Administration* on page 149
- *Application Services (Mobiliser) Administration* on page 261

## Setting Up a Production System

---

Install SMS Builder on a Linux or Windows machine to run in a production system.

Some steps vary depending on enterprise IT policies and deployment landscape; use them as a starting point for consultation and discussion with your system architect.

### Hashing the Admin Password

Run the password-hashing tool for the admin user password, and insert it into the database script that creates the admin user. The hashed password is stored in the database.

To comply with SAP® security requirements, the script for creating the admin user is disabled. The admin password in the `/sql/common/current/04-BrandMobiliser-Base-Data.sql` script is a hashed version of `Brand!23`. Replace this password with a new hashed password.

1. To generate a hashed password, navigate to the `/bin` directory, and run:

```
./passwordtool.sh password
```

where *password* is the password to hash.

2. Edit the `04-BrandMobiliser-Base-Data.sql` script, uncomment these lines, and replace `$2a$10$xVTSvw3hcCFtZ2DnMav.Te/WsOMBtLC1MV0QLi/z.ziUyJY/T.d0i` with the newly hashed password:

```
--INSERT INTO M_USERS (ID, USERNAME, PASSWORD, DISABLED)
--VALUES (1, 'admin', '$2a$10$xVTSvw3hcCFtZ2DnMav.Te/WsOMBtLC1MV0QLi/
z.ziUyJY/T.d0i', 0);
```

### Configuring New Database Installations

SMS Builder installs with an embedded Apache Derby database that you can use for development and testing. For production, use either an IBM DB2 or an Oracle database.

#### Prerequisites

Install a DB2 or an Oracle database. SAP Product Availability Matrix (PAM) <http://service.sap.com/pam>. Click the **Mobile** link at the top of the page. Scroll to find the appropriate product and version in the product list.

#### Task

A database administrator (DBA) should install, configure, and support a production database. Installation procedures vary by database platform; review them with your DBA. Run database scripts to create a tablespace, database objects, and the initial data.



1. Review the database scripts that are included with the product for compliance with corporate policies.
2. Perform the appropriate tasks for your database:

Database	Perform These Tasks
IBM DB2	<ol style="list-style-type: none"> <li>1. Configure the DB2 Database.</li> <li>2. Enable the DB2 JDBC Driver.</li> </ol>
Oracle	<ol style="list-style-type: none"> <li>1. Configure the Oracle Database.</li> <li>2. Install the Oracle JDBC Driver.</li> <li>3. Enable the Oracle JDBC Driver.</li> </ol>

---

**Note:** Derby scripts are provided for a standalone Derby installation, also known as a network server. Test and certify a Derby network server installation before deploying it to a production environment.

---

### See also

- *Configuring DB2 Databases* on page 343
- *Enabling the DB2 JDBC Driver* on page 345
- *Configuring Oracle Databases* on page 345
- *Installing Oracle JDBC Drivers* on page 346
- *Enabling the Oracle JDBC Driver* on page 347

### **IBM DB2 Database**

You can use an IBM DB2 database for a production system.

Install an IBM DB2 database version that has been tested and certified for SMS Builder.

#### Configuring DB2 Databases

Configure an IBM DB2 database for new SMS Builder installations.

### Prerequisites

1. Install the DB2 database.
2. Create a new hashed password for the admin user, and save it in the `/sql/common/current/04-BrandMobiliser-Base-Data.sql` script.

### Task

To complete this task, you must either be a root user or have sudo privileges. DB2 database scripts are located in the `SMSBUILDER_HOME/sql/db2/current` directory.

1. To add system groups, on the command line, run:

```
groupadd -g 999 db2iadml
groupadd -g 998 db2fadml
groupadd -g 997 dasadml
```

**2. To add system users, run:**

```
useradd -u 1004 -g db2iadml -m -d /home/mwiz2 mwiz2
useradd -u 1003 -g db2fadml -m -d /home/db2fenc1 db2fenc1
useradd -u 1002 -g dasadml -m -d /home/dasusr1 dasusr1
```

**3. For each new user, change the password:**

```
passwd mwiz2
passwd db2fenc1
passwd dasusr1
```

The default password for each user is “sql.”

**4. Create a DB2 instance:**

```
$ pwd
/opt/IBMDB2/V9.7/instance/
$ ./db2icrt -a server -u db2fenc1 mwiz2
```

**5. Verify the database installation:**

a) Log in as `mwiz2`, using the password set in step 3.

b) Start the database manager:

```
db2star
```

c) Create the sample database:

```
db2sampl
```

d) Start the DB2 command line processor:

```
db2
```

e) Connect to the sample database:

```
connect to sample
```

**6. Create the **brandmob** database and grant DBA privileges to the `mwiz2` user, by running:**

```
01-BrandMobiliser-DB_DB2.sql
```

Ensure that the current DB2 instance has write privileges to the data directory, which the script assumes is `/home/db2`.

**7. Connect to the newly created **brandmob** database as the `mwiz2` user:**

```
db2 connect to brandmob user mwiz2 using sql
```

**8. Insert initial data and create the admin user:**

```
04-BrandMobiliser-Base-Data.sql
```

### Next

Enable the DB2 JDBC driver.

### See also

- *Configuring New Database Installations* on page 342
- *Enabling the DB2 JDBC Driver* on page 345

### Enabling the DB2 JDBC Driver

To use an IBM DB2 database, enable the JDBC driver that is installed with SMS Builder.

#### **Prerequisites**

Configure the DB2 database.

#### **Task**

1. Open the `SMSBUILDER_HOME/conf/cfgbackupservice.dsprovider.properties` file.
2. Remove the comment indicators from the DB2 configuration, and enter the password you created for the `mwiz2` user:
  - If the password is encrypted, replace `XXX` with the encrypted password.
  - If the password is plain text, replace `{enc}XXX` with the password.

```
# DB2
driverClassName=com.ibm.db2.jcc.DB2Driver
url=jdbc:db2://localhost:60000/brandmob
username=mwiz2
password={enc}XXX
validationQuery=select 1 from sysibm.sysdummy1
```

3. Comment out the Derby configuration, and save the file:

```
# DERBY EMBEDDED - For development and Testing only!!!!
#driverClassName=org.apache.derby.jdbc.EmbeddedDriver
#url=jdbc:derby:mwiz2
#username=mwiz2
#password=sql
#validationQuery=select 1 from sysibm.sysdummy1
```

#### **See also**

- *Configuring New Database Installations* on page 342
- *Configuring DB2 Databases* on page 343

### **Oracle Database**

You can use an Oracle database in a production system.

Install an Oracle database version that has been tested and certified for SMS Builder.

### Configuring Oracle Databases

Configure an Oracle database for new SMS Builder installations.

#### **Prerequisites**

1. Install an Oracle database.

2. Create a new hashed password for the admin user, and save it in the `/sql/common/current/04-BrandMobiliser-Base-Data.sql` script.

### Task

Consult your DBA for a custom installation using an existing tablespace or user for the SMS Builder schema.

1. Edit the `SMSBUILDER_HOME/sql/oracle/current/02-BrandMobiliser-Users.sql` script, and replace XXX with a password for the mwiz2 user:

```
create user mwiz2 identified by XXX;
```

2. As a user with DBA privileges, log in to the database.
3. Create a tablespace:

```
/sql/oracle/current/01-BrandMobiliser-Tablespaces.sql
```

4. Create the mwiz2 user:

```
/sql/oracle/current/02-BrandMobiliser-Users.sql
```

5. As the mwiz2 user, create a schema:

```
/sql/oracle/current/03a-BrandMobiliser-Objects.sql
```

Perform all remaining steps as the mwiz2 user.

6. Create another schema:

```
oracle/current/03b-BrandMobiliser-Objects.sql
```

7. Insert initial data and create the admin user:

```
/sql/common/current/04-BrandMobiliser-Base-Data.sql
```

8. Configure the JDBC connection in `/conf/cfgback/service.dsprovider.properties`.

### Next

Install the Oracle JDBC driver.

### See also

- *Configuring New Database Installations* on page 342
- *Installing Oracle JDBC Drivers* on page 346
- *Enabling the Oracle JDBC Driver* on page 347

### Installing Oracle JDBC Drivers

To use an Oracle database, install an Oracle JDBC driver.

1. Download the Oracle JDBC driver from the Oracle Web site at <http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html>.
2. Unzip the downloaded package and find `ojdbc6.jar`.

3. Copy `ojdbc6.jar` to the `thirdparty/oracle` directory.

There should be a manifest file in the directory called `oraclemanifest`.

4. Update the manifest in `ojdbc6.jar` with `oraclemanifest`:

a) Change to the `thirdparty/oracle` directory.

b) Run:

```
jar -umvf oraclemanifest ojdbc6.jar
```

5. Copy the modified `ojdbc6.jar` to the `bundle/application` directory, and rename it `oracle-jdbc-osgi_11.2.0.2.0-1.0.0.jar`:

```
cp ojdbc6.jar ${BRAND_HOME}/bundle/application/oracle-jdbc-osgi_11.2.0.2.0-1.0.0.jar
```

where `BRAND_HOME` is set to the SMS Builder installation directory.

6. Configure the location of the Oracle JDBC driver:

a) Open the `conf/config.properties` file.

b) Under the `#JDBC drivers` section, add this line:

```
${aims.app.dir}/oracle-jdbc-osgi_11.2.0.2.0-1.0.0.jar
```

### Next

Enable the Oracle JDBC driver.

### See also

- *Configuring New Database Installations* on page 342
- *Configuring Oracle Databases* on page 345
- *Enabling the Oracle JDBC Driver* on page 347

### Enabling the Oracle JDBC Driver

To use an Oracle database, enable the Oracle JDBC driver that you installed.

### Prerequisites

- Configure an Oracle database.
- Install an Oracle JDBC driver.

### Task

1. Open the `SMSBUILDER_HOME/conf/cfgbackup/service.dsprovider.properties` file.
2. Remove the comment indicators from the Oracle configuration, and enter the password you created for the `mwiz2` user:
  - If the password is encrypted, replace `XXX` with the encrypted password.

- If the password is plain text, replace {enc}XXX with the password.

```
# Oracle
driverClassName=oracle.jdbc.driver.OracleDriver
url=jdbc:oracle:thin:@localhost:1521:xe
username=mwiz2
password={enc}XXX
validationQuery=select 1 from DUAL
```

### 3. Comment out the Derby configuration, and save the file:

```
# DERBY EMBEDDED - For development and Testing only!!!!
#driverClassName=org.apache.derby.jdbc.EmbeddedDriver
#url=jdbc:derby:mwiz2
#username=mwiz2
#password=sql
#validationQuery=select 1 from sysibm.sysdummy1
```

### See also

- *Configuring New Database Installations* on page 342
- *Configuring Oracle Databases* on page 345
- *Installing Oracle JDBC Drivers* on page 346

## Enabling Encryption

After you enable encryption, you can encrypt any property value. By default, encryption is disabled.

1. Open the `SMSBUILDER_HOME/conf/system.properties` file.
2. Uncomment this line, and replace `samplesecret` with a new value for `decryptionkey`:

```
# com.sybase365.arf.container.system.decryptionkey=samplesecret
```

3. (Optional) Uncomment this line, and set `decryptionkeylength` to a higher value, for example, 256 or 512, and save the file:

```
# com.sybase365.arf.container.system.decryptionkeylength=128
```

- If the line remains commented out, the default value, 128, is used.
  - If you uncomment the line, and set the value to 256 or higher, you must install the Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 6. See <http://www.oracle.com/technetwork/java/javase/downloads/jce-6-download-429243.html>. Do not set the decryption key length to a value greater than 2048.
4. Stop and restart the server.

### See also

- *Encrypting Property Values* on page 349

## Encrypting Property Values

You can encrypt property values in configuration files using an encryption tool. Passwords are the most commonly encrypted values, but you can encrypt any value.

### Prerequisites

Enable encryption.

### Task

1. Go to `SMSBUILDER_HOME/bin`, and run:

```
../encrypt.sh decryptionValue password
```

where:

- `decryptionValue` is the value of `decryptionkey`, defined in `system.properties`.
- `password` is the property value to encrypt.

The result of running the `encrypt.sh` command is an encrypted and Base64-encoded value, for example:

```
x9CJt8qwgXSuDn7FY21mVOIZ1WiwIRiJoX9CatjqQiYRPj7NdeAchigFmxQKX4
```

2. In the configuration file, enter the encrypted value, prepended with `{enc}`, for example:

```
password={enc}x9CJt8qwgXSuDn7FY21mVOIZ1WiwIRiJoX9CatjqQiYRPj7NdeAchigFmxQKX4
```

`{enc}` alerts the configuration manager to unencrypt the property value before using it.

### See also

- *Enabling Encryption* on page 348
- *Enabling SSL* on page 350

## Configuring Authentication

By default, when you log in to the Web UI, authentication is performed using the built-in database model (`authentication.bean=AuthenticationManager`). You can reconfigure authentication to use an LDAP system.

1. Open the `SMSBUILDER_HOME/conf/cfgbackup/service.webui.security.properties` file.
2. Set the value of `authentication.bean` to either:
  - `AuthenticationManager` – database authentication, database-role authorization, or

- `LdapAuthenticationManager` – LDAP authentication, database-role authorization.
3. If you set the value of `authentication.bean` to `LdapAuthenticationManager`:
    - a) Add these lines to the file:

```
username=admin  
pwd=brandldap
```
    - b) Reconfigure the `ldap.*` properties to connect to the LDAP system provided by your enterprise IT administrator.

```
ldap.host=localhost  
ldap.port=389  
ldap.userpath=uid={uid},ou=people,o=sybase365  
ldap.security.authentication=SIMPLE
```
  4. Stop and restart the server.

