Ubisense

Ubisense Installation Guide

for Migrating ACS 2.6 + Series 7000 with Site connector to SmartSpace

Version 3.6

Part Number: SS_MIG_ACS/S7K/SC_3.6_EN

Ubisense Limited, St Andrew's House, St Andrew's Road, Cambridge CB4 1DL, United Kingdom.

Telephone: +44 (0)1223 535170. Website: www.ubisense.com.

Contents

Ubisense Migration Guide for ACS 2.6 + Series 7000 with Site connector

The following changes were made with the introduction of SmartSpace:

- Services were renamed to be clearly grouped according to the component to which they belong and to use names that more clearly identify their functionality
- Services which were previously required from a separate Ubisense Platform release were bundled into SmartSpace, and have also been renamed for clarity

When an ACS 2.6 dataset is upgraded to version 2.7, the data files must be migrated to the appropriate new layout so they are found by the new services. Otherwise most data will be lost during upgrade.

To support this, a process to migrate from ACS version 2.6 to ACS version 2.7 has been implemented.

You can only migrate services for which you have SmartSpace licenses. If you have services for functionality you have not licensed in SmartSpace, these services will be removed automatically during migration. Make sure you have installed SmartSpace licenses for *all* services you require to ensure their continued availability.

Use the following instructions to migrate ACS 2.6 with Series 7000 + Site connector to SmartSpace + Series7000 + ACS with Site connector.

Migrating your existing system is very like installing a new SmartSpace system. When you create a new installation, you install software onto one or more server machines, and start the core servers and controller(s); you then install and deploy licensed services; and, finally, download client machines and relevant tools and documents. Additionally when you migrate to SmartSpace, partway through the installation steps, you run the migration tool to start migrating your installation. You leave the migration tool running and then continue with the installation process. When all the required packages installed and deployed, you go back to the migration tool and allow it to complete the migration process. Finally you can complete any additional steps required to get your client machines running.

The steps you take to migrate to your new installation are listed below in the order they should be carried out:

1. *Back up the running dataset*.
2. *Install SmartSpace licenses*.
3. *Remove the existing Site connector server package and install the new Site connector server service*.
4. *Stop the core server and service controller machines*.
5. Install the SmartSpace 3.4 software for *server*, *admin* and *client machines*.
6. *Install the Ubisense Security Manager software*.
7. *Start the server machines and the application updater*.
8. *Check services are deployed and running in the expected configuration*.
9. *Start the ubisense_smartspace_migration_helper tool*.
10. *Install the SmartSpace packages*.
11. *Install the Series 7000 packages*.
12. *Install the ACS packages*.
13. *Continue with the migration using ubisense_smartspace_migration_helper*.
14. *Manually remove the Location platform integration service.*
15. *Remove empty folders from the dataset (optional)*.
16. *Run the ubisense_acs_monitor_spatial_relations tool*.
17. *Download software to client machines*.
18. Install *ACS text translations* and *audit translations*.
19. *Run the client software*.
20. *Configure Security Manager*.
21. *Restrict audit messages for automated processes*.
22. If required, install SmartSpace Web for *Windows* or *Linux*.

Migration requirements

Ubisense software is supplied in zipfiles with the name of the package followed by numbers indicating the version of the software, for example SmartSpace_3_4_7147.zip. Before you install the software required for the migration, you need to unzip these files into a *distribution directory* accessible to the machines on which you will be installing the software.

To migrate from ACS 2.6 to ACS 2.7 you will need to unzip software packages for the following:

- SmartSpace 3.4
- ACS 2.7.0
- ACS_Migration
- Series 7000
- Site connector

You will also need to unzip the Series 7000 sensor bootfiles you require. These are supplied in zipfiles with the name **Sensor_Bootfiles** followed by an identifier for the bootfile and numbers indicating the version of the software, for example **Sensor_Bootfiles_standard_2_1_11_7222.zip**.

Fix for installations where old services have been manually removed

In certain rare scenarios, generally where a dataset has been running for many years, files may have been removed manually from the dataset, for example to make the backup smaller. If old service files have been manually deleted from your installation without the removal of corresponding services in Service Manager, the migration process will fail with a message such as:

```
...
...
...
Services Ubisense/Unicast/Monitor Proxy needs removal
Services Ubisense/Visualisation/Representation Store needs removal
Total number of old services requiring removal = 48
Undeploying old services before removal
Starting undeploy procedure for 48 services
  waiting for all undeploy procedures to complete
  waiting for all undeploy procedures to complete
Removing migrated services
Failed to clean up file ubisense_objects_configuration_service (Ubisense:ACS:Object
Configuration) 1.8.2
Migration has finished with unresolvable errors.
You might need to restore from backup and retry.
```

Ubisense supply a tool, **ubisense_create_missing_dataset_files**, to allow you to correct this problem. If you have removed old service files manually, you *must* run **ubisense_create_missing_dataset_files** before you run the migration tool. Contact Ubisense Support if you need to use this tool and run it as shown in the example below.

You should take a backup of your dataset before you begin, so that it can be restored should any step fail.

Running ubisense_create_missing_dataset_files

1. From the command line, run **ubisense_create_missing_dataset_files**.

```
Create Missing Files
--------------------

This tool fixes up a dataset for which old service files have been
manually removed, by creating a dummy file for each missing file.
This avoids errors in migration, or Service Manager/CLEANUP SERVICES.
Have you taken a backup? [y/n]
```

2. Press Y followed by Enter to confirm you have taken a backup.

```
creating (Ubisense:SmartSpace core:[Downloadables]Location import and export)
3.4.7147 i586_windows_1.3 ubisense_locations_export.exe
creating (Ubisense:SmartSpace core:[Downloadables]Location import and export)
3.4.7147 i586_windows_1.3 ubisense_locations_import.exe
...
...
...
Attempted to create 16 missing files
Successfully created 16 missing files
Done
```

Migration process

Follow the steps in each of the following sections in turn to perform the migration.

Back up the dataset

Take a backup of the running dataset before you begin, so that it can be restored should any step fail.

Install SmartSpace Licenses

Installing Licenses on Windows

SmartSpace feature licenses are supplied as a zipfile with the name **FeatureSetup.zip**. Before you install the licenses, you need to unzip this file into a directory accessible to a server machine from which **ubisense_core_server** will be executed.

To install SmartSpace licenses:

1. Go to the directory where you unzipped the licenses.
2. Double-click the **FeatureSetup.msi** file and the Ubisense Feature Licenses Setup Wizard appears.
3. Click **Next** and the Ubisense Feature Licenses Setup wizard appears.
4. By default all licenses are selected for installation to the default location **C:\Program Files (x86)\Ubisense 3\bin**.
    - Click on the directory tree of licenses, click on individual features and choose whether or not they are to be installed
    - Click **Reset** to return the licenses selection to its default setting
    - Click **Browse** to navigate to a different directory to install the licenses in
5. When you have selected the files and location you require, click Next and then click **Install**.
6. When installation is complete, click **Finish** to close the wizard.

Installing Licenses on Linux

License files must be placed on the server so that the platform can find them. The default location is in the directory **/etc/ubisense**. If a different location is required, then the **license_search_path** can be defined in **platform.conf** (see *Configuration Parameters* for information on the location of this file). Each program also searches for licenses in the same directory as its executable. Licenses should be readable by both the platform user and by the operations group.

Upgrade Site connector

The Site connector server is an now independent service rather than a package deployed via the Ubisense platform. On Windows, this is installed as a Windows Service. On Linux, it should be started in the same way as core and controller, via a startup script or systemd, depending on the Linux distribution.

In all cases, you should undeploy and remove any existing Site Connector package, using Service Manager, before installing this release:

1. Run Service Manager and go to **Services / Ubisense / Site Connector**.
2. Right click **Site Connector Server** and choose **Undeploy Service**.
3. Right click **Site Connector Server** again, click **Remove Service** and confirm you want to remove the service.

Installing the Site connector server on Windows

1. Go to the **UbisenseSiteConnectorForServers** directory of your Site connector distribution directory.
2. Double-click the UbisenseSiteConnectorForServers.msi file and the Ubisense Site Connector For Servers Setup wizard appears.
3. Click **Next**.
4. Choose the Destination Folder for the software.
   You can accept the default **C:\Program Files (x86)\Ubisense 2.1** or change to another destination.
5. Click **Next** and click **Install**.
6. When the installation is complete, click **Finish** to close the Ubisense Site Connector For Servers Setup wizard.

After installation is complete, start the service using Windows Services manager:

1. Open Services by typing **View local services** in the Start menu.
2. Start the service **UbisenseSiteConnectorServer 2.1**.

   The service is configured to start automatically on reboot.

You can also stop and start the site connector service from the command prompt (as an administrator):

```
net stop "UbisenseSiteConnectorServer 2.1"
net start "UbisenseSiteConnectorServer 2.1"
```

Installing the Site connector server on Linux

For Linux, you can find the Site connector server executable in your distribution directory under **linux/server**. To install the executable:

1. Copy **ubisense_site_connector_server** onto your server
2. Create a startup script or systemd configuration to run the executable on startup.

