TIBCO Data Virtualization Authentication and Security Scenarios

<table>
<thead>
<tr>
<th>Project Name</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Release</td>
<td>2.0</td>
</tr>
<tr>
<td>Date</td>
<td>April 2018</td>
</tr>
<tr>
<td>Primary Author</td>
<td>Michael Gardner</td>
</tr>
<tr>
<td>Document Owner</td>
<td>Deane Harding</td>
</tr>
<tr>
<td>Client</td>
<td></td>
</tr>
<tr>
<td>Document Location</td>
<td></td>
</tr>
<tr>
<td>Purpose</td>
<td>Highlights various usage patterns involving TIBCO Data Virtualization and security, particularly regarding supported authentication methods.</td>
</tr>
</tbody>
</table>
## Revision History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>September 2012</td>
<td>Michael Gardner</td>
<td>Initial revision</td>
</tr>
<tr>
<td>2.0</td>
<td>April 2018</td>
<td>Deane Harding</td>
<td>Updated with TIBCO branding</td>
</tr>
</tbody>
</table>

## Approvals

This document requires the following approvals. Signed approval forms are filed in the project files.

<table>
<thead>
<tr>
<th>Name</th>
<th>Signature</th>
<th>Title</th>
<th>Company</th>
<th>Date of Issue</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Distribution

This document has been distributed to:

<table>
<thead>
<tr>
<th>Name</th>
<th>Title</th>
<th>Company</th>
<th>Date of Issue</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Related Documents

This document is related to:

<table>
<thead>
<tr>
<th>Document</th>
<th>File Name</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDV Administration Guide</td>
<td>TIB_tdv_7.0.6_AdministrationGuide.pdf</td>
<td>TIBCO DV Engineering</td>
</tr>
</tbody>
</table>
Copyright Notice
COPYRIGHT® TIBCO Software Inc. This document is unpublished and the foregoing notice is affixed to protect TIBCO Software Inc. in the event of inadvertent publication. All rights reserved. No part of this document may be reproduced in any form, including photocopying or transmission electronically to any computer, without prior written consent of TIBCO Software Inc. The information contained in this document is confidential and proprietary to TIBCO Software Inc. and may not be used or disclosed except as expressly authorized in writing by TIBCO Software Inc. Copyright protection includes material generated from our software programs displayed on the screen, such as icons, screen displays, and the like.

Trademarks
All brand and product names are trademarks or registered trademarks of their respective holders and are hereby acknowledged. Technologies described herein are either covered by existing patents or patent applications are in progress.

Confidentiality
The information in this document is subject to change without notice. This document contains information that is confidential and proprietary to TIBCO Software Inc. and its affiliates and may not be copied, published, or disclosed to others, or used for any purposes other than review, without written authorization of an officer of TIBCO Software Inc. Submission of this document does not represent a commitment to implement any portion of this specification in the products of the submitters.

Content Warranty
The information in this document is subject to change without notice. THIS DOCUMENT IS PROVIDED “AS IS” AND TIBCO MAKES NO WARRANTY, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO ALL WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. TIBCO Software Inc. shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Export
This document and related technical data, are subject to U.S. export control laws, including without limitation the U.S. Export Administration Act and its associated regulations, and may be subject to export or import regulations of other countries. You agree not to export or re-export this document in any form in violation of the applicable export or import laws of the United States or any foreign jurisdiction.

For more information, please contact:

TIBCO Software Inc.
3303 Hillview Avenue
Palo Alto, CA 94304
USA
# Table of Contents

1. **Introduction** ................................................................. 7  
   1.1 Purpose ........................................................................... 7  
   1.2 Audience ........................................................................ 7  

2. **NTLM for REST Web Services (on Linux)** ................................................... 8  
   2.1 Ingredients ...................................................................... 8  
   2.2 Recipe ............................................................................ 8  
   2.3 Tips .............................................................................. 9  

3. **LDAP for Authentication (Including SSL)** ................................................... 10  
   3.1 Ingredients .................................................................... 10  
   3.2 Recipe .......................................................................... 10  
   3.3 Tips ............................................................................. 11  

4. **Integrated Authentication with SQL Server** .................................................... 12  
   4.1 Ingredients .................................................................... 12  
   4.2 Recipe .......................................................................... 12  
   4.3 Tips ............................................................................. 12  

5. **Security Policies for Web Services** ................................................................. 13  
   5.1 Ingredients .................................................................... 13  
   5.2 Recipe .......................................................................... 13  
   5.3 Tips ............................................................................. 15  

6. **SAML Validator** .............................................................................. 16  
   6.1 Ingredients .................................................................... 16  
   6.2 Recipe .......................................................................... 16  
   6.3 Tips ............................................................................. 18  

7. **Kerberos Single Sign-On With AES256 Encryption and TDV (on Linux)** .......... 20  
   7.1 Ingredients .................................................................... 20  
   7.2 Recipe .......................................................................... 20  
   7.3 Tips ............................................................................. 24  

8. **TDV Studio and SSO with Kerberos** ................................................................ 26  
   8.1 Ingredients .................................................................... 26  
   8.2 Recipe .......................................................................... 26  
   8.3 Tips ............................................................................. 27  

9. **TDV with SSL (CA Signed Certificate) – Server Side Configuration** .......... 29  
   9.1 Ingredients .................................................................... 29  
   9.2 Recipe .......................................................................... 29  
   9.3 Tips ............................................................................. 36  

10. **TDV with SSL (CA Signed Certificate) – Connecting with Studio** ............... 37  
   10.1 Ingredients .................................................................... 37
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.2</td>
<td>Recipe</td>
<td>37</td>
</tr>
<tr>
<td>10.3</td>
<td>Tips</td>
<td>38</td>
</tr>
<tr>
<td>11</td>
<td>TDV with SSL (CA Signed Certificate) – Accessing a Published TDV Web Service from TDV</td>
<td>39</td>
</tr>
<tr>
<td>11.1</td>
<td>Ingredients</td>
<td>39</td>
</tr>
<tr>
<td>11.2</td>
<td>Recipe</td>
<td>39</td>
</tr>
<tr>
<td>11.3</td>
<td>Tips</td>
<td>41</td>
</tr>
<tr>
<td>12</td>
<td>TDV with SSL (CA Signed Certificate) – Accessing a WSDL Data Source over SSL</td>
<td>42</td>
</tr>
<tr>
<td>12.1</td>
<td>Ingredients</td>
<td>42</td>
</tr>
<tr>
<td>12.2</td>
<td>Recipe</td>
<td>42</td>
</tr>
<tr>
<td>12.3</td>
<td>Tips</td>
<td>45</td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Purpose
This document will highlight various usage patterns involving TIBCO Data Virtualization (TDV) and security, particularly regarding supported authentication methods.