### **Configuring the Event Scheduler JDBC Driver**

To manage events that trigger event applications, configure the JDBC database driver that the event scheduler uses. The event scheduler is based on the Quartz scheduler.

The event scheduler persists schedules in the database. The event scheduler uses a JDBC driver different from the one SMS Builder uses, and you must configure it separately.

1. Open the `SMSBUILDER_HOME/conf/cfgbackup/service.event.quartz.properties` file.
2. For your database, enable the JDBC driver, and save the file:
  - For Oracle databases, uncomment this line:

```
#jobStore.driverDelegateClass=org.quartz.impl.jdbcjobstore.oracle.Oracle  
Delegate
```

- For most other databases, use the default driver:

```
jobStore.driverDelegateClass=org.quartz.impl.jdbcjobstore.StdJDBCDelegate
```

- For other databases that do not work with the default driver, see the Quartz documentation at <http://www.quartz-scheduler.org/documentation/quartz-1.x/configuration/ConfigJobStoreTX>.

### **Enabling SSL**

Enable SSL for the Web UI (HTTPS).

SMS Builder embeds Jetty for its `javax.servlet` container capability. Configure Jetty for SSL, and use the X.509 certificate, which SAP® recommends.

1. Create a keystore if one does not yet exist:
  - a) On the command line, enter:



```
keytool -keystore keystore -alias jetty -genkey -keyalg RSA
```

- b) Follow the onscreen instructions. Enter the first and last name to match your machine host name.
- c) Copy the keystore file to the `SMSBUILDER_HOME/conf/keystore` directory.

2. In the `conf/cfgbackup` directory, create an `org.ops4j.pax.web.properties` file (if it does not already exist), and add these lines:

```
# Enable SSL
org.osgi.service.http.secure.enabled=true

# SSL Port
org.osgi.service.http.port.secure=8443

# Keystore created to hold SSL certificate
org.ops4j.pax.web.ssl.keystore=conf/keystore

# Keys to access Keystore and SSL certificate
org.ops4j.pax.web.ssl.password=password
org.ops4j.pax.web.ssl.keypassword=keypassword
```

3. To encrypt the properties `org.ops4j.pax.web.ssl.password` and `org.ops4j.pax.web.ssl.keypassword`, run the encryption tool.
4. Enter the encrypted passwords, as in the example, below:

```
# Keys to access Keystore and SSL certificate
org.ops4j.pax.web.ssl.password={enc}cMYSsdsyRNzhyKlrbZbLIUH1z0tux5jyXWxPn76RlU=
org.ops4j.pax.web.ssl.keypassword={enc}$2a$10$xVTSvw3hcCFtZ2DnMav.Te/WsOMBtLC1MV0QLi
```

5. Stop and restart the server.
6. Verify the connection at `https://hostname:8443/brand`, where *hostname* is the name of the machine on which the server is running.

For more information about configuring Jetty for SSL, see <http://www.eclipse.org/jetty/documentation/current/>

### See also

- *Encrypting Property Values* on page 349

## Communication Channels

---

A communication channel is the conduit for receiving inbound and delivering outbound messages.

Short Message Peer-to-Peer (SMPP) – connections are used to send and receive SMS messages.

Built-in channel types are:

- SmsOutDummy – the loopback channel, which is used for simulation tests in a development environment, and requires no configuration. When a new workspace is created, the SMS outbound channel is automatically assigned to SmsOutDummy.
- Short Message Peer-to-Peer (SMPP) – uses the SMPP protocol to deliver inbound and outbound messages to short message service centers (SMSC) and external short messaging entities.
- Java Message Service (JMS) – delivers inbound and outbound messages via the message-oriented middleware.

The platform administrator can configure SMPP (inbound and outbound) and JMS channels to have one or more connections, and set these connections to active or inactive. An active connection on an inbound channel receives incoming messages. An active connection on an outbound channel delivers outgoing messages. Channels are defined at the platform level, and are shared by all workspaces.

### Configuring SMPP Inbound Channels

Incoming messages on an SMPP inbound connection can target any workspace. The destination workspace is determined by the short code of the incoming message.

1. In the Web UI navigation bar, select **Workspace Administration**, then select **Manage Channel Configurations**.
2. On the Manage Channels screen, select the **SMPP IN** tab, and enter values for these parameters:

Parameter	Description
Name	Unique name for the channel.
URL	Host name of the short message service center (SMSC).
Port	Port number for the listener.
Username	User name for authentication.
Password	Valid password for the user name.
System type	Identifier for SMSC.
Keep Alive (ms)	Frequency for sending enquire-link requests, in milliseconds; required to keep the SMPP connection alive. SMPP protocol setting.
Active	To activate the channel, select the check box; to deactivate, unselect the check box.

---

**Note:** Adding, deleting, or modifying a channel requires that you restart all active channels, which impacts all running applications in all workspaces.

---

3. To restart channels, select **Channel Actions > Restart Channels**.

**See also**

- *Configuring SMPP Outbound Channels* on page 353

**Configuring SMPP Outbound Channels**

SMPP outbound connections are specific to a workspace—one connection per workspace. SMS Builder makes outbound connections to short message service centers (SMSC) and SMS gateways.

1. On the Manage Channels screen, select the **SMPP OUT** tab, and enter values for these parameters:

Parameter	Description
Name	Unique name for the channel.
URL	Host name of the short message service center (SMSC).
Port	Port number to create a connection to.
Username	User name for authentication.
Password	Valid password for the user name.
System type	Identifier for the SMSC.
Dest Ton	Destination-number type.
Dest Npi	Destination numbering plan identification.
Src Ton	Source-number type.
Src Npi	Source numbering plan identification.
Delay (ms)	Message delay before sending next message, in milliseconds.
Keep Alive (ms)	Frequency for sending enquire-link requests, in milliseconds; required to keep the SMPP connection alive. SMPP protocol setting.
Permanent	Whether connections persist or a new connection is created for each message. To persist connections, select the check box; to create new connections, unselect the check box.
Active	Whether this channel is active or not. To activate, select the check box; to deactivate, unselect the check box.

**Note:** Adding, deleting, or modifying a channel requires that you restart all active channels, which impacts all running applications in all workspaces. In addition, when you switch a channel from “active” to “not active,” the channel is detached from its workspaces during the restart process, leaving these workspaces without an outbound channel.

2. To restart channels, select **Channel Actions > Restart Channels**.

**See also**

- *Configuring SMPP Inbound Channels* on page 352

**Configuring JMS Channels**

A single JMS channel supports both inbound and outbound traffic.

The JMS queue names define whether messages on the queue are inbound or outbound. The SAP JMS message format is proprietary. To use this channel mechanism, consult with your SAP Support contact.

1. On the Manage Channels screen, select the **JMS Connector** tab, and enter values for these parameters:

Parameter	Description
Name	Unique name for the channel.
URL	JMS broker to connect to.
Username	User name for authentication.
Password	Valid password for the user name.
In Queue	Queue name to look for inbound messages.
Out Queue	Queue name to look for outbound messages.
Active	To activate the channel, select the check box; to deactivate, unselect the check box.

**Note:** Adding, deleting, or modifying a channel requires that you restart all active channels, which impacts all running applications in all workspaces. In addition, when you switch a channel from “active” to “not active,” the channel is detached from its workspaces during the restart process, leaving these workspaces without an outbound channel.

2. To restart channels, select **Channel Actions > Restart Channels**.

**Workspaces**

A workspace is a logical grouping of applications, users, and other artifacts. When you install SMS Builder, the default workspace, which you can use for development, is created automatically.

Workspaces meet both development and deployment needs. In the development environment, a workspace provides the logical grouping of users who are collaborating on projects or tasks. A workspace can also be used for partitioning development, QA, and production environments

After you create a workspace, you can assign users to it. You can assign a user to more than one workspace. The default installation of the platform has a predefined workspace called

default and a user called `admin`, who is assigned to the `default` workspace. The `admin` user has the `SUPER_ADMIN` role, and is the platform administrator.

---

**Warning!** Do not delete the `default` workspace or the `admin` user.

---

In a deployment environment, you must configure a workspace with at least one unique short or long code. Short codes are special telephone numbers, significantly shorter than full telephone numbers, which you can use to address SMS and MMS messages from some mobile phones and fixed phones, and are limited to national borders. Long codes are longer numbers that you can use for international calls. The processing engine uses unique short codes or long codes to dispatch incoming messages to a corresponding workspace. In the remainder of this topic, code refers to either a short code or a long code.

You can assign more than one code to a workspace, but you cannot use a code in multiple workspaces. If there is more than one code assigned to a workspace, the processing engine uses the code flags, `Default` and `Use for Reply`, to determine which code to assign to the **Origination MSISDN** property in responses (outbound messages). As an example, a workspace has short codes: A, B, and C. When an inbound message is sent to any of these short codes, the processing engine sets the value of the **Origination MSISDN** property in the response, based on the status of the short code's flags. Mobile operators require **Origination MSISDN** values.

Short Code	Selected Short-Code Flag	Value Assigned to Origination MSISDN in Responses	Reason for Origination MSISDN Value
A	Default	A	A is the default short code.
B	Use for Reply	B	The Use for Reply flag is selected.
C	None	A	The Use for Reply flag is not selected, so the system selects the default short code.

Each workspace can have only one outbound channel connection. Inbound channels are not workspace specific. An incoming message from any channel is evaluated and routed to its corresponding workspace, based on the destination short or long code in the message.

For best results:

- Do not assign any users to the `default` workspace. Preserve the initial setup of the workspace (assigned to the `admin` user). Treat the `default` workspace as a guest workspace, so when users are inadvertently assigned to it, no damage is done.
- Do not assign any short codes to the `default` workspace, because short codes cannot be reused across workspaces.
- To prevent unintended traffic flow, do not set up any channels on the `default` workspace.

## Creating Workspaces

A workspace is a logical grouping of applications, users, and, other artifacts. In addition to a unique name, a workspace also has a unique short or long code.

1. To create a workspace, on the Web UI navigation bar, select **Workspace Administration**.
2. In the Workspace and User list, select **Manage Workspaces**.
3. On the Workspaces screen, select **Add New Workspace**.
4. On the Manage Workspace screen, enter values for these parameters, and click **Save**:
  - Name – unique name of the workspace.
  - Short Name – name and short name can be the same, but they must be unique in the system.
  - Select an outbound channel from the list.
5. Select the **Shortcode** tab, and enter a short code.  
After you add a short code to a workspace, all incoming messages with a destination MSISDN equal to the short code are sent to the workspace.
6. (Optional) To determine the value of the **Origination MSISDN** property in outbound messages, set one of these flags:
  - **Default** – if this flag is set and the Use for Reply flag of the incoming-message's short code is not set, the value of **Origination MSISDN** in outbound messages is set to this short code
  - **Use for Reply** – if this flag is set for the short code to which the inbound message is sent, the value of **Origination MSISDN** in outbound messages is set to this short code.
7. Click **Add**.  
The short code is added to the workspace. To add another short code, repeat steps 5–7.

You can edit an existing workspace by selecting it from the Manage Workspace screen.

### **See also**

- *Opening Workspaces* on page 356
- *Configuring Default Menus* on page 358
- *Default Menu* on page 357

## Opening Workspaces

Workspace administrators can assign Web UI users to multiple workspaces, and users can open their workspaces individually. After logging in to the Web UI, one of your workspaces opens. The name of the current workspace appears on the Login Status Bar.

1. To open a different workspace, on the Login Status Bar, expand the **Workspace** list, and select the workspace.

2. To confirm the change, click **OK**.

The new workspace opens, and the main window (Dashboard) appears. Each workspace has unique artifacts, such as short codes, channels, and applications. By design, you cannot see or access resources from multiple workspaces at the same time.

### See also

- *Creating Workspaces* on page 356
- *Configuring Default Menus* on page 358
- *Default Menu* on page 357

## Default Menu

A default menu responds to keywords that are sent to a workspace but not assigned to an application. Each workspace must have a default menu. Typically, the workspace administrator sets up the default menu.

Responses on a default menu are in menu form. For example, in a workspace with three interactive applications, the response message for an unrecognized keyword may be:

Choose: 1 - for banking; 2 - for payment; 3 - for weather.

When you set up a default menu, you associate applications with the menu. The menu is generated automatically. When a workspace receives an unrecognized keyword, it sends an automatically generated response to the mobile consumer, using default-menu settings. Mobile consumers can respond with options in the menu index.

Functionalities available on the Setup Default Menu screen are:

- **Default Menu** – displays the menu that is sent in response to unrecognized keywords. The order of the menu items determines the generated menu indexes. You can change the order using the up and down arrows to move applications. Application names are links to the corresponding application screens.
- **Add Application to Default Menu** – provides a list of all applications that can be added to the default menu. The list displays all interactive applications that are currently active. For information about developing applications, see *SMS Application Development*.
- **Response Messages** – displays two messages: the text that is prepended to the default menu, and the message that is sent as a response when the default menu is empty. The prepended text can be as simple as “Choose,” or a welcome message. When there are no applications assigned to the default menu, an alternate response is sent to mobile subscribers, for example, a keyword that is valid for one of the applications in the workspace.
- **Activate Default Menu** – when you create a new workspace, you must activate the default menu.

