pitney bowes

Customer Engagement

EngageOne Vault

7.6

Vault REST API for EngageOne Server 4.4 SP9

Table of Contents

1 - Vault REST API EngageOne Server installation

The Vault REST API is a RESTful service that allows a web-based application service to access a Vault server using REST calls.

In this section

Vault REST API EngageOne Server 4.4

The Vault REST API is used by the Archive panes to access the associated Vault system.

Before starting, identify the Vault REST API related properties you will need to override (see Vault Properties section below) and the associated values. At a minimum, this will include the host name or IP address of the Vault server and the port number that will be used for requests from the Vault REST service to the Vault server.

Note:

- This version of the Vault REST API is compatible with EngageOne 4.4. It is not compatible with earlier versions of EngageOne.
- The Vault REST API war file is bundled with EngageOne Server 4.4 and is installed using the EngageOne Server 4.4 installer. Please refer to the EngageOne Server 4.4 installation instructions on how to install this optional component of EngageOne Server.

Vault REST API properties file

The Vault REST properties file defines some operational values that the Vault REST service requires.

The format of the file is:

```
Property_name=property_value
```

The following table lists the available properties and its default value:

Property Name	Description	Default Value
`vault.hostname`	Host name or IP address of the Vault connection that the Vault REST API uses.	127.0.0.1
`vault.portnumber`	Port number that the Vault REST API will use to connect to the Vault server.	6003
`vault.rest.sslconnection`	SSL connection flag. True enables SSL connection and false for non-SSL connection.	false

Property Name	Description	Default Value
`vault.rest.certificate.truststoreFilePath`	File path to the Java trust store (for trusted certificate authorities). Default: `C:/Program Files (x86)/Java/jdk1.8.0_25/jre/lib/security/cacerts`	
`vault.rest.certificate.truststorePw`	Password of Java trust store The password may be encrypted using Vault password encryption.	(blank)
`vault.rest.certificate.keystoreFilePath`	File path to the Java key store (used for SSL for certificates and private keys). Default: `C:/Program Files (x86)/Java/jdk1.8.0_25/jre/lib/security/cacerts`	
`vault.rest.certificate.keystorePw`	Password of Java key store The password may be encrypted using Vault password encryption.	(blank)
`vault.rest.certificate.keyPw`	Password of the keys in the SSL certificate file The password may be encrypted using Vault password encryption.	(blank)
`vault.defaultPageno`	Results page number used when a results page number is not specified by the requestor. **Note:** This is not related to document page number.	1
`vault.defaultPagesize`	The number of results that will be returned in response to a query when a pagesize is not specified in a query. This is not related to a document page size.	10
`vault.defaultPDFPageCount`	The default maximum page count for PDF files created.	1000
`vault.defaultReprintStartpage`	The starting page in a document reprint.	1

Property Name	Description	Default Value
`vault.defaultReprintTotalpages`	The maximum number of pages that will be included in a document reprint starting from the `vault.defaultReprintStartpage`	1000
`vault.defaultDocumentMaxhits`	The maximum number of results that will be returned from a document or account query. This is also the maximum number of records that will be used for consolidating document records into folders for folder based queries.	1000
`vault.defaultDocumentFolderMaxhits`	The maximum number of document records that will be used when consolidating documents records into folders for folder based queries.	1000
`vault.defaultDocumentFolderMaxhitsOutput`	This is the maximum number of folders that will be returned in a folder based query.	1000
`vault.sortlocale`	This is the locale used when arranging the display order of folders and documents within folders.	en-US
`vault.minimumRenderTime`	This is the minimum amount of time (in milliseconds) that a successful render request should take. Some browser and PDF reader combinations require a delay between request and reply to function correctly.	100
`vault.rest.key`	The path to the key file used to encrypt Vault passwords that are encrypted.	
`vault.rest.serviceUser`	The user name used to access the Vault service (render or router) when Vault authentication is enabled.	
`vault.rest.servicePW`	The password used to access the Vault service (render or router) when Vault authentication is enabled. This password may be encrypted using Vault password encryption.	