Sample systemd startup script for the Site connector server

The following is a sample startup script for systemd.

```
[Unit]
Description=ubisense_site_connector_server daemon
After=multi-user.target

[Service]
User=platform
Group=platform
Type=simple
ExecStart=/home/platform/bin/ubisense_site_connector_server

[Install]
WantedBy=ubisense_service.target
```

Stop the core server and service controllers

Stopping the core server and service controllers on Windows

To stop the server software:

1. From a server machine, run Platform Control.
2. You stop the core server and service controller by:
   a. Selecting **UbisenseCoreServer 2.1** and then clicking **Stop**.
   b. Selecting **UbisenseServiceController 2.1** and then clicking **Stop**.

   The status of the service changes to **to be stopped**.
3. Click **Apply**. The status of the service changes to **not running**.

Stopping the core server and service controllers on Linux

First stop the Ubisense services. Either:

- Run the following command:

  ```
  ubisense_service_admin stop
  ```

  or

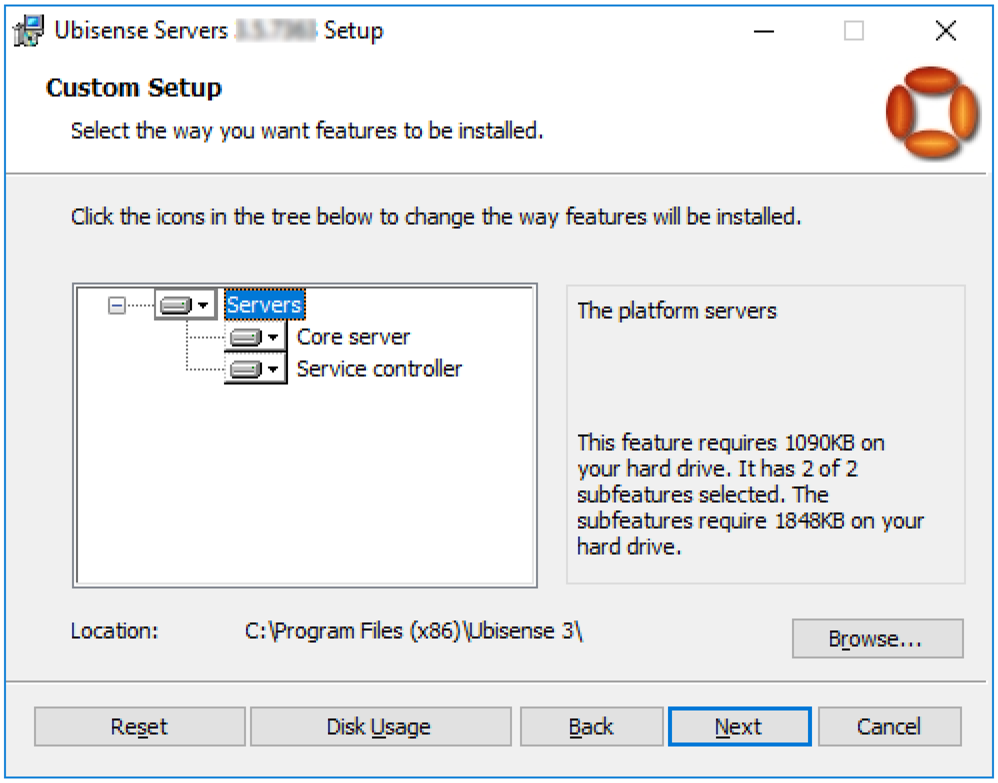
1. Run Service Manager.
2. Open the **Services** folder.
3. Right-click **Ubisense** and choose **Stop Service**.

After that stop the scripts you use to run the **ubisense_core_server** and **ubisense_local_control** services.

Install the Ubisense server software

Installing the server software on Windows

1. Go to the **servers\windows** directory of your SmartSpace distribution directory.
2. Double-click the UbisenseServers.msi file and the Ubisense Servers Setup wizard appears.
3. Click **Next** to display the Custom Setup dialog.



Ubisense Servers [redacted] Setup

Custom Setup

Select the way you want features to be installed.

Click the icons in the tree below to change the way features will be installed.

- Servers
  - Core server
  - Service controller

The platform servers

This feature requires 1090KB on your hard drive. It has 2 of 2 subfeatures selected. The subfeatures require 1848KB on your hard drive.

Location: C:\Program Files (x86)\Ubisense 3\

4. Choose the components to install. For each server machine you can choose to install either the core server or service controller or both. If you intend to run SmartSpace on a single server, you need to install both the core server and service controller on that machine. For an installation with more than one server, you need to run the core server on one machine only and the service controller on the rest, and you can install the components accordingly.

   By default, all features are selected. Choose whether to install or exclude items using the dropdowns beside their names. **Reset** returns you to the default selection.

5. Choose the Destination Folder for the software.
   You can accept the default **C:\Program Files (x86)\Ubisense 3** or change to another destination.

6. Click **Next** and click **Install**.

7. When the installation is complete, click **Finish** to close the Ubisense Servers Setup wizard.

Installing the server software on Linux

For Linux servers, there are two executables: **ubisense_core_server** and **ubisense_local_control**. You can find them in the following locations in the distribution directory:

```
servers/linux/ubisense_core_server
servers/linux/ubisense_local_control
```

If you want to run SmartSpace on a single server, copy both of these files to that machine.

If you want to run SmartSpace on several servers, copy **ubisense_core_server** onto one server machine only and **ubisense_local_control** onto the remainder of the machines.

Install software for admin machines

Installing the Service Manager software on Windows

1. Go to the **clients\windows** directory of your SmartSpace distribution directory.
2. Double-click the UbisenseServiceManager.msi file and the Ubisense Service Manager Setup wizard appears.
3. Click **Next**.
4. Choose the Destination Folder for the software.
   You can accept the default **C:\Program Files (x86)\Ubisense 3** or change to another destination.
5. Click **Next** and click **Install**.
6. When the installation is complete, click **Finish** to close the Ubisense Service Manager Setup wizard.

Installing the administrative software on Linux

Administrative executables, used to configure and maintain the running state of the Ubisense platform, should be executable by the operations group.

Your distribution directory contains the following admin executables:

```
    tools/linux/ubisense_backup
    tools/linux/ubisense_cache_service_credentials
    tools/linux/ubisense_configuration_client
    tools/linux/ubisense_file_downloader
    tools/linux/ubisense_installer
    tools/linux/ubisense_machine_id
    tools/linux/ubisense_multicast_test
    tools/linux/ubisense_proxyconfig_admin
    tools/linux/ubisense_restore_dataset
    tools/linux/ubisense_save_dataset
    tools/linux/ubisense_service_admin
    tools/linux/ubisense_service_ping
    tools/linux/ubisense_trace_receiver
    tools/linux/ubisense_transfer_config
```

Copy the executables from the distribution directory onto your admin machine.

Install the Ubisense Application Manager Software (Windows only)

1. Go to the **clients\windows** directory of your SmartSpace distribution directory.
2. Double-click the UbisenseApplicationManager.msi file and the Ubisense Application Manager Setup wizard appears.
3. Click **Next**.
4. Choose the Destination Folder for the software.
   You can accept the default **C:\Program Files (x86)\Ubisense 3** or change to another destination.
5. Click **Next** and click **Install**.
6. When the installation is complete, click **Finish** to close the Ubisense Application Manager Setup wizard.

Install the Security Manager Software

1. Go to the **clients\windows** directory of your SmartSpace distribution directory.
2. Double-click the UbisenseSecurityManager.msi file and the Ubisense Security Manager Setup wizard appears.
3. Click **Next**.

4. Choose the Destination Folder for the software.
   You can accept the default **C:\Program Files (x86)\Ubisense 3** or change to another destination.
5. Click **Next** and click **Install**.
6. When the installation is complete, click **Finish** to close the Ubisense Security Manager Setup wizard.

Start the core server and service controllers

Starting the core server and service controllers on Windows

To start the server software:

1. From a server machine, run Platform Control 3.
2. You start the core server and service controller by:
   a. Selecting **UbisenseApplicationUpdater 3** and then clicking **Start**.
   b. Selecting **UbisenseCoreServer 3** and then clicking **Start**.
   c. Selecting **UbisenseServiceController 3** and then clicking **Start**.

   The status of the service changes to **to be started**.
3. Click **Apply**. The status of the service changes to **running**.

Starting the core server and service controllers on Linux

Use your start up scripts to start either the **ubisense_core_server** or **ubisense_local_control** services on each of your server machines. (Example scripts are given in the Ubisense installation guides.)

Check services are deployed and running

Run Service Manager 3 to check services are deployed and running in the expected production configuration.

1. Run Service Manager 3 and open the MANAGE SERVICES tab.
2. Open **Controllers** under **Cells & Controllers**:
   - Ensure all registered controllers are running.
   - If you have any controllers that are no longer used, select them and click **Remove**.