While TDV contains a robust set of security features designed to work together as an integrated system to protect data and access to it, this document does not aim to describe that system. Rather, it will highlight a subset of the features available, with accompanying information that will give the reader an insightful glimpse into:

- **integration** – a description of how the integration between TDV and the authentication technology is achieved.
- **implementation** – not necessarily a how-to guide, but some detail on the level of configuration required and what prerequisite technologies may possibly need to exist.
- **tips** – things to consider, often based on expertise about previous successes or known limitations.

This document will serve both to highlight the wide range of scenarios supported by TDV, as well as a reference guide to quickly identify whether an existing environment’s authentication practice may already be supported by TDV.

If a particular use case is not highlighted in this document, please refer to the bundled documentation that ships with the product for a comprehensive list of security features and supported authentication methods.

1.2 Audience
This document is intended to provide guidance to the following users:

- All audiences – for referencing supported authentication methods.
2 NTLM for REST Web Services (on Linux)

2.1 Ingredients

- TDV version 7.0.0 or later on Linux
- Winbind (a component of Samba)
- NTLM configuration for TDV

2.2 Recipe

1. Configure Samba (winbind) on the TDV host.

   Enabling NTLM authentication while using Linux as the host operating system for TDV requires a level of interoperability between Windows and Unix account management. “winbind” (a component of Samba) provides that unification and allows a Unix box to become a member of a Windows domain. Therefore, the first step in achieving this type of authentication with TDV involves installing and configuring winbind on the TDV Linux host.

   More information on winbind can be found at [http://www.samba.org/samba/docs/man/Samba-HOWTO-Collection/winbind.html](http://www.samba.org/samba/docs/man/Samba-HOWTO-Collection/winbind.html)

2. Implement NTLM authentication for TDV on Unix.

   a. Create an LDAP (Active Directory) domain using the TDV Web Manager
b. Configure TDV using the Studio client (Administration > Configuration option):
   
i. Enable NTLM authorization – Set the following setting to true:
      
      Server > Configuration > Security > Authentication > Allow NTLM Authentication
   
   ii. Provide the “external domain” for TDV. This value will be the same value used when
        creating the LDAP (Active Directory) domain using the TDV Web Manager. To do this, edit
        the following setting:
        
        Server > Configuration > Security > Authentication > NTLM External Domain
   
   c. Enable NTLM authorization in the applicable web service

Please note: More detail on this process can be found in the TDV Administration Guide.

3. At this point you can launch a client (e.g. Internet Explorer) and execute the web service. Your request should
   authenticate seamlessly (no login prompts) and your response should appear.

2.3 Tips

- Ensure you are running a minimum TDV version of 7.0.0.00.00 on both the TDV Server and TDV Studio.

- There is a known issue with using a Samba version that is less than 3.2 with Active Directory 2008. Ensure
  you upgrade to at least Samba 3.3.

- There are multiple clients that you can use to test whether authentication works from the TDV Linux server to
  Active Directory. It is possible that one may work (authenticate successfully) while another will not, so it is
  recommended to use (at the least) both “ntlm_auth” and “wbinfo” utilities. Ensure you are logged into the
  server as the same account that TDV will be running under

- Installing and configuring winbind on the TDV Linux host will be specific and unique for every environment,
  and will be done by using the authconfig command or modifying the smb.conf file. While this is documented in
  the TDV Administration Guide, please note that you may have to make the following changes:

   o The security type may be “ADS” instead of “domain”

   o To join the Linux server to the Windows domain, you may need to use the “net ads join” command as
     opposed to “net rpc join” (e.g. net ads join -U Administrator -S server.domain.com)

- If the setup is not working, ensure Samba permissions allow TDV to perform authentication. For example, if
  Samba was installed as root then you might change the group ownership of /var/samba/winbindd_privileged
  from “root” to the group of the user account that TDV runs as (e.g. ‘tibco’). The bottom line is that at runtime,
  the TDV service needs appropriate privileges so it can perform authentication activities via Samba.
3 LDAP for Authentication (Including SSL)

3.1 Ingredients
- TDV on any supported platform
- A supported LDAP server
- Credentials for TDV to use to query LDAP

3.2 Recipe
The process of configuring TDV to use LDAP for authentication is divided into two main steps – configuring an LDAP domain in TDV and then adding LDAP groups to TDV. Optionally, there is a third step when SSL is included.

Please note: Additional detail can be found in the TDV Administration Guide under the topic “LDAP Domain Administration”.

1. Configure an LDAP domain within TDV.
   a. Add an LDAP domain using TDV Web Manager.
   b. Configure the ldap.properties as appropriate for your environment.

2. Add LDAP groups to TDV
   a. Add the desired LDAP groups to TDV using TDV Web Manager.
   b. Assign rights and privileges as appropriate to the LDAP groups.

3. Configuring SSL (if applicable)
   a. If your LDAP server serves a certificate that is signed by a well-known Certificate Authority (CA) then just change the url from ldap:// to ldaps://.
   b. If the LDAP server is serving a self-signed certificate or a certificate signed by an internal CA, then:
      i. Get the Java class InstallCert from http://code.google.com/p/java-use-examples/source/browse/trunk/src/com/aw/ad/util/InstallCert.java, remove the package line from this file (i.e. the first line that looks similar to "package com.aw.ad.util;"), and compile using the javac command.
      ii. Copy the TDV truststore file (cis_server_truststore_strong.jks if using strong encryption or cis_server_truststore.jks if not) to the directory where InstallCert.class exists and rename the truststore file to jssecacerts
      iii. Execute java -cp . InstallCert <ldaphost>:<sslport> and accept when prompted to add the certificates to the truststore.
iv. Backup the original truststore file and copy the new jssecacerts to $TDV_HOME/conf/server/security/cis_server_truststore.jks (or cis_server_truststore_strong.jks, as appropriate).

v. Copy $TDV_HOME/jre/lib/security/cacerts to the directory where InstallCert.class exists and rename cacerts to jssecacerts.

vi. Execute `java -cp . InstallCert <ldaphost>:<sslport>` and accept when prompted to add the certificates to the truststore.

vii. Backup the original cacerts file and copy the new jssecacerts to $TDV_HOME/jre/lib/security/cacerts.

4. When creating the LDAP domain, prefix the url with ldaps:// (i.e. add an “s” to ldap://)

3.3 Tips

- You must be a TDV administrator to add an LDAP domain to TDV.

- The LDAP login must be the full distinguished name (DN) of the user that TDV will use to authenticate to Active Directory. An ordinary, non-administrator user can be used. For example, the login might look like CN=TDV Acct,CN=Users,DC=MyDomain,DC=local.

