This guide explains how to install and configure the Fusion Registry.
Contents

1 Introduction .......................................................................................................................... 5
  1.1 Scope of this document ................................................................................................. 5
  1.2 Fusion Registry Distribution ......................................................................................... 5
  1.3 Java (required) ................................................................................................................ 5
  1.4 Servlet Container (required) ...................................................................................... 5
  1.5 Database (optional) ....................................................................................................... 5
  1.6 Fusion Security (optional) ............................................................................................ 5
  1.7 LDAP Active Directory (optional) ................................................................................ 6
  1.8 Rabbit MQ (optional) ..................................................................................................... 6

2 Deployment ......................................................................................................................... 7
  2.1 Choice of Java Servlet Container .................................................................................. 7
  2.2 Deployment Using Tomcat ............................................................................................ 7
  2.3 Configuring Tomcat Memory ......................................................................................... 7
  2.4 Configuring Tomcat HTTPS ......................................................................................... 7

3 Installing Fusion Registry .................................................................................................. 8
  3.1 Install Wizard ................................................................................................................ 8
  3.2 Step 1 – Database Connection ...................................................................................... 8
  3.3 Step 2 – Server Settings ............................................................................................... 9
  3.4 Step 3 – Appearance .................................................................................................... 10
  3.5 Step 4 – Security Settings ............................................................................................ 11

4 Installation Completion ...................................................................................................... 12

5 Fusion Registry Properties File .......................................................................................... 13
  5.1 Introduction ................................................................................................................... 13
  5.2 Changing the location of the properties file .............................................................. 14
  5.3 Overriding Server URL Property ................................................................................ 14

6 Security Roles and Fusion Registry Access ....................................................................... 15
  6.1 Overview ....................................................................................................................... 15
  6.2 Enforce Login ............................................................................................................... 15
  6.3 Data Consumer Access ................................................................................................. 16
  6.4 Registry Security ........................................................................................................... 17

7 Fusion Security .................................................................................................................. 18

8 Configuring X509 Certificate Authentication ..................................................................... 19
  8.1 Authentication via Certificates .................................................................................... 19
## Version History

<table>
<thead>
<tr>
<th>Version #</th>
<th>Implemented By</th>
<th>Revision Date</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>20190321</td>
<td>Phil Lazarou</td>
<td>21 March 2019</td>
<td>Root User security section added</td>
</tr>
<tr>
<td>20190424</td>
<td>Matt Nelson</td>
<td>24 April 2019</td>
<td>Updates to Certificate Authentication Set Up</td>
</tr>
<tr>
<td>20190501</td>
<td>Phil Lazarou</td>
<td>1 May 2019</td>
<td>Updates to “Custom Database Connection”</td>
</tr>
<tr>
<td>20190813</td>
<td>Phil Lazarou</td>
<td>12 August 2019</td>
<td>Updated Client Debug settings</td>
</tr>
<tr>
<td>20190822</td>
<td>Phil Lazarou</td>
<td>22 August 2019</td>
<td>Updated Data Validation settings</td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Scope of this document
The purpose of this document is to provide an overview of how to install Fusion Registry in a Java Servlet Container. This document explains the Fusion Registry install process, how to configure the Fusion Registry and specific advice on configuring aspects of the Java Servlet Container, Apache Tomcat.

It is recommended that readers have familiarity with the appropriate required dependencies which are listed in the following sections.

1.2 Fusion Registry Distribution
Fusion Registry is distributed as a single Web Application Archive (war) file: FusionRegistry.war it should be deployed to a servlet container such as Apache Tomcat (a free product).

1.3 Java (required)
A Java Runtime Environment (JRE) version 1.8 or higher.

1.4 Servlet Container (required)
Fusion Registry is deployed to a Servlet Container. Apache Tomcat is a popular, open source servlet container, download links and installation instructions can be found at the following URL. http://tomcat.apache.org/

It is recommended to use the latest version of Apache Tomcat as it will include the latest security patches.

Apache Tomcat 8 or above is supported.

The rest of this document will only give servlet container information regarding Apache Tomcat.

1.5 Database (optional)
Fusion Registry makes use of an Object Relational Mapping (ORM) library called Hibernate. This allows Fusion Registry to communicate with any SQL-92 compliant database. This distribution has only been set up to connect to MySQL, Oracle, and SQL Server databases, the minimum tested version for these databases are: MySQL 5.6, Oracle 10g and SQL Server 2010.

If your database is not one of these types, please contact Metadata Technology, as we may be able to add your database to the list of supported database management systems.

1.6 Fusion Security (optional)
By default, Fusion Registry provides the ability to secure the Registry only allowing a single trusted user to perform changes. If you wish to have a number of user accounts with different access credentials, then Fusion Security is required to create and administer users. Please refer to the Fusion Security Setup and User guides as well as the section on Security in this document (see section 3.5).
1.7 LDAP Active Directory (optional)
As an alternative Security mechanism, it is possible to use an LDAP Active Directory to manage user authentication with Fusion Registry. Please refer to the section on Security in this document (see section 0).

1.8 Rabbit MQ (optional)
If you wish to scale a number of Fusion Registry instances, you may choose to use RabbitMQ as a communications mechanism to allow the different Registry instances to notify the other instances of structure and data changes. An alternative option is to use the database polling strategy which is discussed in this guide.
2 Deployment

2.1 Choice of Java Servlet Container
Fusion Registry must be run within a Java Servlet Container. Metadata Technology recommends using Apache Tomcat as the Java Servlet Container, as this has been used during the testing lifecycle of Fusion Registry. The Fusion Registry has only been tested in Apache Tomcat and therefore we cannot guarantee that the Fusion Registry will work with other Java Servlet Containers.

2.2 Deployment Using Tomcat
Fusion Registry consists of a single .war file called FusionRegistry.war. This file needs to be copied into the directory: `<TOMCAT_HOME>/webapps` then the Tomcat server should be started. As the Tomcat application server starts, the contents of the Fusion Registry war file will be unpacked into the directory:

`<TOMCAT_HOME>/webapps/FusionRegistry`

Please check the Tomcat log files to ensure that Fusion Registry has deployed correctly. Once it has then you may navigate to the URL:

`http://[server]:[port]/FusionRegistry/`

The values for server and port must be replaced with the IP address and port number that the web application server is running on. For example, if the web browser is running on the same machine as the web application server and the Apache Tomcat has not had its default port settings modified, then the following address can be used:

`http://localhost:8080/FusionRegistry/`

2.3 Configuring Tomcat Memory
It is important to override the default Tomcat server memory settings as the default Tomcat settings will not be adequate to run the Fusion Registry. Unless you are running Tomcat as a Windows service (see section 12) overriding the memory settings can be achieved by placing a `setenv.bat` (Windows) or `setenv.sh` (Unix) file into the Apache Tomcat bin folder. The recommended minimum settings are:

- 2Gb Heap Memory

The Fusion Registry distribution contains a `setenv.bat` (Windows) and `setenv.sh` (Unix) with the recommended minimum settings configured. These files can be copied (and optionally modified) to the Tomcat/bin folder before starting the Tomcat instance.