3. If Series 1 License Support is present, it will be flashing red, and you should use Service Manager to undeploy and remove it.
    a. Select **Series 1 License Support** and click **Undeploy**.
    b. Select **Series 1 License Support** again, click **Remove** and confirm you want to remove the service.
4. If Series 7000 is present, logging needs to be switched on.
    - Run Location Engine Configuration and open the Monitoring tab to enable logging.
5. Clean the dataset services (optional).

    After the old services are migrated they can no longer be cleaned up by this tool, so it is advisable to clean them up before proceeding.
    - In Service Manager, click on **CLEANUP SERVICES** and, if there are services listed, click **Remove**.

Start the migration tool

The **ubisense_smartspace_migration_helper** is a command-line tool distributed with SmartSpace 3.4 or higher. It does not require installation and can be run directly from the distribution directory.

1. Start the migration tool.

    From the command line, run **ubisense_smartspace_migration_helper**.

```
SmartSpace Migration Helper
---------------------------

IMPORTANT INFORMATION.

This tool supports reorganization and migration of Ubisense datasets by
copying data for discontinued services to new services that manage the
same schemas.

This is done by undeploying old services (which copies their data to a
folder called 'migration' at the root of the dataset), installing new
services (which replace the old services), deploying the new services
with the data retrieved from the 'migration' folder, and then removing
the old services.

This procedure will only work if all services are deployed and running
in their expected production configuration before migration starts. If
this is not the case, or if anything fails during the migration process,
the dataset may be left in an invalid state, so you must take a backup
of the dataset and store it in a separate location before proceeding.

Have you taken a backup? [y/n]
y

Are all services deployed and running in their expected production configuration?
[y/n]
y
```

2. Press Y followed by Enter to confirm you have taken a backup.
3. Press Y followed by Enter to confirm you are happy with your setup and wait until you are prompted to install new services.

```
Setting core and controller to start migration mode
  waiting for core and controller to be in correct state
  waiting for controller <controllername>
Undeploying all migratable services -- this will store data files ready for
migration
Starting undeploy procedure for 64 services
  waiting for all undeploy procedures to complete
  waiting for all undeploy procedures to complete
  waiting for all undeploy procedures to complete
  waiting for all undeploy procedures to complete
Ready for installation of new services
Please use the Ubisense Service Manager to install and deploy new services.
  NOTE: installed services will not start yet - this is normal
Have you finished using Ubisense Service Manager to install services? [y/n]
```

Do not press Y at this stage. Leave the Command window open and go onto the next steps. (You will return to the prompt later.)

Install new services

You are now ready to install the services for the features you have licensed.

> At this point in the migration process, you might see services in unexpected states in the **MANAGE SERVICES** tab of Service Manager. However, all services are currently controlled by the migration helper and it will ensure the correct services are deployed.

1. From an admin machine, run Service Manager 3.
2. Click on **INSTALL SERVICES**.
3. Specify the directory from which to install.

   This is generally the **packages** folder in each of your distribution directories.
4. Select the features you want to install.

   For SmartSpace, which features are available depends on the licenses you have installed. Select the features you require from the list.

   For Series 7000, select the Series 7000 RTLS feature.

   Also for Series 7000, go to the sensor bootfiles distribution and select **Everything (no feature dependencies found)** to install the Sensor boot services.

   For ACS there are four features:

   - ACS (*required*: installs ACS and legacy platform services)
   - ACS Protocol (*optional*: installs a protocol service implementing the simple ACS protocol)
   - ACS Support (*optional*: installs a service for transferring external object information between installations)
   - Atlas Copco Open Protocol (*optional*: installs a protocol service implementing the Atlas Copco Open Protocol)
5. Click **Install**.
6. When installation is complete, click **Close** to close the Installing Services dialog.

You have now installed your SmartSpace features and you can continue with the migration process. **Right now you do not need to take any further action in Service Manager**.

Continue with the migration

1. Go back to the Command window where you started **ubisense_smartspace_migration_helper** to continue with the migration.
2. Press Y followed by Enter to confirm you have installed the services you require.

   The migration tool will proceed to copy service state into the correct new service locations and remove services that have been superseded by newly installed services.

```
...
Have you finished using Ubisense Service Manager to install services? [y/n]
y
Working out which old services should be removed
Services Ubisense/ACS/Association needs removal
Services Ubisense/ACS/External Object Information needs removal
Services Ubisense/ACS/Factory Layout needs removal
...
...
...
Services Ubisense/UTCP/External Systems needs removal
Services Ubisense/Unicast/Monitor Proxy needs removal
Services Ubisense/Visualisation/Representation Store needs removal
Total number of old services requiring removal = 48
Undeploying old services before removal
Starting undeploy procedure for 48 services
  waiting for all undeploy procedures to complete
  waiting for all undeploy procedures to complete
Removing migrated services
Setting core and controller to finish migration mode
  waiting for core and controller to be in correct state
  waiting for controller <controllername>
  waiting for controller <controllername>
Saving all service state back to core
  waiting for services to finish backup
  waiting for services to finish backup
  waiting for services to finish backup
Setting core and controller back to non-migration mode
  waiting for core and controller to be in correct state
  waiting for controller <controllername>
Done
```

Manually remove the Location platform integration service

During the upgrade process, a service called **Location platform integration** may have been left in place. This service is not required after upgrading and, if it is present, you must remove it from your installation.

To verify if the service is present and to remove it:

1. Run Service Manager 3 and click the MANAGE SERVICES tab.
2. Locate **Ubisense:Location system:Location platform integration**.

   You can use the hierarchy of folders in the SERVICES pane to find it by clicking on the > beside **All**, **Ubisense**, and **Location system** in turn to refine the list.
3. If **Location platform integration** is present, select it, and then click **Remove**. Click **Remove** once again to confirm and the service is deleted.

Remove empty folders from the dataset

When the migration process is complete, you can remove any empty folders from the dataset.

- If you have the Linux or Cygwin **find** tool you can use:

```
cd <datasetfolder>
find . -type d -empty -delete
```

- If you have PowerShell you can use:

```
Get-ChildItem -recurse | Where {$_.PSIsContainer -and `
@(Get-ChildItem -Lit $_.Fullname -r | Where {!$_.PSIsContainer}).Length -eq 0} |
Remove-Item -recurse
```

Run the ubisense_acs_monitor_spatial_relations tool

To ensure that the spatial relations monitored in your ACS 2.6 installation continue to be monitored after migration to SmartSpace, you must run the **ubisense_acs_monitor_spatial_relations** tool. (You can run this tool at any time to ensure all spatial relations required by ACS are monitored.)

You can find **ubisense_acs_monitor_spatial_relations** in the **linux** or **windows** folder in the **ACS_Migration** distribution directory. To use the tool, from the command line, run **ubisense_acs_monitor_spatial_relations**. (Note: The tool does not display a message on completion.)

Downloading ACS programs to client machines (Windows)

Managing applications

To create shortcuts to ACS applications:

1. Run the Ubisense Application Manager and click on **APPLICATIONS**.
2. Available applications are listed, with their version numbers and, where applicable, location on the Start menu.

   Choose the applications you want to install.

   - Double-click a single application
   - Select several applications and press Enter

   The following SmartSpace client program is available:

   - SmartSpace Config (the main SmartSpace configuration GUI)

   The following Series 7000 client program is available:

   - Location Engine Configuration

   The following ACS client programs are available:

   - ACS Main GUI (main ACS configuration GUI)
   - Product Tag Association GUI (standalone GUI for associating Ubisense tags to products)
   - Pulsed Line Product Feeder GUI (standalone GUI for feeding pulsed assembly lines with products)
3. Click **Create shortcuts for selected applications**.

   Shortcuts are created in the Start menu in the locations indicated.

Managing tools and documents

To download ACS command-line tools and documents to a selected directory:

1. Run the Ubisense Application Manager and click on **DOWNLOADABLES**.
   Command-line tools and documents are listed in different categories.
2. Choose the tools or documents you want to download:

   - Double-click a single file name
   - Select several files and press Enter
3. For SmartSpace, the tools and documents available to you depend on the features you have installed.

4. Under **Ubisense Generation 2.X>Series 7000 RTLS>Administration tools and documentation**, the following command-line tools and documents are available.

   Series 7000 tools

   - ubisense_battery_monitor_config
   - ubisense_le_ident
   - ubisense_reboot_sensor
   - ubisense_sensor_ip_configuration
   - ubisense_tag_cleanup_config

   Series 7000 documents

   - MulticellOperation.pdf
   - UbisenseTagCleanupService.pdf

5. Under **Ubisense Generation 2.X>ACS>Tools**, the ACS command-line tools are available.

   ACS tools

   - ubisense_acs_protocol_clientserver (simulate a tool controller using the simple ACS protocol)
   - ubisense_create_config_translation_template (write a template for configuration translations)
   - ubisense_device_tag_simulator (simulate tag location updates in assembly line stations)
   - ubisense_open_protocol_clientserver (simulate a PF controller using the Atlas Copco Open Protocol)
   - ubisense_product_data_dumper (write current product instance data to a file)
   - ubisense_product_data_loader (load product instance data from a file)
   - ubisense_product_instance_cleanup (remove product instances that have no tag association)
   - ubisense_product_tag_simulator (simulate product tags moving along assembly lines)
   - ubisense_text_translation_reader (read .utf translation files and store them in the dataset)

6. Under **Ubisense Generation 2.X>ACS > SmartSpace backwards compatibility with ACS > ACS SmartSpa** **tools**, the ubisense_acs_smartspace_sync_admin command-line tool is available. If you intend to synchronize ACS parameters with SmartSpace properties, download this tool. See *ACS Configuration* for information on syncing ACS and SmartSpace properties.
7. Specify the directory to install the files in and click **Start download**.