- If the LDAP login name includes a comma, the comma should be escaped by a preceding “\” character in the LDAP login string.

- Some organizations configure one or more aliases in their internal DNS systems in order to make knowing the specific hostname of any domain controller unnecessary – if such an alias hostname is available in your organization, you will want to use it for the “Server URL” property.

- All of the configuration values in the ldap.properties file are relative to the domain information provided in the Server URL.

- To address the fact that TDV users can be located in more than one area within LDAP, use the pipe character “|” to indicate the logical or condition.

- When defining search contexts for users and groups, TDV searches recursively starting at the parent container so it is not necessary to explicitly specify children containers.

- It is not necessary to restart TDV for changes to the ldap.properties file to take effect.

- LDAP users that are members of the group(s) granted access to TDV will not exist until that user has authenticated to TDV for the first time.

- If there are many LDAP groups, this can result in lengthy delays populating the Add External Groups dialog box and potentially hundreds of pages to scroll through. There are two possible solutions: filtering the results listed in the dialog box, or configuring a search filter in the ldap.properties file.
4 Integrated Authentication with SQL Server

4.1 Ingredients
- TIBCO Data Virtualization 7.0.0 or later
- SQL Server 2005 or 2008 data source
- jTDS JDBC Driver

4.2 Recipe
Integrated authentication will allow TDV to use the credentials of a connected user when connecting to a SQL Server database.

1. Create a new data source adapter.
   a. Create a new data source adapter that inherits from a TDV SQL Server data source adapter.
   b. Add the jTDS driver to the new adapter’s configuration folder (e.g. $TDV_HOME/conf/adapters/custom/<$ADAPTER_NAME>)
2. Restart the TDV server process.
3. Create a new data source, choosing the newly created adapter from the list of data source types.
4. Add a new JDBC connection property to the data source.
   a. The name of the property is „domain”
   b. The value will be the NT domain of the connecting user
5. Test the connection.
   a. The operation should execute using the supplied user’s credentials on the SQL Server database

4.3 Tips
- jTDS is an open source pure java JDBC 3.0 driver for Microsoft SQL Server and Sybase. It can be downloaded from [http://jtds.sourceforge.net/](http://jtds.sourceforge.net/). jTDS is an open source driver and thus is not supported by TIBCO Support or TIBCO Professional Services.

- Ensure you are running a minimum TDV version of 7.0.0.00.00 on both the Server and Studio.
- Integrated authentication can only be configured to make use of users from a single domain at a time
- A dedicated user account for the TDV server is still required on a SQL Server database that has integrated authentication configured to enable execution of system-initiated operations such as cache refresh operations or data source validation.
5 Security Policies for Web Services

5.1 Ingredients
- TDV version 7.0.0 or later
- A published TDV SOAP-based web service
- Optionally, a custom security policy to import

5.2 Recipe
1. To use a predefined policy:
   a. Within Studio, Expand localhost > policy > security > system.
   b. Determine the policy you want to use.
   c. Open a published SOAP web service and navigate to the SOAP tab.
   d. Choose the policy from the list of options.

2. To create a custom security policy:
   a. Within Studio, expand localhost > policy > security > user.
   b. Right-click on user and select New > New Security Policy
c. Type a name for the security policy.

d. The security policy editor opens in the workspace pane.
e. Type or import your security policy

f. Edit your policy if necessary

g. Save your changes

5.3 Tips
- Ensure the policy conforms to XML standards.
6 SAML Validator

6.1 Ingredients
- A published TDV web service
- X.509 certificate (to validate the key used to sign the token)
- SAML-specific security policy
- SAML Validator
- SAML Subject Mapper

6.2 Recipe
TDV is able to validate SAML tokens and map SAML Subjects to TDV users. To do so, you must configure a SAML Validator, Subject Mapper, and web service. There are several steps to processing a SAML token and verifying that it is acceptable. Those steps are:

a) Validate the timestamp of the token.

This functionality ensures that an expired token is rejected. This check is performed by TDV out of the box.

b) Validate the key used to sign the token.

This function ensures that the public X.509 certificate matches the private key. Installation of the key is required in the TDV Server trust store. This check is performed by TDV out of the box.

c) Validate the signature of the token.

This function ensures that the certificate was not altered after it was signed. This check is performed by TDV out of the box.

d) Validate the audience of the token.

This check will ensure that a token is being presented to a system that can accept and process the token. The list of audience members is set and processed by a custom SAML validator.

e) Validate the issuer of the token.

This check will ensure that a token being processed comes from a trusted issuer. The list of issuers is set and processed by a custom SAML Validator.

This validation requires a mixture of core TDV functionality and custom java programming. The following steps outline the process:

1. Import the certificate to the TDV truststore.
   a. On the TDV server machine, copy the certificate into the $TDV_HOME/conf/server/security folder.
b. Open a command line window and navigate to this same folder. Issue a command similar to the following to import the certificate: `keytool -import -file $CERT_NAME.cer -keystore cis_server_truststore.jks -storepass $PASSWORD` When asked whether to trust the certificate, type „y“ and press enter.

2. Repeat the process to import into the strong trust store:
   a. Open a command line window and navigate to this same folder. Issue a command similar to the following to import the certificate: `keytool -import -file $CERT_NAME.cer -keystore cis_server_truststore_strong.jks -storepass $PASSWORD` When asked whether to trust the certificate, type „y“ and press enter.

3. Create or identify a TDV user account for use by the web service.

4. Create a SAML Validator using Java code.

5. Create a Subject Mapper using Java code.

6. The SAML Validator, Subject Mapper, and any supporting classes need to be bundled into a jar file and delivered to the TDV server. This jar file, along with any other required external classes and jar files, must be copied to the `$TDV_HOME/app/server/lib/` directory.

7. Restart the TDV server process.

8. Generate and publish a web service.

9. Right click and open the web service and view the properties named Security Policy, SAML Validator class, and Subject Mapper class.
   a. Select a SAML-specific security policy
   b. Enter a fully qualified java class name of a custom SAML Validator
   c. Enter a fully qualified java class name of a custom SAML Subject Mapper

These items tell TDV to perform core SAML token validation as outlined in a, b, and c above, and then to pass the SAML token to the custom SAML Validator Class for d and e above. The final step is to map the SAML Subject to a TDV user, which is performed by the Subject Mapper class.
10. Test your web service