2.4 Configuring Tomcat HTTPS
An HTTPS connection provides a secure connection to the Fusion Registry server, by using the Secure Sockets Layer (SSL) protocol. It is strongly recommended to enforce a HTTPS connection to ensure username and password details are encrypted between client and server. HTTPS connections can terminate at a load balancer, in which case Tomcat can remain on a standard HTTP connection.

Please refer to the Apache Tomcat guide if configuring HTTPS.
3 Installing Fusion Registry

3.1 Install Wizard
Once the Apache Tomcat instance has been configured and started, the Fusion Registry will be accessible via a web browser. On first use, the Fusion Registry will show the Install Wizard, each step of the wizard must be completed in order to configure the Fusion Registry. Once the wizard is complete, the Fusion Registry home page will load, and the install wizard will no longer be accessible. Future modifications to the Fusion Registry configuration can be performed by logging in, and visiting the Settings menu. All the configuration options available in the Install Wizard are accessible via the Settings pages in the Fusion Registry.

3.2 Step 1 – Database Connection

Figure 1 showing step 1 of the install process: Database Connection

The first step of the Install Wizard is to configure a database connection. For relational databases (if MySQL, SQL Server, Oracle) the Fusion Registry will automatically create the required database tables on connection if they do not already exist.

The in-memory database is not a persistent storage mechanism. Registry settings information is stored in a properties file (discussed in section 5) and as such these settings will be loaded back into memory on tomcat start-up. Any SDMX Structural content loaded into the Fusion Registry is stored in memory and not persisted to any external store, and as such this information will not be preserved on tomcat shut down.
An in-memory database can be used in conjunction with the Structure File Sweeper (Install Wizard Step 4) which will automatically to load structure files from the file system on tomcat start up. This provides a dissemination solution which does not require a persistent database.

In the database type drop-down there is a choice for “Custom”. This allows the specification of a database connection via a custom string (see section 11.2 for further details on this feature).

3.3 Step 2 – Server Settings

The second step allows the user to define server and general properties, these include:

1. **Server URL** – This is the fully-qualified URL which the Fusion Registry will be access by, and is used by the Fusion Registry when it needs to communicate the URL to a user, for example on the web service builder.

   **NOTE:** It is very important that the Server URL is set to reflect the public URL that the application will be accessed by, as this will be used as the base URL for any redirects (such as login, logout).

2. **Parent Registry** – The Parent Registry allows the specification of a parent Fusion Registry instance. The structures in the parent Fusion Registry will be automatically obtained and stored in this local instance on modification, this is achieved by the local instance polling the RSS feed of the parent registry. Any structures obtained from the parent registry will be marked as external and therefore will not modifiable in the local registry instance. This value must be the fully qualified URL of the parent Fusion Registry instance, for example: https://registry.sdmx.org/FusionRegistry

3. **Registry Name** – This is the name which is presented to the user in the header of the User Interface. It is for display purposes only. The default value is “Fusion Registry”.

Figure 2 showing step 2 of the install process: Server Settings
4. **Sender Id** – This Id will be sent in the ‘SenderId Header’ field of any SDMX messages. The default value is Unknown.

5. **Support Email** – If you wish to publicise an e-mail address for users of your Fusion Registry to contact you by, then provide a value in this field. This email address will be present on the footer of the Fusion Registry User Interface. There is no default value.

6. **Support URL** – If you wish to publicise a URL for users of your Fusion Registry to go to as a point of contact then provide a value in this field. This support URL will be present on the footer of the Fusion Registry User Interface. There is no default value.

7. **Default Agency** – When authorized users are creating new structures, the value in this Agency drop-down will be initially set to the value specified here. This drop-down will show all agencies stored in the Fusion Registry, on first install the only Agency contained in this list will be SDMX.

On clicking the Next button, the Fusion Registry will validate the Server URL and the Parent Registry (if set) to ensure that both are legal values.

### 3.4 Step 3 – Appearance

![Figure 3 showing step 3 of the install process: Appearance](image)

This step allows aspects of the Registry appearance to be set.

The Registry Logo is used at the top-left of most of the User Interface screens and also on the login page. A custom image may be specified, which must be no larger than 200 by 200 pixels. The following file types are supported: tif, jpg, cur, tiff, ico, bmp, gif, png, jpeg, wbmp and rle.

A favicon may be applied which must be an .ico file. Depending on the browser the user is using to access the Registry, this may be displayed in the browser’s address bar, bookmarks list and possibly the browser’s history.

The colour scheme affects a number of elements across the Registry UI such as the title bar and menu items.
3.5 Step 4 – Security Settings

The Fusion Registry supports a number of security mechanisms (see section ???). On the install pages, the “local root” user credentials need to be specified. This is a single local account that can be used to access the Registry at the highest privilege level of “root”. After the install process, only the root user can modify the root user settings, so it is important that the credentials for this account are kept safe.

This step has settings for the root username, password and the maximum number of attempts that can be made to login as root before the account is locked. If this value is set to a zero or negative number, this states there the account will never become locked from repeated failures to login.

If the root user becomes locked, see section 15 for instruction on how to unlock the root user.
4 Installation Completion

On the final step, the Finish button completes the Fusion Registry configuration. The browser will redirect the user to the Fusion Registry home page. Fusion Registry configuration settings can be performed by logging in as a user with Root or Admin privileges, and using the Settings menu located on the left hand menu bar.

Note: The settings menu is not visible on the page for non-Admin users

Figure 5 showing the front page of an empty Registry
5 Fusion Registry Properties File

5.1 Introduction

Whilst all of the installation is performed via the Web User Interface, database connection details are stored in a local properties file. If running Fusion Registry with an in-memory database, other settings are also stored in the same properties file.

Unlike previous versions of the Fusion Registry it is recommended that you do not modify the values in this file, as all settings are configured via the web User Interface.

The properties file is only read at start-up and changing any of the values in the Fusion Registry properties file whilst Fusion Registry is running will have no effect.