   The files are downloaded to the specified directory.

Whenever you upgrade your ACS installation, you must follow the process described above to replace your existing tools and documents with upgraded versions.

Adding Text Translations to the Dataset

ACS requires server side text translation to be loaded into the dataset. This step requires the **ubisense_text_trans** which you can download using the Ubisense Application Manager (in DOWNLOADABLES > Ubisense Generation 2.X > ACS > Tools). Ensure you have downloaded the tool and that it is on your path.

To load the text translations:

1. Open a Command Prompt window.
2. Change to the ACS-2.7.0 distribution directory (where you unzipped the ACS software).
3. Run the following command:

```
ubisense_text_translation_reader.exe translations
```

Customized Labels in the Product Tag Association Tool

You can configure the Product Tag Association GUI to display additional text for Product ID and Tag ID. You add the configurable text to the relevant text translation file(s), and then add the file (s) to the dataset by following the steps described above.

See *Example: Configuring Labels in the Product Tag Association GUI* in the ACS Help for an example.

Adding Audit Translations to the Dataset

ACS requires server side translation for the Audit UI to be loaded into the dataset. This step requires the ubisense_translation_tool which you can download using the Ubisense Application

Manager (in DOWNLOADABLES > Ubisense > Platform > Translation tool). Ensure you have downloaded the tool and the DLLs that accompany it and that it is on your path.

1. Open a Command Prompt window.
2. Change to the ACS-2.7.0 distribution directory (where you unzipped the ACS software).
3. Run the following command:

```
ubisense_translation_tool import translations\for_ubisense_
translation_tool\acs_2_7_0_en_de.xliff
```

Running the Client Software (Windows Only)

To run the client software, use the Windows shortcuts you downloaded with the Ubisense Application Manager:

- Start > Ubisense 3 > ACS Main GUI
- Start > Ubisense 3 > Location Engine Configuration
- Start > Ubisense 3 > Product Tag Association GUI
- Start > Ubisense 3 > Pulsed Line Product Feeder GUI
- Start > Ubisense 3 > SmartSpace Config

These shortcuts run the UbisenseApplicationUpdater program installed as part of Ubisense SmartSpace which will download and start the correct version of a selected program.

Settings in Ubisense Security Manager

If required, use Security Manager to restrict users from performing certain actions in clients such as ACS Main GUI.

Audit Messages for Automated Processes

Because ACS logs audit messages for various automated processes, a large number of audit messages can be generated. As a result, ACS can eventually become unusable and possibly fill the disk. To avoid this, you can set a configuration parameter—**audit_ignore_user_substring**—to prevent the generation of certain audit messages by one or more authenticated users.

By default, services that are authenticated with a user name containing the text string `[Automated]` do not produce audit messages. By specifying a different text string you can "ignore" audit messages from processes that are authenticated with user names containing that string of text.

Use the following command to exclude audit messages for a set of users:

```
ubisense_configuration_client set audit_ignore_user_substring <username_
to_be_ignored>
```

where any user name that contains <username_to_be_ignored> will be ignored.

For example,

```
ubisense_configuration_client set audit_ignore_user_substring notaudited
```

would prevent the generation of audit messages by services authenticated by the *notaudited* user and any other user whose name contains the string *notaudited*.

Installing SmartSpace Web on Windows

If you have licensed SmartSpace features that are accessed in a browser, such as Web maps or Web forms, you need to set up a web server before installing these features.

To install and configure the Web Server:

1. Enable Internet Information Services.
2. Install the Windows windows-core-req.
3. Install the websites required.
4. Modify the website configuration files, if required.

Enable Internet Information Services (IIS) on Windows Server 2016

1. Open the Server Manager, click Add roles and features.
2. Click through to Server Roles, and select Web Server (IIS).
3. Click through to Web Server Role (IIS) and under Role Services, ensure that, in addition to the default features, you have enabled Security/Windows Authentication.
4. Click through to Confirm the installation.

Due its increased vulnerabilities, NTLM is no longer added as a provider for Windows authentication, when SmartSpace Web is installed. If you need to use NTLM, you can manually add it as a provider in IIS Manager.
Firefox does not support Windows authentication by default without NTLM. If you need to support Firefox, you can either add NTLM manually or use this workaround:
The workaround for Firefox requires the browser settings to be changed:

1. Open Firefox and enter **about:config** in the address bar. Click to confirm advanced configuration.
2. In the Filter field, enter **negotiate**.
3. Double-click **network.negotiate-auth.trusted-uris**. This preference lists the trusted sites for Kerberos authentication.

4. Enter your domain (e.g. **company.local**)
5. Click Save (the tick button).

Install the Windows ASP.NET Core Runtime Hosting Bundle

1. Download the Microsoft windows-core-req from *https://dotnet.microsoft.com/en-us/download/dotnet/6.0* which includes the .NET Runtime and IIS support.
2. Run the installer.
3. If .NET was not previously installed on the server, then a reboot is required for IIS to pick up the path to the .NET Runtime.

Install the Website and/or REST API

Follow these instructions to install the SmartSpace Web application.

When you install the SmartSpace Web application, the following components are created as part of the installation process:

Component	Description
Application Pool:	SmartSpace has its own application pool.
Website:	The entry point to SmartSpace via a browser.

To install the SmartSpace Web application:

1. Go to the **web\windows** directory of your SmartSpace distribution directory.
2. Double-click the **SmartSpaceWeb.msi**.
3. Enter a Website Name: this name will form part of the URL when accessing SmartSpace in a browser.
4. Choose the location for the software.

   You can accept the default **C:\Program Files (x86)\Ubisense 3\SmartSpace\** or click **Change** to select another destination.
5. Enter an Application Pool name.
6. Click **Next** and **Install**.

To install the SmartSpace Web API:

1. Go to the **web\windows** directory of your SmartSpace distribution directory.
2. Double-click the **SmartSpaceWebApi.msi**.
3. Enter a Website Name: this name will form part of the URL when accessing SmartSpace in a browser.
4. Choose the location for the software.

   You can accept the default **C:\Program Files (x86)\Ubisense 3\WebApiCore\** or click **Change** to select another destination.
5. Enter an Application Pool name.
6. Click **Next** and **Install**.

By default, the SmartSpace website can be accessed by navigating to **http://localhost/smartspace** and the REST API can be accessed by navigating to **http://localhost/smartspaceapi**.

Workaround for Failed Installation due to PowerShell Configuration

Installation of SmartSpace Web or the Application REST API can sometimes fail and rollback. This can be because the local machine has been configured to require signed PowerShell scripts. The workaround is to temporarily remove this configuration from the Windows Registry in order to run the installer successfully.

In **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\PowerShell**, ensure you have the following values set:

Key	Type	Value
EnableScripts	REG_DWORD	1
ExecutionPolicy	REG_SZ	Unrestricted

Modify the website configuration files if required

Advanced configuration of the websites is done by creating and editing configuration files in the installation folders. By default, on Windows, these files are in:

- SmartSpace website: **C:\Program Files (x86)\Ubisense 3\SmartSpace\Web\localsettings.json**
- REST API website: **C:\Program Files (x86)\Ubisense 3\WebApiCore\Web\localsettings.json**

These files should be created if you wish to make advanced configurations, rather than modifying the installed defaults in **appsettings.json**. This is because **appsettings.json** can be overwritten on a software upgrade.

Using OpenID Connect to enable external authentication

OpenID Connect support allows an external authentication authority to handle login and provide user ID and roles to SmartSpace Web. The external authority takes care of ensuring that the user is allowed to access the application, and sends claims about the user back to SmartSpace Web. SmartSpace can then be configured to extract the user ID, and optionally the roles, from the claims provided by the authority.

To use OpenID Connect, SmartSpace must be accessed via https, so make sure you have configured SSL in IIS.

There are two places where OpenID Connect must be configured: at the authority, and in SmartSpace Web.

Configuration at the Authority

The authority provides an application client ID and secret, and is configured to accept the redirect URIs that will be used during authentication. The authority will also be configured to control which users/groups are allowed to access SmartSpace, and which properties should be sent to SmartSpace for it to use as the user ID and/or roles. Generally this involves creating a new "application" instance or registration at the authority, and then configuring the application to work with SmartSpace.

In order to ensure that only valid granted authorities are used, both the authority and SmartSpace share a configured "client id" and "client secret". The ID identifies the specific SmartSpace web site application at the authority, and the secret is used by SmartSpace to verify that the returned claims are valid. So the first step to setting up the authority is to create a new client ID in the authority, and to generate the client secret. The application must have the "code flow" or "code" response type enabled (also known as "Authorization code grant").