Property Name	Description	Default Value

`vault.rest.pool.enabled`	Controls whether the Vault REST API should use a socket connection pool when communicating with the Vault renderer.
	If set to true, a connection pool is used. You can control the pool with the settings:
	`vault.loader.pool.maximum`
	`vault.loader.pool.startup`
	`vault.loader.pool.connectionCheckInterval`
	The default is false (use a new connection is used for each message between the REST API and the Vault loader).
		false

`vault.rest.pool.maximum`	Maximum number of socket connections the socket pool will use,	5
`vault.rest.pool.startup`	Number of connections allocated to the socket pool when the connection to the Vault system is created.	5
`vault.rest.pool.connectionCheckInterval`	Time interval (in seconds) between echo messages on an idle socket. An echo message keeps the connection to Vault loader active to keep it from being closed for inactivity.	30

Vault REST properties file sample lines

```
Sample properties entries (using all default values)
#####################################
#for Vault servers
#Vault Render IP address or host name vault.hostname=127.0.0.1
#Vault Render port number vault.portnumber=6003
#####################################
vault.rest.sslconnection=true
vault.rest.certificate.keystoreFilePath=C:/Program Files
(x86)/Java/jdk1.8.0_25/jre/lib/security/cacerts
vault.rest.certificate.keystorePw=changeit
```

```
vault.rest.certificate.keyPw=<password>
vault.rest.certificate.truststoreFilePath=C:/Program Files
(x86)/Java/jdk1.8.0_25/jre/lib/security/cacerts
vault.rest.certificate.truststorePw=changeit
# URL pagination control
#default URL page number vault.defaultPageno=1
#default URL page size vault.defaultPagesize=10
#####################################
#PDF rendering control :
# maximum number of pages allowed in a generated PDF file
vault.defaultPDFPageCount=1000
############################
# reprint control
# starting page when reprinting document vault.defaultReprintStartpage=1
# maximum total page count when reprinting document
vault.defaultReprintTotalpages=1000
#####################################
```

REST API related index flags

Two new flags, `f` and `g`, are now available for the IndexN settings in the `database.ini` file.

Use the f flag to indicate that the index is a date index. This date index gathers documents under the same month for the REST API month folder view.

Use the g flag to indicate that the index is a Communication Type attribute index. The Communication Type index is used by the Vault REST API to gather documents that have the same communication type into a Communication Type view.

An example of what should be in the Vault `database.ini` file for the date index is:

```
Index4=invlink,doc.date,xdhasbf,Dates under this account
Render4=Communication Type;Date,COMTYPE;doc.date
```

Where `COMTYPE` is the name of the Vault job journal field for Communication Type.

Note: The `b` flag is required to force search results to be presented in descending date order.

An example of what should be in the Vault database.ini file for the attribute index is:

```
Index11=communicationtype,?COMTYPE+int.null+doc.date,sadhg,Communication
 Type
Render11=Date;Communication Type,doc.date;int.match
```

The `?COMTYPE+int.null+doc.date` syntax causes an index entry to be created even if `COMTYPE` is not defined in the journal entry for a document.

2 - Vault REST SSL set-up

The following steps cover setting up Vault REST SSL.

In this section

Vault REST SSL set-up

1. Create a private key.
   a) Enter `openssl genrsa -des3 -out vault.key 1024`
   b) Enter pass phrase for vault.key.
2. Create a certificate signing request (CSR) file.
   a) Enter `openssl req -new -key vault.key -config openssl.cnf -out vault.csr`
   b) Enter pass phrase for `vault.key`.
   c) Enter values for these fields. Some fields show a default value. If you enter '.', the field will be left blank.