The properties file is called:

`fusion_registry.properties`

By default, the Fusion Registry has a copy of this file located in the directory:

`<Tomcat HOME>\webapps\<Web AppName>\WEB-INF\classes`

If you make any changes using the maintenance tool, Fusion Registry will attempt to save a new properties file to the directory:

`<user home>\MetadataTechnology\FusionRegistry`

Therefore, on a Windows 7 Operating System this will typically be:

`C:\users\<your user name>\MetadataTechnology\FusionRegistry`

Whereas on a Unix Operating System, it is more likely to be located at:

`/users/\<your user name>/MetadataTechnology/FusionRegistry`

If you are running Tomcat as a Windows service, then the Home directory is likely to be:

`C:\Windows\System32\config\systemprofile`

And the file is likely to be located at:

`C:\Windows\System32\config\systemprofile\MetadataTechnology\FusionRegistry`

On Fusion Registry start-up, the Fusion Registry will load the properties file from the WEB-INF\classes directory first, and then search for a properties file in your home directory. If it locates a properties file in your home directory, the values in this file will be read and will override values from the default properties file.

If you are unsure about which of the files Fusion Registry is using to obtain system information, please look at the start-up log in your web application server. There will be entries like the following:

```
INFO localhost-startStop-1
org.springframework.beans.factory.config.PropertyPlaceholderConfigurer -
Loading properties file from class path resource
[fusion_registry.properties]
```
5.2 Changing the location of the properties file

The location of your properties file can be changed. This is useful if you either do not want the Registry to store information in the computer’s home directory or if you wish to run multiple Registries on one server.

To specify a new location you need to set a Java System variable called “RegistryProperties” to the URI value of the location where you wish the properties file to be. If you are running Apache Tomcat as a service please refer to Section 9 of this document. Otherwise the easiest way to achieve this is to create a new file called setenv.bat (or setenv.sh on Unix environments) and place it in the tomcat’s bin directory. The contents of this file should state the full location of the properties file which must be in the URI format. To illustrate this:

```
SET JAVA_OPTS=-DRegistryProperties=file:///c:/dir/AFile.txt
```
(For Windows systems)

```
export JAVA_OPTS=-DRegistryProperties=file:///dir/AFile.txt
```
(For Unix systems)

It is important to note that Fusion Registry will NOT start if this value is incorrect or if this file cannot be created.

To check that this value is being used by the system, check the Registry log during Registry startup and look for a line similar to the following:

```
Property RegistryProperties has been specified as file:///c:/dir/AFile.txt
```

5.3 Overriding Server URL Property

When deploying multiple load-balanced Fusion Registry servers, there may be a requirement for each server to maintain its own value for the “Server URL” property, instead of each accessing a single value from the shared database.

In order to override the Server URL property, edit the Fusion Registry properties file, local to each instance, and modify the file to include a value for ‘registry.url’. For example:

```
registry.url=https://localhost:8443/FusionRegistry
```

This property will now be used instead of the database property. In addition, if using Fusion Security, the Fusion Security server domain will be ignored when authenticating a user.
6 Security Roles and Fusion Registry Access

6.1 Overview
The Fusion Registry has 5 distinct user roles, these are:

1. Root
2. Admin
3. Agency
4. Data Provider
5. Data Consumer

Users with Root and Admin roles are able to access all parts of the Fusion Registry application. Only the Root user is able to change the Root user credentials.

A user has a role of Agency if their user account is linked to one or more Fusion Registry Agencies. An Agency user has the following privileges:

- Create, modify, and delete SDMX Structures for their own Agency.
- View log files, and audit information for transactions related to structures maintained by their Agency.
- View User Activity Audit information.
- Link Provision Agreements to data stores, if the Agency owns the Provision Agreement.
- Register data against any Provision Agreement which is owned by their Agency.
- Author and upload Reference Metadata.

A user has a role of Data Provider if their user account is linked to one or more Fusion Registry Data Providers. A Data Provider user has the following privileges:

- Register data using the User Interface or SDMX Web Service.
- Load data to the publish web service, and poll the publish status.

**Note:** A Data Provider can only supply data against Provision Agreements which are assigned to the Data Provider.

A user has a role of Data Consumer if their user account is linked to one or more Fusion Registry Data Consumers. By default a Data Consumer has no special privileges except for in the case where the Fusion Registry has been configured to only accept authenticated access. This is explained further in the next sections.

6.2 Enforce Login
The Fusion Registry content can either be public or private, this is controlled using the enforce login toggle. A private registry restricts access to all user interfaces and web services, only allowing access to authenticated users. A public registry allows anonymous access to the user interface and web services. There are some important points to note:

- This requires: Fusion Security version 2.0.7 or higher.
- Fusion XL will not be able to communicate with a Registry in this state.
- Fusion Security will operate correctly whilst “Enforce Login” is set however the modification or assignment of users to new Organisations cannot be performed. It is recommended to create your users and permissions before turning “Enforce Login” on.
To enable “Enforce Login”, navigate to Admin -> Security -> Server Security. On the Server Security page there is a control labelled “Enforce Access”. Setting this control “on” will prevent unauthorised users from Registry access.

![Server Security](image)

Figure 6 showing the Server Security page with the Enforce Login control

Once the Registry is in this mode, all unauthorised requests to the Registry will result in the login page being returned.

NOTE: If using Enforce Login in conjunction with a Fusion Security server, the Fusion Security server must be configured to use Certificate Authentication against the Fusion Registry. This is documented in the Fusion Security Setup Guide.

6.3 Data Consumer Access

If a user logs into the Registry as a Data Consumer when the Registry is not enforcing login, the user will experience no differences as to an unauthorised user.

For a Registry with Enforce Login enabled, a Data Consumer has the ability to query the public SDMX web service. A Data Consumer may not access the Registry via the Registry User Interface and attempting to log in to the Registry will inform the user of this.

A Data Consumer can see all of the following structures without any restriction:

- Agencies
- Concepts Schemes
- Category Schemes,
- Codelists
- Data Structures
- Metadata Structures
- Hierarchical Codelists

The following structures can also be viewed but there are restrictions:

- Dataflows - only those assigned to the Data Consumer.
- Categorisations - only those which attach to a structure the Data Consumer can see.
- Content Constraints. The Data Consumer can only see those attached to a DSD, or attached to a Dataflow the Data Consumer has been given permission to see.
Permissions can be applied to a Data Consumer from the Security Settings page “Data Consumer Dataset Access”. This page allows for individual Data Consumers to be assigned permission to view one or more Dataflows.

![Data Consumer Dataset Access](image)

Figure 7 showing the settings page allowing the assignment of Data Consumers to Dataflows

Activating the “Add Permissions” control will display a modal showing all of the Dataflows in the Registry. Any number of dataflows may be assigned to a Data Consumer. Once assigned these are shown on the right-hand side of the “Data Consumer Dataset Access” page.

### 6.4 Registry Security

The Fusion Registry provides the following security mechanisms for authentication:

- **Local Security** – Local security authenticates the user within the Fusion Registry application. Only one root user account exists and this user has unrestricted access to the whole application. The local root credentials are encrypted and stored in the Fusion Registry database. Local Security is always active but the account can be locked by repeated submissions of incorrect passwords.