6.3 Tips
- Common exceptions are noted during execution and can be used to tell the caller what is wrong with their security token. Common exceptions include:
  - Bad Timestamp
    A bad timestamp is where the SAML token generated has expired. More concretely, the current time is outside the expiration time of the token, even after adding a clock skew (i.e. the amount of time to account for clocks being out of sync.) The exception message received in this case is:
    “The creation time is older than currenttime - timestamp-freshness-limit - max-clock-skew”
  - Bad certificate
    A bad certificate may be evidence that the X509 section of the security token has been faked or tampered with. In this case, the exception message received is:
    “Validation of self signed certificate failed”
  - Bad Signature
    Each SAML token is signed with a checksum signature that can be used to validate if the token has been modified at all. If there is modification of the SAML token, or if the token has been faked, the signature will not match. The exception message in this case will be:
    “WSS1614: The signature in the SAML Assertion is invalid”
• You can disable the stack trace from SAML exception messages if you are concerned with giving away any information about what error happened to the end user in case that information could be used to hack into the web service by bypassing some of the validation code by setting the following option in your server command:

```-D com.sun.xml.ws.fault.SOAPFaultBuilder.disableCaptureStackTrace=false```

• TDV supports SAML 1.1.

• Built-in SAML support includes:
  
  o Saml1.1-Bearer-Wss1.1.xml scheme for defining the Method in which the bearer assertion is used to facilitate single sign on in a web browser.

  o Saml1.1-HolderOfKey-Wss1.0.xml scheme for defining the Method that establishes a correspondence between a SOAP message and the SAML assertions added to the SOAP message.

  o Saml1.1-SenderVouches-Wss1.1.xml scheme for the Subject-confirmation method that enables an attesting entity to vouch for the identity of a subject to a party that trusts the sender.
7 Kerberos Single Sign-On With AES256 Encryption and TDV (on Linux)

7.1 Ingredients
- TDV 7.0.0 or later on Linux, with TDV Already configured to use Active Directory
- Kerberos V5 KDC Server on Linux
- Kerberos configuration for TDV
- Active Directory on Windows Server 2008 R2 with SP1 or later
- ODBC connection configuration on supported Windows client

7.2 Recipe
The steps in this section all refer to placeholder settings that will need to be swapped for actual values applicable to the implementation being worked on:

<table>
<thead>
<tr>
<th>Placeholder</th>
<th>Description</th>
<th>Sample value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;dv.server.name&gt;</td>
<td>Name of the server where TDV is installed</td>
<td>server01</td>
</tr>
<tr>
<td>&lt;dv.full.server.name&gt;</td>
<td>Full DNS server name where TDV is installed.</td>
<td>server01.company.com</td>
</tr>
<tr>
<td>&lt;TDV_HOME&gt;</td>
<td>Full path to where TDV is installed on the server</td>
<td>/opt/TIBCO/TDV_Server_7.0</td>
</tr>
<tr>
<td>&lt;dv.service.name&gt;</td>
<td>Name of the TDV service as it will be referred to when creating the SPN associated with the AD service user during the keytab file creation. Good practice is to use some reference to the environment</td>
<td>tdvdev</td>
</tr>
<tr>
<td>&lt;ad.domain.name&gt;</td>
<td>Domain name as defined in Active Directory</td>
<td>dev.company.local</td>
</tr>
<tr>
<td>&lt;ad.dc.server.name&gt;</td>
<td>Name of the Active Directory Domain Controller server</td>
<td>ldap01</td>
</tr>
<tr>
<td>&lt;ad.service.user&gt;</td>
<td>Name of a user account in Active Directory associated with the service user used in TDV Kerberos authentication</td>
<td>svc_acct</td>
</tr>
<tr>
<td>&lt;ad.service.user.pw&gt;</td>
<td>Password associated with Active Directory user</td>
<td>passwordABC</td>
</tr>
<tr>
<td>&lt;kerberos.realm&gt;</td>
<td>Name of the realm to be configured in the Kerberos KDC</td>
<td>DEV.COMPANY.LOCAL</td>
</tr>
<tr>
<td>&lt;keytab.file&gt;</td>
<td>Name of the keytab file that is generated with the ktpass utility</td>
<td>httpservice.keytab</td>
</tr>
</tbody>
</table>

1. Active Directory setup
a. Create a user in Active Directory that will be used to service TDV Kerberos authentication requests, this user will be referred to as <ad.service.user>. This user needs to be set up so that DES encryption types cannot be used for this account but also so that AES128 and AES256 will be supported. To achieve this, set the checkboxes for the user in Active Directory as per the following screenshot:

![Active Directory screenshot showing checkbox settings for DES encryption]

b. Generate a keytab file for the service:

```
ktpass -princ HTTP/<tdv.service.name>.<ad.domain.name>@<kerberos.realm> -mapuser <ad.service.user> -pass <ad.service.user.pw> -crypto AES256-SHA1 -pType KRB5_NT_PRINCIPAL -out <keytab.file>
```

Example:

```
ktpass -princ HTTP/tdvdev.dev.company.local@DEV.COMPANY.LOCAL -mapuser dev\svc_acct -pass <password> -crypto AES256-SHA1 -pType KRB5_NT_PRINCIPAL -out httpservice.keytab
```

c. Copy the generated keytab file from the Active Directory server to a suitable directory on the server where TDV is installed (e.g. `<TDV_HOME>/keytab.file`).

2. TDV server set up.

a. In order to use the AES256 encryption type, you will need to install the JCE crypto policy with the unlimited version into the TDV native JRE on the server where TDV is installed (see [http://docs.oracle.com/javase/7/docs/technotes/guides/security/jgss/jgss-features.html](http://docs.oracle.com/javase/7/docs/technotes/guides/security/jgss/jgss-features.html) for more information).

ii. Extract the zip file to your desktop, this will create a folder called UnlimitedJCEPolicy. The folder will contain the following files:

- README.txt
- local_policy.jar
- US_export_policy.jar

iii. Copy the above 3 files (NOT including the folder) into `<TDV_HOME>/jre/lib/security` on the machine identified by `<tdv.server.name>`.

b. Assuming Kerberos V5 KDC Server has already been installed onto the Linux server, a configuration file called `krb5.conf` should exist. The default location for this file is `/etc/krb5.conf`. Below is the exact content that should be placed into the file (log file locations can be altered as appropriate).