      Country Name (2-letter code) [AU]

      Enter State or Province Name (full name) [Some-State]

      Enter Locality Name (for example, city) []

      Organization Name (for example, company) [Internet Widgits Pty Ltd]

      Organizational Unit Name (for example, section) []

      Common Name (for example, YOUR name) []

      Email Address []

      A challenge password []

      An optional company name []

3. Remove the password for the private key.
   a) Enter `copy vault.key vault.key.org`
   b) Enter `openssl rsa -in vault.key.org -out vault.key`
   c) Enter pass phrase for `vault.key.org`
4. Create a certificate.
   - Type `openssl x509 -req -days 365 -in vault.csr -signkey vault.key -out vault.crt`
5. Create a PFX file.
   - Enter `openssl pkcs12 -in vault.crt -inkey vault.key -export -out vault.pfx`

     Where:

     `vault.crt` = certificate

`vault.key` = private key

`vault.pfx` = resulting PFX file (containing BOTH the key and cert)

6. Transform the PFX file into a PEM file.
   a) Enter `openssl pkcs12 -in vault.pfx -out vault.pem`
   b) To remove the encryption, enter `openssl pkcs12 -nodes -in vault.pfx -out vault.pem`
   P12 and PFX are the same format. PEM is the format that combines the CERTIFICATE and PRIVATE KEY information.

Register a certificate in Java environment

1. Before you start:
   - Open Windows Prompt (requires administrator privileges)
   - Set up your Java running environment, for example:

     `C:/Program Files (x86)/Java/jdk1.8.0_25/jre/`

     or

     `C:/Program Files (x86)/Java/jre1.8.0_25/`

     Certificates are located in `c:\certificates`

     The default password (for java default certificate key store) is changeit

2. Change to the directory with the certificates `C:\certificates`
3. Enter `keytool -import -v -alias VaultServer -file vault.crt -keystore "C:/Program Files (x86)/Java/jdk1.8.0_25/ jre/lib/security/cacerts"`
4. Enter keystore password: `changeit` (default)
   The certificate information displays.

Setup SSL for VaultREST

Create the certificates and register them in Java environment. For more information, see **Setup SSL for VaultREST** on page 11.

Note:

- `vault.key` is a self-signed private key file, no password for the key file.
- `vault.crt` is a self-signed certificate file.

1. Set up SSL enabled on server/render side. For more information, see the "Vault Customizing Guide".
2. Copy the certificates to the server and render working directory, for example `<server>` and `<render>`.
3. Set up SSL enabled for Repository server (`e2serverd.exe`)

   Edit `e2serverd.ini`

```
[server1]
service=*:6001
ssl=1
sslcertificate=vault.crt
sslprivatekey=vault.key
```

4. Set up SSL enabled for Rendering Engine (`e2renderd.exe`)

   Edit `e2renderd.ini`

```
[server1]
service=*:6003
ssl=1
sslcertificate=vault.crt
sslprivatekey=vault.key
```

5. If SSL is enabled in `e2serverd.ini`, you need to set up SSL enabled for [connection1] in `e2renderd.ini`

   Edit `2renderd.ini`

```
[connection1]
service=127.0.0.1:6001
ssl=1
sslcertificate=vault.crt
sslprivatekey=vault.key
```

6. Set up SSL for VaultREST component.

   The VaulREST SSL parameters are described in “Vault REST API properties file” on page 6.

   a) Use the `viewpoint.config.externalPropertyFilePath` parameter to set up an external property file. (`VaultREST.properties`). A sample is contained in the VaultREST war file.