### See also

- *Creating Workspaces* on page 356

- *Opening Workspaces* on page 356
- *Configuring Default Menus* on page 358

## **Configuring Default Menus**

Set up and activate a default menu for each workspace. When a consumer selects an option in the default menu, the associated application starts.

A default menu can have a maximum of five menu items.

1. In the Web UI navigation bar, select **Actions > Set Up Default Menu**.
2. Add an application to the default menu:
  - a) Select an application from the list.  
The selected application appears as the last item in the default menu.
  - b) Click **Add to Menu**.  
When there are five applications in the menu, Add to Menu is disabled.
3. (Optional) Change the order of applications in the menu:
  - a) In the Default Menu list, select an application, and click the up or down arrow to change the order.
  - b) To remove an application from the list, select the **X** that corresponds to the application.
4. (Optional) Edit the response message:
  - a) Under Text Prepended to Menu, enter explanatory text that mobile consumers see above the list of applications in the menu.
  - b) Under Message when Default Menu is Empty, enter the text that mobile consumers see when there are no applications in the menu, for example, a keyword that is assigned to an application in the workspace.
  - c) Click **Save**.
5. Click **Activate**.

Any subsequent changes you make to the default menu require you to reactivate the menu before the changes take effect.

---

**Note:** The default menu is designed to be active at all times. If there is an issue, you must either fix it, or remove all linked applications, so that no menu is created. As a last resort, stop the default menu by disconnecting the outbound channel from the workspace.

---

### **See also**

- *Creating Workspaces* on page 356
- *Opening Workspaces* on page 356
- *Default Menu* on page 357



## **Deleting Workspaces**

Before you delete a workspace, remove all applications running in the workspace and unassign all users from the workspace.

### **Prerequisites**

Back up the database.

### **Task**

---

**Note:** This task can be performed only by a system administrator or a DBA.

---

1. In the Web UI, log in as a user who has the ADMIN role, and who is assigned to the workspace you want to delete.
2. Navigate to the Set Up Default Menu page.
  - a) Remove all applications assigned to the default menu.
  - b) Approve the default menu.
3. Go to the Assets page, and remove all applications from the workspace.
4. Navigate to the Manage User page, and unassign all users from this workspace, including the user who logged in.

If the logged-in user is assigned only to this workspace:

- a) Assign the logged-in user to a different workspace.
  - b) Unassign the logged-in user from the current workspace.
5. Using the database management tool, log in to the database, and delete the workspace data.
    - a) In the M\_CLIENTS table, find the workspace ID (for example, 200). The workspace name is in the NAME column.
    - b) In the M\_CLIENT\_MSISDNS table, find the ID where CLIENTS\_ID = 200. In this example, M\_CLIENT\_MSISDNS.ID = 222.
    - c) In the M\_MESSAGE\_LOG table, delete all rows where CLIENTS\_MSISDNS\_ID = 222.
    - d) In the M\_CLIENT\_MSISDNS table, delete the row where ID = 222.
    - e) In the M\_SESSIONS\_ACTIVE table, delete all rows where CLIENTS\_ID = 200.
    - f) In the M\_SESSIONS table, delete all rows where CLIENTS\_ID = 200.
    - g) In the M\_MESSAGE\_RECEIVERS table, delete all rows where CLIENTS\_ID = 200.
    - h) In the M\_MENU\_PAGES table, get the value of ID where CLIENTS\_ID = 200. In this example, M\_MENU\_PAGES.ID = 444.
    - i) In the M\_MENU\_PAGES\_LANGS table, delete all rows, where ID = 444.
    - j) In the M\_MENU\_PAGES table, delete all rows where CLIENTS\_ID = 200.

k) In the `M_CLIENTS` table, delete the row, where `ID = 200`.

## Users

---

The admin user is automatically created in the embedded Derby database when you install SMS Builder. The admin user is created in a production database when you run the database configuration scripts. The admin user has the `SUPER_ADMIN` role and unlimited access to the system.

The initial password for the admin user is `Brand!23`. In a production system, the admin user should change the password. The admin user is the platform administrator. Some tasks, such as creating workspaces, and restarting and configuring channels, can be performed only by the platform administrator. In practice, the platform administrator generally creates and configures workspaces, and grants the `ADMIN` role to users.

A user with the `ADMIN` role can administer workspaces, create users, and assign workspaces to users. A user with the `ADMIN` role who is assigned to more than one workspace is the administrator for all those workspaces.

You can deactivate users in the Manage User screen. When inactive users try to log in, they see `Invalid user ID or password`. Only a system administrator or a DBA can delete users from the database.

## Adding Users

On the Manage Users screen, administrators can add new users, and assign roles and workspaces to them.

1. On the Web UI navigation bar, select **Workspace Administration**.
2. Select **Manage Users**, then select **Add New User**.
3. Enter:

Property Name	Description
User name	Login name for the user.
Password	Password for the user.
Reenter Password	Reenter the password.
Roles	<ul style="list-style-type: none"> <li>• To assign a role to the user, select the role from the Available list, and click the right arrow.</li> <li>• To remove a role, select it in the Selected list, and click the left arrow.</li> </ul>

Property Name	Description
Workspaces	<ul style="list-style-type: none"> <li>To assign a workspace to the user, select the workspace in the Available list, and click the right arrow.</li> <li>To unassign a workspace from the user, select the workspace in the Selected list, and click the left arrow.</li> </ul>

- Click **Save**.

#### See also

- Editing User Properties* on page 361
- Deactivating Users* on page 362
- User Roles* on page 362

## Editing User Properties

You can edit properties for users in the current workspace.

Although an administrator can have access to multiple workspaces, only users in the workspace that he or she is currently logged in to appear.

- On the Web UI Navigation bar, select **Workspace Administration**.
- Select **Manage Users**.
- Select the user whose properties you want to edit.
- You can edit these properties:

Property Name	Description
User name	Login name for the user.
Password	Password for the user.
Reenter Password	Reenter the password.
Roles	<ul style="list-style-type: none"> <li>To assign a role to the user, select the role from the Available list, and click the right arrow.</li> <li>To remove a role, select it in the Selected list, and click the left arrow.</li> </ul>
Workspaces	<ul style="list-style-type: none"> <li>To assign a workspace to the user, select the workspace in the Available list, and click the right arrow.</li> <li>To unassign a workspace from the user, select the workspace in the Selected list, and click the left arrow.</li> </ul>

- Click **Save**.

#### See also

- Adding Users* on page 360

- *Deactivating Users* on page 362
- *User Roles* on page 362

### Deactivating Users

Deactivate users to prevent them from accessing applications.

1. On the Web UI Navigation bar, select **Workspace Administration**.
2. Select **Manage Users**.
3. Select the user you want to deactivate.
4. Select **Actions > Disable User**.

You can also deactivate users by assigning them to the default workspace, which removes their assigned roles. Users can still log in, but cannot access an active workspace.

#### **See also**

- *Adding Users* on page 360
- *Editing User Properties* on page 361
- *User Roles* on page 362

### User Roles

User roles restrict access to screens and controls.

#### *SUPER\_ADMIN*

The default user (admin) has the SUPER\_ADMIN role. Do not change this configuration, but do change the password. The default password is `Brand!23`. The admin user is created in the production database when you run the database configuration scripts.

A user with the SUPER\_ADMIN role is the platform administrator, has unlimited access to the system, and exclusive access to the Workspace Administration screen, which can be used to:

- Create and manage workspaces
- Create the QA workspace, and assign it to the workspace administrator
- Configure channels
- Start, stop, and restart the processing engine

A SUPER\_ADMIN user can also perform any function available to other roles.

#### *ADMIN*

A user with the ADMIN role is called the workspace administrator. Each workspace should have an assigned workspace administrators who can create users for the workspace, set up the default menu, and manage sessions.

In a development environment with no active channels, you can assign the ADMIN role to an application developer. A user with the ADMIN role can:

- Access all Workspace Administration screens, except Manage Workspace and Manage Channel Configuration.
- Create and manage workspace users in the Manage User screen.
- Manage active sessions that are created for interactive applications, using the Managing Sessions screen.
- Monitor the processing engine log using the Processing Engine Logs screen.
- Generate traffic reports for a workspace.

ADMIN users can also perform any function available to the APP\_ADMIN and APP\_OWNER roles.

### *APP\_ADMIN*

The APP\_ADMIN role is assigned to team members of the QA team who are working in a QA environment.

Users with this role work with applications, events, and subscribers. They have full access to development functions, including create, modify, approve, simulate, and delete. They also can access to the Manage Categories screen.

Assign the APP\_ADMIN role to testers, so they can import, activate, and test applications.

### *APP\_OWNER*

Typically, the APP\_OWNER role is assigned to QA team members and application developers in the production environment, when necessary.

Application owners can create applications, but they cannot activate them. Access is restricted to read-only functionalities.

To allow business analysts to view traffic reports from testing, and to troubleshoot reported bugs, assign the APP\_OWNER role to them.

### **See also**

- *Adding Users* on page 360
- *Editing User Properties* on page 361
- *Deactivating Users* on page 362

## **Deleting Users**

To delete a user, first unassign all roles and workspaces, except the default workspace, from the user.

### **Prerequisites**

Back up the database.

### **Task**

---

**Note:** This task can be performed only by a system administrator or a DBA.

---

1. In the Web UI, log in as a user who has the ADMIN role, and who is assigned to the workspace.
2. Navigate to the Manage User page, and unassign all roles and workspaces, except the default workspace, from the user.
3. Using the database management tool:
  - a) In the M\_USERS table, find the ID of the user you want to delete (for example, ID = 100).
  - b) In the M\_CLIENTS\_USERS table, delete the row where USERS\_ID = 100.
  - c) In the M\_USERS table, delete the row where ID = 100.

## Categories

---

You can assign categories to applications and events. Categories are primarily used to filter results in the Web UI.

You can create new categories, and edit and delete existing categories on the Manage Category screen.

In the Web UI navigation bar, select **Actions > Manage Categories**.

You cannot delete a category if it is assigned to an application or event.

## Subscribers

---

The data model for mobile-subscriber storage is called subscribers. A group of subscribers and their attributes is called a set. The Subscribers screen lists all subscriber sets in the workspace.

Subscribers storage is available to all SMS Builder applications. You can use subscribers storage as the system of record for small-scale implementations. Ideally, use subscribers storage as temporary storage, for staging, or as in-transit storage, pending batch transfer to a system of record. The database schema used by subscribers storage is not fully optimized for large-scale or more domain-specific purposes, such as for customer relationship management (CRM), or enterprise resource planning (ERP).

A subscribers set name is not required to be unique, but to avoid confusion, SAP recommends that it is. A set is made up of a list of rows with 21 fields. The **KEY** field is the unique key, and the remaining 20 fields are free form. The first free-form field, **ATTRIB1**, can store up to 1000 characters; the remaining free-form fields can store up to 320 characters each.

The set model is designed to store lists of subscribers. In a subscriber list, the **KEY** field stores a mobile subscriber's unique MSISDN. You can use the remaining free-form fields to store additional attributes related to the subscriber. You can use these attributes in different ways; for example, to dynamically "mail-merge" into the SMS message template, to send customized SMS messages based on attributes, or to filter the number of SMS messages sent,

based on specific attributes. Since SMS messages are limited to about 160 characters (varies based on the character-encoding type), SMS Builder automatically breaks long messages into smaller pieces, and sends each piece as a separate SMS message.

You can use the Web UI to generate a subscriber-set report, by exporting the set to a comma-separated value (CSV) file. You can also use a CSV file to transfer a set to a permanent system of record. CSV files can be read by spreadsheet programs, uploaded to databases, imported into reporting applications, and processed by custom-built applications.

If you need an automated process to transfer a set, you can develop an application using custom plug-in states, to send or upload the set in batch mode to the target system. To schedule a batch process, use the SMS Builder event system. You can populate a set by uploading a CSV file on the Upload Subscriber screen, or you can create an empty set on the New Subscriber Set screen, and populate it automatically using an application. For example, you can develop applications to pull subscribers from:

- A CRM system
- Twitter followers
- Facebook friends
- LinkedIn connections

You can also add mobile subscribers who respond to a short code to opt in. You can use subscriber lists to collect customer data for real-time analytics and reporting.

Create and populate subscriber sets by:

- Creating an empty set – then populate using either the Upload Merge or the Add Subscriber application state.
- Uploading a subscriber file – either a CSV file or a compressed CSV file.

### See also

- *Generating Subscriber Reports* on page 367

## **Creating Empty Subscriber Sets**

A subscriber set is a group of subscribers. The Subscribers screen lists all the subscriber sets in the workspace. A set has a unique ID and a name.

Although a set name is not required to be unique, SAP recommends that you use unique names to distinguish each set from others in the system.

1. In the Web UI Navigation bar, select **Subscribers**.
2. Select **Create Empty Set**.
3. On the New Subscriber Set window, select the **Set Details** tab, and enter:
  - Set Name – a unique name for the subscriber set.

- Attributes Metadata – (optional) a comma-separated list of attribute names to assign to the subscriber set free-form fields; maximum number of fields is 20.

4. Click **Save**.

### **Uploading Subscriber Sets**

You can upload subscriber sets from both comma-separated value (CSV) files and compressed CSV files. CSV files contain comma-separated attribute names on the first row, and comma-separated values in remaining rows. Each set has a unique ID and a name.

