



TIBCO® Data Virtualization

Installation Guide

Version 8.7.0 | October 2023

Contents

Contents	2
Preparing for Installation	8
Software Components for Installation	8
Overview of Installation	9
Tracking and Documenting Your Install or Upgrade	9
Installation Requirements and Support Information	10
Disk Space and Physical Memory Requirements	11
Minimum Installation Requirements	11
Sizing Guidelines for TDV	13
Port Requirements	15
Studio and Server Connectivity and Installer Limitations	18
TDV Supported Platforms	19
Java Support	20
Operating System Support for Studio	20
Operating System Support for Server	21
Support for Virtualization Environments	24
Driver Support	24
SNMP Support	25
Web Service Protocols	26
Directory Services Support for LDAP	27
Options and Features Supported for Use with TDV	28
Kerberos Support	28
Web Browser Support	29
Browsers and Kerberos Support	30
Directory Services Support for Kerberos	30
Supported Data Sources	31

Supported Add-On Adapters	36
Supported Advanced Data Source Adapters	37
Supported Cache Targets	42
Data Ship Source and Target Support	48
TDV DDL Feature Support	51
Supported Client Applications	52
Client Application Target Framework	53
Enterprise Service Buses	54
Client-Side ADO.NET Driver Support	54
Data Sources Supported for Kerberos Token Pass-through	54
Security Features	55
Support and Maintenance Policies for TIBCO Products	56
Limitations for TDV Discovery	57
Installing TDV, Studio, and Drivers	59
Overview of Installation Steps	59
Installation Overview for New TDV Software Customers	60
Installation Overview for Existing Customers Upgrading from a Previous Release	60
Preparing Your System for Installation	61
Preparing UNIX for TDV Installation	63
Preparing Microsoft Windows for TDV Installation	65
Installing on Windows	65
Running the TDV Server Installer	66
Running the Studio Installer	68
Installing the Drivers	68
Installing on UNIX	69
Installing TDV Server on UNIX	69
Installing Drivers on UNIX	72
Setting the TDV Server to Start Automatically on UNIX	73
Installing on Amazon Web Service	74
About TDV Software Patches	75
About the Installed TDV Services	76

Importing Metadata into the New TDV Instance	76
Tips from an Expert if the Server Does Not Start	76
Where to Go After Installation	77
Silent Mode Installation	78
TDV Docker Container	83
Prerequisites	83
Docker	85
TDV	86
Building TDV Docker Images	86
Using Quick Start Script	87
Using Docker Build	88
Publishing TDV Docker Images	89
Launching TDV Containers	89
Launching TDV Containers (Single Node) - Using Quick Start Script	91
Launching TDV Containers (Single Node) - Using Docker Compose	94
Launching TDV Containers (Single Node) - Using Docker Run	95
Launching TDV Containers (Cluster Nodes) - Using Quick Start Script	103
Launching TDV Containers (Cluster) - Using Docker Run	106
Runtime TDV Container Configuration - Common Examples	115
Change TDV Admin Password (while container is running)	116
Change TDV Base Port	116
Change TDV Server Memory Setting	118
Configure External Volume For Local Persisted File Data Sources	119
Configure Data Source With 3rd party JDBC Driver (type 4)	120
Configure Data Source With 3rd party JDBC Driver (type 3)	121
Configure JVM Settings from Docker	123
Configure LDAP Properties From Docker	124
Configure Truststore and Keystore Files From Docker	125
Limitations	125
Upgrading to a New Version of TDV Server	126

Tips from Expert	126
Useful Docker Commands for TDV Containers	127
TDV Container Orchestration Using Kubernetes	129
Introduction to Container Orchestration	130
TDV Container Orchestration Architecture	130
Prerequisites	133
Pre-Configuration of the Runtime Environment (On-Premises/Private Cloud only):	133
Building the Container Image	134
Configuring Kubernetes Using Quick Start Script	135
Deploying TDV as a Single Mode instance	139
Deploying TDV as a Cluster Mode Instance	141
Installing the TDV HAProxy Service using the Quick Start Script	143
Removing a TDV application via Helm	145
Running TDV on a Public Cloud - Microsoft Azure	146
Setting up the Resource Group	146
Running TDV on a Public Cloud - Amazon Web Service	149
Creating a Container Registry	150
Creating a Cluster	151
Deploying the TDV Helm Chart	152
TIBCO Data Virtualization(R) Container Distribution	155
Useful Kubernetes Commands	155
Generic Kubernetes Commands	156
Useful TDV-Specific Commands	156
TDV Deployment Examples	158
TDV for AWS Marketplace	166
Prerequisites	166
Launching TDV Server on AWS Marketplace	166
Launching a TDV Windows Image on AWS Marketplace	167
Launching a TDV Linux Image on AWS Marketplace	169

TDV Server Configuration	171
TDV Instance Id	171
TDV Admin Password	171
Default TDV Security Group Configuration	171
TDV Security Group	172
Data Source Driver Management	175
TDV Updates and Bundled TDV Software	175
Installing Optional TDV Products	177
Version Support	177
Installation Requirements	177
Add-On Adapter Installation Requirements	177
Active Cluster Installation Requirements	178
Installing an Optional TDV Product	179
Installing the Advanced Adapters	180
Auto Deployment	180
Manual Deployment	180
Installing the TDV Client Drivers that are Distributed with TDV	182
Importing Resources Defined in an Earlier Release	183
Manage Active Cluster Security	183
Updating the Digital Certificate to Secure Cluster Communication	183
Set Access Privileges	184
TDV and Business Directory Product Maintenance	185
Upgrade, Downgrade, and Rollback	185
About Hotfix Maintenance	186
Applying the Hotfix to TDV Server, Studio, and Business Directory	186
About Service Pack Maintenance	187
Applying the Service Pack to TDV Server, Studio, and Business Directory	187
Applying the Service Pack or Hotfix to Active Cluster	188
Upgrading from an Earlier Release and Migrating The Metadata	189
Downgrade/Rollback	194

Maintaining TDV-to-Client Application Connections	198
Updating an ODBC Client Application	199
Updating a JDBC Client Application	199
Uninstalling TDV	200
Uninstalling TDV on Windows	200
Uninstalling TDV on UNIX	200
Preparing for Uninstalling on UNIX	201
Uninstalling TDV On UNIX	201
TIBCO Documentation and Support Services	202
Legal and Third-Party Notices	204

Preparing for Installation

Read the sections described in this chapter, prior to installing the Data Virtualization Platform:

- [Software Components for Installation](#)
- [Overview of Installation](#)
- [Tracking and Documenting Your Install or Upgrade](#)

Software Components for Installation

TDV provides the following installers for the Data Virtualization software components:

Installer	Included in the installer	
Server	Server Studio Deployment Manager Repository Java Monitor Discovery Web UI	Active Cluster Salesforce.com Adapter SAP Adapter SAPBW and BEx Adapters Oracle EBS Adapter Siebel Adapter Default cache database Advanced Data Sources Adapters
Studio	Studio	Java
Client	ODBC ADO.Net SSIS Power BI	JDBC
Business Directory Server	BD Server BD Repository	BD web application Java

Overview of Installation

Before you install TIBCO® Data Virtualization, review the new features and bug fixes as documented in the *TDV and Business Directory Release Notes*. You can view it at <https://docs.tibco.com/products/tibco-data-virtualization>

Choose one of the options below:

1. For a new TDV installation, refer to the chapter [Installing TDV, Studio, and Drivers](#).
Note: After TDV 8.x is installed, you can proceed with installation of hotfix or service packs. Refer to the sections [About Hotfix Maintenance](#) and [About Service Pack Maintenance](#) for more information on how to do this.
2. If you are upgrading within the same major version (for example, 8.x to 8.(x+1)), refer to the section [About Service Pack Maintenance](#).
3. If you are upgrading from an earlier TDV major version (for example, 7.x to 8.x), and also want to migrate the metadata created in the earlier TDV version, refer to [Upgrading from an Earlier Release and Migrating The Metadata](#).
Note: If you do not need to migrate the metadata, follow the steps as outlined in the chapter [Installing TDV, Studio, and Drivers](#) to install a new version of TDV.
4. If you need to apply a hotfix, follow the steps as outlined in the section [About Hotfix Maintenance](#).

Tracking and Documenting Your Install or Upgrade

We recommend that you document the issues that you encounter during an upgrade and document all customizations made to your new TDV system, to help ensure that your next upgrade goes smoothly.

Before beginning your install, be aware that Java and PostgreSQL customizations are not preserved through the installation process. After install you must redo the customizations.

Installation Requirements and Support Information

This topic describes installation requirements and what TIBCO® Data Virtualization (TDV) supports. It includes the following topics:

- [Disk Space and Physical Memory Requirements](#)
- [Port Requirements](#)
- [Studio and Server Connectivity and Installer Limitations](#)
- [TDV Supported Platforms](#)
 - [Java Support](#)
 - [Operating System Support for Studio](#)
 - [Operating System Support for Server](#)
 - [Support for Virtualization Environments](#)
 - [Driver Support](#)
 - [SNMP Support](#)
 - [Web Service Protocols](#)
 - [Directory Services Support for LDAP](#)
- [Options and Features Supported for Use with TDV](#)
 - [Kerberos Support](#)
 - [Web Browser Support](#)
 - [Supported Data Sources](#)
 - [Supported Add-On Adapters](#)
 - [Supported Advanced Data Source Adapters](#)
 - [Supported Cache Targets](#)
 - [Data Ship Source and Target Support](#)
 - [Client Application Target Framework](#)

- [Enterprise Service Buses](#)
- [Client-Side ADO.NET Driver Support](#)
- [Data Sources Supported for Kerberos Token Pass-through](#)
- [Security Features](#)
- [Support and Maintenance Policies for TIBCO Products](#)
- [Limitations for TDV Discovery](#)

Disk Space and Physical Memory Requirements

TDV performance depends on processor speeds, available memory, network bandwidth, data source response times, query join types, the complexity of views, and many other implementation factors. Fast response times and support for a large active user base and large loads are achieved with:

- Fast multi-core CPUs
- Large amounts of RAM
- Ample disk space
- GB Ethernet network connections on the same subnet as the most heavily trafficked data source

If hardware configurations are less than optimal, TDV functions equally well, although more slowly, for most development tasks.

Minimum Installation Requirements

TDV has these resource requirement

- **Disk Space:** Minimum of 8 GB read/write, persistent storage.
 - 2.0 GB for TDV installation
 - 1.0 GB for the TDV repository database
 - 5.0 GB for future growth (e.g. default TDV temp directory, TDV VCS data, logs, file cached data, and future TDV patch/Java/repository software updates)
- **Memory:** Minimum of 8 GB memory.

- **CPU:** Minimum of 2 CPUs/cores.

For larger TDV deployments, please refer to [Sizing Guidelines for TDV](#). Additionally, larger TDV deployment above the “small” configuration will need to consider if changes to the following areas are required or not.

Additional storage configuration

This section discusses the additional storage configuration required for TDV, beyond the default installation configuration.

TDV temp directory

By default, TDV creates a temp directory under the TDV installation directory.

For production TDV configurations, you should ensure the TDV temp directory has access to at least 10GB of storage or more in order to better handle large query processing, file caching, etc. For information about those configuration parameters, see the TDV Administration Guide.

TDV Version Control System (VCS) directory:

By default, TDV creates a VCS directory under the TDV installation directory.

For development or production DV deployments that use VCS, the following additional storage requirements need to be met.