One of the following other mechanisms may also be specified:

- **Fusion Security** – Fusion Security is used to manage user accounts and authenticate users. The Fusion Registry connects to the Fusion Security web application for user authentication. Fusion Security connects provides the ability to set up and maintain user accounts, and link the user to any number of Fusion Registry organisations.

- **LDAP Active Directory** – Active Directory (AD) is used to connect to a Microsoft Active directory service or equivalent. This allows the registry to both authenticate and authorise users when they log into the registry.
7 Fusion Security

Fusion Security can be specified as an authentication service for Fusion Registry, allowing the management of users and permissions to be controlled from the Fusion Security application.

Navigate to the Authentication Service page via Admin -> Security -> Authentication Service. This page has a drop-down allowing the choice of Security Service to be “none” (no additional authentication service beyond the local root user), “Fusion Security” and “LDAP Active Directory”.

Choosing “Fusion Security” as the additional security service requires that you specify the URL of the Fusion Security Service as well as the root credentials to that Security service (note that these are not the root credentials of the local root user of the Fusion Registry). Once “Apply Settings” is clicked, these credentials are checked and if correct then Fusion Security will be configured to be providing security to Fusion Registry.

With Fusion Security as the authentication service, it is possible to configure Fusion Registry to support certificate authentication. Certificate authentication allows external users to authenticate with the Fusion Registry Tomcat instance using a certificate. The certificate must contain a CName which must match a username of a Fusion Security user account. When the Fusion Registry is presented with an authenticated certificate, it will load the user account from Fusion Security. For more information see section 8 (Configuring X509 Certificate Authentication).
8 Configuring X509 Certificate Authentication

8.1 Authentication via Certificates
The Fusion Registry supports authentication via the use of an X509 certificate (in the PKCS12 format).

The Web Application Container WAC (e.g. Tomcat) must be configured to trust the certificate as the WAC is responsible for authenticating the Certificate. **Certificate Authentication will only work if HTTPS is enabled.**

When the request is passed onto the Fusion Registry, the Fusion Registry will attempt to map the Certificate to a Fusion Registry user.

To support a new Certificate, log into the Fusion Registry as an Admin user, and navigate to Security -> Certificate Manager. New Certificates are added by clicking on the cogs icon and selecting **Create New Certificate Definition.**

8.2 Certificate Authentication Setup Procedure

8.2.1 Generating Self-Signed Certificates

8.2.1.1 Overview
The following gives an example of how to create self-signed certificates these certificates will be used for applications to authenticate with the tomcat server. Whilst trusted authorities such as Verisign are typically used to generate Certificates for website to run under https, a self-signed Certificate can be generated locally using applications such as Java Keytool, which is freely available as part of the Java JDK distribution (under Java/bin). Self-signed certificates will not be trusted by web browsers, as they have not been created by a trusted authority, however for internal use, to distribute to users of an application, they are perfectly adequate (if you trust yourself).
The certificates created in this example are not the same certificates that are used to enable Tomcat to run under https. It is assumed that the Certificate to enable https will be generated by a trusted authority, and that Tomcat has already been configured to run under https.

This process makes use of **Java Keytool** this is a command line application for certificate creation and management.

### 8.2.1.2 Generate a certificate for the client of Fusion Registry

This process creates a self-signed certificate to be given to the user of the Fusion Registry. This certificate contains the user’s username in the Certificates Common Name (CN). This process consists of two steps as follows:

1. Generate a p12 file which contains the private key and certificate. This generation step prompts a number of questions on the command line. The first question is ‘what is your first name’ this value here should be the username of the user that the certificate will be issued to.

   ```
   keytool -genkeypair -alias regClient -keyalg RSA -validity 365 -keystore regClient.p12 -storetype PKCS12
   
   What is your first and last name?
   [Unknown]: melson
   What is the name of your organizational unit?
   [Unknown]: IT
   What is the name of your organization?
   [Unknown]: MT Data Provider
   What is the name of your City or Locality?
   [Unknown]: Godalming
   What is the name of your State or Province?
   [Unknown]: Surrey
   What is the two-letter country code for this unit?
   [Unknown]: UK
   Is CN=melson, OU=IT, O=MT Data Provider, L=Godalming, ST=Surrey, C=UK correct? [no/yes]: yes
   
   2. Export the certificate from the p12 file to a .cer file
   
   ```
   ```
   keytool -exportcert -alias regClient -keystore regClient.p12 -storetype PKCS12 -file regClient.cer
   ```

### 8.2.2 Configuring Tomcat for Certificates

Tomcat can be configured to use certificate authentication. It is important to note that Tomcat can be configured to enforce all clients use a certificate, or it can be configured to allow certificates to be used, but not enforce that they are. The latter use-case enables clients to use the Registry anonymously or sign in using Basic Authentication, whilst also supporting certificates. The former use case will reject any access from a user who does not have a certificate.

To support certificate authentication, modify the server.xml file located in Tomcat/conf. The Connector is used to set the keystore, truststore, and if the client must show a certificate (clientAuth=true) or if a certificate is optional (clientAuth=want).

To configure Tomcat, perform the following steps:

1. Create a trust store for the Registry, containing the Registry client certificate file:

   ```
   keytool -importcert -alias regClient -storetype PKCS12 -keystore regtomcat.truststore -file regClient.cer
   ```
2. Place both the registry p12 file and the Registry truststore inside the tomcat/conf folder for Fusion Registry.

3. Modify the [tomcat]/conf/server.xml

   ```xml
   <Connector port="8443"
      maxThreads="150"
      SSLEnabled="true"
      scheme="https"
      secure="true"
      clientAuth="want"
      sslProtocol="TLS"
      SSLVerifyClient="require"
      SSLVerifyDepth="2"
      keystoreType="PKCS12"
      keystoreFile="/conf/httpsregistry.p12"
      keystorePass="password"
      truststoreType="PKCS12"
      truststoreFile="/conf/truststore.truststore"
      truststorePass="password" />
   ```

8.2.3 Link Certificate to Organisations

Once a Certificate has been added to the tomcat truststore, it will be used to authenticate the request. Tomcat acts as the authentication service, and therefore no other authentication is required (LDAP and Fusion Security play no role in authentication in this instance). The Fusion Registry receives the pre-authenticated request, and needs to determine which Organisation(s) the certificate is linked to. The link between the certificate an Organisation(s) is based on the certificate’s CName. The management of this connection is performed in the Fusion Registries’ Certificate Manager, which can be accessed under Admin -> Security. The Certificate Manager is used to define a Certificate by its CName and connect this to an email address, and any number of Organisations.