Uploading compressed files is the preferred method, especially for files containing a large number of subscribers. Compression can significantly reduce the upload streaming time. Compressed CSV files have a .zip extension.

All uploaded files are checked to verify that they conform to specific formats; otherwise, they are not processed; no virus-scanning capability exists.

---

**Tip:** Scan for viruses before uploading files.

---

1. In the Web UI Navigation bar, select **Subscribers**.
2. Select **Upload Subscriber**.
3. Enter the set name.
4. Click **Browse**, select the file, and click **Open**.
5. Click **Upload**.
6. Wait until streaming is complete and a success message appears beneath the Upload button, then click **Subscribers**.

Uploading a file is a two-step process: streaming the file to the server, then processing and uploading the file contents to the database. Depending on the file size and system utilizations, the latter may take several minutes. However, once the file is stored on the server, the subscriber set is created, enabling users to track the uploading process from the Subscribers screen. The process states change in this order: Not Started, Load In Progress, and Load Success. During the Load In Progress state, you can see the number of subscribers increase.

### **Reports**

---

You can access the Reports screen from the Web UI navigation bar.

In the Reports screen, you can export data to comma-separated value (CSV) files, which can be imported into more sophisticated reporting tools or applications. You can also import CSV files into permanent systems of record.



## Generating Traffic Reports

The Traffic Reports screen provides simple search and filter tools to extract reports of incoming and outgoing messages, as well as acknowledgements from SMS providers.

The Traffic Reports screen is a tool for filtering messages, and displays a preview of up to 2000 results. You can export traffic reports to comma-separated value (CSV) files, in which case all rows are exported. CSV files can be read or imported by analysis and reporting tools.

1. In the Web UI navigation bar, select **Reports**.
2. On the Standard Reports screen, select **Traffic Report**.
3. Select the sender in the list, or select **Advanced** to narrow results:
  - Start and End Dates – narrows results to the dates messages were sent.
  - Application Name – filters messages based on the interactive or event application name.
  - Message Direction – filters incoming and outgoing messages, based on their direction, in, out, or in-and-out.
  - Receiver – returns messages with a specific customer MSISDN. This is helpful for customers who use interactive applications.
4. Click **Search**.  
You see a maximum of 2000 rows.
5. Select **Export CSV**.  
This link appears only when there are results to show.
6. Select **Save File**, and click **OK**.

### **See also**

- *Monitoring Sessions* on page 374
- *Processing Engine Performance* on page 374

## Generating Subscriber Reports

You can export subscriber sets to comma-separated value (CSV) files. In the Subscribers screen, you can preview and export subscriber sets. You can export all rows in a set; the maximum number of rows you can preview is 2000.

Exporting lets you create subscriber sets by merging multiple files, and save the combined content. A subscriber set may have been created as an empty set, and later populated by applications, using options such as opt-in, subscribe, or log for reporting. Exporting lets you transfer data to the system of record, for reporting. You can also develop an automated system to perform the transfer, using applications and custom states.

1. In the Web UI navigation bar, select **Reports**.
2. On the Standard Reports screen, select **Subscriber Report**.

3. Click the set you want to export.  
You see attributes for each subscriber in the set.
4. Click **Export CSV**.
5. Select **Save File**, and click **OK**.

**See also**

- *Subscribers* on page 364

## Maintenance and Tuning

---

You can monitor, maintain, and tune SMS Builder components. The frequency of required maintenance depends on traffic volume (number of messages).

### Default Ports

The SMS Builder installation sets up default ports. Consult your IT system architect, and adjust values if conflicts occur, or disable if not needed.

Port Value	Port Key
5366	JMX remote connection port enables monitoring and management from a remote system.  In the <code>/bin</code> directory, set the <code>RMI_PORT</code> in the start-up script, <code>run.sh</code> or <code>run.bat</code> .  See Monitoring and Management Using JMX Technology at: <a href="http://docs.oracle.com/javase/6/docs/technotes/guides/management/agent.html">http://docs.oracle.com/javase/6/docs/technotes/guides/management/agent.html</a> .
5365	Telnet port from localhost enables connecting to OSGi shell console using Telnet.  In the <code>/bin</code> directory, edit the start-up script, <code>run.sh</code> or <code>run.bat</code> , and set the value of <code>TELNET_PORT</code> .
8080	Set the HTTP port for Web UI in the <code>conf/org.ops4j.pax.web.properties</code> file.

**See also**

- *Processing Engine Performance* on page 374
- *Editing Configuration Files* on page 370
- *Monitoring Sessions* on page 374

### **Configuring the HTTP Port**

The default HTTP port number is 8080, which is the same port that SAP Mobile Platform uses for its OData proxy. If SMS Builder is running on the same machine as SAP Mobile Platform, set the HTTP port number to a value that is not being used by SAP Mobile Platform.

1. Verify which port numbers SAP Mobile Platform is using, by checking the `SMP_HOME\Server\config_master\org.eclipse.gemini.web.tomcat\default-server.xml` file.
2. In the SMS Builder installation, edit the `SMSBUILDER_HOME\conf\org.ops4j.pax.web.properties` file, and set the HTTP port to a value that SAP Mobile Platform is not using.

## **System Configuration Files**

Configure system components in SMS Builder configuration files.

### *Server Configuration Files*

Server configuration files are located in the `SMSBUILDER_HOME/conf` directory. `config.properties` and `system.properties` are required to start the server:

- `config.properties` contains messaging server configurations, which are based on the Apache Felix implementation. You need not change most of these settings, as they pertain to the SMS Builder platform. If you apply a patch, follow the step-by-step instructions provided with the patch.
- `system.properties` contains subsystem configurations, such as logging, the Jetty HTTP server, and encryption settings. If you plan to encrypt configuration property values, enable encrypting in `system.properties`.

### *SMS Builder Web UI*

The user interface for SMS Builder is configured in the `service.brand_webapp.properties` file.

### *AIMS System Web Console*

During development, use the AIMS System Web console to inspect the OSGi container, deployed bundles, and register configurations. The Web console does not meet production standards, so to prevent unintended deployment to production, it is by default, disabled.

### *Jetty Servlet Container*

The Jetty servlet container hosts Web applications, and is configured in the `org.ops4j.pax.web.properties` file. As described on the Apache Jakarta Web site, AJP13 is a connector component that communicates with a Web connector via the AJP protocol. AJP13 may be useful if you want to invisibly integrate Tomcat 4 (or Jetty) into an Apache installation, and you want Apache to handle the static content contained in the Web application, or to use Apache SSL processing. In many application environments, using an

AJP13 component results in better overall performance than running your applications under Tomcat alone using the HTTP/1.1 connector. For details about the `jetty.xml` file, see <http://www.eclipse.org/jetty/documentation/current/>.

### Log4j

Logging is controlled by the `org.ops4j.pax.logging.properties` file; the properties are dynamic and use the log4j configuration format. See <http://logging.apache.org/log4j/1.2/manual.html>.

### Editing Configuration Files

You can copy configuration files to a location where the server reloads them automatically. You can also track configuration-file changes.

1. To make changes to a system configuration file, and have those changes take effect without restarting the server:

- a) Change the configuration file in the `conf/cfgbackup` folder.
- b) Copy the modified file to the `conf/cfgload` folder.

The server loads the file automatically, and moves it back to the `conf/cfgbackup` folder. By default, changes are not logged.

2. To track configuration-file changes:

- a) Edit the `conf/cfgbackup/org.ops4j.pax.logging.properties` file.
- b) Change the value of **log4j.logger.com.sybase365.arf.container.system** from WARN to INFO:

```
log4j.logger.com.sybase365.arf.container.system=INFO, main,
osgi:VmLogAppender
log4j.additivity.com.sybase365.arf.container.system=false
```

When the property file is reloaded, the log is written to `log/felix.log`. The following example displays log entries that resulted from reloading the `service.webui.security.properties` file.

```
16:15:53,507 | INFO | Thread-1 | cm-loader | ldap.port:389
16:15:53,513 | INFO | Thread-1 | cm-loader |
ldap.userpath:uid={uid},ou=people,o=sybase365
16:15:53,513 | INFO | Thread-1 | cm-loader |
ldap.security.authentication:SIMPLE
16:15:53,513 | INFO | Thread-1 | cm-loader | ldap.host:localhost
16:15:53,513 | INFO | Thread-1 | cm-loader |
service.pid:service.webui.security
16:15:53,513 | INFO | Thread-1 | cm-loader |
authentication.bean:AuthenticationManager
16:15:53,513 | INFO | Thread-1 | cm-loader |
arf.filename:service.webui.security.properties
```

### See also

- *Monitoring Sessions* on page 374
- *Default Ports* on page 368

- *Processing Engine Performance* on page 374

## Log Files

If you encounter issues running SMS Builder, check the log files in the `SMSBUILDER_HOME/logs` directory.

Log File	Description
<code>brand.log</code>	Primary log file for SMS Builder applications
<code>felix.log</code>	Server kernel and miscellaneous log for all nonapplication issues
<code>pax-web.log</code>	Pax-Web and Jetty servlet container log
<code>persist.log</code>	Persistence log
<code>spring.log</code>	Dependency injection (DI) application framework used by SMS Builder applications

## Database Table Maintenance

Database tables store logins, and persist configurations, transactions, and session data. Logging tables require regular archiving when traffic volume is high.

The information presented here is intended to help the DBA incorporate application tables into the overall database maintenance plan. As always, regular backups are essential for recovery in the event of unexpected failure.

---

**Note:** The DBA should review these guidelines, and define archiving and purging schedules based on company policy.

---

**Table 30. Logging Tables**

Table Name	Description	Archive and Purge
<code>M_CORE_STATUS_LOG</code>	Engine logs	Purge based on the <code>TIMESTAMP2</code> column.
<code>M_LOGIN_HISTORY</code>	User logins	Purge based on the <code>TIMESTAMP2</code> column.
<code>M_MESSAGE_LOGS</code>	Size increases with number of subscribers and traffic	Purge based on the <code>TIMESTAMP2</code> column.
<code>M_MESSAGE_RECEIVERS</code>	Size increases with number of subscribers and traffic	Purge based on <code>MARKED_DELETED = 1 AND</code> if there are no references in the <code>M_MESSAGE_LOG</code> table.

Table Name	Description	Archive and Purge
M_MESSAGE_ATTRIBUTES	Number of rows increase with traffic and session data	Purge before purging the M_SESSIONS table, and based on the SESSIONS_ID column.
M_SESSIONS	Number of rows increase with traffic and session data	Purge based on the CLOSED_DATE column.
M_SMAPP_TRANSACTION_LOG	Number of rows increase with traffic and session data	Purge based on the TIMESTAMP2 column.

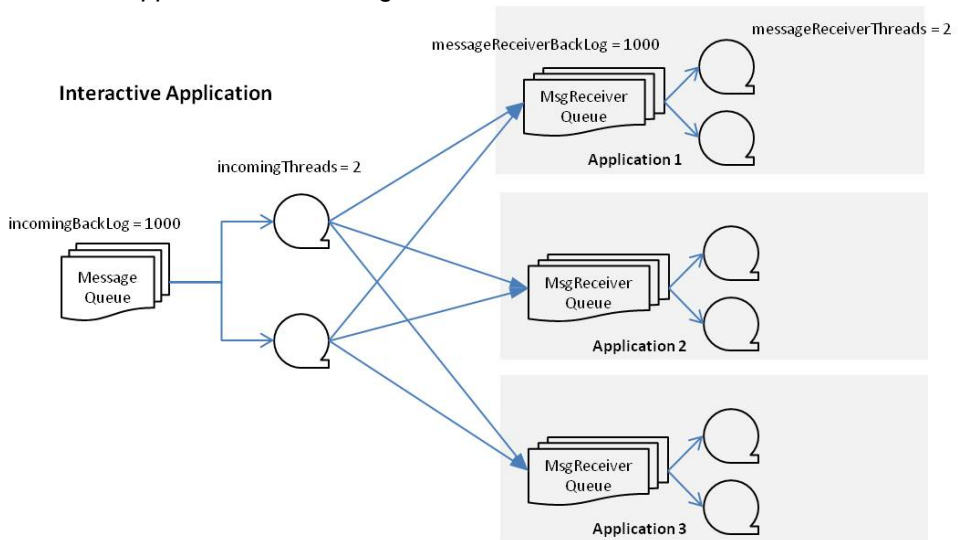
## Processing Engine

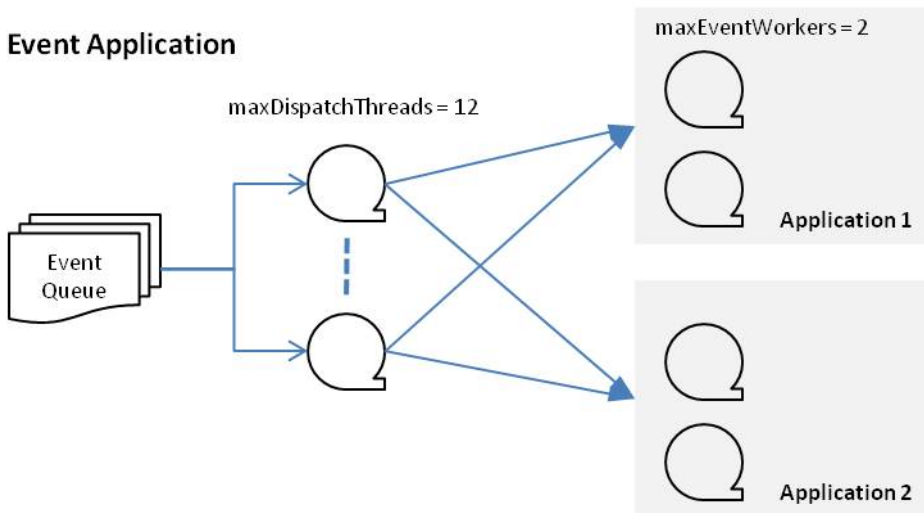
Understanding the internal mechanism and configuration parameters of SMS Builder may help you tune the processing engine for optimal performance.