Note: If your TDV deployment doesn't meet the storage requirements for VCS then you will need to reconfigure it to point to another storage location. Refer to the Administration Guide, chapter “Using Version Control and TDV” for instructions on how to manage the TDV data directory.

- 1.2 GB of additional free space per TDV user is required for this setting. Different types of resources require different amounts of space to store. We recommend that you use 12KB per resource as a rough storage guideline.
- The following guidelines will help you figure out how to calculate your VCS directory storage needs.
 - If you have 100,000 resources, we recommend 1.2GB of space available for storing resources.
 - If you expect a large amount of check-ins to the VCS directory, we recommend that you allocate additional space in the VCS directory area which lives under the

TDV installation (INSTALL_DIR\data\vcs). Typically, changes within version control systems are stored as textual diffs. Textual diffs can add up over time.

Sizing Guidelines for TDV

On-Premises

For TDV on-premise deployment, use the following TDV sizing guidelines. Keep in mind that these are starting point recommendations for each configuration. It is OK to exceed these recommendations for a given configuration.

Small configuration (poc/demo)

2 CPUs/cores, 8 GB of memory, 8 GB of readable/writable persistent storage

Medium configuration (development)

4 CPUs/cores, 16 GB of memory, 16 GB of readable/writable persistent storage

Large configuration (production)

8 CPUs/cores, 32 GB of memory, 32 GB of readable/writable persistent storage:

X-Large configuration (production with MPP Engine)

16 CPUs/cores, 64 GB of memory, 64 GB of readable/writable persistent storage

Note: For x-large TDV instances running MPPE, the minimum requirement calls for 2 clustered 8-core instances.

Docker

For TDV deployment in Docker environment, please use the following TDV sizing guidelines. Keep in mind that these are starting point recommendations for each configuration. It is OK to exceed these recommendations for a given configuration.

Small configuration (poc/demo)

TDV - 2 CPUs/cores, 8 GB of memory, 8 GB of readable/writable persistent storage

TDV Repository - 1 CPU/core, 2 GB of memory, 2 GB of readable/writeable persistent storage

TDV Cache- 1 CPU/core, 2 GB of memory, 2 GB of readable/writeable persistent storage

Medium configuration (development)

TDV - 4 CPUs/cores, 16 GB of memory, 8 GB of readable/writable persistent storage

TDV Repository - 1 CPU/core, 2 GB of memory, 4 GB of readable/writeable persistent storage

TDV Cache- 1 CPU/core, 2 GB of memory, 4 GB of readable/writeable persistent storage

Large configuration (production)

TDV - 8 CPUs/cores, 32 GB of memory, 16 GB of readable/writable persistent storage

TDV Repository - 1 CPU/core, 4 GB of memory, 8 GB of readable/writeable persistent storage

TDV Cache- 1 CPU/core, 4 GB of memory, 8 GB of readable/writeable persistent storage

X-Large configuration (production with MPP Engine)

TDV - 16 CPUs/cores, 64 GB of memory, 32 GB of readable/writable persistent storage

TDV Repository - 1 CPU/core, 8 GB of memory, 16 GB of readable/writeable persistent storage

TDV Cache- 1 CPU/core, 8 GB of memory, 16 GB of readable/writeable persistent storage

Note: For X-Large TDV instances running MPPE, the minimum requirement calls for 2 clustered 8-core instances.

Public Cloud

AWS and Azure provides a wide selection of instances optimized to fit different use cases.

Since the TDV images on the public cloud platforms are not light-weight (as the case of docker containers), the minimum recommended configuration is a Medium resource configuration. You may choose a different setting based on your organizational needs.

Refer to the following links for the public cloud platform guidelines:

<https://aws.amazon.com/ec2/instance-types/>

<https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-sizes-specs>

Kubernetes

TDV Kubernetes distribution is packaged with helm charts with the following default CPU and memory settings:

<Default settings>

You can change these settings by using the `--set` flag while installing the TDV Helm chart. Alternately, if you are familiar with Helm charts, you can also modify the YAML file that is delivered via the edelivery site.

Port Requirements

By default, TDV Server listens to port 9401 for ODBC connections. The ODBC port number is always one greater than the server's web services HTTP base port which by default, is 9400. So the ODBC default port number is 9401. If SSL is used (encrypt is set to true), the ODBC driver automatically adds 2 to the port value so that the 9403 port is used. To determine the actual ODBC port settings, refer to the Client Interfaces Guide, section "TDV Port Settings for Client Connections to TDV".

Changing the HTTP base port value also changes the value of all derived ports after the next TDV restart (with the exception of the Repository and Cache database ports, which will remain the same).

Port number availability for TDV and Business Directory:

TDV ports

TDV Ports Default	Description
9400	
9401	JDBC, ODBC, and ADO.NET ← port needs to be exposed for non SSL TDV client access
9402	
9403	JDBC SSL, ODBC SSL, and ADO.NET SSL ← port needs to be exposed for SSL TDV client access
9404	
9405	JMX/RMI port for Monitor collector
9406	
9407	
9408	
9409	Monitor RMI registry + JMX/RMI port for Monitor daemon

BD ports

Business Directory Ports Default	Description
9500	

Business Directory Ports Default	Description
9501	
9502	
9503	
9504	
9505	JMX/RMI port for Monitor collector
9506	
9507	
9508	
9509	Monitor RMI registry + JMX/RMI port for Monitor daemon

MPP Engine ports

TDV Ports Default	Description
9300	Zookeeper Quorum port
9301	Zookeeper Election port
9302	Zookeeper Client Port
9303	Drill HTTP (web console) Port

TDV Ports Default	Description
9304	Drill User Port
9305	Drill Server Bit Ports
9306	Drill Server Bit Ports

Note: The above ports are only active on operating systems that support MPP engine.
(Refer to [MPP Engine OS Support](#))

Studio and Server Connectivity and Installer Limitations

You can sometimes mix versions of Studio and Server as follows within a major release.

Studio Version	Server Version	Support
older	newer	Active
newer	older	Not active

For example:

- Connecting a 8.0.0 Studio with a 8.2.1 or 8.2.2 Server is supported.
- Connecting a 8.2.1 Studio with a 8.2.0 Server is not supported.

Business Directory and Deployment Manager Limitations

You can sometimes mix versions of Business Directory, Deployment Manager, and TDV as follows.

BD/DM Web UI Version	TDV Version	Support
older	newer	Not Supported
newer	older	Supported

For example:

- Business Directory 8.4, 8.5 clients are not compatible with published resources from TDV 8.6.
- The use of Business Directory 8.6 clients with published resources from TDV 8.4 is supported.
- When trying to connect to 8.5/8.6 TDV servers from 8.3/8.4 DM to create a site, for example, the keystore and truststore files must match between the servers if default TDV ones are used. This is to ensure backward compatibility

Installer Limitations

- 64-bit installers are supported only on 64-bit platforms.
- Linux and Windows installers are available only on the x86 hardware platform..

Type of Client	Requirements
64-bit Studio client	<ul style="list-style-type: none"> • They are at the same TDV version and patch level. • The Server version is newer than the Studio version and they are both within the same major TDV version.

TDV Supported Platforms

Studio can be installed and run on all Microsoft Windows platforms, but is not available for any UNIX platforms. Business Directory is supported on Windows and UNIX platforms only.

64-bit installers are provided for each of the Windows and UNIX platforms. In addition, separate java versions are provided for each platform (see [Java Support](#)).

— [Java Support](#)

- [Operating System Support for Studio](#)
- [Operating System Support for Server](#)
- [Support for Virtualization Environments](#)
- [Driver Support](#)
- [SNMP Support](#)
- [Web Service Protocols](#)
- [Directory Services Support for LDAP](#)

Java Support

The JDK required for TDV for each platform is listed in the following table.

Platform	Required
AIX (TDV Server)	openjdk version "11.0.14.1" 2022-02-08
Docker + Docker K8s images	openjdk 11 (version depends on when you build the image.)
Linux (TDV Server, TDV Business Directory)	java "11.0.15" 2022-04-19 LTS
Windows (TDV Server, TDV Business Directory, TDV Studio)	java version "11.0.15" 2022-04-19 LTS

Operating System Support for Studio

Client-platform operating system support and patch levels are listed in the following table.

Operating System (Client)	Patch	TDV Support	Notes
Microsoft Windows 10		Active	x64
Microsoft Windows Server 2012 Standard		Active	x64.
Microsoft Windows Server 2012 R2 Standard		Active	
Microsoft Windows Server 2016 Standard		Active	x64
Microsoft Windows Server 2019 Standard		Active	x64

Operating System Support for Server

Server platform operating system support and patch levels are listed in the following table:

Operating System (Server)	Release Date/SP	TDV and BD Support	Notes
AIX 7.2 TL4 or higher	November 2019	Active	Not supported for Business Directory
AIX 7.1 TL5 or higher	October 2017	Active	Not supported for Business Directory
CentOS version 7.9 or higher in 7.x		Active	64-bit versions are supported. TDV deploys in native 64-bit JVM on all supported 64-bit operating systems. x64 architecture. Also supported for BD.

Operating System (Server)	Release Date/SP	TDV and BD Support	Notes
CentOS version 8.5 or higher in 8.x		Active	64-bit versions are supported. TDV deploys in native 64-bit JVM on all supported 64-bit operating systems. x64 architecture. Also supported for BD.
Microsoft Windows 10		Active	Also supported for BD.
Microsoft Windows Server 2012 Standard		Active	x64. Also supported for BD.
Microsoft Windows Server 2012 R2 Standard		Active	
Microsoft Windows Server 2016 Standard		Active	x64 Also supported for BD.
Microsoft Windows Server 2019 Standard		Active	x64 Also supported for BD.
Oracle Linux 7.9 or higher in Red Hat compatibility mode		Active	x64. Also supported for BD.
Oracle Linux 8.5 or higher in Red Hat compatibility mode		Active	x64. Also supported for BD.
Red Hat Enterprise Linux v7.9 or higher in 7.x	N/A	Active	64-bit versions are supported. TDV deploys in native 64-bit JVM on all supported 64-bit operating systems. x64. Also supported for BD.
Red Hat Enterprise Linux	N/A	Active	Red Hat provides a 64bit OS image

Operating System (Server)	Release Date/SP	TDV and BD Support	Notes
v8.5 or higher in 8.x			for RHEL 8 that provides 64bit application support for TDV. Also supported for BD.
SLES 12	SP5 or higher	Active	
SLES 15	SP3 or higher	Active	

There is a known limitation of Windows OS that can result in a UNC error when using TDV. The known issue is that:

- The Windows service process can't see any mapped network driver of a front end user session, because the Windows service is running under a different credential, and the mapped network driver is valid in the user session only.
- The Windows service process can see SYSTEM ODBC DSN only, any USER ODBC DSN is not visible to the Windows service.

To work around for this known issue, use the UNC path for TDV to access remote files.

MPP Engine OS Support

MPP Engine is supported on Linux platforms only with the following pre-installed 3rd party software:

- a. Network Security Service (NSS) Package version 3.28.4 x86_64 or higher (package name: nss).
- b. CentOS/RedHat/Oracle Linux versions 6.5 or higher are supported for this feature.
- c. SUSE Linux version 12 and above is supported for this feature.