8.2.4 Testing

To test Certificate Authentication:

1. Launch Fusion Registry web application.
2. Open a web browser, and in the settings navigate to ‘Manage Certificates’.
3. Load the registry client certificate into the web browser (regClient.p12 for example)
4. Navigate to the Fusion Registry web page using https. The web browser may ask which certificate you wish to use: select 'regClient'. The Fusion Registry application should show the user as being logged in, and it will not be possible to log out.

Alternatively a test can be performed in code such as Java, see the following code snippet which requests information from a secure resource. The expected response will simply be an empty JSON array. If the user associated with the certificate does not have admin permissions or if the certificate is not being checked correctly, then the response will be an error or HTML text representing a redirect to the HTML home page.
8.2.5 Troubleshooting

If there are issues getting certificate authentication working try the following:

1. Remove the following from `tomcat/conf/server.xml`

```xml
    <Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />
```

2. Ensure the Java version used to create the self-signed certificate is the same as the Java version used to run Tomcat. Tomcat may report Java version on startup, if not it will likely be set in your `JAVA_HOME` or `JRE_HOME` environment variable
9 Active Directory

9.1 Overview
Active Directory may be used as the Authentication and Authorisation provider for Fusion Registry. This requires an Active Directory server running on a network that Fusion Registry can communicate to. Fusion Registry can communicate with Active Directory either using a Role Template or else a Role Mapping can be defined.

This document does not explain how to deploy Active Directory or create the required users or groups within it.

9.2 Connection Settings
To establish the connection between Fusion Registry and the Active Directory server, the following information needs to be supplied:

1. Protocol – either LDAP or LDAPS.
2. URL of Active Directory – either the IP address or server name of the Active Directory server.
3. Active directory domain – the domain that Active Directory resides on. This can also be the NetBIOS.
4. Base group DN (optional) - this is the root Distinguished Name (DN) for the Registry to search for groups under. If this is left blank then the search will be performed on the highest level of the Active Directory forest.

![Security Method: LDAP Active Directory](image)

Figure 9 showing the LDAP settings

**Important Note:** if using the LDAPS protocol then the Registry will be unable to validate this connection fully. This means that if the certificate is not valid for the connection, but the URL is correct, the connection is still considered valid. When you attempt to login to the Registry, it will not work, since the certificate is invalid. It is vital that you ensure you have a valid certificate when using LDAPS.

9.3 Role Template
If no Role Mappings are defined then the Registry will communicate to Active Directory using the Role Template. In this scenario, users are given permission by assigning them to groups and the names of the groups follows a specific pattern to provide authorisation.
To set up roles permitting Agency level authorisation, groups for the appropriate agency must be named “ACY_” and then be followed by the agency name. E.g. A group which permits users assigned to that group to modify SDMX structures, must be named “ACY_SDMX”

To assign a Data Provider role to a user, then the group must be named “DP_” followed by the AgencyId of the Data Provider, an underscore, and then the Data Provider ID. For example to create a group of Data Providers for the Data Provider “DP1” owned by the Agency “ACY”, the group must be named: “DP_ACY_DP1”

To assign a Data Consumer role to a user, then the group must be named “DP_” followed by the AgencyId of the Data Provider, an underscore, and then the Data Provider ID. For example to create a group of Data Providers for the Data Provider “DP1” owned by the Agency “ACY”, the group must be named: “DP_ACY_DP1”.

To create a group that permits Administrator access to the Registry, the group needs to be named “Administrators” (this is not case sensitive).

9.4 Role mapping

If you do not want to map roles to users using the default naming template, you can define your own mapping values. Custom Role Mapping allows an association between a group in Active Directory to permissions in the Registry. For example, you can specify that any users assigned to the group named “ABC” have permission to be Agencies for “ACY1”.

To setup a mapping, click the “Define Role Mappings” button and you will be presented with a dialog titled: “Define Role Mappings”. This modal has controls allowing the creation of custom mappings.

![Figure 10 showing the LDAP role mappings](image)

Clicking “Add Role Mapping” presents the user with an input field, and a dropdown:

![Figure 11 showing the input field with dropdown](image)

The key is what will be looked up when parsing your roles on AD, the key needs to match the name of the group in Active Directory. The second part is the type of role this mapping value equates to. If for example you want to map “ABC” with the Agency “ACY1”, then you would type “ABC” as the key, and select agency “ACY1” as the value.
When you have added all your required mapping values, clicking “OK” will inform the Registry to use your mapping values and not use the template. Any groups that your Active Directory is connected to and not defined in the mapping will be ignored.

If you wish to define a custom name for an Administrator mapping, click the “Add Administrator Role Mapping”. This will only allow you to specify a key since an admin does not associate with organisations.
10 Single Sign-On (SQL Server only)

Fusion Registry 9 supports Single Sign-On (SSO) when connecting to a SQL Server database. This can be activated via the Registry UI. In order for this feature to work a DLL is also required.

The DLL can be obtained from Microsoft. You will need to download the "Microsoft SQL Server JDBC Drivers" package which contains a number of drivers named “sqljdbc_auth.dll” but for different systems (e.g. x86, 64 bit, etc.). You need to locate the appropriate DLL for your system.

Figure 12 showing the database settings page for a SQL Server database type

This DLL needs to be supplied to the Java Runtime running your Web Application Server. There are a number of ways in which this can be achieved. Two of the simplest methods are listed below:

1. Copy the DLL file to the Java Runtime “bin” directory that is running your Web Application Server. It is important to place the DLL in the correct directory (for example: C:\Java\jdk1.8.0_92\jre\bin). Note: that modifying a Java Runtime in this manner means that all applications that use this Java Runtime will be affected.

2. Pass the DLL location to the Web Application Server on server startup. If you are running Apache Tomcat as a service, please refer to section 12 of this document. Otherwise this can be achieved by modifying the “setenv.bat” file located in the Tomcat bin directory. Locate the directory with the DLL you wish to add (e.g. c:\temp) then add the following line to setenv.bat and the Java library path will be modified allowing Tomcat to access the DLL file:

   set CATALINA_OPTS=%CATALINA_OPTS% -Djava.library.path=C:\temp\SSO_DLL

Once your Web Application Server has started and can access the correct DLL, SSO can be enabled via the database settings. When attempting to enable SSO, if you receive an error like the following, then the DLL could not be located or is the wrong version for your system:

   java.lang.UnsatisfiedLinkError: no sqljdbc_auth in java.library.path

In this scenario, please double-check the actions you performed and ensure that you are using the correct driver.
11 Additional Settings

11.1 Registry Settings

11.1.1 Accessing Registry Settings

Settings modification can be performed by any Admin user. In the left-hand menu of the Registry, click Admin -> Settings to get to the sub menu of Registry settings.