Authorities have different ways to do this, but for example, in Microsoft Azure Active Directory (AD), at the time of writing:

1. Go to App Registrations.
2. Click New registration.
3. Give the application a name and the set of users who can log in, and then click Register.

4. Click Certificates and secrets, and create a new secret.
5. Take a copy of the Value (it can only be viewed/copied on creation).

Set up the redirect URIs used during login and logout. These must match the redirect requests that the SmartSpace website generates when it attempts to login or logout a user by directing the browser to the authority.

- Sign in or callback URI: **https://<your server host as seen by the end user>/SmartSpace/signin-oidc**
- Sign out or front channel logout URI: **https://<your server host as seen by the end user>/SmartSpace/**

You can also define roles, or put a set of AD groups as claims. See the Azure AD documentation for how this is done.

Configuration in SmartSpace Web

There is an OpenID Connection configuration section in the web settings JSON files. Depending on your operating system this file is either **localsettings.json**, or **/etc/ubisense/web.json**.

```
"OpenIDC" : {
         "ClientId": "865b3f07-3f30-4881-895b-a831d11917ac",
         "ClientSecret": "<your secret shared with the authority goes here>",
         "MetadataAddress": "https://login.microsoftonline.com/fec9c4e9-d7de-4363-a4e3-
039b6f2606d2/v2.0/.well-known/openid-configuration",
         "IdClaim": "preferred_username",
         "RoleClaim": "http://schemas.microsoft.com/ws/2008/06/identity/claims/role",
         "LogoutURL": "",
         "DebugClaims": true
   },
```

The settings in this section are case sensitive, and have the following meanings:

Setting	Meaning
ClientId	The identity of this app at the OpenID Connect authority
ClientSecret	The shared secret used to generate and validate claims about the logged-in user
MetadataAddress	The OpenID Connect configuration meta-data document from your authority. For Azure AD, for example, go to Endpoints under your application registration, and copy "OpenID Connect metadata document".
IdClaim	The claim type to be used as the logged-in user in SmartSpace

Setting	Meaning
RoleClaim	The claim type to be used as a role for the user in SmartSpace. Optional
LogoutURL	Some OpenID Connect authorities do not provide an "end_session_endpoint", so logout will be unsuccessful. In this case, provide a URL to log out the user from the client ID at the authority. Optional
DebugClaims	If set true, this will provide a page within SmartSpace Web that you can visit once authentication has redirected back to SmartSpace, and see the claims provided. The page can be found at SmartSpace/Auth/Claims. It is recommended you disable this configuration in production. The default is false. Optional

After the configuration has been saved, restart the web site. Go to IIS Manager -> Application Pools -> SmartSpaceCore -> Start.

You can test log in by visiting the SmartSpace Web site and clicking Login. You should be redirected to the authority login page. On successfully logging in, you will be asked to confirm sharing your profile with SmartSpace, and then be redirected back to SmartSpace Web. Note that if you are already logged in at the authority (for example if you are logged into Microsoft 365), there may be no need to provide any user/password – you may not even see the redirects as they can happen very quickly.

On success, the user ID should be displayed in the top left of the SmartSpace Web site. If this is not the case, enable DebugClaims, restart SmartSpace Web, and authenticate again. Then visit /SmartSpace/Auth/Claims to see what the authority returned, and pick a suitable IdClaim.

Using Roles from OpenID Connect

Claims returned by the authority that have a claim type configured in RoleClaim will be treated as user roles. These will be passed transparently through to SmartSpace. To use them within SmartSpace they should also be created as roles using the USERS / ROLES task in SmartSpace Config. You do not need to add members of the role within SmartSpace Config (though this is allowed if necessary). Users that have the role according to the OpenID Connect authority will automatically be treated as having the role within SmartSpace Web.

For example, if the authority passes a claim "SmartSpace Supervisor", you should create a "SmartSpace Supervisor" role in the USERS / ROLES task in SmartSpace Config. After you have created the role within SmartSpace, you can:

- assign this role as a member of other roles
- specify the searches/views/properties that this role can access directly

- assign the role to reports, HMIs, and in ObjectView API views

Again, you can use *DebugClaims* to see what the authority has returned to SmartSpace for the currently authenticated user. Remember to disable DebugClaims and restart the web site before going into production.

Header Authentication (SiteMinder)

The websites can read the authenticated user from a header passed to them by a proxy server, such as SiteMinder. To configure this, set "AuthOptions/UserHeader" to be the name of the header from which the logged in user is to be extracted.

```
{
    "AuthOptions": {
       "UserHeader": "SITEMINDER_USER"
   }
}
```

If you configure this option, it is vital that users cannot access the website except through the proxy server, because otherwise they could add their own header as part of the request and gain unauthorized access. Typically in this case you would configure IIS bindings to only listen on the loopback interface, or configure IP Address and Domain Restrictions. See Microsoft IIS documentation for details.

Header Authentication with Other External Authentication Systems

Upstream reverse proxies can provide some other external authentication system, and pass both user and roles from that external system to SmartSpace.

AuthOptions.RolesHeader can be used with AuthOptions.UserHeader to specify a header from which to read roles for passthrough to SmartSpace. Roles are comma separated in the value of the header provided. The config is optional: if not specified in the settings file then no roles will be passed through. The option is ignored if UserHeader is not set.

For example:

```
  "AuthOptions": {
       "UserHeader": "AUTH_USER",
       "RolesHeader": "AUTH_ROLES",
        ...
   }
```

This would read the user (authenticated by the up-stream reverse proxy) from header AUTH_USER, and the roles from header AUTH_ROLES. If these are used, the web site should not be accessible except from the reverse proxy, which should filter out any user-supplied values for these headers.

As with roles in *OpenID Connect*, the roles that can be passed in the header should also be created in the USERS / ROLES task in SmartSpace Config so they can be assigned as members of other roles, or given searchs/views, etc.

Switching to Forms Authentication

By default the Windows website installation uses Windows authentication for login. You can switch to forms authentication by configuring LDAP parameters and setting up "AuthOptions/UseCookiesOnWindows". For production or integration deployment, you will need an SSL certificate signed by a suitable root authority for your users. For development or test deployment, a test certificate can be used instead (e.g. generated locally using OpenSSL). If you configure forms authenticatication without SSL, it will not work.

In the following examples, the first example shows the configuration for an Active Directory server, whilst the second shows the configuration for an LDAP server that does not require a login for searching.

LDAP authentication with an Active Directory server

```
"LDAPAuth": {
      "Server": "adserver.company.com",
      "Port": "389",
      "User": "",
      "Password": "",
      "SearchStart": "dc=company,dc=com",
      "AccountId": "sAMAccountName"
   },

   "AuthOptions": {
      "UseCookiesOnWindows" : true,
      "ExpiryTimeSpan": "00:30",
      "SlidingExpiry":  true
   }
}
```

LDAP authentication with no login for searching

```
"LDAPAuth": {
      "Server": "ldap.company.com",
      "Port": "389",
      "User": "",
      "Password": "",
      "SearchStart": "ou=people,dc=company,dc=com",
      "AccountId": "uid",
      "ObjectClass": "account"
   },

   "AuthOptions": {
      "UseCookiesOnWindows" : true,
      "ExpiryTimeSpan": "00:30",
      "SlidingExpiry":  true
   }
}
```

Note: The example given above works with the default OpenLDAP schema: other servers and schema might require different parameters.

How the LDAP validator behaves

The LDAP validator does the following:

1. If User option set, bind using this user/password.
2. If SearchStart is set, use the SearchStart, AccountId and ObjectClass to search for the DN of the entered user name.
3. Bind using the DN, if the search succeeded, or the entered user name. Use the entered password. If this bind succeeds, the user is authenticated successfully.

The validator reports what it is doing on the website trace (always on). for example:

```
[Tue Sep 24 14:23:03 2019, 127.0.0.1:42943] website: LDAPValidator: Is user valid
marcin
[Tue Sep 24 14:23:03 2019, 127.0.0.1:42943] website: LDAPValidator: searching (&
(objectclass=nsAccount)(uid=marcin)) at base ou=people,dc=ubisense,dc=aws
[Tue Sep 24 14:23:03 2019, 127.0.0.1:42943] website: LDAPValidator: binding as user
uid=marcin,ou=People,dc=ubisense,dc=aws
```

Enabling Secure LDAP (LDAPS)

To enable secure LDAP (LDAPS) for the connection you must use port 636. By using this port, SSL/TLS is enabled for the connection to the LDAP server (all other port numbers use TCP).

```
"LDAPAuth": {
      "Server": "adserver.company.com",
      "Port": "636",
      "User": "",
      "Password": "",
      "SearchStart": "dc=company,dc=com",
      "AccountId": "sAMAccountName"
   },

   "AuthOptions": {
      "UseCookiesOnWindows" : true,
      "ExpiryTimeSpan": "00:30",
      "SlidingExpiry":  true
   }
}
```

Configuring the Authentication Timespan

The cookies authentication uses an expiry time of 30 minutes and a sliding timespan. This means that the authentication will expire 30 minutes after the user closes the website, but will continue to be refreshed while the user is still visiting the website.