Support for Virtualization Environments

The TDV Server is fully supported and can be hosted in operation systems run on virtualization platforms such as Red Hat Enterprise Virtualization, Hyper-V, VMware Fusion and Oracle VirtualBox as long as the Server is run on a supported OS platform and version that is listed in the section [Operating System Support for Server](#)

Driver Support

Driver	Server Version	TDV Support
ODBC	iODBC Driver Manager v3.52.12 for AIX and Linux	Active
ODBC	<p>Windows Driver Manager</p> <p>The Windows Driver Manager is part of your Windows Operating System. Refer to the section <i>Operating System Support for Server</i> in the <i>TDV Installation and Upgrade Guide</i> for a list of supported OS.</p>	Active
ODBC	<p>DataDirect Driver Manager v8.0</p> <p>Note: DataDirect Driver Manager is supported on Linux and AIX platforms. There is no extra configuration needed for using TDV ODBC driver with DataDirect Driver Manager</p>	Active
ODBC	<p>unixODBC Driver Manager(v2.3.1)</p> <p>Note: unixODBC Driver Manager is supported on Linux and AIX platforms. For using TDV ODBC driver with the unixODBC Driver Manager, set the environment variable TDV_ODBC_UODBC to TRUE for the applications that use TDV ODBC Driver.</p>	Active

Driver	Server Version	TDV Support
	<pre>set TDV_ODBC_UODBC=TRUE</pre>	
JDBC	JRE v11 (csjdbc.jar) and conforms to JDBC API 4.0	Active
JDBC	<p>JRE v1.8 (csjdbc8.jar) and conforms to JDBC API 4.0</p> <p>Example: If you are running a client using JRE 8 (also known as 1.8), you would include csjdbc8.jar in the CLASSPATH as shown below:</p> <p>"C:\Program Files\Java\jdk1.8.0_211\bin\java" -classpath .;\csjdbc8.jar Test</p>	Active
ADO.NET	ADO.NET (32-bit and 64-bit)	Active
Power BI Data Connector	20.0.7656	Active
SSIS	20.0.7668	Active
ADO.Net	ADO.Net 2021 Data Provider - v20.0.7656	Active

SNMP Support

The TDV system supports SNMP v3.

Web Service Protocols

Web Service Protocols	TDV Support
SOAP v1.1	Active
SOAP v1.2	Active
WSDL v1.1	Active
WSI-Basic Profile v1.0	Active
WSI-Basic Profile v1.1	Active
XPath v1.0	Inactive
XPath v2.0	Inactive
XPath v3.0	Active
XQuery v1.0	Inactive
XQuery v3.0	Active
XSLT v1.1	Active
XSLT v2.0	Active

Directory Services Support for LDAP

The following LDAP directory services are compatible for the TIBCO Data Virtualization to use as a secure authentication service.

[illegible]

Options and Features Supported for Use with TDV

The TDV product suite supports a large collection of data sources, connection protocols, features, and client interfaces that grows with each service pack and release. The following topics catalogs many of these items:

- [Kerberos Support](#)
- [Web Browser Support](#)
- [Directory Services Support for Kerberos](#)
- [Supported Data Sources](#)
- [Supported Add-On Adapters](#)
- [Supported Cache Targets](#)
- [Data Ship Source and Target Support](#)
- [TDV DDL Feature Support](#)
- [Supported Client Applications](#)
- [Client Application Target Framework](#)
- [Enterprise Service Buses](#)
- [Client-Side ADO.NET Driver Support](#)
- [Data Sources Supported for Kerberos Token Pass-through](#)

Kerberos Support

Enterprise users can leverage Kerberos infrastructure to authenticate just once to secure access to TDV-defined resources. The duration of an authenticated session is set by the Kerberos administrator. TDV supports pass-through of the Kerberos tokens from the authenticated client through TDV to the Kerberos server and to the data sources. The TIBCO Data Virtualization Server, data sources, and clients of the TDV Server must be configured to support Kerberos token pass-through and SSO.

The KDC Kerberos v5 Server must already be installed and running in your environment before you install TDV Server and Studio. You then configure the Kerberos system to use

with TDV, establishing a security context in which Kerberos and the TDV identify each other.

Web Browser Support

Online help (and long lists in Manager) might not display as expected in Chrome.

Web Browsers	TDV Support	Notes
Mozilla Firefox	Active	Business Directory, Deployment Manager, Web Manager and Web UI support Mozilla Firefox v94.0 on Windows 10. Business Directory, Deployment Manager, and Web Manager support Mozilla Firefox v94.0 on macOS Catalina.
Chrome	Active	Business Directory, Deployment Manager, Web Manager and Web UI support Chrome v95.0 on Windows 10. Business Directory, Deployment Manager, and Web Manager support Chrome v95.0 on macOS Catalina.
Safari	Active	Not supported for web service API calls. Not supported for TDV Web UI. Business Directory, Deployment Manager and Web Manager support v13.1.
Edge	Active	Business Directory, Deployment Manager, Web Manager and Web UI support v95.0 on Windows 10.

For the Deployment Manager client web applications to function properly, the machine that is running a compatible browser must be running on a machine with Windows 7 or higher. Occasionally the login screen for these web applications does not close automatically, you can close it and continue using the product or you can choose to run in a different browser.

For best results, when running Business Directory and Deployment Manager concurrently, use different browsers.

The TDV and Business Directory servers require a secure connection. So when you first connect a browser to any TDV web-based application, you might get a warning about connecting to an untrusted site.

Depending on your browser:

- You might be asked to allow the connection process to continue.
- You might want to configure it to trust the site so that warning messages no longer appear. For some site configurations this might require configuration of SSL connections for your entire TDV environment.

OAuth 2.0 Compatible Browsers

- OAuth 2.0 is compatible with the Chrome browser.

Browsers and Kerberos Support

Different browsers have different settings to enable Kerberos support. TIBCO recommends that you search the web to confirm the instructions to enable Kerberos SPNEGO authentication and credential delegation for your browser and operating system.

For example in Firefox, add the url to both `network.negotiate-auth.trusted-uris` and `network.negotiate-auth.delegation-uris` and switch `network.negotiate-auth.allow-non-fqdn` to true.

Directory Services Support for Kerberos

The following LDAP directory services are compatible for the TIBCO Data Virtualization to use as a secure authentication service.

Directory Service	TDV Support	Notes
Active Directory 2019	Active	LDAP, LDAPS, Kerberos

Supported Data Sources

TDV supports these data sources.

TDV supports OAuth 2.0 for HTTP-based data sources: SOAP, REST, WSDL, and XML-HTTP. It is also available for several Advanced Adapter data sources.

For other supported data sources and applications, see these sections:

- [Supported Add-On Adapters](#)
- [Supported Advanced Data Source Adapters](#)

Refer to the Adapter Guides for more details about each of the adapters.

Select Data Source Adapter field	TDV Support	Versions, Compatibility, and Notes
Apache Drill	Active	CAST AS functions are not supported.
Amazon AWS S3	Active	
Azure Data Lake Storage	Active	
Composite	Active	
ComputeDB	Active	Version 1.2
Custom Java Procedure	Active	
DB2 V10.5 (Type4)	Active	
DB2 z/OS Version 10 (Type 4)	Active	
DB2 z/OS Version 11	Active	
Data Direct Mainframe	Active	The Shadow RTE Server (version 6.1.4.7606 or later) must be installed on the DataDirect Mainframe computer and the Shadow RTE Client (version 6.1.1.1080 or later) must be installed locally on the computer hosting the TDV Server.

Select Data Source Adapter field	TDV Support	Versions, Compatibility, and Notes
File (cache, delimited, and XML)	Active	For data sources that access a file share, the TDV service user account needs to have permission to read the file share.
Greenplum 3.3	Active	TDV capabilities with Greenplum have been developed and tested with a single node license.
Greenplum 4.1	Active	
Greenplum 4.3	Active	
HBase 0.98 (Apache Phoenix Driver)	Active	<p>Introspection of HBase databases retrieves information from the system tables. User created tables are only introspected if they have been created using the Apache Phoenix shell.</p> <p>Requires installation of Apache Phoenix JDBC drivers, specifically those in phoenix-4.1.0-bin.tar.gz. For more information see the TDV Administration Guide.</p>
HSQldb 2.2.9	Active	
Hive 2.1.1	Active	<p>Kerberos is supported.</p> <p>Trusted Delegation is not supported.</p> <p>For Hive data sources, TDV introspects tables and columns only.</p>
Impala 2.x	Active	
Informix 9.x	Active	
Local File Storage	Active	
LDAP	Active	v3

Select Data Source Adapter field	TDV Support	Versions, Compatibility, and Notes
Microsoft Access	Active	Windows platforms only
Microsoft Access (non-ODBC)	Active	Windows platforms only
Microsoft Excel	Active	2000
Microsoft Excel (non-ODBC)	Active	2000
Microsoft SQL Server 2008	Active	Kerberos authentication is supported.
Microsoft SQL Server 2012	Active	Kerberos authentication is supported with the 2008 driver.
Microsoft SQL Server 2014	Active	Kerberos authentication is supported.
Microsoft SQL Server 2016	Active	Kerberos authentication is supported.
Microsoft SQL Server 2019	Active	Kerberos authentication is supported.
MySQL 5.1	Active	
MySQL 5.5	Active	
MySQL 8.0.23	Active	Use MySQL5.5 adapter to connect to Mysql8.0.X databases.
Neoview 2.3	Active	
Neoview 2.4	Active	

Select Data Source Adapter field	TDV Support	Versions, Compatibility, and Notes
Netezza 6.0	Active	NPS
Netezza 7.0	Active	NPS
OData	Active	Provides for some limited access to SharePoint data.
Oracle 11g (OCI Driver)	Active	11g R1, 11g R2, Oracle RAC Kerberos authentication is supported with thin driver version 11.2.0.4.
Oracle 11g (Thin Driver)	Active	11g R1, 11g R2, Oracle RAC Kerberos authentication is supported with thin driver version 11.2.0.4.
Oracle 12c (OCI Driver)	Active	Oracle RAC
Oracle 12c (Thin Driver)	Active	Oracle RAC
Oracle 19c (OCI Driver)	Active	
Oracle 19c (Thin Driver)	Active	
PostgreSQL 12	Active	
PostgreSQL 9.0	Active	
PostgreSQL 9.1	Active	
PostgreSQL 9.2.3	Active	
REST	Active	Kerberos authentication is supported. NTLM authentication is supported. OAuth 2.0 authentication is supported.

Select Data Source Adapter field	TDV Support	Versions, Compatibility, and Notes
RSS	Active	
Redshift	Read-only	<p>New datasources cannot be created.</p> <p>The following trigonometric functions should not be pushed to Redshift data sources, because they might return incorrect results: SIN, COS, TAN, ASIN, ACOS, COT.</p>
SOAP	Active	<p>1.1, 1.2</p> <p>Kerberos authentication is supported.</p> <p>NTLM authentication is supported.</p> <p>OAuth 2.0 authentication supported.</p>
SAP HANA SPS 09	Active	Support is for on premise SAP HANA deployments.
Sybase 12	Active	12.5 ASE
Sybase 15	Active	<p>15 and 15.5 ASE</p> <p>Kerberos authentication is supported.</p>
Sybase IQ	Active	15
Sybase IQ (Type 2)	Active	15.2
Teradata 13	Active	<p>13 and 13.10</p> <p>Support for query band.</p>
Teradata 14	Active	<p>14.10</p> <p>Might require installation of a Teradata 15 driver.</p> <p>Support for query band.</p>
Teradata 15	Active	FastExport is not supported.