```
[logging]
default = FILE:/var/log/krb5libs.log
dns_lookup_realm = true
dns_lookup_kdc = true
ticket_lifetime = 24h
forwardable = yes
default_tgs_enctypes = aes256-cts
default_tkt_enctypes = aes256-cts
permitted_enctypes = aes256-cts

[libdefaults]
default_realm = <kerberos.realm>
dns_lookup_realm = true
dns_lookup_kdc = true
ticket_lifetime = 24h
forwardable = yes
default_tgs_enctypes = aes256-cts
default_tkt_enctypes = aes256-cts
permitted_enctypes = aes256-cts

[realms]
<kerberos.realm> = {
    kdc = <ad.dc.server.name>.<ad.domain.name>
    admin_server = <ad.dc.server.name>.<ad.domain.name>
    default_domain = <ad.domain.name>
}

[domain_realm]
<ad.domain.name> = <kerberos.realm>
.<ad.domain.name> = <kerberos.realm>

[appdefaults]
pam = {
    debug = false
ticket_lifetime = 36000
renew_lifetime = 36000
forwardable = true
krb4_convert = false
}
c. Alter `<TDV_HOME>/conf/server/ldap.properties` to include the location of the `<ad.service.user>` service account user. At this point we are assuming TDV has already been configured to work with Active Directory and thus this file should already contain entries for locating the TDV users, but is unlikely to contain an entry for the service account. We are also assuming the domain name set up in TDV has the same name as the actual AD domain. Below is an example snippet of an ldap.properties file, items in red are an example of entries to locate TDV users, added when setting up TDV to use AD; items in blue are an example of entries to locate the service account:

```
activedirectory.all.users.search.context=ou=Users,ou=Company|ou=_System
activedirectory.all.users.filter=(objectclass=user)
activedirectory.all.users.username.attribute=samaccountname
activedirectory.all.users.search.timeout=0

activedirectory.user.username.comparison.is.case.sensitive=false
activedirectory.user.search.context=ou=Users,ou=Company|ou=_System
activedirectory.user.filter=(samaccountname=USERNAME)(objectclass=user)
activedirectory.user.username.attribute=samaccountname
activedirectory.user.search.timeout=1000
```

d. Various configuration changes need to be made to the TDV server. The easiest way to do this is to log into TDV Studio as the admin user and do the following:

- Select Configuration from the Administration menu.
- In the tree pane, navigate to `Server->Configuration->Security->Authentication`.
- Select the Windows Domain Mapping setting.
- In the right-hand pane select “Add Key Value Pair” and enter a Key of `<kerberos.realm>` and a Key Value of `<ad.domain.name>`
- In the tree pane, navigate to `Server->Configuration->Security->Authentication->Kerberos`.
- Make the following changes to the configuration items beneath the Kerberos node:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Action and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow Kerberos</td>
<td>Change this value to true. A warning is presented to avoid inadvertent changes made without full implementation.</td>
</tr>
<tr>
<td>Debug Output Kerberos</td>
<td>When set to true, TDV writes the JDK’s Kerberos implementation output messages to a cs_server_client.out file in the logs directory.</td>
</tr>
<tr>
<td>KeyTab File</td>
<td>Enter the value that points to the generated keytab file on the TDV Server (e.g. <code>&lt;TDV_HOME&gt;/keytab.file</code>).</td>
</tr>
</tbody>
</table>
Kerberos Configuration File  Enter the value that points to the krb5.conf file on the server hosting Kerberos (e.g. /etc/krb5.conf).

Required Principal Name  Enter the SPN value established by invocation of the Kerberos setspn utility. TDV must know the SPN to address the Kerberos domain server (e.g. HTTP/<tdv.service.name>.<ad.domain.name>@<kerberos.realm>)

5. Restart TDV

3. Configure a TDV ODBC data source (this assumes the TDV ODBC driver has been successfully installed on a Windows 7 or Windows Vista client).

   a. Configure a DSN. In the ODBC Manager configuration window, enter the following values for each setting:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Action and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSN Name</td>
<td>Set this to the name you wish to use to identify this ODBC connection (e.g. kerberosConnection).</td>
</tr>
<tr>
<td>TDV Host</td>
<td>Set this to the name of the host where TDV is installed (e.g. &lt;tdv.full.server.name&gt;).</td>
</tr>
<tr>
<td>Port</td>
<td>Set this to the TDV ODBC port (e.g. 9401).</td>
</tr>
<tr>
<td>Integrated Authentication</td>
<td>Select Kerberos. This will disable the User Name, Password and Domain fields.</td>
</tr>
<tr>
<td>Kerberos SPN</td>
<td>Set this to the SPN associated with the service account to be used for Kerberos Single Sign On (e.g. HTTP/&lt;tdv.service.name&gt;.&lt;ad.domain.name&gt;@&lt;kerberos.realm&gt;)</td>
</tr>
<tr>
<td>Datasource</td>
<td>Set this to the name of a published TDV data source</td>
</tr>
</tbody>
</table>

7.3 Tips

   • Ensure you are running a minimum TDV version of 7.0.0.00.00 on both the Server and Studio.

   • Sample problems with solutions:

     o Problem: Error message “KDC has no support for encryption type (14)” displayed either when running the TIBCO development tool “TestKerberos.jar” or in the cs_server_client.log file.

       Solution: Ensure the service user has been set up as per part 1b of this recipe (AD service user setup)

     o Problem: Error message “Unable to obtain password from user” displayed on screen when attempting ODBC configuration and also in the cs_server_client.log file. This started appearing after making the changes described in part 1b. of this recipe.
Solution: Restart TDV after making changes to the service user.

- Problem: Error message “Client not found in Kerberos database (6)” displayed on screen when attempting ODBC configuration and also in the cs_server_client.log file. This occurs because Kerberos cannot find a unique user in AD whose userPrincipalName (UPN) matches the principal specified in the keytab file. This can occur if no users with that UPN are found or if multiple users with that UPN are found.

  Solution: Ensure EXACTLY one user in AD contains a UPN that matches the principal specified in the keytab file. The AD user must be the service user.

- Problem: As a result of the service user password changing in AD, the following error message gets displayed on screen when attempting ODBC configuration: “Failure unspecified at GSS-API level (Mechanism level: Specified version of key is not available (44))”

  Solution: The keytab file needs regenerating. The problem occurs because changing the password increments the KVNO value in AD and this makes the KVNO in the keytab file out of step with the KNVO in AD.
8 TDV Studio and SSO with Kerberos

8.1 Ingredients
- TDV version 7.0.0 or later
- TDV Studio patched to the same level as the TDV Server
- Administrative access on the PC that Studio is running on
- Environment-specific krb5.ini file
- The SPN associated with the TDV Server Kerberos setup

8.2 Recipe
1. TDV server setup.
   a. Verify TDV is correctly setup for Kerberos authentication. TDV Kerberos authentication setup involves first setting up an external LDAP Active Directory domain, creating an AD user and associating a SPN, updating the user properties for proper Kerberos encryption algorithm, generating a tab file and installing it on TDV server, updating local_policy.jar, installing a krb5.conf file etc. To verify proper configuration of TDV Kerberos authentication, create an ODBC DSN and verify it can connect to the TDV server with Kerberos Integrated authentication using a SPN and without using a User Name/Password.

2. Copy Kerberos configuration file to the client.
   a. The Kerberos configuration file (krb5.ini on a Windows platform, krb5.conf on other platforms) should be copied from the TDV server and placed on the client where the Studio runs. It should be named krb5.ini and copied to C:\Windows\. If it exists, set it to 1. If it does not exist, create a REG_DWORD type value and set it to 1.