You can disable this sliding expiry and set an absolute time after which login will need to be repeated. For example, to log out after three hours:

```
{
   "AuthOptions": {
      "ExpiryTimeSpan": "03:00",
      "SlidingExpiry":  false
   }
}
```

Disable Hardened Headers

By default the website injects headers in each response for penetration security. These disable cross-site/cross-frame scripting, prevent content type sniffing, etc. If necessary, these headers can be disabled, and IIS configured manually to add appropriate headers instead.

```
{
    "SecurityOptions": {
       "HardenHeaders": false
    }
}
```

Enable Cross Origin Scripting

CORS is supported for the SmartSpace REST API. For features using SmartSpace Web where CORS is not supported, there are workarounds such as making configuration changes so that the browser treats localhost requests and the remote site as if they come from the same domain, disabling hardened headers (see above), or using IIS or Apache.

By default, no CORS headers are sent, so browsers will refuse to execute the API web methods from a page served from a different web server.

The option AllowOrigins in the **appsettings.json** file which overrides settings in **localsettings.json** enables cross origin scripting:

```
{
...
   "SecurityOptions": {
   "HardenHeaders": true,
   "AllowOrigins": [ "http://example.com", "https://*.mydomain.com" ]
   }
...
}
```

If AllowOrigins is set, and matches the origin of a request, the API will respond with suitable headers, for example:

```
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: X-Requested-With
Access-Control-Allow-Methods: PUT
Access-Control-Allow-Origin: https://server.mydomain.com
Access-Control-Max-Age: 3600
```

The browser will now allow the request. Note that this allows the browser to cache the pre-flight OPTIONS responses for up to an hour, to reduce load on the API server. Thus changes to the allowed origins may not be picked up by browsers for an hour.

Adding a custom theme for the website

You can customize the look of the SmartSpace website to better reflect your corporate identity, for example by replacing the Ubisense logo with your own or using a custom stylesheet. To prevent such customization from being overwritten during website upgrades, you should store your local theme in an external folder on the host machine, for example **C:\Ubisense\localtheme**. You specify the folder using the website configuration setting ContentOptions / LocalThemePath.

```
{
    "ContentOptions": {
       "LocalThemePath": "C:\\Ubisense\\localtheme\\"
    }
}
```

If the custom theme folder contains any **.min.css** files, they are loaded in place of the **wwwroot/bundle/base.css**. If the folder contains **custom.css**, it will be loaded in addition to other **css** files.

The following files can also be overridden by adding corresponding files in the custom theme folder:

- **images/logo.png**
- **images/background.png**
- **images/ubisense_large.png**
- **images/ubisense_small.png**
- **manifest.json**

Errors

When the website is loaded, you may see 502.5 "ANCM Out-of-Process Startup Failure".

This is normally because the .NET Runtime could not be found. Make sure you restarted the server after installing the windows-core-req.

Also, make sure that you included the Security/Windows Authentication feature when deploying IIS, as this is required by the websites on Windows unless forms or header authentication is configured.

Security Configuration for SmartSpace Web with Security Manager

If you are using a non-trivial security manager configuration to force authentication for services (as is the case for ACS installations) then you must run the **ubisense_cache_service_credentials** tool on the web server host. This is because the **credentials.dat** file created by the tool (in the latest version) allows IIS_IUSRS as a reader. Without this, the web site code cannot read the credentials, and therefore cannot connect to the platform services it needs.

For a Windows server, you *must* use the version of the **ubisense_cache_service_credentials** tool from the 3.4 sp1 distribution or above.

Installing SmartSpace Web on Linux

If you have licensed SmartSpace features that are accessed in a browser, such as Web maps or Web forms, you need to set up a web server before installing these features.

In this section, we will describe configuring the websites using Apache2 as a reverse proxy. *Advanced configuration* options will then be covered later.

Linux Requirements for SmartSpace Web

We support recent enterprise Linux distributions, such as SUSE Linux Enterprise Server 11+ or Red Hat® Enterprise Linux® v7+.

The following instructions assume you are configuring a reverse proxy in Apache 2.4.23 or above. For Red Hat® Enterprise Linux® Apache 2.4.23 is only available for version 8+. Whilst configuring a reverse proxy on earlier versions of Red Hat® Enterprise Linux® is possible, instructions for this are beyond the scope of this guide.

The following packages must be installed on the server.

- ldap 2.4 libraries

For production or integration deployment, you will need an SSL certificate signed by a suitable root authority for your users. This certificate will be installed for Apache2. SSL (TLS) is required for Linux installation because the website uses forms-based authentication, and so must have transport level security configured. For development or test deployment, a test certificate can be used instead (e.g. generated locally using OpenSSL).

Microsoft .NET Runtimes

See *Microsoft .NET Runtimes* for information on .NET runtimes for Linux.

Server Configuration

Ensure the web server is connected to the platform, using multicast, unicast cluster (see Smart Space Using Unicast Cluster Setup Guide on the Ubisense Documentation Portal), or a site connector.

If you have not already done so, on the web server install the platform servers for Linux, configure a dataset directory, and start a local controller. The website will be deployed on this controller. See *Installing the Ubisense SmartSpace software on Linux*.

If linux-core-req is already installed on the server, the websites will use that runtime (follow the instructions for Linux on https://dotnet.microsoft.com/). If a server-wide installation of .NET Core is not provided, then an isolated local runtime will be used, shared only by the platform services that use it.

Authentication Options

There are two authentication methods supported in the web sites when deployed on Linux: Forms authentication, and Header authentication.

Forms authentication directs the user to a login page when they attempt to access a part of the web sites that requires authentication. This login page gathers the user and password, which are then validated using a configured LDAP server. If this succeeds, a cookie is returned to the user's browser, which is used to authenticate in subsequent requests. Forms authentication requires the web site to be accessed via HTTPS for all authenticated or login traffic, to protect the credentials and cookies. This is configured in the reverse proxy that handles the HTTPS protocol.

Header authentication relies on another system, such as SiteMinder, doing authentication before passing the request to the website. A special header is set by this up-stream system on each request that reaches the web site, indicating the authenticated user. The web site simply assumes that the given header is authoritative. This is a commonly used method in enterprise environments.

Either method can be used for the SmartSpace website, but the Rest API only supports Header authentication (or no authentication).

Configuration Files

Set up the configuration files for the SmartSpace website and REST API. On Linux these are placed in **/etc/ubisense**. The files should all be readable only by the user that runs the platform controller. The following configuration files are used:

web.json

This contains configuration specific to the website. In the following examples, we will set up the parameters to access the LDAP server used to validate the user's credentials. We also set a proxy base path matching the reverse proxy path we will configure in Apache2 for the website. The first example shows the configuration for an Active Directory server, whilst the second shows the configuration for an LDAP server that does not require a login for searching.

LDAP authentication with an Active Directory server

```
{
    "LDAPAuth": {
      "Server": "adserver.company.com",
      "Port": "389",
      "User": "",
      "Password": "",
      "SearchStart": "dc=company,dc=com",
      "AccountId": "sAMAccountName"
   },

     "ProxyOptions": {
        "Base": "/SmartSpace"
    }
 }
```

LDAP authentication with no login for searching

```
{
    "LDAPAuth": {
      "Server": "ldap.company.com",
      "Port": "389",
      "User": "",
      "Password": "",
      "SearchStart": "ou=people,dc=company,dc=com",
      "AccountId": "uid",
      "ObjectClass": "account"
   },


     "ProxyOptions": {
        "Base": "/SmartSpace"
    }
 }
```

Note: The example given above works with the default OpenLDAP schema: other servers and schema might require different parameters.

How the LDAP validator behaves

The LDAP validator does the following:

1. If User option set, bind using this user/password.
2. If SearchStart is set, use the SearchStart, AccountId and ObjectClass to search for the DN of the entered user name.
3. Bind using the DN, if the search succeeded, or the entered user name. Use the entered password. If this bind succeeds, the user is authenticated successfully.

The validator reports what it is doing on the website trace (always on). for example:

```
[Tue Sep 24 14:23:03 2019, 127.0.0.1:42943] website: LDAPValidator: Is user valid
marcin
[Tue Sep 24 14:23:03 2019, 127.0.0.1:42943] website: LDAPValidator: searching (&
(objectclass=nsAccount)(uid=marcin)) at base ou=people,dc=ubisense,dc=aws
[Tue Sep 24 14:23:03 2019, 127.0.0.1:42943] website: LDAPValidator: binding as user
uid=marcin,ou=People,dc=ubisense,dc=aws
```

Enabling Secure LDAP (LDAPS)

To enable secure LDAP (LDAPS) for the connection you must use port 636. By using this port, SSL/TLS is enabled for the connection to the LDAP server (all other port numbers use TCP).