Select Data Source Adapter field	TDV Support	Versions, Compatibility, and Notes
		<p>The JDBC driver does not support CLOB columns with NULL values when using TDV to cache data into a Teradata 15 target.</p> <p>Support for query band.</p>
Teradata 16	Active	<p>16.20</p> <p>The JDBC driver does not support CLOB columns with NULL values when using TDV to cache data into a Teradata 16 target.</p> <p>Support for query band.</p>
TIBCO Streaming	Active	11.0.0
Vertica 6.1	Active	
WSDL	Active	<p>1.1</p> <p>Kerberos authentication is supported.</p> <p>NTLM authentication is supported.</p> <p>OAuth 2.0 authentication supported.</p>
XML/HTTP	Active	<p>Flat files or over HTTP.</p> <p>Kerberos authentication is supported.</p> <p>NTLM authentication is supported.</p> <p>OAuth 2.0 authentication supported.</p>

Supported Add-On Adapters

Consult your vendor specific documentation for detailed documentation of the objects and fields that have changed from version to version. These SAP adapters require the SAP JCo driver. Configuration steps can be found in the TDV SAP BW Adapter Guide. OLAP Cube Support—With TDV 5.1.0.1 and later, you can create dimensional OLAP views in TDV.

TDV supports the following application data sources.

Adapter	Version Support
Oracle E-Business Suite Adapter	11.5.8, 11.5.10 , 12.1, 12.2 on Oracle 9i and 10g
Salesforce.com Adapter	Version 37 You can install and use the Salesforce.com Adapter on all platforms that TDV supports. See Installation Requirements and Support Information .
SAP Adapter	5.0, 6.0, and above SAP R/3 v4.7
SAP BW Adapter	3.5 and 7.4 SP 9
SAP BW BEx Adapter	3.5 and 7.4 SP 9
Siebel Adapter	7.7, 7.8, 8.0

For installation and licensing instructions, consult [Installing Optional TDV Products](#).

Supported Advanced Data Source Adapters

TDV supports the following application data sources.

Data Source Adapter	Versions, Compatibility, and Notes
Active Directory	LDAP v2 and v3 servers
Amazon DynamoDB	DynamoDB REST API Version 2012-08-10
Amazon Redshift	Version 1.0.7562

Data Source Adapter	Versions, Compatibility, and Notes
Cassandra	Versions 2.1.7 and 3.0.0
CosmosDB	2019
Couchbase	Version 4.0 of the API
DynamicsCRM	Windows server 2016, Windows server 2012 R2, Windows Server 2012. Supports OAuth.
DynamicsGP	Dynamics GP 2010, 2013, and 2015
DynamicsNAV	Dynamics NAV 2013, 2015, and 2016
Eloqua	Eloqua REST API and Bulk API version 2.0 Supports OAuth.
Elasticsearch	Version 2.0 and above
Email	Standard IMAP client as specified in RFC 1730 and RFC 2060
Sharepoint Excel Services	Excel data from SharePoint Online, and SharePoint Server 2010 and 2013
Facebook	Facebook Graph API 2.0, 2.1, 2.2, 2.3 Supports OAuth.
Google Ads	API v201809 Supports OAuth. Note: Google AdWords has been re-branded as GoogleAds. Google AdWords datasource created in TDV version 8.0 or earlier cannot be opened in TDV version 8.1 or later. You need to create a new datasource by choosing GoogleAds from the new datasource

Data Source Adapter	Versions, Compatibility, and Notes
	dialogue in TDV Studio and introspect it again.
Google Analytics	Google Analytics Management API v3.0, Google Analytics Core Reporting API v3.0. Supports OAuth.
Google Contacts	API v3.0 Supports OAuth.
Google Calendar	API v3.0 Supports OAuth.
Google Drive	API V3.0 Supports OAuth.
Google BigQuery	Google BigQuery API v2.0 Supports OAuth.
Google Sheets	Google Sheets API v3.0 Supports OAuth.
HubSpot	HubSpot REST API Supports OAuth.
JDBC-ODBC Bridge	ODBC 2.x and 3.x drivers
JSON	Standard JSON format as specified in RFC 7519
MarkLogic	2019
Marketo	Marketo REST API v1, Marketo SOAP API v2.6 Supports OAuth.

Data Source Adapter	Versions, Compatibility, and Notes
MongoDB	MongoDB 2.6 and 3.0
NetSuite	NetSuite SOAP APIs 2011-2015 Supports OAuth.
OData	OData 2.0, 3.0, and 4.0 Supports OAuth.
RSS	RSS 2.0 feeds
Salesforce with SSO	Supports OAuth.
SharePoint	SharePoint Online, SharePoint Server 2007, 2010, 2013. Supports OAuth.
Single Store	21.0.7810.0
Snowflake	Version 3.25.5. Supports OAuth.
SparkSQL	Version 1.0 and above
Splunk	2019
Twitter	Twitter REST API v1.1. Supports OAuth.

Limitations:

- Sometimes, instead of returning an empty value, “Select * from table where columnname = 'value'” may throw an exception, if there is no value in the column.
- Some adapters support ORDER BY, but sometimes there are only a few objects within that data source that support ORDER BY. TDV displays a message if the tables do not support ORDER BY.

- Tables might need to be filtered with mandatory inputs for querying the contents for table scans to work as expected. For example for google apps directions, the starting location and ending location might be needed to retrieve the results.
- Sharepoint adapters support direct Kerberos authentication.
- Bulk inserts are not supported.
- GoogleSheets does not support client side filtering.
- Table names or column names with the period character are not supported.
- Eloqua data sources where the password value is entered when creating the data source will persist the password and it cannot be changed.
- For the Sharepoint Excel Services adapter, during introspection all String data types are mapped to VARCHAR.
- The DynamicsCRM, DynamicsNAV, DynanoDB, GoogleBigQuery, and SharePoint adapters do not support "is not null" syntax.
- Queries that contain "LIMIT" are not supported.
- For MongoDB, updating schema files within a running instance of TDV is not supported.
- Deployment Manager is case sensitive when using it with these adapters.
- Deployment Manager attributes for these adapters can cause plans to fail.
- “Ignore case sensitivity mismatch between CIS and data source” and “Ignore trailing space mismatch between CIS and data source” override the server side setting for a data source.
By default these two overrides are enabled so that queries are always pushed. This is the case even when there is a mismatch and the query does not contain UPPER or RTRIM or similar options.
- Set these attributes to false or disable the push to get the consistent results as when the query is run with in the TDV.

For installation and licensing instructions, consult Installing Optional TDV Products, page 49.

For OAuth descriptions, see the User Guide, section Configuring OAuth 2.0 for TDV Advanced Adapters.

Supported Cache Targets

TDV supports the following as cache targets:

Cache Target	TDV Support	Parallel Cache Target Support	Native Cache Target Support	Notes
Amazon Redshift	Active	Active	Active	
Apache Hive 2.x	Active		Active	
Apache Impala 2.x	Active		Active	
File	Active	Active		Typically best for demonstrations or caching of a few hundred rows.
Google BigQuery	Active	Active	Active	
Greenplum 4.1	Active	Active	Active	
Greenplum 4.3	Active	Active	Active	
HSQLDB 2.2.9	Active	Active		
IBM DB2 LUW v10.5	Active	Active	Active	Native load with insert and select, and DB2 Load are supported.
Microsoft SQL Server 2008	Active	Active	Active	The DBO schema must be selected and introspected as a resource prior to attempting to cache data.

Cache Target	TDV Support	Parallel Cache Target Support	Native Cache Target Support	Notes
Microsoft SQL Server 2012	Active	Active	Active	The DBO schema must be selected and introspected as a resource prior to attempting to cache data.
Microsoft SQL Server 2014	Active	Active	Active	The DBO schema must be selected and introspected as a resource prior to attempting to cache data.
Microsoft SQL Server 2016	Active	Active	Active	The DBO schema must be selected and introspected as a resource prior to attempting to cache data.
Microsoft SQL Server 2019	Active	Active	Active	The DBO schema must be selected and introspected as a resource prior to attempting to cache data.
MySQL 5.1	Active	Active	Active	
MySQL 5.5	Active	Active	Active	
Netezza 6.0	Active	Active	Active	Native load with insert and select is supported. Parallel

Cache Target	TDV Support	Parallel Cache Target Support	Native Cache Target Support	Notes
				<p>cache processing is achieved using the native DISTRIBUTE syntax.</p> <p>Procedure caching is supported.</p>
Netezza 7.0	Active	Active	Active	<p>Native load with insert and select is supported. Parallel cache processing is achieved using the native DISTRIBUTE syntax.</p> <p>Procedure caching is supported.</p>
Oracle 11g and 11g R2	Active	Active	Active	
Oracle 12c	Active	Active	Active	
Oracle 19c	Active	Active	Active	
PostgreSQL 9.1	Active	Active	Active	<p>Bulk load is supported.</p> <p>Native loading is supported when the source and target are the same database. If not then Parallel loading is used.</p>

Cache Target	TDV Support	Parallel Cache Target Support	Native Cache Target Support	Notes
PostgreSQL 9.2.3	Active	Active	Active	Bulk load is supported. Native loading is supported when the source and target are the same database. If not then Parallel loading is used.
SAP HANA SPS 09	Active	Active		
Singlestore	Active	Active	Active	
Snowflake	Active	Active	Active	
Sybase ASE 12.5	Active			
Sybase ASE 15.5	Active			
Sybase IQ 15.2	Active		Active	
Teradata 13	Active		Active	Supported, but with limitations. If source and target tables are co-located within the same Teradata instance native loading (using INSERT/SELECT statements) will be used, else bulk loading using Teradata FASTLOAD

Cache Target	TDV Support	Parallel Cache Target Support	Native Cache Target Support	Notes
				will be attempted.
Teradata 13.10	Active		Active	<p>Supported, but with limitations.</p> <p>If source and target tables are co-located within the same Teradata instance native loading (using INSERT/SELECT statements) will be used, else bulk loading using Teradata FASTLOAD will be attempted.</p>
Teradata 14.10	Active		Active	<p>Supported, but with limitations. Might require Teradata 15 driver.</p> <p>If source and target tables are co-located within the same Teradata instance native loading (using INSERT/SELECT statements) will be used, else bulk loading using Teradata FASTLOAD will be attempted.</p>
Teradata 15	Active		Active	Choose tables For Caching is not

Cache Target	TDV Support	Parallel Cache Target Support	Native Cache Target Support	Notes
				supported. If source and target tables are co-located within the same Teradata instance native loading (using INSERT/SELECT statements) will be used, else bulk loading using Teradata FASTLOAD will be attempted.
Teradata 16.20	Active		Active	Choose tables For Caching is not supported. If source and target tables are co-located within the same Teradata instance native loading (using INSERT/SELECT statements) will be used, else bulk loading using Teradata FASTLOAD will be attempted.
ComputeDB	Active	Active	Active	
Vertica 6.1	Active	Active	Active	Supports the use of native load and parallel cache load together. Native load

Cache Target	TDV Support	Parallel Cache Target Support	Native Cache Target Support	Notes
				with INSERT AND SELECT is supported.

Data Ship Source and Target Support

Data ship optimization is supported for following data source types.