The processing engine powers mobile-commerce solutions, and lets you scale your services. The processing engine is designed to serve the continued growth of mobile traffic, which demands instant interactions with mobile applications.

The processing engine processes interactive applications somewhat differently than it processes event applications, as shown below.

### Interactive Application Processing



*Event Application Processing***Event Application***Processing Engine Configuration File*

In the `/conf/cfgbackup/service.coreprocessing.properties` configuration file, you can tune the processing engine and queues for increased throughput, based on specific hardware configurations and channel throughput.

```
# Configuration for Incoming Queue - Interactive Application
# Number of threads to handle the dispatch of the incoming queue.
# Note that this should not exceed the messageReceiverThreads
incomingThreads=2

# Configuration for Incoming Queue - Interactive Application
# Maximum number of messages allowed in the incoming queue. This is a
global queue
# that handles all incoming messages before dispatching to a specific
message
# receiver.
incomingBackLog=1000

# Configuration for Message Receiver - Interactive Application
# A message receiver is created for each active Brand Mobiliser
application
# Configure the number of threads to handle the message queue for
each message
# receiver.
messageReceiverThreads=2

# Configuration for Message Receiver - Interactive Application
# Maximum number of messages allowed in each message receiver
queue
# There is one queue for each message receiver.
messageReceiverBackLog=500
```

```
# Configuration for Event Application, used in Campaign
# Maximum number of dispatching thread to process the event queue
maxDispatchThreads=12

# Configuration for Event Application, used in Campaign
# Number of Processor per event application
maxEventWorkers=1

# Batch size in processing the subscriber file
batchLoadSize=100
```

### **Monitoring Sessions**

The SMS Builder Processing Engine creates a session for each incoming message. On the Manage Sessions screen, the workspace administrator can monitor active sessions and terminate them if necessary.

1. To display the currently active sessions, click **Refresh**.
2. To end an active session, select the session, and click **Close Session**. Active sessions are automatically terminated when the application ends, or if the session times out.

---

**Warning!** The list of active sessions includes all sessions, including those for other workspaces. Close sessions carefully.

---

### **See also**

- *Generating Traffic Reports* on page 367
- *Processing Engine Performance* on page 374
- *Editing Configuration Files* on page 370
- *Default Ports* on page 368

### **Viewing Processing Engine Logs**

The server logs both start-up and shutdown messages.

1. On the Web UI navigation bar, select **Workspace Administration**.
2. Select **Processing Engine Logs**.

### **Processing Engine Performance**

Depending on the hardware configuration and the host server, you may be able to tune the processing engine to increase throughput if necessary.

The default performance configuration, which is defined in the `service.coreprocessing.properties` file, works for typical scenarios.

---

**Note:** Always run performance tests on a new configuration before deploying it to the production system. Changes to the `service.coreprocessing.properties` file do not take effect until you restart the server.

---

Performance testing has been conducted using this system configuration:



- Hardware – HP DL 360 G6; 1U standalone; 2x Intel X5450; 8x 2GB (16GB).
- Server configurations – one dedicated server for the database and another for SMS Builder.

#### *Software Configuration for Interactive Application Performance Test*

Processing Engine	Datasource
messageReceiverThreads=10	maxIdle=50 (50 maximum performance)
messageReceiverBackLog=50000	maxActive=25 (50 maximum performance)
incomingThreads=10	
incomingBackLog=50000	

#### *Results for Interactive Application Performance Test*

Load	Result
10,000 messages	110 messages/second

#### *Software Configuration for Event Application Performance Test*

Processing Engine
maxDispatchThreads=12
maxEventWorkers=4

#### *Results for Event Application Performance Test*

Messages	Rate
Up to 88	88 messages/second
Up to 3,520	93 messages/second
Up to 52,695	92 messages/second
Up to 174,668	94 messages/second
Up to 3,576,594	82 messages/second
Up to 4,999,899	97 messages/second

#### **See also**

- *Generating Traffic Reports* on page 367
- *Monitoring Sessions* on page 374
- *Default Ports* on page 368
- *Editing Configuration Files* on page 370



# Troubleshooting

Review information about common problems that arise when working with SAP Mobile Platform Server.

## Quick Fixes to Simple Server Problems

Quick fixes are usually common, single-cause problems that you can solve with minimal overhead or additional support.

### Management Cockpit Stops Working After Enabling Mobiliser Services Feature

#### *Problem*

After enabling the Mobiliser Services feature (com.sap.mobile.platform.server.build.feature.mobiliser.feature.group), the Management Cockpit displays blank screens and `Session Timeout` messages.

#### *Workaround*

1. After waiting a few minutes, check the `SMP_HOME\Server\log\hostname-smp-server.log` for this message: The SMP server has initialized and is ready.
2. Close the browser window and restart the Management Cockpit.

### Mobiliser Features Not Included in Enabled Features List

#### *Problem*

After enabling a Mobiliser feature in Management Cockpit, the feature does not appear in the Enabled Features list.

#### *Workaround*

1. Make sure that you have waited long enough for Mobiliser features to be enabled. This can take anywhere between a few seconds and a few minutes. Click **Refresh** to display the Enabled Features list.
2. Check the `SMP_HOME\Server\log\hostname-smp-server.log` for this message: The SMP server has initialized and is ready.

3. If you are enabling the Mobiliser Services feature (com.sap.mobile.platform.server.build.feature.mobiliser.feature.group), close the browser window and restart Management Cockpit.

### **Connection Failure Outside Corporate Network**

#### *Problem 1*

Receive an error 404 (Not found) while accessing an URL (for which 'Use system proxy' option is enabled) which is outside of the corporate network.

#### *Problem 2*

Receive java.io.exception in the logs while interacting with the push notification provider server such as APNS, GCM, or BIS exists in the network, although it shows 200 (OK) during push operation.

#### *Workaround*

Above mentioned problems occur when the proxy settings are not configured in the server. Due to which, from corporate network, server fails to communicate with the outside network. You need to enter the proxy settings manually in `props.ini` file.

### **Unknown Host Exception While Accessing BIS**

#### *Problem*

While accessing BlackBerry Internet Service (BIS) for push notification, you receive an unknown host exception.

#### *Workaround*

It happens when the SAP Mobile Platform Server is unable to validate the BIS server. Ensure that the BIS certificate is placed in the server.

1. Enter the BIS URL in the browser and fetch the BIS certificate.
2. Open the keystore of the SAP Mobile Platform Server.
3. Place the BIS certificate in the keystore.
4. Restart the server.

### **Browser Caching Issue**

#### *Problem*

Session timeout happens when you try to access Management Cockpit and SAP Mobile Platform Server 3.0 web applications on the same browser instance.

#### *Workaround*

Use different browser instances.

## Impersonator Role Missing Error

### *Problem*

This error can be caused due to any of the following problems:

- When SAP Mobile Platform Server fails to trust the certificate it receives from the reverse proxy. Reverse proxy redirects the client certificate which is received from SSL\_Client header. In this case, reverse proxy does not hold the private key of the client certificate. So, for security reasons you must mention the certificate subject under the impersonator role in the role mapping XML file of the security configuration.
- When SAP Mobile Platform Server is unable to trust the push generator.

### *Workaround*

During certificate onboarding in HTTPS channel via reverse proxy, you need to:

1. Create a security configuration (SC) with the certificate authentication login module. It creates a role-mapping XML file for that SC under CSI folder.
2. Map the reverse proxy certificate's subject under the impersonator role in the XML file.

During certificate push in HTTPS channel, you need to:

1. Map the push generator's subject in the Notification role-mapping.xml.
2. Add push generator's certificate subject to the Notification role-mapping.xml under the CSI folder.

## Unauthorized Error for First Mutual Authentication Request

### *Problem*

When using mutual authentication with SAP Netweaver back end systems, the first request might fail with 401 error (unauthorized).

### *Workaround*

Subsequent requests do work successfully.

## HTML Code Appears in Server Log

### *Problem*

While reviewing the server logs, you see HTML code along with the standard log file format, for example:

```
YYYY MM DD HH:MM:SS03#0-800
#DEBUG#com.sybase.security.http.HttpAuthenticationLoginModule##anonymous#http-bio-8080-exec-10###
<html>
  <head>
    <title>
```

```
Test Config
</title>
</head>
<body>
  <h1>
    <font
      color="#000080">
      SiteMinder WebAS Agent Configuration Test
    </font>
  </h1>
  ...
</body>
</html>
```

### *Fix*

This typically indicates there is a problem with the back-end connection or system. The message displayed comes directly from the back-end system and is not translated. Investigate the back-end connection, service, datasource, or system for additional information.

## Issues Requiring Product Support

---

Your SAP support ID gives you access to enterprise-level incident support as part of your support plan on SAP Service Marketplace.

Product Support can help you resolve new undocumented incidents with software installation, start-up, and overall use, as well as providing diagnostic and troubleshooting assistance for known problems with a new or undocumented cause.

## Product Support Engagement Requirements

If you use SAP Service Marketplace to engage with Product Support, you must meet certain requirements.

### *Service Marketplace Case Creation Requirements*

Be prepared to provide:

- A valid installation number for SAP Mobile Platform
- A valid service contract with SAP
- A valid system ID (S-User ID)
- An enabled NetViewer connection.

### *SAP Mobile Platform Incident Requirements*

- Configure your logs to an appropriate level for your issue. Product Support requires details from one or more of the system logs.
- Capture these basic incident details to help Product Support analyze the problem, and determine any next steps:

- Environment summary: product version, back end, client type (device and OS), proxy connections. These details help isolate component that is causing the failure. If you have an architecture diagram, share it with SAP.
- Problem description: what were the actions preceded the incident. Capture all details that allow Product Support to reproduce the issue.
- Locate the server version in the `SMP_HOME\Server\version.properties` file.

## Creating an Incident on SAP Service Marketplace

If you cannot resolve problems with the troubleshooting documentation for SAP Mobile Platform, go to SAP Service Marketplace for additional help.

Use SAP Service Marketplace to create an incident message for Product Support. Keywords from this message return related articles from the Knowledge Base. Before you submit a message, review these articles to see if they resolve your problem.

1. Go to <http://service.sap.com/message>.
2. Create a message using the wizard.

---

**Note:** You must know the component ID for SAP Mobile Platform to return the correct scope of Knowledge Base Articles and to correctly route the message to Product Support. On-premise installations of SAP Mobile Platform use a different ID than cloud instances. See Knowledge Base Article *1915061- How to Choose a Component for SAP Mobile Platform 3.x in Service Marketplace*.

---

3. Once the message is processed, you receive an e-mail notification of the solution.

## Increasing Server Status Logging Levels

By default, minimal output is recorded in the `startup.log` for the SAP Mobile Platform Server. You may need to change the logging to verbose when working with Product Support.

---

**Note:** Logging startup messages also appear in the Windows console, unless the server is running as a Windows service.

---

1. Create a backup copy of `SMP_HOME\Server\props.ini`.
2. Using a text editor, open the original `SMP_HOME\Server\props.ini`.
3. Change  
`com.sap.mobile.platform.server.general.status.SMPServerStatusLogging` to `true` to start verbose logging.
4. Save the change.

### **Next**

Reconfigure the SAP Mobile Platform Server Windows service for the changes to take effect.

### **See also**

- *Reconfiguring the Windows Service After Server Configuration Changes* on page 123

## **Server Fails to Initialize**

### *Problem*

A serious issue causes one of the server components to fail to initialize. The server is not in a working state.

### *Workaround*

Work with Product Support to determine the cause of the initialization failure.

Product Support may ask you to help analyze and resolve the error:

- Review errors in the following logs, located in `SMP_HOME\Server\log`:
  - `hostname-smp-server.log`
  - `osgi.log` (if it exists)

For Linux systems, you may also be asked to review errors in `SMP_HOME/Server/log/server_daemon.log`.

- Make a backup copy of `SMP_HOME\Server\props.ini`, and then add the following line in the `jvm` section of the original file:

```
-Dcom.sap.mobile.platform.disableServerHealthValidation=true
```

---

**Warning!** Add this line only when directed to do so by Product Support. Setting this property may cause the server to operate incorrectly.

---

## **Statistics Data Loss During Server Crash or Shut Down**

### *Problem*

Reporting tab does not show number of connections per request that is made at the time server crashes.

### *Workaround*

None.

Statistics calculation is done in memory. Because the server has crashed or shut down unexpectedly, the data is lost and cannot be recovered for that particular time.



# Glossary: SAP Mobile Platform

Defines terms for all SAP® Mobile Platform components.

**Application Configuration Profile (ACP)** – the package that holds extensibility-related and other metadata, and additional resources.