Note: If the LDAP server uses LDAPS (port 636), then the web site server must be able to verify the certificate that the LDAP server uses. For example, in active directory, the root certificate used to sign the LDAP server certificate should be imported into the web server host as a certificate authority (CA). How this is done depends on the specific operating system, but if it is not done, then LDAPS authentication will fail.

```
"LDAPAuth": {
      "Server": "adserver.company.com",
      "Port": "636",
      "User": "",
      "Password": "",
      "SearchStart": "dc=company,dc=com",
      "AccountId": "sAMAccountName"
   },

   "AuthOptions": {
      "UseCookiesOnWindows" : true,
      "ExpiryTimeSpan": "00:30",
      "SlidingExpiry":  true
   }
}
```

Using OpenID Connect to enable external authentication

OpenID Connect support allows an external authentication authority to handle login and provide user ID and roles to SmartSpace Web. The external authority takes care of ensuring that the user is allowed to access the application, and sends claims about the user back to SmartSpace Web. SmartSpace can then be configured to extract the user ID, and optionally the roles, from the claims provided by the authority.

To use OpenID Connect, SmartSpace must be accessed via https, so make sure you have configured SSL in your reverse proxy.

There are two places where OpenID Connect must be configured: at the authority, and in SmartSpace Web.

Configuration at the Authority

The authority provides an application client ID and secret, and is configured to accept the redirect URIs that will be used during authentication. The authority will also be configured to control which users/groups are allowed to access SmartSpace, and which properties should be sent to SmartSpace for it to use as the user ID and/or roles. Generally this involves creating a new "application" instance or registration at the authority, and then configuring the application to work with SmartSpace.

In order to ensure that only valid granted authorities are used, both the authority and SmartSpace share a configured "client id" and "client secret". The ID identifies the specific SmartSpace web site application at the authority, and the secret is used by SmartSpace to verify that the returned claims are valid. So the first step to setting up the authority is to create a new client ID in the authority, and to generate the client secret. The application must have the "code flow" or "code" response type enabled (also known as "Authorization code grant").

Authorities have different ways to do this, but for example, in Microsoft Azure Active Directory (AD), at the time of writing:

1. Go to App Registrations.
2. Click New registration.
3. Give the application a name and the set of users who can log in, and then click Register.
4. Click Certificates and secrets, and create a new secret.
5. Take a copy of the Value (it can only be viewed/copied on creation).

Set up the redirect URIs used during login and logout. These must match the redirect requests that the SmartSpace website generates when it attempts to login or logout a user by directing the browser to the authority.

- Sign in or callback URI: **https://<your server host as seen by the end user>/SmartSpace/signin-oidc**
- Sign out or front channel logout URI: **https://<your server host as seen by the end user>/SmartSpace/**

You can also define roles, or put a set of AD groups as claims. See the Azure AD documentation for how this is done.

Configuration in SmartSpace Web

There is an OpenID Connection configuration section in the web settings JSON files. Depending on your operating system this file is either **localsettings.json**, or **/etc/ubisense/web.json**.

```
"OpenIDC" : {
         "ClientId": "865b3f07-3f30-4881-895b-a831d11917ac",
         "ClientSecret": "<your secret shared with the authority goes here>",
         "MetadataAddress": "https://login.microsoftonline.com/fec9c4e9-d7de-4363-a4e3-
039b6f2606d2/v2.0/.well-known/openid-configuration",
         "IdClaim": "preferred_username",
         "RoleClaim": "http://schemas.microsoft.com/ws/2008/06/identity/claims/role",
         "LogoutURL": "",
         "DebugClaims": true
   },
```

The settings in this section are case sensitive, and have the following meanings:

Setting	Meaning
ClientId	The identity of this app at the OpenID Connect authority
ClientSecret	The shared secret used to generate and validate claims about the logged-in user
MetadataAddress	The OpenID Connect configuration meta-data document from your authority. For Azure AD, for example, go to Endpoints under your application registration, and copy "OpenID Connect metadata document".
IdClaim	The claim type to be used as the logged-in user in SmartSpace
RoleClaim	The claim type to be used as a role for the user in SmartSpace. Optional
LogoutURL	Some OpenID Connect authorities do not provide an "end_session_endpoint", so logout will be unsuccessful. In this case, provide a URL to log out the user from the client ID at the authority. Optional
DebugClaims	If set true, this will provide a page within SmartSpace Web that you can visit once authentication has redirected back to SmartSpace, and see the claims provided. The page can be found at SmartSpace/Auth/Claims. It is recommended to disable this configuration in production. The default is false. Optional

After the configuration has been saved, restart the web site by using Service Manager to restart the Ubisense/Visibility/Web site service.

You can test log in by visiting the SmartSpace Web site and clicking Login. You should be redirected to the authority login page. On successfully logging in, you will be asked to confirm sharing your profile with SmartSpace, and then be redirected back to SmartSpace Web. Note that if you are already logged in at the authority (for example if you are logged into Microsoft 365), there may be no need to provide any user/password – you may not even see the redirects as they can happen very quickly.

On success, the user ID should be displayed in the top left of the SmartSpace Web site. If this is not the case, enable DebugClaims, restart SmartSpace Web, and authenticate again. Then visit /SmartSpace/Auth/Claims to see what the authority returned, and pick a suitable IdClaim.

Using Roles from OpenID Connect

Claims returned by the authority that have a claim type configured in RoleClaim will be treated as user roles. These will be passed transparently through to SmartSpace. To use them within SmartSpace they should also be created as roles using the USERS / ROLES task in SmartSpace Config. You do not need to add members of the role within SmartSpace Config (though this is allowed if necessary). Users that have the role according to the OpenID Connect authority will automatically be treated as having the role within SmartSpace Web.

For example, if the authority passes a claim "SmartSpace Supervisor", you should create a "SmartSpace Supervisor" role in the USERS / ROLES task in SmartSpace Config. After you have created the role within SmartSpace, you can:

- assign this role as a member of other roles
- specify the searches/views/properties that this role can access directly
- assign the role to reports, HMIs, and in ObjectView API views

Again, you can use *DebugClaims* to see what the authority has returned to SmartSpace for the currently authenticated user. Remember to disable DebugClaims and restart the web site before going into production.

Header Authentication with Other External Authentication Systems

Upstream reverse proxies can provide some other external authentication system, and pass both user and roles from that external system to SmartSpace.

AuthOptions.RolesHeader can be used with AuthOptions.UserHeader to specify a header from which to read roles for passthrough to SmartSpace. Roles are comma separated in the value of the header provided. The config is optional: if not specified in the settings file then no roles will be passed through. The option is ignored if UserHeader is not set.

For example:

```
"AuthOptions": {
     "UserHeader": "AUTH_USER",
     "RolesHeader": "AUTH_ROLES",
      ...
 }
```

This would read the user (authenticated by the up-stream reverse proxy) from header AUTH_USER, and the roles from header AUTH_ROLES. If these are used, the web site should not be accessible except from the reverse proxy, which should filter out any user-supplied values for these headers.

As with roles in *OpenID Connect*, the roles that can be passed in the header should also be created in the USERS / ROLES task in SmartSpace Config so they can be assigned as members of other roles, or given searchs/views, etc.

restapi.json

This contains configuration specific to the REST API. In this example, we will set up a proxy base path matching the reverse proxy path we will configure in Apache2 for the API:

```
{
    "ProxyOptions": {
        "Base": "/SmartSpaceApi"
   }
}
```

Here we are allowing anonymous access to the API. For header authentication see the *advanced configuration*.

shared.json

shared.json is a configuration file that is loaded by both web sites, where options shared by the two can be set up. This is not used in our example configuration.

Deploy the Platform Services

The platform services should be deployed using the Service Manager client or the **ubisense_installer** command-line tool.

To deploy the services using Service Manager:

1. Run Service Manager 3 on a Windows client connected to the platform.
2. Select the INSTALL SERVICES taskClick **Install services...**, click **Browse** and navigate to the extracted SmartSpace release, then go to the **web\linux\packages** folder.
3. If you already have a .NET Core Runtime installed on the web server, unselect **Shared runtime** in the list of features to install.
4. Click **Install** and wait for the dialog to finish.

5. Click **Install** and the services are installed. You can click **Finish** when the "Service installation complete" message displays to return to the main Service Manager screen.
6. If you have multiple Linux controllers, deploy the website services onto the prepared Linux server:
    a. Select the MANAGE SERVICES task.
    b. Expand the **Controllers** entry under CELLS & CONTROLLERS, so you can see the web server controller.
    c. Under SERVICES, find the “Ubisense/Visibility/Web site” service. Drag this service and drop it onto the web server controller. It should deploy onto that controller.
    d. Repeat the above for the service “Ubisense/Application integration/RestAPI site”.
7. If you have multiple Linux controllers, deploy the website services onto the prepared Linux server:
    a. Expand the **Controllers** entry under HIERARCHY, so you can see the web server controller.
    b. Expand the **Services** entry under HIERARCHY, find the “Ubisense/Visibility/Web site” service. Drag this service and drop it onto the web server controller. It should deploy onto that controller.
    c. Repeat the above for the service “Ubisense/Application integration/RestAPI site”.
8. The Web site and Rest API site services should now be running on the Linux web server, but will not be visible from the wider network.