Data Source Type	Data Ship Source Support	Data Ship Target Support	Performance Option	Notes
DB2 v10.5	Active	Active	Bulk Load using the LOAD utility	LUW
Greenplum 3.3	Active	Active		
Greenplum 4.1	Active	Active		
Greenplum 4.3	Active	Active		
Microsoft SQL Server 2008	Active	Active	Bulk import/export using BCP	
Microsoft SQL Server 2012	Active	Active	Bulk import/export using BCP	
Microsoft SQL Server 2014	Active	Active	Bulk import/export using BCP	

Data Source Type	Data Ship Source Support	Data Ship Target Support	Performance Option	Notes
Microsoft SQL Server 2016	Active	Active	Bulk import/export using BCP	
Microsoft SQL Server 2019	Active	Active	Bulk import/export using BCP	
Netezza 6.0	Active	Active	external tables	
Netezza 7.0	Active	Active	external tables	
Oracle 11g	Active	Active	Database Links	<p>To use an Oracle data source for data ship, the DBA must install the DBMS_XPLAN package in the database and create an area for temporary tables.</p> <p>For this data source to participate in data ship, it must be specified as a data ship source. Participation as a data ship target is optional. If Oracle is both source and target, DB Link needs to be set up between the Oracle databases.</p>
Oracle 12c	Active	Active	Database Links	
Oracle 19c	Active	Active	Database Links	

Data Source Type	Data Ship Source Support	Data Ship Target Support	Performance Option	Notes
PostgreSQL 9.1	Active	Active	Database Links	
PostgreSQL 9.2.3	Active	Active	Database Links	
Sybase IQ 15	Active	Active	Location: iAnywhere JDBC driver	<p>For a Sybase IQ data source to participate in data ship, the QUERY_PLAN_TEXT_ACCESS database option must be set to ON.</p> <p>For this data source to participate in data ship, it must be specified as a data ship source. Participation as a data ship target is optional.</p>
Teradata 13.00	Active	Active	FastLoad/ FastExport	<p>For this data source to participate in data ship, it must be specified as a data ship source. Participation as a data ship target is optional.</p> <p>Teradata Fastload mode doesn't work correctly using the 14.10 JDBC driver</p>

Data Source Type	Data Ship Source Support	Data Ship Target Support	Performance Option	Notes
Teradata 13.10	Active	Active	FastLoad/ FastExport	when Teradata is the Target Data Source. To workaround this issue, use a later version of the Teradata JDBC driver.
Teradata 14.10	Active	Active	FastLoad/ FastExport	
Teradata 15	Active	Active	FastLoad	
Teradata 16.20	Active	Active	FastLoad	
Vertica 5.0	Inactive	Inactive		
Vertica 6.1	Active	Active	Bulk load utility Export to another Vertica database	

TDV DDL Feature Support

TDV DDL (Data Definition Language) feature to CREATE and DROP tables directly is supported in the following data sources:

- Amazon S3
- Azure Data Lake Storage
- Composite
- ComputeDB
- DB2
- Greenplum
- HBase
- HSQLDB

- Apache Hive
- Local File Storage
- MSSQL
- MySQL
- SybaseIQ
- Netezza
- Oracle
- Postgres
- Sybase
- Teradata
- Vertica

Support for DDL Clauses

TDV supports certain DDL extension clauses for the following data sources. See the *TDV Reference Guide*, chapter *TDV SQL Keywords and Syntax* for a list of these extensions.

- ComputeDB
- Teradata
- Vertica

Supported Client Applications

All other client applications are supported through the standard communication protocols that include JDBC and ODBC.

Client-Side Applications	TDV Support	Notes
Cognos 11 R3	Active	
Cognos v10.2.2 fixpack 5	Active	

Client-Side Applications	TDV Support	Notes
MicroStrategy 9.0.2	Active	TDV supports these data sources for use with MicroStrategy: Oracle 10g or 11g, Netezza 5 or 6, SQL Server 2008, and for mixed data coming from Oracle 11g and Netezza 6.
MicroStrategy 9.2.1/9.2.1m on Windows I-Server	Active	Because MicroStrategy can create and delete data directly, you must have used Studio configured one of the following as a temporary tablespace to hold the created and deleted data: Oracle 10g and 11g, Netezza 5 and 6, SQL Server 2008, Teradata 13, MySQL 5, and DB2 v9.
Tableau Desktop Professional Edition Version 7.0.13	Active	
TIBCO Spotfire	Active	

Client Application Target Framework

Client-Side Web Services	TDV Support
.NET Standard 2.0	Active

Refer to the following link for a list of .Net Framework versions supported by .Net Standard 2.0:

<https://dotnet.microsoft.com/en-us/platform/dotnet-standard>

Enterprise Service Buses

Enterprise Service Bus	TDV Support
Sonic 7.5	Active
TIBCO EMS 4.4	Active
OpenMQ 4.4	Active

Client-Side ADO.NET Driver Support

The TDV ADO.NET driver can be installed, uninstalled, or re-installed. It is delivered as a DLL and .NuGet package. It supports native ADO.NET driver functionality on the operating systems supported by .Net Standard 2.0.

Refer the following link for a list of OS and .Net versions supported by .Net Standard 2.0:

<https://dotnet.microsoft.com/en-us/platform/dotnet-standard>

TDV supports communication and use with Visual Studio version 2019 that the tool extension supports. You will need to install the extension by executing the following file:

```
TDV_INSTALL_DIR\ADO.NET Driver\Composite.VisualStudio.vsix
```

The SKUs supported by the Visual Studio Extension are Community, Professional and Enterprise. The Extension essentially has a Provider for DDEX(Data Designer Extensibility). Refer the following link for further details:

[https://docs.microsoft.com/en-us/previous-versions/bb163760\(v=vs.140\)](https://docs.microsoft.com/en-us/previous-versions/bb163760(v=vs.140))

Data Sources Supported for Kerberos Token Pass-through

- IBM DB2 LUW version 9
- Oracle

With these Kerberos authentication modes:

- Microsoft memory-based
- Ticket cache file-based
- Specified data source name and password

...these Oracle data sources are supported for Kerberos:

- Database version 11gR2 with an Oracle 11g driver
- Database version 19c
- Microsoft SQL Server 2008, 2012, 2014, 2016 and 2019
- SOAP 1.1 and 1.2
- REST
- Sybase ASE v12 and v15
- WSDL 1.1
- XML over HTTP

TDV Operating Systems Support

- 64-bit Windows Server 2012, 2016 and 2019
- 64-bit RHEL AS 6.6 and 7.0

Communication Interfaces and Protocols

- ADO.NET
- JDBC
- OData
- ODBC
- Web Services

Security Features

Security features are discussed throughout this guide:

- Kerberos can be used when connecting to several data sources ([Supported Data Sources](#)).
- Password protection is available for operations like installing and starting TDV and registering with data sources like SAP (see Registering with the SAP System Landscape Directory, in the TDV User Guide).

Support and Maintenance Policies for TIBCO Products

TIBCO provides support and maintenance for major/minor releases of TDV.

Support Policies for Third-Party Environments

All versions stated of an environment presume the initial release of a Third-party product without any need for patches, service packs or equivalent terms unless stated. Equally, unless stated, we presume that patches or service packs and minor version releases are upward compatible for our products. Whenever a new release of TDV requires deployment of a patch or service pack or is compatible only with a minor version of an environment, TDV will highlight these requirements in release notes and will require customers to install a patch or service pack or minor version to receive support and maintenance on the product.

The following classifications indicate the level of support for the current release.

Classification	Description
Active	All aspects (design/creation and runtime) are supported in Studio and Server.
Desupported Not Supported	Design/creation of platform version is no longer supported, runtime will persist until the next major or minor version. OR: This platform version has not been added to TDV yet.
Deprecated	Runtime removed from TDV. Old data sources will need to be upgraded to platform versions that are supported

Classification	Description
Inactive	Design/creation and runtime are allowed in Studio and Server, no active testing or development of new features will be performed to the platform version

Support Policies for Third-Party Application Virtualization Environments

Customers deploying TIBCO's products in third-party application virtualization environments from VMWare, Xen, and others should first consult the list of native host environments supported by TDV to verify compatibility. Support issues arising from deploying TDV in any Third-party application virtualization environments will be reviewed and resolved only on the native host operating system to remove any incompatibilities that might be introduced by the application virtualization environment itself.

Limitations for TDV Discovery

Servlets are not supported and cannot be imported from previous TDV versions.

Data Sources Not Supported by Discovery

Discovery supports all data sources and TDV Adapters except the following:

Data Sources Not Supported by Discovery

- Custom Java procedures—Not supported because they are procedural.
- DB2 z/OS
- Hive
- HP Neoview
- IBM DB2 z/OS Version 8, Version 9, Version 10
- Impala
- Netezza
- PostgreSQL

- Relational data sources—Procedural objects in relational data sources are not supported.
- SAP BW
- SAP HANA
- Teradata
- Vertica
- WSDL
- XML (flat files or over HTTP)

Installing TDV, Studio, and Drivers

This topic describes how to install TDV on both Windows and UNIX computers and then verify that the installation was successful.

Topics include:

- [Overview of Installation Steps](#)
- [Preparing Your System for Installation](#)
- [Installing on Windows](#)
- [Installing on UNIX](#)
- [About the Installed TDV Services](#)
- [Tips from an Expert if the Server Does Not Start](#)
- [Where to Go After Installation](#)

Refer to the following sections for the other TDV distribution platforms:

- [TDV for AWS Marketplace](#)
- [TDV Docker Container](#)
- [TDV Container Orchestration Using Kubernetes](#)

Overview of Installation Steps

This section includes the following topics:

- [Installation Overview for New TDV Software Customers](#)
- [Installation Overview for Existing Customers Upgrading from a Previous Release](#)

Installation Overview for New TDV Software Customers

If you are installing TDV Data Virtualization products for the first time, here is an overview of how you would proceed:

1. Review the new features and bug fixes as documented in the *TDV and Business Directory Release Notes*.
2. Review the information in the following topics:
 - [Installation Requirements and Support Information](#)
 - [About TDV Software Patches](#)
 - [Preparing Your System for Installation](#)
3. Install TDV as described in:
 - [Installing TDV, Studio, and Drivers](#)
 - [Silent Mode Installation](#)
4. If a TDV service pack release exists for your TDV installation, then it is recommended to apply that to your new installation. Instructions for how to install a patch or service pack are subject to change with each service pack. For instructions, see [TDV and Business Directory Product Maintenance](#).

Installation Overview for Existing Customers Upgrading from a Previous Release

To install a major upgrade for TDV

1. Review the new features and bug fixes as documented in the *TDV and Business Directory Release Notes*.
2. Review the information in the following topics:
 - [Installation Requirements and Support Information](#)
 - [About TDV Software Patches](#)
 - [Preparing Your System for Installation](#)

3. Review and follow the steps in [Upgrading from an Earlier Release and Migrating The Metadata](#).

Preparing Your System for Installation

To prepare your systems for installation

1. Review the new features and bug fixes as documented in the *TDV and Business Directory Release Notes*.
2. Review any README file included in your installation, patch, or service pack bundle.
3. Review the following requirements:
 - [Installation Requirements and Support Information](#)
 - You must have administrator privileges on the host computer to install TDV Server.
 - You can have multiple JVMs running on the installation machine.
 - Server requires a block of nine ports for use by TDV and associated services. The port setting for Web services HTTP communication serves as the “base port”. By default, the base port is 9400, but you can change it after installation using configuration parameters.
4. Make sure that any LIBPATH or LD_LIBRARY_PATH environment variable that you might have does not begin with a "/" slash or end with a ":" colon. Those characters may keep the repository from starting successfully.
5. Review your firewall settings and verify that they allow access to the ports that TDV (Business Directory, Deployment Manager, and TDV) products need to use.
6. To see the current base port setting, choose Configuration from the Administration menu and navigate to Server > Web Services Interface > Communications > HTTP > Port (Current).