![Registry Settings Interface](image)

The sub menu items on the left of the screen allow the user to specify settings for specific sections. Some of these sections have been discussed previously in this document, in the install section:

- Server – see section 3.3
- Appearance – see section 3.4
- Database – see section 3.2
- Security – see section 3.5

The remaining settings pages are discussed in the following sections.
11.1.2 Email Server

The Fusion Registry does not require an email server. If an email server is configured then the Fusion Registry will use it for the following:

1. To email reset password details on request
2. To support users subscribing to changes in the Fusion Registry

Before the email settings can be applied, you must click the ‘Test Email Settings’ button and send a test email. This is to ensure that the Registry is able to determine if it can communicate with the email server with the specified credentials.

11.1.3 Sweeper

11.1.3.1 Sweeping Settings

The Fusion Registry can be set up to sweep folders on the local file system. There are two types of sweeper available, one for Structures and one for Data.

11.1.3.2 Structure Sweeper

If the Structure Sweeper is enabled, then the Fusion Registry will sweep the provided directory periodically to check for the presence of new files. On detecting new files, the Fusion Registry will
process the file as a Structure import. The file will not be modified in any way on completion. If the file fails to process, then an error file will be written to the same directory.

### 11.1.3.3 Data Sweeper

If the Data Sweeper is enabled, then the Fusion Registry will create subdirectories for each Data Provider in the Fusion Registry which has been assigned to provide data for each Dataflow. These subdirectories will be kept up to date as Fusion Registry content changes.

The directory structure created conforms to the following tree structure:

1. Level 1 Folders – Agency Id
2. Level 2 Folders – Dataflow Id
3. Level 3 Folders – Data Provider Id

An example of this structure is shown below:

![Image of directory structure]

In the above example, Agency1 have defined a Balance of Payments (BOP) Dataflow, and linked the following Data Providers: United Kingdom (UK); France (FR); Germany (DE).

The Data Sweeper will process any files put into these sub-directories as a data import for the given Data Provider/Dataflow combination. In the above example the file ‘UK-BOP-2012.xml’ is processed for the United Kingdom (UK) against the Balance of Payments (BOP) Dataflow belonging to Agency1.

Once a file has been processed and the data loaded into the Fusion Registry, the file is renamed to have the prefix ‘.processed’. So ‘UK-BOP2012.xml’ will be renamed to ‘UK-BOP2012.xml.processed’.

If processing the file resulted in error, the file will be renamed to include the prefix ‘.errored’ and a corresponding file is created which contains the error text. This has the same name as the failed data file suffixed with “.error.txt”. So ‘UK-BOP2012.xml’ will be renamed to ‘UK-BOP2012.xml.errored’ and the error text will be in file ‘UK-BOP2012.xml.error.txt’.

### 11.1.4 Web Services

The web services settings allow for the Fusion Registry to make the web services accessible whilst maintaining a private User Interface.

The image below shows a public server, with public URLs which are then mapped by an Apache server which is acting as an HTTP Reverse Proxy on the Fusion Registry server. The Fusion Registry is not directly accessible via the public, Apache is used to direct traffic only on the URLs it has defined which are then mapped to Fusion Registry URLs.
The settings controls on the Web Services page allow for this type of mapping to be defined. This is shown in the image below:

The Fusion Registry uses this information when generating a public SWAGGER and WADL. It is also used when the Fusion Registry places links in its outputs (for example if querying using the SDMX REST structure query with detail=stubs then the output will use the public URL information when generating the URL of where the full structure definition can be found). This information is also used by Fusion JS and Fusion XL so that they know which URLs to use when making server requests.
NOTE: The Apache Server (or other proxy) will still need to be configured correctly. Please refer to the vendor specific instructions on how to support this use case.

11.1.5 Load Balance
It is possible to run multiple load balanced instances of the Fusion Registry, however it is important to note that the Fusion Registry maintains a number of internal caches to improve performance. When load balancing Fusion Registry instances each instance needs to be made aware when changes take place (such as a structural changes or data being loaded into the Registry).

The Fusion Registry supports 2 mechanisms for keeping its internal cache in sync:

1. Database Polling – if configured, the Fusion Registry will poll the database every ‘x’ seconds. If it detects a change, it will update the relevant parts of its cache. This mechanism will also work if deploying the Fusion Registry across geographical regions with a read replica database for each region.

2. Rabbit MQ PUSH Notification – if configured Rabbit MQ message broker will be used to inform all instances that there was a change, and each instance will update accordingly. Note: it is important to apply the MQ connection on the first Registry instance before starting the subsequent instances. This will ensure that when the subsequent Registry instances start that they will have the correct MQ settings, since this is read on startup, and this will ensure all Registries are kept synchronized.

11.1.6 Caching
As well as the internal cache, the Fusion Registry provides the means to support HTTP 304 (not modified) to allow clients to cache the response to structures, data, and metadata queries.

Figure 17 showing the Caching settings
11.1.6.1 Cache Type

Fusion Registry can make use of a caching solution to cache its SDMX web service responses. Fusion Registry can either use its local file system solution, called Fusion Cache, or it can work with a Varnish server.

If the Fusion Cache is used, the cache limit should be set. The cache will be created in the specified temporary directory.

Varnish setup is explained further in section 12 of this document. For the Varnish Server setting, the fully qualified URL needs to be entered.

11.1.6.2 TTL and Max Series

The Cross Domain TTL can be used to put a “Time To Live” Header on query responses originating from a Cross Domain Origin. The Cross Domain Max Series can be used to restrict the maximum number of series a user can request if the request is originating from a Cross Domain (i.e. if JavaScript is being run locally to query the Registry server).

11.1.7 Data Validation

Figure 19 showing data validation settings
The setting on this page allows Data Validation to be configured. Data validation is performed when data is loaded and registered.

There are a number of validators defined in the Registry and they may relate to different data types (e.g. “Sequential Dates” only affects XLSX files). Each validator can take one of the settings of: “Ignore”, “Report Error”, “Report Error – Prevent Publication”, “Report Error – Prevent Publication & Conversion”, although some validators are so fundamental (e.g. “Duplicate Observations”) that they may not be set to “Ignore” or “Report Error”.

By default, each of these validators will be set to the minimum of “Report Error”. You may find that this is too restrictive (e.g. one validator is for when SDMX header files have the incorrect case), so if this causes issues with data reporting, simply switch these validators to “Ignore”.

A further control on this page allows for the configuring of the maximum number of errors that can be reported per category. If this value is set to 0 or a negative number, then there is no limit to the number of errors that can be reported.

11.1.8 Audit

The Fusion Registry can be used to audit activity, including structure queries, submissions, data registrations, data queries, etc. For each audited event, the associated server side logs can be captured and stored against the event. The Audit tab in the settings Registry Settings enables the level of auditing to be controlled, and the definition of which log events are captured can also be set. Audit and log information can be given a lifetime after which time they will be deleted.