   Here is an example under Windows 7 with Tomcat 6.0.32:

```
run-VaultREST.bat:
set JAVA_OPTS=%JAVA_OPTS%
```

```
-Dviewpoint.config.externalPropertyFilePath=c:/VaultRESTConfig/VaultREST.properties

startup.bat
```

Note: Alternatively, you can create or modify `bin\setenv.bat` which is run automatically by `startup.bat` (if present) to set system properties.

a) Set up parameters in `VaultREST.properties`

For the path separation, please use the forward slash sign "/"

```
vault.rest.sslconnection=true
vault.rest.certificate.keystoreFilePath=C:/Program Files
(x86)/Java/jdk1.8.0_25/jre/lib/security/cacerts
vault.rest.certificate.keystorePw=changeit
vault.rest.certificate.keyPw=<password>
vault.rest.certificate.truststoreFilePath=C:/Program Files
(x86)/Java/jdk1.8.0_25/jre/lib/security/cacerts
vault.rest.certificate.truststorePw=changeit
```

For more information, see “Vault REST API properties file” on page 6.

Note: If you are using Vault password encryption for SSL based passwords, you will have to set up the `vault.default.key` Java system property, or define the `vault.rest.key` property in the external properties file.

How to process PKCS12 Keystore/Truststore

Java keytool.exe can be used to process JKS format Keystore/Truststore. For PKCS12 format Keystore / truststore, you need to use openssl commands. Below scripts are based on windows platform.

```
#####################################
create keystore in PKCS12 format:
type vault.key vault.pem > vault-key-cert.txt
openssl pkcs12 -export -in vault-key-cert.txt -out vault-keystore.pkcs12
 -name vault -noiter -nomaciter -passout pass:changeit
#####################################
create truststore in PKCS12 format: openssl pkcs12 -export -in vault.crt
 -inkey vault.key -out vault-truststore.pkcs12 -name vault -noiter
-nomaciter -passout pass:changeit
#####################################
# import key/certificate into PKCS12 keystore / truststore
openssl pkcs12 -export -in vault.crt -inkey vault.key -out
vault-keystore.pkcs12 -name vault -noiter -nomaciter -passout
```

```
pass:changeit
#####################################
list info inside a keystore / truststore:
openssl pkcs12 -in vault-keystore.pkcs12 -info -password pass:changeit
```

3 - Localizing search and database descriptions

Vault servers provide a series of low-level functions that list available database and search properties.

In this section

Overview

Inside this information are text descriptions that client applications can present to the end-users to indicate what the purpose of each database and search is. These descriptions are configured centrally in the database sections in the `server\database.ini` configuration file.

Locales

Some Vault installations need to support users in multiple languages. In this case the applications need descriptions specific to the user locale. Locale aware applications can tell the server which locale they are interested in and the server will return locale specific descriptions if available. A sample locale set up is in the **Database configuration example** on page 17 below.

Configuration

The database section in the `database.ini` file contains a number of configuration parameters that contain text descriptions. Later sections detail how to set up each key specifically.

Text	API Function	Parameter	Localized Parameter
database description	`database.list`	`Description=`	`Description_<locale>=`
search description	`database.info`	`Index<N>=`	`Index<N>_<locale>=`
search output column name	`database.search`	`Render<N>=`	`Render<N>_<locale>=`

The default descriptions are configured through the Description, `Index<N>` and `Render<N>` parameters. There are localized variants of the setting that have a locale name appended to the parameter name. Locale names should follow the format specified in RFC-1766 "Tags for the Identification of Languages".

These are some example locale names:

Tag	Locale
en-CA	English (Canada)
fr-CA	French (Canada)
el-GR	Greek (Greece)
ja-JP	Japanese (Japan)
zh-Hans	Chinese (Simplified)
zh-Hant	Chinese (Traditional)

The default description strings are returned when using the default locale, when the application does not specify a locale or when the requested locale is not configured.

Applications that support locales (including the Vault REST API) will send the `request.locale` parameter with the locale name as part of the command.

> **Note:** The locale parameter needs to be a specific tag. The server does not support weighted locale selection used by the HTTP Accept-Language header.
>
> The locale parameters will appear in the server log file: `09:54:10 127.0.0.1:56708 {2} <database1> database.list request, locale [fr-CA]`

Encoding

The `database.ini` file is normally encoded in the host default code page. For Windows servers, you can alternatively save the database.ini file as Unicode (UTF-16). The encoding of the `database.ini` file will determine what scripts are supported.