Note: Changing the HTTP base port value also changes the value of all derived ports after the next TDV restart. When the base port is changed, you must update all data sources with the new port information.

These example ports are reserved or are derived from the base port:

9400	Web services HTTP port
9401	JDBC, ODBC, and ADO.NET
9402	Web services HTTP SSL
9403	JDBC SSL, ODBC SSL, and ADO.NET SSL
9404	Default caching database port
9405	JMX/RMI port for Monitor collector
9406	Monitor Daemon
9407	Active Cluster - JGroups (when installed)
9408	Repository
9409	Monitor RMI registry + JMX/RMI port for Monitor daemon
9500	Business Directory
9502	Business Directory (reserved)
9508	Business Directory

7. Stop Server if an earlier version is running.
8. Restart databases, especially those used for your caches and repositories.
9. Shut down all other application programs running on the installation machine.
10. Make sure you know the hostname or the IP address of the installation machine.

11. If you are installing on a Linux operating system, see [Preparing UNIX for TDV Installation](#).
12. If you are installing on a Windows operating system, see [Preparing Microsoft Windows for TDV Installation](#).

Preparing UNIX for TDV Installation

This section applies only if you are installing TDV on a machine running a supported UNIX operating system. Examples of valid and invalid `/etc/hosts` file entries are shown in the following table.

Validity	<code>/etc/hosts</code> File Entry
Valid	127.0.0.1 localhost IP hostname.domain hostname
Valid	127.0.0.1 localhost localhost.localdomain IP hostname.domain hostname
Valid	127.0.0.1 localhost localhost.localdomain localhost IP hostname.domain hostname
Invalid	127.0.0.1 localhost.localdomain IP hostname.domain hostname
Invalid	127.0.0.1 localhost.localdomain localhost IP hostname.domain hostname

To prepare your UNIX machine for installation of TDV products

1. Review the new features and bug fixes as documented in the *TDV and Business Directory Release Notes*.
2. Run the following command to determine if localhost can be resolved on the target installation machine before attempting an installation:

```
ping localhost
```

3. If the ping results look like the following, localhost is being resolved and the machine is ready for TDV installation. You can continue with the instructions in other sections.

Linux Ping Example with Valid localhost

```
$ ping localhost
```

```
PING localhost (127.0.0.1) from 127.0.0.1 : 56(84) bytes of data.
```

```
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=255 time=0.071 ms
```

```
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=255 time=0.063 ms
```

```
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=255 time=0.043 ms
```

```
--- localhost ping statistics ---
```

```
3 packets transmitted, 3 received, 0% loss, time 1999ms
```

```
rtt min/avg/max/mdev = 0.043/0.059/0.071/0.011 ms
```

4. If the ping results look like the following, localhost is not correct. You must edit your `/etc/hosts` file.

Linux Ping Example with Invalid localhost

This example of `/etc/hosts` files shows where Server is unable to connect to the repository database because of the `localhost.localdomain` entry preceding the `localhost` entry (assuming the `localhost` entry exists at all).

```
$ ping localhost
```

```
PING localhost.localdomain (127.0.0.1) from 127.0.0.1 : 56(84) bytes of data.
```

```
64 bytes from localhost.localdomain (127.0.0.1): icmp_seq=1 ttl=255 time=0.080 ms
```

```
64 bytes from localhost.localdomain (127.0.0.1): icmp_seq=2 ttl=255 time=0.071 ms
```



```
64 bytes from localhost.localdomain (127.0.0.1): icmp_seq=3 ttl=255 time=0.044 ms
```

```
--- localhost.localdomain ping statistics ---
```

```
3 packets transmitted, 3 received, 0% loss, time 1998ms
```

```
rtt min/avg/max/mdev = 0.044/0.065/0.080/0.015 ms
```

5. Edit the `/etc/hosts` file to add a localhost entry, directly after the 127.0.0.1 entry, with the following syntax:

```
127.0.0.1 localhost <optional host name>
```

6. Save your changes and rerun the ping for localhost.

Preparing Microsoft Windows for TDV Installation

If you are installing TDV on Windows Server 2012 R2, you must disable User Account Control before installing TDV Server. Optionally, if you run into permission issues when running the installer, you can use this procedure to attempt to solve the issue.

To disable the User Account Control

1. From the Windows Start menu, select Control Panel > User Accounts > Change User Account Control Settings.

Change the setting to Never notify.

Installing on Windows

This installation process is used to install one or more components of TDV. You install TDV for Windows using the InstallAnywhere installer wizard.

Note: If you installed TDV on Microsoft Windows Server 2012 R2, see [Preparing Microsoft Windows for TDV Installation](#).

- [Running the TDV Server Installer](#)

- [Running the Studio Installer](#)
- [Installing the Drivers](#)

Running the TDV Server Installer

This installer installs the following components:

• Server	• Deployment Manager	• Repository
• Java	• Monitor	• Discovery
• Active Cluster	• Salesforce.com Adapter	• SAP Adapter
• SAPBW and BEx Adapters	• Oracle EBS Adapter	• Siebel Adapter
• Studio	• Default caching database	• Advanced Data Sources Adapters

To install TDV on a Windows computer

1. Read the README files included with or associated with the download file.
2. Run the installer executable for your platform.

Note: Make sure your Windows OS has all the necessary updates installed, so that TDV installation can run smoothly. TDV installer and patch are packaged with VC++ redistributables which are needed for the postgres and svn connections to run smoothly.

- The installer installs Microsoft Visual VC++ version 2015-2019. This is required by postgres.
- TDV installer also installs Microsoft Visual VC++ version 2013. This is required for svn connections to be successful. When using a patch, this version of VC++ (2013) needs to be manually installed. Execute the file vcredist_x64.exe which can be found in the directory <TDV_install_dir>\bin

- Follow the prompts on the screen. Special characters and spaces are not supported for <TDV_install_dir>.

You can select the defaults for the <TDV_install_dir> and the TDV Server base port number.

You will be prompted for the following passwords.

TDV Password Type	Description
TDV Server Application Password	The TDV Server application password is used to login to web manager.
TDV Repository	This is the database that will be used to store all of the data and metadata about the items that you create within TDV. It also stores your configuration and other environment settings. Passwords with special characters that are supported by your operating system shell are fine to use.
Default Caching Database	This is the database that will be created for you to hold data that you want to cache using the default caching method. There are multiple caching options. You might want to note the password for future use of this database.

The installation process might take a few minutes, during which progress windows are displayed.

Note: TDV will generate a new encryption key when the installation is in a new destination. If it is an existing installation TDV uses the existing encryption key. In case of any errors encountered, the administrator may have to investigate if the encryption key file location and content are correct and then contact TDV support team for assistance.

- Select a password for the default caching database.
- Finish to exit the installer when the installation is completed.

The Server starts automatically at the completion of the installation process. You can also start and stop the services as described in the TDV Administration Guide.

Install and uninstall logs are called bitrock_installer_<number>.log while the installer is running. After installation is complete, the logs are named <product>_install or <product>_uninstall.log. The log files can be found in the following directories:

Platform	Default Location of Log Files
Unix	/tmp
Operating System Support for Server	C:\Users\<username>\AppData\Local\Temp

6. Optionally, download and install the latest TDV patch as described in TDV and Business Directory Release Notes.

Running the Studio Installer

This installer installs the following components:

- Studio
- Java

This installer can be run on each Windows machine that needs access to the TDV Server.

To install Studio on a Windows computer

1. Read any README files included with or associated with the download file.
2. Run the installer executable for Studio.
3. Follow the prompts on the screen.
4. When the installation is complete, click Finish to exit the installation program.

Studio automatically runs and prompts you for login information.

Installing the Drivers

This client distribution (driver zip) file includes the following components:

- ODBC
- ADO.NET
- JDBC

- SSIS
- Power BI

This zip file can be unpacked on each machine that has client application that needs access to the TDV Server.

To install the drivers distributed with TDV

1. Read any README files included with or associated with the download file.
2. Locate and extract the drivers zip file.
3. When installing the ODBC Win 64-bit driver on Windows 10, make sure to select Run as Administrator. Select the client EXE file, right click and select Run as Administrator. When prompted, select Yes and allow the installation to run to completion.
4. Follow the instructions in the TDV Administration Guide for details on how to complete configuration of each driver.
5. When the installation is complete, click Done to exit the installation program.

Installing on UNIX

Your TDV Server can be installed on a UNIX machine. Studio is not available for UNIX and must be installed on a Windows machine. You can then connect the Studio client to the Server on the UNIX machine.

- [Installing TDV Server on UNIX](#)
- [Installing Drivers on UNIX](#)
- [Setting the TDV Server to Start Automatically on UNIX](#)

Installing TDV Server on UNIX

This installer installs the following components:

• Server	• Deployment Manager	• Repository
• Java	• Monitor	• Discovery
• Active Cluster	• Salesforce.com Adapter	• SAP Adapter
• SAPBW and BEx Adapters	• Oracle EBS Adapter	• Siebel Adapter
• Default caching database	• Advanced Data Source Adapters	

To install TDV on a UNIX computer

1. Make sure you have reviewed and completed any necessary preparation as discussed in [Installation Requirements and Support Information](#).
2. For CentOS, Red Hat Enterprise Linux, and Oracle Red Hat Enterprise Linux systems Security-Enhanced Linux (SELinux) must be enabled. Refer to the section *Configuring Security Enhanced Linux Environments* in the Security Features Guide.
3. If necessary, log into the installation machine as a non-root user. Change your working directory to the user's home directory.
4. Run the following command for your platform:

```
chmod 755 <installer file name>
```

5. Make sure that the directory and path that you expect to use for TDV does not contain any spaces.
6. Make sure that you have READ and WRITE permissions on the installation directory.
7. Run the following command to start the installation:

```
./<installer file name>
```

8. Follow the prompts on the screen. Special characters are not supported for <TDV_install_dir>.

You can select the defaults for the <TDV_install_dir> and the TDV Server base port number. The value you use for <TDV_install_dir> cannot contain a space.

You will be prompted for the following passwords...

Password Type	Description
TDV Server Application Password	The TDV Server application password is used to login to web manager.
TDV Repository	This is the database that will be used to store all of the data and metadata about the items that you create within TDV. It also stored your configuration and other environment settings. Passwords with special characters that are supported by your operating system shell are fine to use.
Default Caching Database	This is the database that will be created for you to hold data that you want to cache using the default caching method. There are multiple caching options. You might want to note the password for future use of this database.

Note: TDV will generate a new encryption key when the installation is in a new destination. If it is an existing installation TDV uses the existing encryption key. In case of any errors encountered, the administrator may have to investigate if the encryption key file location and content are correct and then contact TDV support team for assistance.

9. Finish to exit the installer when the installation is completed.

The Server starts automatically at the completion of the installation process. For information about automatically restarting TDV, see [Setting the TDV Server to Start Automatically on UNIX](#). You can also start and stop Server as described in [About the Installed TDV Services](#) and the TDV Administration Guide.

Install and uninstall logs are called bitrock_installer_<number>.log while the installer is running. After installation is complete, the logs are named <product>_install or <product>_uninstall.log. The log files can be found in the following directories:

Platform	Default Location of Log Files
Unix	/tmp

10. If installing TDV on AIX, make sure that MAX_MEMORY >1500MB is in the <TDV_install_dir>/conf/server/server.properties.