3. Modify Windows registry to enable TGT Session Keys.
   a. In the Windows registry, check if a value named allowtgsessionkey exists at HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\Kerberos\Parameters.
      If it exists, set it to 1. If it does not exist, create a REG_DWORD type value and set it to 1.

4. Create the krb5.properties file.
   a. Using the sample krb5_sample.properties in ${TDV_STUDIO_HOME}\conf\studio\ create krb5.properties in ${TDV_STUDIO_HOME}\conf\studio\. Uncomment all the properties in the Default User section and set the spn property based on the SPN the TDV server was configured.
      
      **Important: The format of the SPN here is different from the TDV Server.**

      The TDV Server SPN has the following generic form:

      HTTP/FullyQualified_TDVServicename@realm

      TDV Studio instances that connect to the TDV will have a SPN in the form:
HTTP@FullyQualified_TDV_ServiceName

As an example, if the TDV Server SPN is
HTTP/dev-krb5-win.support.net@SUPPORT.NET

Then the SPN property for TDV Studio in the krb5.properties file should be
HTTP@dev-krb5-win.support.net

5. Running Studio for SSO. When you launch Studio you’ll now see a new checkbox called SSO. Check this box and Click Connect to sign on.

8.3 Tips
• There is an issue with Windows Vista/7 that may cause Single Sign On to fail if all of the following conditions are true:
  o Studio is running on Windows Vista or Windows 7.
  o User executing Studio has administrative privileges.
  o User Access Control (UAC) is enabled.

The workaround is to execute Studio using the “Run as Administrator” option in Windows which can be accessed by right clicking on the Studio shortcut icon.

• If you get an error when connecting to TDV with SSO, check the exception stack trace. If the source of the exception is “java.io.IOException: FULL head”, You need to configure a larger header buffer size. The default size is 4096. The setting is available in the Configuration dialog in Studio at
Server > Web Services Interface > Communications > HTTP > Header Buffer Size

and

Server > Web Services Interface > Communications > HTTPS > Header Buffer Size

The configuration value should be incremented by 4096 until you no longer get this error. TDV should be restarted each time the configuration value is changed.

- To verify that TDV is properly configured for Kerberos authentication. You can verify this by making an ODBC SSO connection. Support/Engineering also has a Java testing tool called TestKerberos. You can contact Support to get access to this tool. This will verify that TDV is correctly configured for Kerberos authentication and you have a good krb5.ini file.

- Problem: You get an error with an Exception “No CallbackHandler available to garner authentication information from the user”
  
  Solution: Ensure you have the right SPN in the correct format in the krb5.properties file. Also ensure all the properties in the krb5.properties have been set in the default user section.

- Problem: You get an error with Exception “java.security.PrivilegedActionException: GSSException: No valid credentials provided (Mechanism level: No valid credentials provided (Mechanism level: Fail to create credential. (63) - No service creds))”
  
  Solution: Make sure you have added the allowtgsessionkey registry value to 1. Ensure it is of type REG_DWORD and it is set to 1.

- Problem: You get an Exception “java.security.PrivilegedActionException: GSSException: No valid credentials provided (Mechanism level: No valid credentials provided (Mechanism level: kdc.domain.com)). The root of this exception is “Caused by: java.net.UnknownHostException: kdc.domain.com”
  
  Solution: Ensure you have the right krb5.ini file. Ensure you have network connectivity to the kdc server of your realm.
9 TDV with SSL (CA Signed Certificate) – Server Side Configuration

9.1 Ingredients
- TDV Server 7.0.0 or later
- Certificate signed by a certificate authority (CA)
- The CA root certificate which trusts the signed certificate (most major browsers are preinstalled with the most commonly used root certificates, but this can be obtained from the CA that signed your certificate), as well as any intermediate certificates if applicable

9.2 Recipe
There are multiple scenarios that are possible using TDV and certificates, including connecting from various clients and protocols (e.g. Studio, jdbc, web services, etc.) as well as connecting to data sources which require SSL. This recipe focuses on the configuration necessary on the server side, beginning with acquiring a certificate using a certificate request. Depending on your target scenario, perform the steps applicable and be aware that further configuration may be necessary (e.g. client side set up).

1. Obtain a certificate from the CA.
   a. Create a certificate keystore and a key pair (a public key and an associated private key). You can use the java keytool utility to perform this action:
b. Create a certificate signing request (CSR). You can use the java keytool utility to perform this action:

```
C:\Program Files\TIBCO\TDV Server 7.0\jre\bin>keytool -genkeypair -keystore RSA -alias casigned -keyalg RSA -pass password -keystore C:\Certs\Verisign\verisign.jks -storepass password -keysize 2048 -validity 305
What is your first and last name? 
[Unknown]: training.tibco.com
What is the name of your organizational unit?  
[Unknown]: Training
What is the name of your organization?  
[Unknown]: TIBCO Software
What is the name of your City or Locality?  
[Unknown]: Palo Alto
What is the name of your State or Province?  
[Unknown]: California
What is the two-letter country code for this unit?
[Unknown]: US
Is CN=training.tibco.com, OU=Training, O=TIBCO Software, L=Palo Alto, ST=California, C=US correct?
[no]: yes
Enter key password for <casigned>
(RETURN if same as keystore password):
```

You will be prompted for the password of the keystore that you created earlier. If your key and keystore passwords are different, ensure you enter the password for the keystore and not the key.

c. Open the CSR using a text editor and copy the contents onto the form at the CA’s website when prompted for a CSR.

d. Receive the certificate from the CA and generate a certificate file. Often, it is sent via email in plain text. Copy the certificate into a file and save with a .cer file extension.

1. If applicable, do the same for any root and intermediary certificates that you need (e.g. if you are using a trial certificate from Verisign). In many cases, you will already have the root certificate for the most common CAs.

2. Install any root and intermediate certificates into the keystore, if applicable.

   a. You can use the java keytool utility to perform this action:
The option `-trustcacerts` adds the certificate to the keystore as a trusted certificate. It is called a trusted certificate because the keystore owner trusts the public key in the certificate belongs to the identity of the subject (owner) of the certificate. The issuer of the certificate vouches for this by signing the certificate.

3. Install the CA issued certificate into the keystore.
   a. You can use the java keytool utility to perform this action:

   ```
   Command Prompt
   C:\Program Files\TIBCO\TDV Server 7.0\jre\bin\keytool -import -trustcacerts -alias casigned -keystore C:\CACerts\Verisign\verisign-trialIntermediate\keystore
   Enter keystore password:
   Certificate reply was installed in keystore
   ```

   The alias must be identical to the alias we used when originally creating the keystore. This is because we are replacing our self-signed certificate with the certificate signed by the CA.