Audit events are split into two categories:

1. **Transactional** – these audit events are ones which result in the Fusion Registry modifying structures, data, or metadata. A transactional event includes structure submissions, data registrations, and reference metadata submissions.

2. **Non Transactional** – these audit events are ones where information is being read from the Fusion Registry, but no information is being modified. These events include (amongst others) data queries, structure queries, and schema queries.

Audit events are used to capture additional information for structure modifications. They are also used to provide an overview of user activity as shown in the image below.
The Fusion Registry enables auditing to be enabled or disabled. If auditing is disabled, no audit events will be captured. The audit level can be set to audit all events, or transactional events only. **Delete Audits Every** input field can be set how many days the audit events should be kept for, if the field is left empty then the audits will never be deleted. It is important to note that this setting is only relevant for non-transactional events. Transactional audit events will not be deleted even if this setting is set.

Logging can be enabled or disabled. If logging is enabled, the **Logging Level** can be used to define that logs should only be captured for transactional events. The **Delete Logs Every** can be used to specify the number of days or hours that the server logs should be kept for, leaving this field blank will be used to indicate the logs should never be deleted. Note log events will only be deleted for non-transactional audit events.
11.1.9 CSV Format

The Fusion Registry can output datasets in CSV format. This settings page specifies what the delimiter will be; either comma, semicolon, space or tab. The output format can be ID only, name only or ID and Name. There is an option to output the series key – a colon separated field with the values for each of the series elements for that row.

![Registry Administration](image)

**Figure 23 showing the Registry CSV Format settings**

There is an example representation of a CSV output shown on this page and changing the options will update this example allowing a clearer understanding of what will be output.

11.1.10 CSV Locale

The Fusion Registry can provide CSV data localised for the end user. To obtain this a user simply needs to request CSV data with a locale parameter within the URL. However there are further customisations to how locales can affect the output which are configurable from this settings page.
HTTP header support is not enabled by default and can be enabled through the control at the top of this page. Web browsers often supply this information by default and so users may now find that requesting a CSV through the browser now returns it in the user’s locale.

A default locale can be specified, so that any users not explicitly specifying a locale will be supplied with data formatted to that locale.

To redefine a locale (for example removing a group separator or changing the decimal separator) there are controls at the bottom of the screen. A decimal separator and a group separator may only be a single character each. The group separator is optional.

11.1.11 Debug Settings

This page allows an administrator to specify the debug levels on both the Client and Server side of the Registry. Unless you have good reason to modify these settings, it is recommended to leave Client Side Debug set to “Debug Off” and Server Side Debug set to “Low”.

Figure 24 showing the CSV Locale settings
If Client Side Debugging is enabled, it will affect all users accessing the HTML User Interface of Fusion Registry. When Client Side debug is off, some of the required artefacts to make the HTML pages work correctly are supplied in a single minified file. This has the effect of reducing network traffic between the server hosting the Registry and any clients accessing it. If Client Side debug is enabled, instead of a single minified file, the files are supplied individually and non-minified. This makes the Registry easier to debug, but is generally not a feature that users require.

**Important Note:** Client Side Debugging should not be used with Internet Explorer since certain pages in the Registry will not work with Client Side Debugging enabled. Also, the “Content Security” pages will not work in any browser with Client Side Debugging enabled. Client Side Debugging should only be enabled when instructed by a member of the Metadata Technology support staff and should be set back to "Debug Off" as soon as the investigation has been completed.

The Server Side Debugging control affects how much information is stored in the Registry logs on the server side. This pertains to how the Java code is logged by the logging managers. The three values are:

1. **Low** – the default setting. Only Log levels of INFO or above (WARN, ERROR, etc.) are recorded in the logs.
2. **Medium** – All libraries pertaining to Metadata Technology will be logged at DEBUG level. All other libraries will be at INFO level.
3. **High** – Everything is logged at DEBUG level, including all third-party libraries.

Please note, that logging can introduce a performance impact. This is because logging information is persisted to files and to the database and so almost every interaction with the Registry could be affected.

Modifying these settings is only recommended if there is good reason to do so.

### 11.2 Custom Database Connection

If you wish to make a connection to a database but wish to specify values which are not available from the Registry User Interface, you can make use of the “Custom” database connection. This
allows a connection to a database of your choice by specifying the JDBC connection string. This can permit connection to other database types not listed in the drop-down, as well as using advanced features of a database not supported from the Registries User Interface but supported from the connection string.

![Database Connection String](image)

**Figure 26 showing a Custom database connection**

The database page can be accessed by navigating to Admin -> Settings -> Database. In the drop-down, select “Custom” and an entry field for “Database Connection String” will be displayed. The value that needs to be entered into this field must be a valid JDBC connection string (please refer to your Database vendor’s definition of how to specify the appropriate JDBC connection String).

An example of a valid custom string (to connect to an Oracle database):

`jdbc:oracle:thin:@localhost:1521/orcl12c`

This page includes fields for username, password, dialect and driver class name. The dialect is the Hibernate dialect to use when communicating with the database. The Registry uses database specific SQL generator code where necessary and the type of SQL generated is determined by the dialect. The driver class name specifies which Java driver to use and this information is provided by the database connector vendor. The following table gives examples of values to use for various databases:

<table>
<thead>
<tr>
<th>Database</th>
<th>Dialect</th>
<th>Driver Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>MySQL (5.x)</td>
<td>org.hibernate.dialect.MySQL5Dialect</td>
<td>com.mysql.cj.jdbc.Driver</td>
</tr>
<tr>
<td>SQL Server</td>
<td>org.hibernate.dialect.SQLServerDialect</td>
<td>com.microsoft.sqlserver.jdbc.SQLServerDriver</td>
</tr>
<tr>
<td>Oracle (12)</td>
<td>org.hibernate.dialect.Oracle10gDialect</td>
<td>oracle.jdbc.driver.OracleDriver</td>
</tr>
</tbody>
</table>

For other database types, please refer to the database vendor’s information.

Note that the Registry is only distributed with drivers for MySQL and SQL Server databases. If you wish to connect to another database type, you will need to provide the appropriate drivers for your system.
12 Running Apache Tomcat as a Service

12.1 Tomcat as a Service

On Windows environments you may wish to run Apache Tomcat as a service. If you wish to do this there are three important issues to be aware of:

- By default the amount of memory that is allocated to Tomcat as a service will almost certainly not be enough to run Fusion Registry with anything beyond a trivial amount of structures and data. It is recommended that if you wish to run Tomcat as a service, then you increase the “Maximum Memory Pool” value (see section 12.3).
- It may be desirable to explicitly state the location of the Fusion Registry properties file.
- Configuration of Apache Tomcat is not performed by modifications to the setenv.bat file and must be performed by configuring the service.