**admin user** – a user with a defined admin security profile.

**administrators** – SAP Mobile Platform users to which an administration role has been assigned.

**Afaria®** – an enterprise-grade, highly scalable device management solution with advanced capabilities to ensure that mobile data and devices are up-to-date, reliable, and secure. Afaria is a separately licensed product that can extend the SAP Mobile Platform in a mobile enterprise. Afaria includes a server (Afaria Server), a database (Afaria Database), an administration tool (Afaria Administrator), and other runtime components, depending on the license you purchase.

**Agency application hierarchy** – The order and structure of, and the relationship between, the various definition types that comprise any Agency application project.

**Agency application project** – The definitions and references to synchronization logic that comprise the encapsulated business logic of an Agency mobile application.

**Agency Client** – The executable client software within the Agency paradigm that consumes and processes the business logic defined in the Agency application project, and presents that business logic and behavior to the mobile user.

**Agency Client Branding SDK** – A toolset provided to allow partners and customers developing Agency applications to brand those applications for packaging and delivery.

**Agency Editor** – The primary development tool for an Agency application project, built as a plug-in to the Eclipse IDE and presenting a 4GL development interface to define and modify business logic and rules within the application project.

**Agency Server** – The component of the Agency paradigm that interfaces between the Agency Clients and the back end system or systems with which the mobile application synchronizes data. The Agency Server is also responsible for serving up the business logic of the mobile application to the Agency Clients during the initial implementation and in subsequent updates to the application logic.

**Agency Test Environment (ATE)** – An Agency Client wrapped in numerous testing, debugging and monitoring tools, used primarily by developers to mimic various client device platforms during the development life cycle of a mobile application.

**anonymous user** – a user type who can access the system without identification.

**Apple Push Notification Service (APNS)** – service provided by Apple for devices running the iOS operating system. The APNS acts as a intermediary to push notifications from the provider to the device rather than have the application operate as an active listener for those notifications.

**application** – in SAP Mobile Platform Server, the runtime entity that can be directly correlated to a native or hybrid application. The application definition on the server establishes the relationship among packages used in the application, the user activation method for the application, and other application-specific settings.

**application activation** – from the SAP Mobile Platform standpoint, all activities that allow an identified or anonymous user to be paired with an application and its connections and customizations. Or, all activities that allow an application to be activated. An activated application creates an instance that is known by SAP Mobile Platform.

From the application user standpoint, application activation is the automated series of events by which a user, without administrative intervention, can start consuming services.

**application connection** – a unique connection to the application on a device.

**Application Data** – The data that is the business logic served up by the Agentry Server to the Agentry Client during a transmit; term used to distinguish between this data and Production Data.

**application ID** – the unique ID that identifies an application (automatic or manual).

**application instance** – represents a client application on a single device.

**application node** – in SAP Mobile Platform, a registered application with a unique ID. The main entity that defines the behavior of device and back end interactions.

**application provisioning** – placing a client application on a device which includes:

1. Copying the application to the device.
2. Installing the application on the device.
3. Configuring the application on the device.
4. Securing the device.

**application registration** – configuring an application to work with SAP Mobile Platform. Registration requires a unique identity that defines the properties for the device and back-end interaction with SAP Mobile Platform Server.

**application user** – the distinct set of identities (identified or anonymous) that have ever been in contact with the system by utilizing the application. In Management Cockpit, an application user is the distinct list of names under which a user has been identified to the system. An application user may also be a user (identified or anonymous) that has been associated with an application ID.

**artifacts** – client-side or automatically generated files; for example: .xml, .cs, .java, .cab files.

**availability** – indicates that a resource is accessible and responsive.

**back end** – a system that provides a datasource, such as a database or Web service.

**binary large object (BLOB)** – a collection of binary data stored as a single entity in a database management system. A BLOB may be text, images, audio, or video.

**binding** – represents the datasource associated with the given tile.

**certificate** – a digital security mechanism attached to an electronic message and used to verify the identity of a specific user.

**certificate provisioning** – placing digital certificates on a device for user authentication.

**client application** – in SAP Mobile Platform, the software that runs on a smart phone, tablet computer, or other mobile device. See *mobile application*.

**client device** – general term used to describe the device upon which the Agency Client is installed and running; can apply to any of the devices to which the Agency Client can be installed, including PCs, laptops, tablets, and smart phones.

**client resources** – also known as resource bundles. Containers used by applications to download dynamic configurations, styles, or content from SAP Mobile Platform Server.

**command line interface (CLI)** – the standard term for a command line tool or utility.

**connection** – configuration details and credentials required to connect to a database, Web service, or other back end.

**connection pool** – a cache of back-end system connections maintained by SAP Mobile Platform Server, so that the connections can be reused when SAP Mobile Platform Server receives future requests for data. Or a collection of proxy connections pooled for their respective back ends, such as SAP Gateway

**Consumer Portal** – a reference Web application. The Consumer Portal allows consumers to manage their mobile-banking accounts, with multiple payment instruments, such as bank accounts, credit cards, stored value accounts (SVA) and Offline SVAs. They can pay bills, send money to family or friends, and add to airtime.

**custom control** – a special type of UIControl subclass defined in the extensibility metadata. Manage custom controls within the client code.

**data manipulation language (DML)** – a group of computer languages used to retrieve, insert, delete, and update data in a database.

**data points** – using performance harnesses and measuring KPIs, specific points defined to take and report measurements. Data points are required to implement end-to-end tracing features.

**data synchronization** – the process of establishing consistency among data from a source to a target data storage and vice versa, and the continuous harmonization of the data over time.

**data vault** – a secure store across the platform that is provided by an SAP Mobile Platform client.

**Definition** – A finite component of an Agency application project, encapsulating a general functional type within an application, and that exists within the Agency application hierarchy and may have both a parent definition and one or more child definition types, as well as attributes that comprise its makeup.

**demilitarized zone (DMZ)** – also known as a perimeter network. The DMZ adds a layer of security to the local area network (LAN), where computers run behind a firewall. Hosts running in the DMZ cannot send requests directly to hosts running in the LAN.

**deploy** – uploading a deployment archive or deployment unit to an SAP Mobile Platform Server instance. SAP Mobile Platform Server can then make these units accessible to users via a client application that is installed on a mobile device.

**device application** – a software application that runs on a mobile device. See *mobile application*.

**device provisioning** – making an out-of-the box corporate device or bring your own device (BYOD) secure and ready for synchronization.

**device user** – the user identity tied to a device.

**end-to-end tracing (E2E tracing)** – supportability feature that allows developers to add specified libraries and code to applications to enable tracing during runtime.

**export** – the SAP Mobile Platform administrator can export mobile objects, then import them to another server on the network.

**features** – groups of individual bundles (also known as Eclipse plugins) and static resources that together form an installable function or set of functions.

**generic business object** – a data entity that is independent of the underlying model.

**Google Cloud Messaging (GCM)** – a free service for sending messages to Android devices. GCM requires an API Key to allow SAP Mobile Platform Server to send push notifications over GCM.

**Initial Transmit** – The term applied to the first transmit performed by an Agency Client which, at the onset of the transmit, contains no business logic.

**Introscope** – a third-party tool that can be integrated into a system landscape to quickly isolate and resolve performance issues wherever they arise in each stage of the application life cycle.

**JCA** – J2EE Connector Architecture. A Java-based solution for connecting an application server with a back end.

**Java development environment (JDE)** – IDE specific to the Java programming language that is used in SAP Mobile Platform to create and test BlackBerry Java applications.

**key performance indicator (KPI)** – used by SAP Mobile Platform monitoring. KPIs are monitoring metrics that are made up for an object, using counters, activities, and time which jointly for the parameters that show the health of the system. KPIs can use current data or historical data.

**keystore** – the location in which encryption keys, digital certificates, and other credentials in either encrypted or unencrypted keystore file types are stored for SAP Mobile Platform Server runtime components. See *truststore*.

**layout configuration file** – an XML file that holds the extensibility descriptors (or metadata).

**Lightweight Directory Access Protocol (LDAP)** – an application protocol for accessing, querying, and modifying data in distributed directory services.

**mobile application** – (mobile app) is a software application designed to run on smart phones, tablet computers and other mobile devices. For Agentry, a general term used to refer to the mobile application built in Agentry and generally used to make the fine distinction between the Agentry Client executable itself, and the business logic which it is processing and presenting to the user; in essence this business logic is the mobile application.

**mobile data model** – shows the relationship between back-end enterprise data and the data on a mobile device.

**monitoring** – an SAP Mobile Platform feature that allows administrators to identify areas of weakness or periods of high activity in a particular area, as well as overall system health. It can be used for system diagnostics or for troubleshooting.

**OData for SAP** – provides SAP Extensions to the OData protocol that enables users to build user interfaces for accessing the data published via OData. The interfaces require human-readable, language-dependent labels for all properties and free-text search within collections of similar entities and across (OpenSearch).

**OData metadata document** – describe the entity data model (EDM) for a given service, which is the underlying abstract data model used by OData services to formalize the description of the resources it exposes.

**OData Schema** – defines the structure of the XML files in the OData service.

**OData Service Document** – a document that describes the location and capabilities of one or more collections.

**offline demo data** – collection of XML files used for demonstration purposes, without online connection. The files are created based on real data feeds and must adhere to naming rules.

**onboarding** – the enterprise-level activation of an authentic device, a user, and an application entity as a combination, in SAP Mobile Platform Server.

**Open Data (OData) Protocol** – Web protocol for querying and updating data. It applies and builds upon Web technologies such as HTTP, Atom Publishing Protocol (AtomPub) and JSON to provide access to information from a variety of applications.

**Partner Portal** – a reference Web application. Agents who sell to consumers on behalf of a system provider can use the Partner Portal to manage existing consumers, add to airtime, validate pending consumer registrations, settle commissions, and run reports.

**personalization key** – allows a mobile device user to specify attribute values that are used as parameters for selecting data from a data source and to provide operation parameter values. There are three type of personalization keys: transient, client, and server.

Personalization keys are most useful when they are used in multiple places within a mobile application, or in multiple mobile applications on the same server. Personalization keys may include attributes such as name, address, zip code, currency, location, customer list, and so forth.

**persistent identifier (PID)** – a unique identity for a dictionary that contains configuration properties for a managed service in the Open Services Gateway initiative (OSGi) modular architecture.

**perspective** – an Eclipse term applied as a named tab that groups commonly used resources (such as servers) and UI views associated with those resources. In SAP Mobile Platform the Mobile Development perspective facilitates mobile application development.

**physical role** – a security provider group or role that is controls access to SAP Mobile Platform Server resources.

**Problems view** – displays errors and warnings for the Mobile Application Project in Eclipse. This is a valuable source for collecting troubleshooting information and resolving issues during the development phase, and avoiding device application problems later, for example, device application synchronization or data refresh errors.

**Production Data** – The data stored either in the back end system or on the client device that is synchronized between those two components during a transmit; term used to distinguish between this data and the Application Data.

**provisioning** – See *application provisioning* and *device provisioning*.

**Publish** – The term that describes the deployment and transformation of the business logic built and/or modified within the Agentry Editor to the Agentry Server with the intent of serving that business logic to Agentry Clients when they next synchronize.

**push synchronization** – the server-initiated process of downloading data from SAP Mobile Platform Server to a remote client, at defined intervals, or based upon the occurrence of an event.

**queue** – a list of pending activities, made up of in-flight messages for a messaging application. The server sends messages to specific destinations based on message order in the queue. The depth of the queue indicates how many messages are waiting to be delivered.

**recovery** – performing the activities required to bring a system to a usable/functional state after a failure (populating CDB, initializing client, and so on).

**remote function call (RFC)** – used to write applications that communicate with SAP R/3 applications and databases. An RFC is a standalone function. Developers use SAP tools to write the Advanced Business Application Programming (ABAP) code that implements the logic of a function, and then mark it as "remotely callable," which turns an ABAP function into an RFC.

**Representational State Transfer (REST) Web services** – a style of software architecture for distributed hypermedia systems, such as the World Wide Web.

**resource** – a unique SAP product component (such as a server) or a subcomponent.

**restoration** – returning a system to its prefailure state using one or more methods of recovery. Restoration does not guarantee a return to a usable state.

**role** – controls access to SAP Mobile Platform resources.

**SAP** – one of the back-end types that SAP Mobile Platform supports. SAP Business Suite applications (such as ERP, CRM, SRM, SCM, Industry Solutions and so on) consist of many technologies and components. Unless stated otherwise, the term "SAP" means a backend business application that is based on the SAP NetWeaver ABAP application server, for example ECC 6.0.

**Management Cockpit** – in SAP Mobile Platform, a Web-based interface that allows you to administer your installed SAP products.

**SAP Messaging Service** – the synchronization service that facilitates communication with device client applications.

**SAP Mobile Platform Server** – the application server included with the SAP Mobile Platform product that manages mobile applications, back-end synchronization, communication, security, transactions, and scheduling.

**SAP NetWeaver Gateway** – enables people-centric applications to consume SAP Business Suite data through popular devices and platforms in an easy and standards-based fashion.

**SAP Passport** – medium for transporting technical data in a request from the client to the server. Used for collecting trace and reporting information for chains of requests (RFC, HTTP) across system borders.

**schedule** – the definition of a task (such as the collection of a set of statistics) and the time interval at which the task must execute in SAP Mobile Platform.

**security profile** – part of the application user and administration user security. A security profile determines the scope of user identity, authentication and authorization checks, and can be assigned by the platform administrator in SAP Mobile Platform. A security profile contains a set of configured security providers (for example, LDAP) to which authentication, authorization, and attribution are delegated. It also can include encryption metadata to capture certificate alias and the type of authentication used by server components. By using a security profile, the administrator creates a secured port for component communication

**security provider** – a security provider and its repository holds information about the users, security roles, security policies, and credentials used to provide security services to SAP Mobile Platform. A security provider is part of a security configuration.