4. We need to export the signed certificate from the keystore and import it into the existing truststore that the TDV server is using. These instructions assume this is the out-of-the-box cis_server_truststore_strong.jks truststore, but you can equally create your own brand new truststore if required.
   a. Export the signed certificate from the keystore created earlier. You can use the java keytool utility to perform this action:

   ```
   Command Prompt
   C:\Program Files\TIBCO\TDV Server 7.0\jre\bin\keytool -export -alias casigned -keystore C:\CACerts\Verisign\verisign-trialIntermediate\keystore
   Enter keystore password:
   Certificate stored in file C:\CACerts\Verisign\InstalledCert.cer
   ```

   In this example, the certificate is exported to the file C:\CACerts\Verisign\InstalledCert.cer.
b. Import the signed certificate as a trusted certificate. You can use the java keytool utility to perform this action:

```
C:\Program Files\TIBCO\TDV Server 7.0\jre\bin\keytool -import -alias casigned -keystore "C:\Program Files\TIBCO\TDV Server 7.0\conf\server\security\cis_server_truststore_strong.jks" -trustcacerts -file "C:\ACerts\Verisign\installedCert.cer"
Enter keystore password: Re-enter new password: Certificate was added to keystore
```

This added our signed certificate to the file where we store trusted certificates. This type of keystore is known as a truststore. The keytool command will prompt for a new password for the truststore, but the password need not change. The generated truststore in the example is named cis_server_truststore_strong.jks.

5. Optionally, create a new truststore that contains the root certificate of the CA. This is required when you want to access a TDV published web service over SSL from another TDV instance. That new instance of TDV needs the root certificate of the CA that signed the server certificate. This exactly mirrors the scenario where a web browser has pre-installed certificates from various CAs as trusted authorities.

a. Create a new truststore and import only the root certificate as a trusted certificate. You can use the java keytool utility to perform this action:

```
C:\Program Files\TIBCO\TDV Server 7.0\jre\bin\keytool -import -alias root -keystore C:\ACerts\Verisign\clientTruststore.jks -file C:\ACerts\Verisign\VerisignTrialRoot.cer
```

You will be prompted to trust the certificate that is presented, enter yes. The certificate will then get added to the truststore.

6. Configure TDV to use the CA issued certificate by referencing the previously created keystore.

a. In Studio, use the Administration menu to open the Configuration window. Expand Server > Communications and change the entry “Strong Keystore Alias (On Server Restart)” to the alias of our certificate in our keystore (we used “casigned” in the screenshot examples).
b. Change the entry “Strong Keystore File Location (On Server Restart)” to the keystore we created earlier (we used the value “C:\CACerts\Verisign\verisign.jks” in the screenshot examples). Ensure you use the keystore and not the truststore we optionally created in step 5.
c. Change the entry “Strong Keystore Password (On Server Restart)” to the password used when creating the keystore.

![Configuration screenshot showing configuration settings]

**Properties**
- **Name**: Strong Keystore Password (On Server Restart)
  - **Type**: Password
  - **Value**: *********
  - **Description**: The password of the keystore file and the entries within it. All password protected entries in the keystore file must use the same password as the file itself. Changing this value will have no effect until the next server restart. This value is locally defined. It will not be altered when restoring a backup and will not be replicated in a cluster. This setting will only be active if a server license with strong encryption is enabled. A strong keystore is a keystore file that has key entries with more than 512 bits.

b. The setting “Strong Keystore File Type (On Server Restart)” remains unchanged as “JKS”. PKCS12 key stores are supported as well.

7. Configure TDV to use the CA issued certificate by updating the “Strong Truststore File Location (On Server Restart)” setting. If the certificate was added to the out-of-the-box truststore in step 4, then likely no change is necessary and the value should be automatically set to the correct value as per the below screenshot (your install path may differ of course). Alternatively, if a new truststore was created at step 4 then set this value as path to the newly created truststore.

a. If required, change the entry “Strong Truststore File Location (On Server Restart)” to the correct truststore.
b. The setting “Strong Truststore File Type (On Server Restart)” remains unchanged as “JKS”. PKCS12 key stores are supported as well.

8. If you will be connecting to external data sources that require X.509 (mutual) authentication, then you will need to change the setting “Enable Client Authentication” to true.

9. Restart the TDV Server process.
Remember that these changes take effect upon restart, and also that configuration for the Studio client is required before it is able to make a secure connection to TDV.

### 9.3 Tips

- A common error when importing any of your certificates is “Input not an X.509 certificate”. The most common reason for this error is that the pasted certificate text is in an incorrect format in the certificate file. Ensure there are no extra leading or trailing spaces in any lines of the certificate text and the contents have been copied and pasted entirely. Also be sure there are no blank lines at the beginning or end of the pasted contents of the certificate.

- It is recommended that you take a backup of the file `$TDV_HOME/conf/server/server_values.xml` before making any configuration changes. If a failure occurs upon server restart (or if the server starts but you are unable to access the server from Studio) after making changes to the keystore and truststore references, these configuration values can be modified in that file or it can be restored to its original state. This should allow both the TDV Server and Studio to function normally again.

- TDV does not currently support different passwords for the keystore and an individual certificate. Please ensure you use the same key for both.

- Keep in mind that this only covers server side configuration and you will want to perform the appropriate client side configuration, as applicable.
10 TDV with SSL (CA Signed Certificate) – Connecting with Studio

10.1 Ingredients
- TDV Server 7.0.0 or later, with completed server-side certificate configuration
- TDV Studio 7.0.0 or later
- Root certificate from CA

10.2 Recipe
This recipe assumes server side configuration has been performed and the aim is to configure the TDV Studio client for secure access to TDV.

1. Configure TDV Studio for secure access to TDV Server.
   a. Import the root certificate into the cis_studio_truststore_strong.jks file. You can use the java keytool utility to perform this action:

   ![Command Prompt](image)

   You will be prompted to trust the certificate that is presented, enter `yes`. The certificate will then get added to the Studio truststore.

2. Log into the TDV Server from TDV Studio.
   a. Check the “encrypt” checkbox on the Studio login dialog for a secure connection.
**Please note:** When you establish a secure connection to Studio, a padlock icon appears on bottom right panel of your Studio window.