The `server.properties` file is processed every time the server is restarted from `composite.sh monitor`.

11. Optionally, download and install the latest TDV patch as described in TDV and Business Directory Release Notes.

Installing Drivers on UNIX

These files contain the following driver components:

- ODBC
- ADO.NET
- JDBC
- SSIS
- Powerr BI

To install the drivers

1. Make sure you have reviewed and completed any necessary preparation as discussed in [Installation Requirements and Support Information](#).
2. If necessary, log into the installation machine as a non-root user. Change your working directory to the user's home directory.
3. Make sure that you have READ and WRITE permissions on the directory for which you want to unzip the contents of the file.
4. Locate and extract the drivers zip file.
5. Follow the instructions in the TDV Administration Guide for details on how to complete configuration of each driver.
6. When the installation is complete, click Done to exit the installation program.

Setting the TDV Server to Start Automatically on UNIX

If at any time after installing the software, you restart the UNIX installation machine, Server and the metadata repository do NOT start automatically (unlike when they start automatically after a successful installation of the software).

To configure the TDV service files `cis.repository` and `cis.server`

1. Log into the installation machine as root.
2. Change the working directory to `<TDV_install_dir>/bin`.
3. Run the following command as the root user:

```
cis_install_services.sh
```

This command prompts for a username, and other details to install and configure the service files `cis.repository` and `cis.server`.

4. Enter the name of the user to start TDV (not the root user) and the other information requested.

The script then installs `cis.repository` and `cis.server` into an appropriate location on the installation machine and configures them. The location will be printed on your screen when the configuration is successful, so make note of this location, because you need this to perform verification of the service files.

Note: Do not run the `cis.repository` or `cis.server` scripts in the `<TDV_install_dir>/bin/` directory. These are template files used by `cis_install_services.sh` only and are not meant to be run.

Running `cis_install_services.sh` does not interrupt any repository or server processes that are running, but prepares the machine for automatically starting those processes during restart of the UNIX-based computer.

5. Run the following commands as the root user:

```
cd <init_directory>
```

```
chmod 550 cis.repository
```

```
chmod 550 cis.server
```

```
chmod 550 cis.cache
```

The value of `init_directory` depends on the operating system:

- Linux: `/etc/rc.d/init.d` or `/etc/rc.d`
- AIX: `/etc/rc.d/init.d`

To verify the TDV service files configuration

6. Go to the location noted previously from running `cis_install_services.sh`.

Note: The console output of the script `cis_install_services.sh` displays the exact location. Choose the location for your operating system.

7. Enter these commands:

```
./cis.repository restart
```

```
./cis.server restart
```

```
./cis.cache restart
```

Now if the machine is rebooted, the monitor, server, and repository processes should automatically start once the machine is ready to go.

Installing on Amazon Web Service

The TDV Server is supported on Windows and UNIX. Studio requires a Windows-based OS to operate.

To install TDV on a Windows-based AWS

1. Install and configure a supported version of Windows for AWS.
2. Select and install the AMI for TDV.

3. Follow the install instructions in [Running the TDV Server Installer](#).
4. Follow the install instructions in [Running the Studio Installer](#).
5. Follow the instructions in the TDV Administration Guide to register you TDV licenses.

To install TDV Server on a UNIX-based AWS

1. Install and configure a supported version of Linux for AWS.
2. Select and install the AMI for TDV Server.
3. Follow the install instructions in [Installing on UNIX](#).
4. Locate the TDV Studio installer that came bundled with your AMI.
5. Move the installer file to a Windows-based AWS or another Windows machine.
6. Follow the install instructions in [Running the Studio Installer](#).
7. Connect to the TDV Server on your Linux AWS.
8. Follow the instructions in the TDV Administration Guide to register you TDV licenses.

About TDV Software Patches

TDV produces service pack patches as needed to update installed products. Patches are applied after the product has been installed. A patch is a zipped package of files that fixes known issues and which often provides enhanced functionality.

You must use the component-specific patch to get fixes for each component (For example, the Studio patch has fixes for Studio, the Client driver patch has fixes for client drivers such as ODBC, ADO.Net). Refer to [Software Components for Installation](#) for a list of components.

After installation of TDV, you might want to apply the latest TDV patch which might be a later version than what you just installed. It is recommended that you install a patch on all computers running TDV products to ensure complete compatibility and minimize unforeseen problems.

For information about how to obtain and install the latest patch, see [TDV and Business Directory Product Maintenance](#).

Note: Instructions for how to install a patch or service pack are subject to change with each service pack. For instructions, see the TDV and Business Directory Release Notes.

About the Installed TDV Services

The installation process installs the following services which are TDV processes that run in the background:

- **server**—the TDV Server process.
- **repository**—the database repository used by TDV.
- **monitor**—a process that monitors the TDV Server and ensures that it is always running.
- **cache**—a process that runs the default caching database.

All processes must be running for TDV to function properly.

For more information on configuring and starting TDV, see the TDV Administration Guide.

Importing Metadata into the New TDV Instance

If you are upgrading your version of TDV from an earlier version and you have completed the instructions in [Exporting Metadata from the Existing TDV Instance](#), follow the instructions in the section [Importing Metadata into the New TDV Instance](#). To verify the installation, follow the instructions in the section [Verifying a Successful Installation](#).

Tips from an Expert if the Server Does Not Start

If the server does not start and the log files indicate that the cause is not enough heap memory, you can modify the default max memory setting.

The `server.properties` file is processed every time the server is restarted from `composite.sh monitor`.

To modify the max memory setting

1. Stop the server.
2. Increase the `MAX_MEMORY` value in the one of the following locations depending on your server:
 - `<TDV_install_dir>/conf/server/server.properties`

- <BD_install_dir>/bd/conf/server/server.properties
3. If adjusting the heap size with MAX_MEMORY is not enough to allow large CAR files to load, you can try setting the following Studio configuration parameters back to their default values:
 - Default Bytes to Fetch—Default value is 100.
 - Default Rows to Fetch—Default value is 1000.
 4. From the process manager for your platform, shut down and restart all TDV processes (such as the TDV Server and monitor).

Where to Go After Installation

For your next steps, particularly if you are new to TDV products, see the information in the following PDFs or online help. You can access the PDFs at <TDV_install_dir>/docs, or from within Studio at Help > Online Help.

Book Title	Description
Getting Started Guide	Contains a simple example to get you familiar with the Studio application.
Administration Guide	Contains procedures for: <ul style="list-style-type: none"> • Completing and configuring your TDV installation • Licensing TDV software • Starting and stopping TDV • Finding and interpreting log files • Setting up security • Setting up JDBC, ODBC, and other drivers
User Guide	Explains Studio features and how to create and publish resources.
Client Interfaces Guide	Contains instructions, guidelines, and examples of how to access TDV resources through various client applications.

Silent Mode Installation

Installations can be run without manual interactive interfaces (i.e. graphical user interface or console based). There are two ways to run the installer in silent mode: 1) using a property file with key/value pairs or 2) command line with key/value pairs.

Topics covered in this chapter include:

- [Creating the Options File for a Silent Installation](#)
- [Running the Installer in Silent Mode](#)

Creating the Options File for a Silent Installation

Optionally, when running a silent mode installation you can use an options file that has specific key-value pairs.

To create the options file for a silent install

1. In a text editor, create a options file similar to the following:

Business
Directory

```
# Modify install directory and all port number references
```

```
# mode=unattended
```

```
install_directory=/opt/TIBCO/BD
```

```
server_port=9500
```

```
repository_admin_password=password
```

```
bd_admin_password=password
```

TDV Server

```
# Modify install directory and all port number references
```

```
# mode=unattended
```

```
install_directory=/opt/TIBCO/TDV
```

```
server_port=9400
```

```
repository_admin_password=password
```

```
database_admin_password=password
```

```
server_admin_password=password
```

2. Edit the values within the file for your installation.

The following table describes the variables in the options file:

Variable	Description and Value
INSTALL_DIRECTORY	<p>Directory in which to install the software referred to as <TDV_Installdir>.</p> <p>The value can be empty, or the directory can be non-existent. On UNIX, there can be no space in the directory name. Examples:</p> <pre>install_directory=/opt/TIBCO/TDV</pre> <pre>install_directory=C:\Program Files\TIBCO\Studio</pre> <pre>install_directory=/opt/TIBCO/BD</pre>

Variable	Description and Value
REPOSITORY_ADMIN_PASSWORD	Password to access the repository database, which is automatically installed during the installation. PostgreSQL requires that the password you choose cannot contain a # or \$.
SERVER_PORT	Defaults to 9400 for TDV and 9500 for Business Directory.
DATABASE_ADMIN_PASSWORD	The password used to access the default caching database, which is automatically created during installation. PostgreSQL requires that the password you choose cannot contain a # or \$.
SERVER_ADMIN_PASSWORD	The password used to login to the web manager and the client applications.

3. Save the file as <installer.properties>.

Running the Installer in Silent Mode

Running the installer via command line options

Option file method:

- a. Create the options file. See [Creating the Options File for a Silent Installation](#) .
- b. Run the installer with the following option: <instFile>.exe/bin --optionfile <OPTION_FILE>

1. Command line (no options file) method - See examples below:

Component	Command Options
TDV Server	<ul style="list-style-type: none"> Windows Installation with all input parameters: <instFILE>.exe --mode unattended --install_directory <TDV_Installdir> --server_port "6400" --server_admin_password "admin1" --repository_admin_password "password" --database_admin_password "password" <i>Note:</i> database_admin_password is only valid for TDV Server. BD does not

Component	Command Options
	<p>use this variable.</p> <ul style="list-style-type: none"> Windows Installation with only the required parameters: <pre><instFILE>.exe --mode unattended --server _admin_password "admin1" -- repository_admin_password "password" --database_admin_password "password"</pre> <p>This command installs TDV Server in the default directory C:\Program Files\TIBCO\TDV Server <version> on default port 9400.</p> Linux/AIX Installation with all input parameters: <pre><instFILE>.bin --mode unattended --install_directory <TDV_Installdir> -- server_port "6400" --server _admin_password "admin1" --repository_admin_ password "password" --database_admin_password "password"</pre> Linux/AIX Installation with only the required parameters: <pre><instFILE>.bin --mode unattended --server _admin_password "admin1" -- repository_admin_password "password" --database_admin_password "password"</pre> <p>This command installs TDV Server in the default directory /opt/TIBCO/TDV_ Server_<version> on default port 9400.</p> <p><i>Note:</i> User should have rwx permissions on /opt</p>
Studio	<ul style="list-style-type: none"> Windows: <pre><instFILE>.exe --mode "unattended" --install_directory <TDV_Installdir></pre> Linux/AIX: <pre><instFILE>.bin --mode "unattended" --install_directory <TDV_Installdir></pre>
Business Directory	<ul style="list-style-type: none"> Windows Installation with all input parameters: <pre><instFILE>.exe --mode unattended --install_directory "<TDV_Installdir>" --bd_ admin_password "admin1" --repository_admin_password "password" -- server_port 9500</pre> Windows Installation with only the required parameters: <pre><instFILE>.exe --mode unattended --bd_admin_password "admin1" -- repository_admin_password "password"</pre> <p>This command installs BD in the default directory C:\Program Files\TIBCO\BD Server<version> on default port 9500.</p> Linux/AIX Installation with all input parameters: <pre><instFILE>.bin --mode unattended --install_directory "<TDV_Installdir>" --bd_ admin_password "admin1" --repository_admin_password "password" -- server_port 9500</pre> Linux/AIX Installation with only the required parameters: <pre><instFILE>.bin --mode unattended --bd_admin_password "admin1" -- repository_admin_password "password"</pre>