To deploy the services using the **ubisense_installer** command-line tool:

1. Go to the **web\linux\packages** directory of your SmartSpace distribution directory.
2. Run the following commands to install and deploy the Web site and REST API site:

```
ubisense_installer -ud SmartSpaceWeb.xml
```

```
ubisense_installer -ud SmartSpaceRestApi.xml
```

3. If you do not have a .NET Core Runtime installed on the web server, you must also run the following command:

```
ubisense_installer -ud SharedRuntime.xml
```

Configure Apache2 Reverse Proxy

Now install and configure Apache2 as a reverse proxy. The instructions below are targeted at SLES, and will need to be adapted for other Linux distributions.

Install Apache2

Use the package management software for your Linux distribution to install Apache2. For example:

```
sudo zypper in apache2
```

You will also need to ensure all required Apache modules are enabled using the following command:

```
sudo a2enmod <module name>
```

The required modules include:

```
mod_proxy
mod_proxy_http
mod_ssl
mod_headers
mod_rewrite
```

On some variants of Linux, drop the “mod_” prefix when enabling a module for Apache. You can see the list of enabled modules with the command:

```
sudo apache2ctl -M
```

Modules may be displayed with slightly different names in the output generated by this command, with a **_module** suffix instead of a **mod_** prefix.

Enable Outbound Connections

For SELinux servers such as RHEL 7, you may find that the Apache service cannot connect to another website. To correct this you need to enable outbound connections by running the following command:

```
/usr/sbin/setsebool -P httpd_can_network_connect 1
```

Install the SSL certificates

Place the SSL certificate and key in a suitable location:

```
/etc/apache2/ssl.crt/localhost.crt
/etc/apache2/ssl.key/localhost.key
```

Creating the service configuration file

Apache is configured by **.conf** files located in **/etc/apache2/vhosts.d/**.

Create a file **smartspace.conf**. The example below matches the proxy paths configured in the **web.json** and **restapi.json** example files above.

```
<VirtualHost *:*>
        RequestHeader set "X-Forwarded-Proto" expr=%{REQUEST_SCHEME}
</VirtualHost>

<VirtualHost *:80>
    # Rewrite http to https
    RewriteEngine On
    RewriteCond %{HTTPS} !=on
    RewriteRule ^/?(.*) https://%{SERVER_NAME}/$1 [R,L]
</VirtualHost>

<VirtualHost *:443>
        SSLProxyEngine on
        ProxyPreserveHost On

        # We proxy SmartSpaceApi to the REST api http port
        ProxyPass /SmartSpaceApi http://127.0.0.1:5002/SmartSpaceApi
        ProxyPassReverse /SmartSpaceApi http://127.0.0.1:5002/SmartSpaceApi

        # We proxy SmartSpace to the web site http port
        ProxyPass /SmartSpace http://127.0.0.1:5000/SmartSpace
        ProxyPassReverse /SmartSpace http://127.0.0.1:5000/SmartSpace

        # Add the forwarded protocol header
        RequestHeader set "X-Forwarded-Proto" expr=%{REQUEST_SCHEME}

       # Using localhost as server hostname
        ServerName mywebserverhost.domain.com
        ServerAlias mywebserverhost

        # Set logging_dir to a suitable logging directory
        ErrorLog /var/log/apache2/smartspace_error.log
        CustomLog /var/log/apache2/smartspace_custom.log common

        # Give away as little as possible on 404.
        ErrorDocument 404 "Not found"

        # Redirect top level to the SmartSpace web site
        RedirectMatch ^/$ /SmartSpace

        # Configure the https options to be suitably secure
        SSLEngine on
        SSLProtocol all -SSLv2
        SSLCipherSuite ALL:!ADH:!EXPORT:!SSLv2:!RC4+RSA:+HIGH:+MEDIUM:!LOW:!RC4
        SSLCertificateFile /etc/apache2/ssl.crt/localhost.crt
        SSLCertificateKeyFile /etc/apache2/ssl.key/localhost.key
</VirtualHost>
```

In this example, **SSLCertificateFile** should be the primary certificate file for the domain name. **SSLCertificateKeyFile** should be the key file generated when CSR is created. **SSLCertificateChainFile** should be the intermediate certificate file (if any) that was supplied by the certificate authority.

Enable SSL for Apache

Additionally SSL must be enabled for Apache by adding the **SSL** flag to **APACHE_SERVER_FLAGS** in **/etc/sysconfig/apache2**:

```
APACHE_SERVER_FLAGS="SSL"
```

You will need to restart Apache to reload any new or changed configuration files, using the following commands:

```
sudo systemctl restart apache2
sudo systemctl enable apache2
```

Check the status of Apache with the command:

```
sudo systemctl status apache2
```

Test the website

Visit the website in a browser. You should be redirected to the top level SmartSpace website page.

Advanced Configuration

Header Authentication (SiteMinder)

The websites can read the authenticated user from a header passed to them by a proxy server, such as SiteMinder. To configure this, in **shared.json**, **web.json** or **restapi.json**, set **AuthOptions/UserHeader** to be the name of the header from which the logged in user is to be extracted. If this option is configured, then the LDAPAuth options do not need to be set.

```
{
    "AuthOptions": {
       "UserHeader": "SITEMINDER_USER"
       "RequireAuth": true
   }
}
```

If UserHeader has been configured, it would be normal to also require authentication for all pages. This is what the RequireAuth configuration setting does. Since the UserHeader assumes that the header passed in the request has been checked, it is important that the user should not be able to bypass the proxy server and pass their own user header directly to the web sites.

Configuring the Authentication Timespan

The default authentication on Linux uses an expiry time of 30 minutes and a sliding timespan. This means that the authentication will expire 30 minutes after the user closes the website, but will continue to be refreshed while the user is still visiting the website.

You can disable this sliding expiry and set an absolute time after which login will need to be repeated. For example, to log out after three hours:

```
{
   "AuthOptions": {
      "ExpiryTimeSpan": "03:00",
      "SlidingExpiry":  false
   }
}
```

Disable Hardened Headers

By default the website injects headers in each response for penetration security. These disable cross-site/cross-frame scripting, prevent content type sniffing, etc. If necessary, these headers can be disabled, and the reverse proxy configured manually to add appropriate headers instead.

```
{
    "SecurityOptions": {
       "HardenHeaders": false
    }
}
```

Enable Cross Origin Scripting

CORS is supported for the SmartSpace REST API. For features using SmartSpace Web where CORS is not supported, there are workarounds such as making configuration changes so that the browser treats localhost requests and the remote site as if they come from the same domain, disabling hardened headers (see above), or using IIS or Apache.

By default, no CORS headers are sent, so browsers will refuse to execute the API web methods from a page served from a different web server.

The option AllowOrigins in the **appsettings.json** file which overrides settings in **localsettings.json** enables cross origin scripting:

```
{
...
   "SecurityOptions": {
   "HardenHeaders": true,
   "AllowOrigins": [ "http://example.com", "https://*.mydomain.com" ]
   }
...
}
```

If AllowOrigins is set, and matches the origin of a request, the API will respond with suitable headers, for example:

```
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: X-Requested-With
Access-Control-Allow-Methods: PUT
Access-Control-Allow-Origin: https://server.mydomain.com
Access-Control-Max-Age: 3600
```

The browser will now allow the request. Note that this allows the browser to cache the pre-flight OPTIONS responses for up to an hour, to reduce load on the API server. Thus changes to the allowed origins may not be picked up by browsers for an hour.

Adding a custom theme for the website

You can customize the look of the SmartSpace website to better reflect your corporate identity, for example by replacing the Ubisense logo with your own or using a custom stylesheet. To prevent such customization from being overwritten during website upgrades, you should store your local theme in an external folder on the host machine, for example **/home/platform/localtheme**. You specify the folder using the website configuration setting ContentOptions / LocalThemePath.

```
{
    "ContentOptions": {
       "LocalThemePath": "//home//platform//localtheme//"
    }
}
```

If the custom theme folder contains any **.min.css** files, they are loaded in place of the **wwwroot/bundle/base.css**. If the folder contains **custom.css**, it will be loaded in addition to other **css** files.

The following files can also be overridden by adding corresponding files in the custom theme folder:

- **images/logo.png**
- **images/background.png**
- **images/ubisense_large.png**
- **images/ubisense_small.png**
- **manifest.json**

Removing the Shared Runtime after Undeploying the Websites

If you deploy the website and/or the REST API without installing the .NET Runtime on the server, and subsequently move these website services to another controller, the shared runtime will be left in the dataset. You can simply delete this directory when it is no longer needed by any locally deployed services. The folder is:

```
<dataset path>/Ubisense/Platform/Shared\ runtime/x.x.x/
```

where `x.x.x` is the .NET Runtime version number.

Security Configuration for SmartSpace Web with Security Manager

If you are using a non-trivial security manager configuration to force authentication for services (as is the case for ACS installations) then you must run the **ubisense_cache_service_credentials** tool on the web server host. This is because the **credentials.dat** file created by the tool (in the latest version) allows IIS_IUSRS as a reader. Without this, the web site code cannot read the credentials, and therefore cannot connect to the platform services it needs.