### 10.3 Tips

- The password for `cis_studio_truststore_strong.jks` is set to “changeit” by default. Also, if the root certificate is not imported into the studio truststore, the SSL handshake will not complete and you will be unable to login. This is very similar to how web browsers function. Many root certificates from common CA’s are installed by default in most web browsers.
11 TDV with SSL (CA Signed Certificate) – Accessing a Published TDV Web Service from TDV

11.1 Ingredients
- TDV version 7.0.0 or later with appropriate server-side certificate configuration and published web service
- A second TDV version 7.0.0 or later instance that will introspect the web service.
- Truststore with a trusted root certificate from CA

11.2 Recipe
This recipe will demonstrate using a web service using SSL. The web service is published from a TDV instance, and will be introspected by a second TDV instance.

1. Within TDV Studio, create a new data source of type REST or WSDL.

![New Physical Data Source](image)

2. Fill in all the relevant fields on the Basic tab, including adding any operations. Click on the Advanced tab and import the truststore containing the root certificate, as shown below. Be sure to enter the password associated with the truststore. Then proceed through the introspection process as normal.
3. Open the applicable web service operation and test it by executing it in Studio
11.3 Tips

- This assumes that you have completed the optional step of creating a new truststore that contains the root certificate of the CA as part of the configuration of the TDV server that is publishing the web service. This is required when you want to access a TDV published web service over SSL from a second TDV instance. The second instance of TDV needs the root certificate of the CA that signed the server certificate. This exactly mirrors the scenario where a web browser has pre-installed certificates from various CA’s as trusted authorities.
12 TDV with SSL (CA Signed Certificate) – Accessing a WSDL Data Source over SSL

12.1 Ingredients

- TDV version 7.0.0 or later with appropriate server-side certificate configuration and published web service
- A published web service that supports SSL
- Keystore file containing the certificate for the data source

12.2 Recipe

This recipe will demonstrate introspecting a WSDL data source that requires an SSL connection for interaction.

1. Within Studio, create a new data source of type WSDL.

2. In the basic tab enter the path to the WSDL and any authentication information necessary.
3. If the remote web service uses NTLM or Kerberos-based authentication, then you must change the value of the Authentication setting on the Basic tab of the New Physical Data Source dialog shown above.
   
a. For NTLM authentication change the property value to NTLM.
   
b. For Kerberos, change the setting to Negotiate.

4. Switch to the Advanced tab and click on the Import Certificate Key Store from File button to import a key store file.
5. In the prompt that appears, enter the following information

   a. Key Store File Path: The path to the key store file containing the certificate for the data source.

   b. Key Store Type: JKS or PKCS12 as appropriate.

   c. Key Store Password: The password of the key store you selected

   d. Click OK to add the key store to the data source configuration. Repeat as appropriate to add any additional key stores to the data source.
6. Click on the Create & Introspect button to introspect the data source.

7. Select the operations to introspect from the web service and complete the introspection process.

   **Please Note:** If you are not able to successfully see operations from the WSDL during this step, you should validate that TDV is able to access the schema xsd file referenced in the schemaLocation attribute of the wsdl.

8. Save the data source.

9. Execute one of the introspected web service operations to confirm that TDV is able to call it correctly.

**12.3 Tips**

- SSL authentication can fail if the keystore file password and the password for all of the individual keys in the keystore are not the same. All keys that have been imported into the TDV keystore must have the same password for SSL configuration to work correctly. This may be the case if SSL authentication is failing and the following error is in the cs_server.log file:

```
Caused by: com.compositesw.common.security.CompositeSecurityException: The keystore could not loaded. [security-1900704]
at com.compositesw.common.security.KeyStoreUtil.loadKeyStore(KeyStoreUtil.java:246)
at com.compositesw.common.security.KeyStoreUtil.loadKeyStore(KeyStoreUtil.java:254)
at com.compositesw.server.security.KeyStoreUtil.loadKeyStore(KeyStoreUtil.java:684)
```

© Copyright TIBCO Software Inc.
TDV Authentication and Security Scenarios

at com.compositesw.server.security.KeyStoreUtil.loadDataSourceTrustStore(KeyStoreUtil.java:502)
at com.compositesw.server.security.KeyStoreUtil.loadEffectiveDatasourceKeystores(KeyStoreUtil.java:780)
at com.compositesw.cdms.ds.ws.WsConnection.configureClient(WsConnection.java:1226)
... 12 more
Caused by: java.io.IOException: Keystore was tampered with, or password was incorrect
at sun.security.provider.JavaKeyStore.engineLoad(JavaKeyStore.java:771)
at sun.security.provider.JavaKeyStore$JKS.engineLoad(JavaKeyStore.java:38)
at java.security.KeyStore.load(KeyStore.java:1185)
at com.compositesw.common.security.KeyStoreUtil.loadKeyStore(KeyStoreUtil.java:244)
... 17 more
Caused by: java.security.UnrecoverableKeyException: Password verification failed
at sun.security.provider.JavaKeyStore.engineLoad(JavaKeyStore.java:769)
... 20 more

• If you see the following error message with a stack trace similar to the following when sending a request to an external WSDL data source:

<Sep 6, 2017 3:35:02 PM EDT> <Error> <CSSecurityProviders.LDAPAtnLoginModuleImpl> <BEA-000000> <Authentication Failed for User ssltest javax.security.auth.login.AccountNotFoundException: User 'ssltest' has no password
at com.csg.cs.security.wls.authentication.ldif.LDIFDelegate.authenticate(LDIFDelegate.java:139)
at com.bea.common.security.internal.service.LoginModuleWrapper$1.run(LoginModuleWrapper.java:110)
at java.security.AccessController.doPrivileged(Native Method)
at com.bea.common.security.internal.service.LoginModuleWrapper.login(LoginModuleWrapper.java:106)

and an error message similar to the following in the cs_server.log:

INFO [Stored Procedure Scan initialize (http-17)] 2017-09-06 15:35:02.300 -0400
AuthChallengeProcessor - basic authentication scheme selected
INFO [Stored Procedure Scan initialize (http-17)] 2017-09-06 15:35:02.300 -0400
HttpMethodDirector - Failure authenticating with BASIC 'WebLogic Server'@server:34000

The exception may indicate that the user account sent by TDV is not recognized by the remote web service. The cs_server.log entry indicates that your WSDL data source is configured to use basic authentication, but the remote data source requires either NTLM or Kerberos. Consult with your Web service administrator to determine the correct authentication method to use and then modify the Authentication method setting in your WSDL data source configuration as required.

• If the user account sent by TDV is not recognized by the remote web service. The cs_server.log entry indicates that your WSDL data source is configured to use basic authentication, but the remote data source requires either NTLM or Kerberos. Consult with your Web service administrator to determine the correct authentication method to use and then modify the Authentication method setting in your WSDL data source configuration as required.

• If you are getting an HTTP 500 error with the message “the Client Authentication certificate user is not authorized on the external data source server” then the certificate user TDV is attempting to connect with has insufficient security permission on the external data source. Contact the data source administrator to request that they add permissions for the certificate user.