To configure Tomcat as a service start the Tomcat Configuration process (on Windows 10, type “Configure Tomcat” from the Windows Start menu).

12.2 Properties file Location

Services on Windows provide the ability to be run as a specified user. This can be changed (see image below) but by default the “Local System account” will be used.

![Figure 27 showing the Apache Tomcat Manager "Log On" settings page](image)

If using the “Local System account” then the home directory is a location in the Windows folder itself. This actual value will vary on different version of Windows, but on Windows 10, it may be:

```
C:\Windows\System32\config\systemprofile\FusionRegistry
```

**Important note:** This location is not accessible to any users except those without Administration access.

This location will be used to store and retrieve Fusion Registry properties. It is probably not desirable to use this location. Modifying this can be achieved in one of two ways:

- Changing the “Log On” user for the Tomcat Service to be another user on the system. This will have the effect of using that user’s home directory for the properties file. E.g. if the user
specified is “user1” then the Fusion Registry properties file will probably be located in C:\Users\user1\MetadataTechnology\FusionRegistry.

- Providing the argument to specify the location of where the properties file will be read from and written to. This is explained further in section 5.2 and section 12.3 of this document.

12.3 Configuring Apache Tomcat

The file “setenv.bat” cannot be used to configure Apache Tomcat when run as a service. Instead the Apache Configuration Tool has a tab labelled “Java”. From this tab the appropriate settings can be applied.

![Figure 28 showing the Apache Tomcat Manager Java Properties page](image)

By default, the Initial memory pool Maximum Memory Pool is set to 128Mb and the Maximum memory pool is set to 256Mb. This Maximum Memory Pool will almost certainly not be sufficient to run Fusion Registry so it is recommended to increase this value to at least 2048 Mb. These two settings are the equivalent of settings –Xms and –Xmx in the setenv.bat file.

Other settings, such as “java.library.path” (for SQL Server SSO) must be entered in the input area labelled “Java Options”. Ensure each setting is on its own line. The example below shows typical Apache settings along with the Fusion Registry properties location and the SSO DLL location being set:

-DCatalina.home=C:\tomcat-service
-DCatalina.base=C:\tomcat-service
-Djava.io.tmpdir=C:\tomcat-service\temp
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
-Djava.util.logging.config.file=C:\tomcat-service\conf\logging.properties
-DRegistryProperties=file:///c:/dir/AFile.txt
-Djava.library.path=C:\temp\SSO_DLL
13 Configuring Varnish

Varnish is an HTTP accelerator allowing for caching of various requests. You may find it useful if you have repeated predictable requests for specific data or structures. Varnish is not provided with the Fusion Registry and must be obtained from the vendor’s website.

Varnish needs to be configured so that requests to your web site are sent directly to Varnish, which will then either serve the request from its cache or request the information from Fusion Registry. Typically it is placed in the following arrangement:

![Diagram of Varnish configuration](image)

When a request is placed to your server, Varnish will check to see if the request exists within its cache. If it does not, then the request is passed onwards and will be served from the Registry. As the request response is passed back, Varnish will cache this request and response so that any equivalent subsequent requests will be served from the cache.

However when structures or data changes within the Fusion Registry, notification must be sent to the Varnish server informing Varnish that any subsequent requests for that particular structure or data must not be served from the cache. This is achieved by informing the Registry of the Varnish server on the “HTTP Cache” page of the settings (see section 11.1.6). The location of the varnish server (and possibly the port if not using port 80) should be specified. So it is likely that the value 'http://127.0.0.1:80' may suffice in many cases. Once this value is set up, on changes to the structures or data, a Varnish BAN request is sent to the URL specified. Varnish will consume this request and remove the appropriate values from its cache.

If the HTTP Cache value is not set correctly, then you will observe that previously cached queries will be served even though the data or structures have changed in the Registry.

The log files of the Registry will show Varnish BAN requests being sent. The log files of Varnish will show BAN requests being received.

With the Fusion Registry distribution there is an example Varnish configuration file that you may wish to use as a starting point for your Varnish configuration. For further instructions on configuring Varnish please refer to the developer’s web sites.
14 Configuring User Agent

When Fusion Registry performs HTTP communication to other servers (e.g. when performing a registration) the HTTP request header will specify a value for “User-Agent”. By default this value will be a string of the form:

    FusionRegistry/<version number of the Registry>

For example:

    FusionRegistry/9.4.0

If you wish to modify the User Agent that Fusion Registry supplies, you will need to edit a properties file within the Registry. The properties file is called:

    metadata.properties

And is located in the directory:

    <Tomcat HOME>/webapps/<Web AppName>/WEB-INF/classes

There will be an entry in this file which states the value for user.agent in the following manner:

    user.agent=FusionRegistry/9.4.0

This value can be set to whatever value you wish the User Agent to be. It is permissible to leave this value blank if you require the Registry to not specify a User Agent in HTTP communication. Please note that since this file is within Fusion Registry itself and not in the Settings file (see section 5) if you upgrade Fusion Registry in the future, you will need to make the change again.

It is recommended that you do not modify any of the other values in this file.
15 Recovering a Locked Local Root Account

15.1 Unlocking
If the root account becomes locked, to unlock it you will need access to the underlying database that stores all of the Registry information. In the database there will be a table with the name 'registry_root_security'. This table contains the following information about the root account: the username; the password (encrypted); the number of times a wrong password can be entered before the account is locked; whether or not the root account is locked.

To unlock a locked root user set the value of the column 'is_locked' to 0. There is no need to restart the Registry after this change, the root user is now unlocked.

15.2 Changing password
In order to change the root password without accessing the Fusion Registry’s GUI, you must run a separate application that is bundled with the distribution of the Registry.

This tool can be found in the folder ‘password reset’, and tool name ‘password reset.jar’ is an executable jar file that can be run by issuing the command “java -jar ‘password reset.jar’” on the command prompt.

Once run, it will ask you for the new password, or if left blank, will generate a pseudo-random password.

The output will show the new Hash and the password used; in order to apply this new password, you must copy the hash generated from the tool and inserted into the “pwd” column of the table 'registry_root_security'.

After this is complete, you will be able to log into the root account using this new password.
16 Reverting to the Install Wizard

If, for whatever reason, you need to return the Registry to the state of displaying the Install Wizard, then you will need access to the underlying database that stores all of the Registry information. In the database there will be a table with the name 'registry_settings'. This table contains information about the Registry in a table with column names of 'name' and 'value'. One of the rows in this table will have the name of 'installed.version' and the value will be the version of the Registry.

Deleting this row in the table, committing this change to the database and then restarting the Registry will ensure that on restart that the Registry is displaying the install wizard.