**server connection** – the connection between SAP Mobile Platform SDK and a back-end EIS.

**Simple Object Access Protocol (SOAP)** – an XML-based protocol that enables applications to exchange information over HTTP. SOAP is used when SAP Mobile Platform Server communicates with a Web service.

**single sign-on (SSO)** – a credential-based authentication mechanism.

**solution** – in Visual Studio, the high-level local workspace that contains the projects users create.

**Solution Explorer** – in Visual Studio, the pane that shows the active projects in a tree view.

**Start Page** – in Visual Studio, the first page that appears when you launch the application.

**statistics** – in SAP Mobile Platform, the information collected by the monitoring database to determine if your system is running as efficiently as possible. Statistics can be current or historical. Use current or historical data to determine system availability or performance. Performance statistics are known as key performance indicators (KPIs).

**structured data** – data in a table with columns and labels.

**subscription** – defines how data is transferred between a user's mobile device and SAP Mobile Platform Server. Subscriptions notify a device user of data changes, then these updates are pushed to the user's mobile device.

**synchronization** – synchronous data delivery using an upload/download pattern. For push-enabled clients, synchronization uses a "poke-pull" model, where a notification is pushed to the device (poke), and the device fetches the content (pull), and is assumed that the device is not always connected to the network and can operate in a disconnected mode and still be productive. For clients that are not push-enabled, the default synchronization model is pull.

**synchronize** – the process by which data consistency and population is achieved between remote disconnected clients and SAP Mobile Platform Server.

**tile** – UI elements or screens, which can be primitive or can embed further tiles. In theory, you can nest tiles to an unlimited level.

**tile container** – the container view controller in iOS terms. Container view controllers group tiles that work together.

**Transmit** – The term that describes the initiation of the synchronization process between the Agency Client and Agency Server.

**truststore** – the location in which certificate authority (CA) signing certificates are stored. See *keystore*.



**user** – SAP Mobile Platform displays the mobile-device users who are registered with the server.

**Web Service Definition Language (WSDL) file** – describes the Web service interface that allows clients to communicate with the Web service. When you create a Web service connection for a mobile business object, you enter the location of a WSDL file in the URL.

**workspace** – in Eclipse, the directory on your local machine where Eclipse stores the projects that you create.



# Index

3-DES encryption 289

401 unauthorized error 379

404 error 378

## A

access

    web portal 312

Accessing LDAP roles 196

Active Directory nested groups 196

adding

    preference nodes 320

ADMIN role 362

admin user 360

    password hashing 342

administrator 172

administrators

    administrator role 172

    developer role 172

    help desk role 172

    notification user role 173

advanced encryption standard (AES) 230

AES (advanced encryption standard) 230

AES encryption 289

Agency applications

    configuring additional settings 72, 107, 131

AIMS System Web console

    configuring 369

Apache Reverse Proxy

    configuring with httpd.conf 210

    decrypting certificates for HTTPS connections  
    212

    preparing for HTTP clients 210

APP\_ADMIN role 362

APP\_OWNER role 362

application

    pinging back-end connections 98

application access 252

application layer 261

application logs

    managing 84

application security 231

Application settings logs 103

applications 66, 72, 107, 131

    configuring applications 36

    creating an application definition 36

    defining back-end connections 99

    deleting an application 86

    deleting back-end connections 101

    editing an existing application 85

    editing back-end connections 100

    managing and monitoring tasks 84

    managing applications 84

    pinging authorization URL status 84

    pinging back-end connections 84, 86, 101

    preferences 318

    provisioning 108

    viewing usage statistics 102

assigning

    roles to users 360

    workspaces to users 360

audit 300

    database audit manager 301

    dispatcher 301

    javascript object notification (JSON) audit  
    manager 302

    JSON audit manager 302

audit dispatcher 301

audit manager

    database 301

    javascript object notification (JSON) 302

    JSON (javascript object notification) 302

authentication

    application, defining 58, 95, 165

    configuring 349

    configuring for SAP Mobile Platform Server  
    219

    how it works 177

authentication providers 177

AuthenticationScope 193

## B

back-end connections

    defining connections 99

    deleting connections 101

    editing connections 100

    managing multiple connections 86, 98, 101

    pinging authorization URL status 86, 101

    pinging back-end connections 86, 98, 101

## Index

- testing authorization URL status (Ping) 86, 101
- testing back-end connections (Ping) 86, 98, 101
- back-end security 163
- bar charts 332
- bonecp 287
- brand.log file 371
- business logic configuration 278
  - audit 300
  - event handler 303
  - framework 278
  - jobs 305
  - messaging 287
  - miscellaneous 308
  - tasks 304
- C**
- card production life cycle (CPLC) 274
- categories 364
- certificate security provider 184
- changing password
  - Mobiliser web portal 317
- channels 325
  - communication 351
  - EmailChannel 293
  - GCMChannel 300
  - Google Cloud Messaging (GCM) 300
  - HtmlChannel 294
  - HttpChannelEnd 295
  - JabberChannel 295
  - JMS, configuring 354
  - messaging 293
  - short message peer to peer channel 296
  - SMPP inbound, configuring 352
  - SMPP outbound, configuring 353
  - SmppChannel 296
- charts 332
- ClamAV
  - configure 337
- client logs
  - examples 90
  - viewing 90
- ClientHttpValuesAsNamePrincipals 244
- ClientHttpValuesAsRolePrincipals 244
- Cockpit
  - starting and stopping Management Cockpit on Linux 27
  - starting and stopping Management Cockpit on Windows 26
- communication channels 351
- communication ports 145
  - HTTP and HTTPS 145
- config.properties file 369
- configuration
  - keystore 269
- configuration files 369
  - editing 370
  - tracking changes in 370
- configure application
  - overview 38
- configuring
  - Agentry applications 72, 107, 131
  - AIMS System Web console 369
  - authentication 349
  - ClamAV 337
  - DB2 database drivers 345
  - default menus 358
  - event schedulers 350
  - hibernate session factory 286
  - HTTP port 369
  - Jetty servlet container 369
  - JMS channels 354
  - log4j 369
  - ports 368
  - SAP NetWeaver 336
  - SMPP inbound channels 352
  - SMPP outbound channels 353
  - SMS Builder Web UI 369
  - virus scan adapter 336
- configuring applications 36
- connection failure 378
- connections
  - managing 84
- control flags 208
- controlFlag 208
- CPLC (card production life cycle) 274
- create
  - encrypted password preferences 266
  - hashed password 265
- create keystore for data encryption 271
- creating
  - categories 364
  - empty subscriber sets 365
  - users 360
  - workspaces 356
- creating an application definition 36
- creating application endpoint URL 38, 42, 50, 53, 57

- creating back-end connection 38, 42, 50, 53, 57
- cron expression
  - examples 338
  - format 338
  - special characters 338
- cron job
  - task handler 305
- cron utility 321
- customer credentials
  - hashing 314
- D**
- data 325
- data archiving 333
- data deletion 334
- data encryption
  - creating keystore 271
- data integration 3
- data retention 334
- database audit manager 301
- database drivers
  - DB2, configuring 345
  - Oracle 346
  - Oracle, enabling 347
- database layer 261
- database scripts
  - DB2 configuration 343
  - Oracle configuration 345
- database table
  - MOB\_JOBS 307
- databases
  - configuring 342
  - DB2 343
  - Oracle 345
  - scheduling events 350
  - subscribers 364
  - table maintenance 371
- DB2
  - database driver, configuring 345
  - database scripts, running 343
- deactivating users 362
- default menus
  - configuring 358
  - defined 357
- default ports 368
- defining
  - application authentication 58, 95, 165
- defining back-end connections 99
- deleting
  - categories 364
  - security profiles 98
  - users 363
  - workspaces 359
- deleting an application 86
- deleting back-end connections 101
- deletion script 335
- demand for payment 311
- deploying
  - hybrid app REST API 68
- deployment model
  - application layer 261
  - database layer 261
  - web layer 261
- Deregistration logs 103
- describing home screen 27
- describing icons 27
- developer 172
- displaying users and applications 27
- E**
- editing
  - categories 364
  - configuration files 370
  - user properties 361
  - workspaces 356
- editing an existing application 85
- editing back-end connections 100
- EmailChannel 293
- enabling
  - encryption 348
  - Oracle JDBC drivers 347
  - SSL 350
- enabling strong encryption
  - Java Development Kit (JDK) 230
  - JDK (Java Development Kit) 230
- encrypt
  - preferences 273
- encrypt preferences 273
- encrypt.sh encryption tool 349
- encrypted password preferences
  - creating 266
  - updating 266
- encrypting property values 349
- encryption
  - 3-DES 289
  - advanced encryption standard (AES) 230
  - AES 289
  - AES (advanced encryption standard) 230
  - CLIEncrypter 230

## Index

- configuration 267
- configuration files 230
- creating keystore 271
- Java Development Kit (JDK) 230
- JDK (Java Development Kit) 230
- messaging 289
- OSGi 230
- preferences 267
- encryption certificates
  - SAP Mobile Platform Server administration 227
- encryption, enabling 348
- event handler
  - business logic configuration 303
- event handlers
  - operations dashboard 328
- event handling
  - performance considerations 277
- event log
  - setting event log file rollover 128
  - setting event log file size 128
  - setting event log levels 126
  - system logging components 126
- event queues
  - physical queues 327
  - virtual queues 327
- event schedulers, configuring 350
- events 326
  - handlers 328
  - physical queue 327
  - queues 327
  - scheduled 327
  - virtual queue 327
- exception mapping 282
- exporting
  - preference nodes 319
- exporting application 92

## F

- felix.log file 371
- files
  - logging 371
- framework 278
  - bonecp 287
  - gateway 278
  - hibernate session factory 286
  - java database connectivity (JDBC) 287
  - JDBC (java database connectivity) 287

## G

- gateway 278
  - exception mapping 282
  - java management extensions authentication 279
  - JMX authentication 279
  - odata interface 283
  - open data protocol interface 283
  - standard security filters 280
  - tcp interface 284
  - transmission control protocol interface 284
- Gateway Management Cockpit 3
- gauge charts 332
- GCMChannel 300
- generate private keys 276
- glossaries
  - SAP Mobile Platform terms 383

## H

- handlers
  - tasks 331
- hashed password
  - creating 265
  - updating 265
- hashing
  - customer credentials 314
- hashing passwords 342
- help desk 172
- hibernate session factory 286
- HtmlChannel 294
- HTTP and HTTPS
  - ports 145
- HTTP cookies 244
- HTTP headers 244
- HTTP port
  - configuring 369
- HTTP proxy 116
- HttpChannelEnd 295
- hybrid apps
  - deploying, REST API reference 68

## I

- Impersonator role missing error 379
- importing
  - preference nodes 318

- importing application
  - guidelines for importing applications 93
  - troubleshooting an imported application 93
- information, server
  - physical memory 322
  - swap space 322
  - virtual memory 322
- installing
  - Oracle JDBC driver 346
- Integration Gateway 3
  - roles 174
- interface
  - odata (open data protocol) 283
  - open data protocol (odata) 283
  - tcp (transmission control protocol) 284
  - transmission control protocol (tcp) 284
- interface management
  - REST 333
  - SOAP 333
- Introscope 139
  - workload analysis 139

## J

- JAAS subjects
  - propagating single sign-on using 244
- JabberChannel 295
- java
  - memory settings 121
- java database connectivity (JDBC)
  - bonecp 287
- Java Development Kit (JDK)
  - enabling strong encryption 230
- java management extensions (JMX) 279
- java.io.exception 378
- javascript object notification (JSON)
  - audit manager 302
- JDBC (java database connectivity)
  - bonecp 287
- JDBC driver, Oracle 346
- JDK (Java Development Kit)
  - enabling strong encryption 230
- Jetty for SSL, configuring 350
- Jetty servlet container
  - configuring 369
- JMS channels, configuring 354
- JMX (java management extensions) 279
- jobs
  - business logic configuration 305
  - cron utility 321

- operations dashboard 321
- JSON (javascript object notification)
  - audit manager 302
- JSON audit manager 302

## K

- Kapsel apps
  - similar to hybrid apps 66
  - uploading new version 66
- keys
  - element 274
  - private 276
- keystore 223
  - configuration 267, 269
  - payment handler 269
- keytool utility 224
- keytool.exe 197

## L

- LDAP
  - configuration properties 197
  - configuring SAP Mobile Platform Server to use
    - 219
    - role computation 194
  - LDAP nested groups 196
  - LDAP security provider
    - modules available 193
  - LDAP SSL configuration 197
  - LDAP trees
    - multiple 193
  - life cycle
    - Agentry applications 72, 107, 131
    - Kapsel apps 66
  - load balancer
    - adding in DMZ 23
  - log files
    - setting and purging 103
  - log4j 369
  - logging 371
    - configuring log4j 369
    - database tables 371
    - processing engine 374
  - logging in
    - Operations Dashboard 317
  - logic
    - messaging 290