For example, if the file is encoded as 'ISO 8859-1 (Latin 1)', you would not be able to represent CJK (Chinese, Japanese, Korean) languages.

Database configuration example

Below is a sample database section from `database.ini` that shows a case where the descriptive text is configured for two locales, one English and one French. The default locale is being used for

the English descriptions and a single alternate “locale fr-CA” (French, Canada) is provided for French speaking users.

```
[oas]
Description=Old Age Security
Description_fr-CA=Sécurité de la Vieillesse
Index1=account,cust.account,wctul,Account Numbers
Index2=name,cust.name,crtul,Names
Index3=address,cust.address,crtul,Addresses
Index4=invlink,doc.date,xdhasb,Dates under this account
Index5=sin,SIN,ctl,Social Insurance Number
Index1_fr-CA=Numéros de Compte
Index2_fr-CA=Noms
Index3_fr-CA=Adresses
Index4_fr-CA=Dates sous ce Compte
Index5_fr-CA=Numéro d'Assurance Sociale
Render1=Account;Name;Address,int.match;cust.name;cust.address
Render2=Name;Account;Address,cust.name;cust.account;cust.address
Render3=Address;Account;Name,int.match;cust.account;cust.name
Render4=Date,doc.date
Render5=SIN;Account;Name;Address,int.match;cust.account;cust.name;cust.address
Render1_fr-CA=Compte;Nom;Adresse
Render2_fr-CA=Nom;Compte;Adresse
Render3_fr-CA=Adresse;Compte;Nom
Render4_fr-CA=Date
Render5_fr-CA=NAS;Compte;Nom;Adresse
```

Description

The text name of the database presented to users.

If an application provides a locale name, the server will look for alternate text descriptions in the `Description_<locale>` settings (see below). If the alternate setting is not present it will use this description as a default.

Databases without descriptions will not be shown to users. If you are not using the default database, you can remove it from the list of databases shown to users by explicitly setting its description to the empty string. Sections of common settings that are intended to be pulled in through inherit= and not used as databases should not set descriptions.

Format

`Description=<description>`

Default

Description=

Example

`Description=Monthly Residential Statements`

Description_<locale>

The text name of the database presented to users for a specific, alternate locale. See **Localizing search and database descriptions** on page 15 above.

Format

```
Description_<locale>=<description>
```

Default

When not configured, the server will fall back to using the text from the `Description=` setting.

Examples

```
Description_de-DE=Monatsrechnungen
```

```
Description_fr-FR=Eau Résidentiel
```

Index<N>_<locale>

An alternate index description corresponding to the `Index<N>` setting but for a specific, alternate locale.

> **Note:** This key is only the description; it does not have the additional fields that appear in the `Index<N>` setting. See **Localizing search and database descriptions** on page 15 for more information.

Format

```
Index<N>_<locale>=<description>
```

Default

When not configured, the server will fall back to using the text from the `Index<N>=` setting.

Example

```
Index5_fr-CA=Numéro d'Assurance Sociale
```

Render<N>_<locale>

An alternate list of search output column titles corresponding to the `Render<N>` setting but for a specific, alternate locale.

> **Note:** This key is only the list of titles; it does not have the list of fields that appear in the `Render<N>` setting. The list must have the same number of entries as the list in `Render<N>`.

See **Localizing search and database descriptions** on page 15 for more information.

Format

```
Render<N>_<locale>=<title>;<title>;<title>...
```

Default

When not configured, the server will fall back to using the text from the `Render<N>=` setting.

Examples

```
Render3_fr-CA=Adresse;Compte;Nom
```

```
Render6_es-ES=Cuenta;Nombre;Dirección
```

```
Render7_sv-SE=Fakturanummer;Datum
```

3001 Summer Street
Stamford CT 06926-0700
USA

www.pitneybowes.com/us