## Index

logs, application  
  viewing 103  
logs, trace 107

## M

maintenance

  and tuning 368  
  database tables 371

Management Cockpit

  setting up SSL certificates 28  
  starting and stopping on Linux 27  
  starting and stopping on Windows 26

managing

  back-end connections 86, 98–101  
  security profiles 94  
  tasks by application type 84  
  users 92

managing applications

  overview 84

managing multiple connections 98

memory settings

  java 121

mer private chargeKey (MPcK) 276

mer private readKey (MPrK) 276

mer private signingKey (MPsK) 276

messaging 287

  channels 293  
  encryption 289  
  engine 287  
  logic 290  
  template 292

messaging engine

  ActiveMQConnectionConfiguration 287  
  JmsReceiveCallback 287  
  PickupJmsMessages 287  
  QueueBasedFutureFactory 287

miscellaneous configuration 308

  demand for payment 311  
  one-time password generation 309  
  OTP generation 309  
  security endpoint 308  
  SMS available on cell authentication 309  
  transaction 310

MOB\_JOBS 307

Mobiliser

  3-DES encryption 289  
  AbstractCardPaymentHandler 269  
  access, web portal 312  
  ActiveMQConnectionConfiguration 287

AES encryption 289

audit 300

audit dispatcher 301

authentication, SMS 309

bonecp 287

business logic configuration 278

card production life cycle 274

changing passwords 317

channels 293

configuration, business logic 278

configure ClamAV 337

configure SAP NetWeaver 336

configure virus scan adapter 336

CPLC 274

create keystore 271

creating hashed password 265

cron job, task handler 305

customer credentials, hashing 314

data archiving 333

data deletion 334

data encryption 271

data retention 334

database audit manager 301

deletion script 335

demand for payment 311

deployment model 261

DummyCardPaymentHandler 269

EmailChannel 293

encrypt preferences 273

encrypted password preferences 266

encryption 267, 271, 289

encryption, configuration 267

event handler 303

event handling threads 277

exception mapping 282

framework 278

gateway 278

GCMChannel 300

generate private keys 276

Google Cloud Message (GCM) 300

handler 277

hashed password, creating 265

hashed password, updating 265

hashing customer credentials 314

hibernate 286

HtmlChannel 294

HttpChannelEnd 295

JabberChannel 295

java database connectivity (JDBC) 287



- java management extensions authentication 279
  - javascript object notification audit manager 302
  - JDBC (java database connectivity) 287
  - JmsReceiveCallback 287
  - JMX authentication 279
  - jobs 305
  - JSON audit manager 302
  - keystore 269
  - keystore configuration 269
  - keystore, configuration 267
  - logic 290
  - mer private chargeKey 276
  - mer private readKey 276
  - mer private signingKey 276
  - messaging 287
  - messaging encryption 289
  - messaging engine 287
  - messaging logic 290
  - messaging templates 292
  - miscellaneous configuration 308
  - MOB\_JOBS table 307
  - MOB\_PREFERENCES table 267
  - MPcK 276
  - MPrK 276
  - MPsK 276
  - near-field communication 274
  - NFC 274
  - odata interface 283
  - ODC 274, 276
  - ODC, configuring 274
  - ODC, installing 274
  - on-device charging 274, 276
  - on-device charging, on-device charging, configuring 274
  - on-device charging, installing 274
  - one-time password generation 309
  - open data protocol interface 283
  - Operations Dashboard 317
  - OTP generation 309
  - password preferences, creating 266
  - password preferences, updating 266
  - payment handler 269
  - performance considerations 277
  - PickupJmsMessages 287
  - prefent unauthorized access 312
  - preferences 267, 305
  - preferences, encrypting 273
  - provision secure element keys 274
  - proxy setup 262
  - QueueBasedFutureFactory 287
  - RsaPrivateKeyLogicImpl 269
  - RsaPublicKeyLogicImpl 269
  - secure element keys 274
  - security endpoint 308
  - security filters, standard 280
  - security fundamentals 312
  - SecurityConfigProvider 269
  - session factory, hibernate 286
  - setup 263
  - short message peer to peer channel 296
  - SmpChannel 296
  - SMS available on cell authentication 309
  - task handler, cron job 305
  - tasks 304
  - tcp interface 284
  - templates 292
  - thread pool 277
  - transaction configuration 310
  - transmission control protocol interface 284
  - unauthorized access, prevent 312
  - universal user 265
  - updating hashed password 265
  - virus protection 336
  - web portal user accounts 316
  - web portal, access 312
  - monitoring
    - tasks by application type 84
  - monitoring sessions 374
  - MPcK (mer private chargeKey) 276
  - MPrK (mer private readKey) 276
  - MPsK (mer private signingKey) 276
  - multiple LDAP trees 193
- ## N
- NamedCredential 244
  - near-field communication (NFC) 274
  - network communication
    - ports 145
  - network proxy server
    - enabling SOCKS proxy, for APNS connections 23
  - NFC (near-field communication) 274
  - Nginx reverse proxy
    - configuring with nginx.conf 214
  - Nginx Reverse Proxy
    - preparing for Agentry clients 213

## Index

- Nginx Web server
  - installing 213
- nodes
  - importing 318
  - preference, exporting 319
- Not found 378
- notification user 173
- O**
- odata (open data protocol)
  - interface 283
- ODC (on-device charging)
  - configuration 274
  - generate private keys 276
  - installation 274
  - provision secure element keys 274
- on-device charging (ODC)
  - configuration 274
  - generate private keys 276
  - installation 274
  - provision secure element keys 274
- one-time password (OTP)
  - generation 309
- open data protocol (odata)
  - interface 283
- opening a workspace 356
- Operations Dashboard 317
  - 3-DES encryption 289
  - adding preference nodes 320
  - AES encryption 289
  - applications, preferences 318
  - audit dispatcher 301
  - bar charts 332
  - channels 325
  - charts 332
  - data 325
  - database audit manager 301
  - EmailChannel 293
  - encrypting preferences 273
  - event handlers 328
  - event queues 327
  - events, servers 326
  - exporting nodes 319
  - gauge charts 332
  - GCMChannel 300
  - Google Cloud Messaging (GCM) 300
  - HtmlChannel 294
  - HttpChannelEnd 295
  - importing nodes 318
  - information 322
  - JabberChannel 295
  - java management extensions authentication 279
  - javascript object notification (JSON) 302
  - JMX authentication 279
  - jobs 321
  - JSON (javascript object notification) 302
  - logging in 317
  - messaging logic 290
  - physical queues 327
  - preference applications 318
  - preference nodes 318, 319
  - preferences 318
  - requests 324
  - scheduled events 327
  - server list 322
  - servers 322
  - short message peer-to-peer channel 296
  - SmppChannel 296
  - SMS available on cell authentication 309
  - standard security filters 280
  - system preferences 319
  - task details 330
  - task handlers 331
  - tasks, servers 329
  - trackers 332
  - virtual queues 327
- Oracle
  - database scripts, running 345
  - JDBC driver, enabling 347
  - JDBC driver, installing 346
- org.ops4j.pax.logging.properties file 369
- org.ops4j.pax.web.properties file 369
- OTP (one-time password)
  - generation 309
- outside corporate network 378
- P**
- partitions 119
- password policy
  - Data Vault login 254
- passwords
  - encrypting 349
  - hashing for the admin user 342
  - initial 360
- pax-web.log file 371
- payment handler 269
- performance 119, 374

- performance considerations 277
  - event handling 277
- persist.log file 371
- physical memory 322
- physical queues 327
- pinging authorization URL status 84, 86, 101
- pinging back-end connections 84, 86, 98, 101
- platform administrator 360
- platform administrators, SMS 341
- port numbers 147
- ports 368
  - HTTP, configuring 369
- ports, communication 145
  - HTTP and HTTPS 145
- preference node
  - adding 320
- preference nodes
  - exporting 319
  - importing 318
- preferences 318
  - applications 318
  - encrypting 273
  - encryption 267
  - nodes 319
- preparing the SAP server
  - SAP single sign-on 243
- prevent unauthorized access 312
- processing engine 372
  - performance 374
- processing-engine log 374
- product support 381
- production deployment model
  - application layer 261
  - database layer 261
  - web layer 261
- production system, setting up 342
- profiles, security
  - managing 94
- propagating
  - single sign-on 244
- property values, encrypting 349
- providers
  - authentication, how it works 177
- provision secure element keys 274
- provisioning applications 108
- proxy setup
  - standard reverse 262
- purging
  - log files 103

## R

- registration 253
- Registration logs 103
- registrations
  - managing 84
- Relay Server installation 209
- reporting
  - usage statistics 84
- reports 366
  - subscribers, generating 367
  - traffic, generating 367
- requests 324
- Resource bundle logs 103
- REST 333
- REST API reference
  - hybrid app, deploying 68
- restarting channels 353
- reverse proxy
  - installation 209
- role mapping 174
- roles
  - assigning to users 360
  - computing 194
  - user 362
- running
  - DB2 database scripts 343
  - Oracle database scripts 345

## S

- SAP
  - configuring SAP Mobile Platform components for 7
- SAP Mobile Platform
  - configuring components for 6
  - enabling connection environment for 6
- SAP Mobile Platform administrators 172
- SAP Mobile Platform Server
  - starting and stopping on Linux 25, 114
  - starting and stopping on Windows 25, 113
- SAP Mobile Platform Server administration
  - replacing default encryption certificates 227
- SAP NetWeaver
  - configure 336
  - virus scan adapter 336
- SAP single sign-on
  - preparing the SAP server 243
- scheduled events 327
- secure element keys 274

## Index

- security
  - application 231
  - enabling encryption 348
  - enabling SSL 350
  - encrypting properties 349
  - filters 280
  - hashing the admin password 342
- security certificates
  - See SSL certificates
- security configuration 164
- security endpoint 308
- security filters 280
- security fundamentals 312
  - hashing customer credentials 314
  - unauthorized access, prevent 312
- security log levels
  - decrease security log levels 220
  - increase security log levels 220
- security migration 260
- security postinstallation checklist 158
- security profile 164
- security profiles
  - deleting 98
  - for administration 219
  - managing 84, 94
- security providers
  - certificate 184
  - No Authentication Challenge 178
- server administration 113
- server crash 382
- server list 322
- server log
  - setting server log file rollover 128
  - setting server log file size 128
  - setting server log levels 126
  - system logging components 126
- server shut down 382
- servers 322
  - channels 325
  - data 325
  - events 326
  - information 322
  - requests 324
  - server list 322
  - tasks 329
- Service Marketplace 381
- service.brand\_webapp.properties file 369
- sessions, monitoring 374
- sets, subscriber 364
- setting and purging
  - log files 103
- setting event log file size and rollover 128
- setting server log file size and rollover 128
- setting server log levels 126
- setting up
  - production system 342
- single sign-on 232
  - propagating 244
- SiteMinder authentication 235
- SiteMinder security configuration 236
- SiteMinder Web agent 240
- Skipping LDAP role lookups 196
- SkipRoleLookup 196
- SMPP channels
  - inbound, configuring 352
  - outbound, configuring 353
- SmppChannel 296
- SMS administrators 341
- SMS available on cell authentication 309
- SMS Builder Web UI
  - configuring 369
- SOAP 333
- SOCKS proxy
  - enabling in a network proxy server, for APNS connections 23
- spring.log file 371
- SSL certificates
  - setting up 28
- SSL, enabling 350
- SSO 232
- SSO integration 232
- standard reverse proxy
  - setup 262
- standard security filters 280
- starting and stopping SAP Mobile Platform Server
  - on Linux 25, 114
- starting and stopping SAP Mobile Platform Server
  - on Windows 25, 113
- statistics data loss 382
- subscriber reports 367
- subscriber sets
  - empty, creating 365
  - uploading 366
- subscribers 364
- SunOne nested groups 196
- SUPER\_ADMIN role 362
- support 381
- swap space 322

- system administrators, SMS 341
- system logging components 126
- system preferences 319
- system.properties file 369

## T

- task details 330
- task handler
  - cron job 305
- task handlers 331
- tasks 329
  - business logic configuration 304
  - details 330
  - handlers 331
- tcp (transmission control protocol) 284
- technical support 381
- template
  - messaging 292
- terms
  - SAP Mobile Platform 383
- testing authorization URL status (Ping) 86, 101
- testing back-end connections (Ping) 86, 98, 101
- thread pool
  - tomcat 122
- tomcat
  - thread pool 122
- Toolkit for Integration Gateway
  - invoking operations 174
- trace logs 107
- trackers 332
- tracking configuration-file changes 370
- traffic reports, generating 367
- transaction configuration 310
- transmission control protocol (tcp) 284
- truststore 223
- tuning and maintenance 368

## U

- unauthorized access 312
- universal user
  - Mobiliser 265
- update
  - encrypted password preferences 266
  - hashed password 265
- uploading
  - Kapsel apps 66

- subscriber sets 366
- usage statistics
  - reporting 84
- user roles 362
- users 360
  - adding 360
  - deactivating 362
  - deleting 363
  - editing properties 361
  - managing 84, 92

## V

- viewing
  - application logs 103
  - viewing application usage statistics 102
  - viewing devices and applications 92
  - viewing statistics 27
  - viewing user connections 92
- virtual memory 322
- virtual queues 327
- virus protection 336
  - configure 336
  - virus scan adapter 336
- virus scan adapter
  - configure 336

## W

- web layer 261
- web portal
  - access 312
  - changing passwords 317
  - user accounts 316
- workspace administrators, SMS 341
- workspaces
  - creating 356
  - default menus 357
  - defined 354
  - deleting 359
  - opening 356

## X

- X.509
  - installing libraries 6, 7