Component	Command Options
	<p>The above command installs BD in the default directory: /opt/TIBCO/TDV_BD_Server_8.0 on port 9500</p> <p><i>Note:</i> User should have rwx permissions on /opt</p>

Note: All the available options can be viewed by executing the command
 <instFILE>.exe/bin --help

The variables used in the above table are as follows:

- <instFILE> is the file name. For example, TIB_tdv_server_8.0.0_win_x86_64.exe for a Windows TDV Server.
 - <OPTION_FILE> is the name of the file where the input parameters are stored.
 - <TDV_Installdir> is the installation directory For example, /opt/TIBCO/TDV_BD_Server_8.0 for Linux/AIX BD
2. Verify that the installation was successful by looking for the TDV installation directory. You can also view success or failure messages in:
- %HOMEDRIVE%\BD_install.log (Windows) or /tmp/BD_install.log (UNIX)
 - %HOMEDRIVE%\TDV_install.log (Windows) or /tmp/TDV_install.log (UNIX)

TDV Docker Container

This section will cover the TIBCO Data Virtualization (TDV) software for the Docker container distribution format. Users will be able to build a TDV Docker image and run it as a Docker container. The following sections are described in this chapter:

[Prerequisites](#)

[Building TDV Docker Images](#)

[Publishing TDV Docker Images](#)

[Launching TDV Containers](#)

[Runtime TDV Container Configuration - Common Examples](#)

[Limitations](#)

[Upgrading to a New Version of TDV Server](#)

[Tips from Expert](#)

[Useful Docker Commands for TDV Containers](#)

Prerequisites

The following section outlines what you need to prepare for building and running the Data Virtualization Docker image.

Note: The TDV product does not provide a Docker image. You must build it explicitly.

Before building the Docker image, ensure the following:

Linux, macOS, and Windows platforms

Install the latest version of the Docker Engine 19.03.5 or higher (<https://docs.docker.com/get-docker/>). This is the build and runtime environment you will use to create the Data Virtualization Docker images and containers.

Use the following link to all non-root users to run Docker commands (<https://docs.docker.com/install/linux/linux-postinstall>) in your Linux Docker runtime environment.

Note: Linux based operating system are the most popular for hosting Docker runtime environments. If you are not using, that then look at the Windows OS or macOS alternatives below.

Windows platform

TDV supports Windows 11 Pro 64bit 21H2 or higher and Windows Server 2022 21H2 or higher. Some of the key considerations when using the docker desktop for windows are given below:

Hyper-V feature must be enabled on Windows (<https://learn.microsoft.com/en-us/windows-server/virtualization/hyper-v/system-requirements-for-hyper-v-on-windows>)

References:

- Windows 11 : <https://techcommunity.microsoft.com/t5/educator-developer-blog/step-by-step-enabling-hyper-v-for-use-on-windows-11/ba-p/3745905>
- Windows Server 2022: <https://learn.microsoft.com/en-us/windows-server/virtualization/hyper-v/hyper-v-technology-overview>

Use the Docker desktop for Windows to create the TDV docker images and containers. The latest version of Docker Desktop for Windows can be installed from here - <https://docs.docker.com/desktop/install/windows-install/>

Listed below are some of the important considerations, If you are using the docker desktop:

- WSL 2 feature must be enabled on Windows (<https://docs.docker.com/desktop/windows/wsl/>)
- During Docker Desktop for Windows installation, choose “Use WSL 2 instead of Hyper-V” option.
- Enable the “use Linux containers” configuration as outlined here: <https://docs.docker.com/desktop/get-started/#switch-between-windows-and-linux-containers>

If you choose to run the TDV docker build and run scripts in Docker Desktop for Windows, then you must open a WSL shell and run with bash.

Note: Please modify the `run_tdv_cluster_container.sh` script before running it. Add "bash" prefix in front of every `"/run_tdv_container.sh"` reference in that script.

Some examples of running the TDV script inside the WSL shell are given below:

Build example

```
$ bash build_tdv_image.sh
```

Run single instance example

```
$ bash run_tdv_container.sh -d --del-vol --env TDV_ADMIN_PASSWORD=123456
```

Run cluster example

```
$ bash run_tdv_cluster_container.sh -d --del-vol --nodes 2 --env TDV_
ADMIN_PASSWORD=123456
```

The TDV Docker container will require a minimum of 8GB of RAM and 8GB of read & write persistent disk storage. For additional resource requirements see [TDV](#)

macOS platform

Latest version of Docker Desktop for Mac (<https://docs.docker.com/desktop/install/mac-install/>)

Note : To allow non-root users to run Docker commands, follow the instructions in <https://docs.docker.com/install/linux/linux-postinstall>

Docker

Docker Engine 19.03.5 or higher is required for building/running a TDV Docker image/container.

Verifying Docker Installation

- In the command prompt, run the command:

```
$ docker run hello-world.
```

- If you cannot run this default Docker container, then check your Docker installation. Being able to run this default Docker container ensures your Docker environment is ready for building and running TDV.

Note: When running TDV with a Docker container, Docker will manage the TDV Server process lifecycle, therefore the TDV Monitor Daemon will not be running inside the TDV container. You will notice that the `cs_csmonitor_server.log` displays errors. This is because the monitor server is attempting to connect to a monitor daemon which does not exist in a docker environment. It is safe to ignore this error message.

TDV

Resource Requirements for TDV

Storage, CPU and memory resources should to be specified when building a TDV Docker image and when running a TDV Docker container.

- **Storage (runtime)** - TDV requires read/write persistent disk storage that is maintained outside of the Docker container. See [Sizing Guidelines for TDV](#) for recommendations.
- **CPU (runtime)** - TDV requires a minimum of 2 CPUs/cores per TDV Docker container. See [Sizing Guidelines for TDV](#) for recommendations.
- **Memory (build time)** - TDV requires a minimum of 8 GB memory per TDV Docker container. See [Sizing Guidelines for TDV](#) for recommendations.

Building TDV Docker Images

TDV is packaged with a quick start container build script “`build_tdv_image.sh`”. You can download this script from the edelivery site:

edelivery.tibco.com.

Alternatively, you can also use build without that script by just directly using the TDV Dockerfiles and corresponding tar.gz file that are provided on - edelivery.tibco.com alongside the other TDV distributions (e.g. installer and patch). Follow the steps given in the sections below, to build the TDV Docker Images.

Once you build the TDV Docker image, you can use private clouds (i.e. on-premises) or public clouds such as AWS and Azure to run the TDV application as a container. For information on deploying docker containers refer to the following documentation:

Manage TDV Image on AWS or Azure:

<https://docs.aws.amazon.com/AmazonECR/latest/userguide/what-is-ecr.html>

<https://docs.microsoft.com/en-us/azure/container-registry/container-registry-get-started-docker-cli?tabs=azure-cli>

Deploying TDV Container on AWS:

<https://docs.aws.amazon.com/AmazonECS/latest/developerguide/docker-basics.html>

<https://docs.docker.com/cloud/ecs-integration/>:

Deploying docker containers on Azure:

<https://docs.docker.com/cloud/aci-integration/>

Using Quick Start Script

Follow these steps, to build the TDV Docker Images using the quick start script:

1. Download the build_tdv_image.sh on your instance for building Docker images.
2. Run the following command:

```
$ ./build_tdv_image.sh [-hv]
[-c] [-f <file>] [-n <name>:<tag>]
```

The table below gives a description of the different parameters used with the script:

Parameter	Description	Comments
-h	Help for using the script.	
-v	Enable verbose mode output.	
-c	Enable build caching.	default is "docker build --no-cache"
-f	Location of Dockerfile.tdv	

Parameter	Description	Comments
-n	Set name:tag for image.	default is "tibco/tdv:<version>"

Using Docker Build

Follow these steps, to build the TDV Docker Images using the docker build command:

1. Download the docker container files Dockerfile.tdv, Dockerfile.tdv.repo, Dockerfile.tdv.cache and TIB_tdv_<TDV_VERSION>_docker.tar.gz on your instance for building Docker images. **The edelivery has separate archive files for each distribution. This archive is only for the Docker distribution.**
2. You should also download the md5 checksum files in order to validate the integrity of these files.

Example:

```
md5sum -c Dockerfile.tdv.md5
```

```
md5sum -c TIB_tdv_<TDV_VERSION>_docker.tar.gz.md5
```

3. Goto to the location where you have downloaded the TDV Docker files (Dockerfile.tdv, Dockerfile.tdv.repo, Dockerfile.tdv.cache) and tar.gz file and run the Docker build command as shown below.

Generic example:

```
$ docker build -t <your-docker-repo-name>/<image_name>[:<image-tag>] -f Dockerfile.tdv
```

TDV default build example: (default TDV base port 9400 and default TDV server memory 7 GB):

```
$ docker build -t myrepo/tdvrepo:<TDV_VERSION> -f Dockerfile.tdv.repo
```

```
$ docker build -t myrepo/tdvcache:<TDV_VERSION> -f Dockerfile.tdv.cache
```



```
$ docker build -t myrepo/tdv:<TDV_VERSION> -f Dockerfile.tdv
```

Publishing TDV Docker Images

If you want to publish your TDV Docker image to a Docker registry then log in to your Docker registry and follow the commands below.

For more information on Docker login, visit <https://docs.docker.com/engine/reference/commandline/login>

After a successful login, you can publish the Docker image using the docker push command.

Note: The example below shows the publishing command for one image. You will have to run the “docker tag” and “docker push” commands for each image (the repo, cache and server).

Generic Command Usage:

```
$ docker login --username <user-name> <remote-repo-name>
```

```
$ docker tag <local-repo-name>/<image-name>:<image-tag> <remote-repo-name>/<image-name>[:<image-tag>]
```

```
$ docker push <remote-repo-name>/<image_name>[:<image-tag>]
```

```
$ docker logout <remote-repo-name>
```

Launching TDV Containers

TDV is pre-packaged with the following deployment tools to help users to launch TDV Containers:

- Quick start container deploy script “run_tdv_container.sh” (for single node)

- Quick start container deploy script “run_tdv_cluster_container.sh” (for cluster modes)
- Docker-Compose TDV example "docker-compose-tdv.yml"

You can download these files from the

edelivery.tibco.com site.

Alternately, you can also use the TDV Docker files and the corresponding tar.gz file that are available in the site - edelivery.tibco.com alongside the other TDV distributions (e.g. installer and patch). Follow the steps given in the sections below, to deploy the TDV Docker Images.

This section will explain how to start a TDV Docker container. If you need to review TDV container sizing guidelines refer [Sizing Guidelines for TDV](#)

TDV Admin Password

There is no default TDV Admin password for the docker container and specifying an Admin Password is required when launching a TDV container. You can specify the password in one of the following ways:

1. By setting an environment variable TDV_ADMIN_PASSWORD prior to running the quick script or the “docker run” command:

```
$ export TDV_ADMIN_PASSWORD=<VALUE>;
```

2. By using the option “-e”:

```
$ docker run -e TDV_ADMIN_PASSWORD
```

Note: Refer to the sections [Launching TDV Containers \(Single Node\) - Using Quick Start Script](#) and [Launching TDV Containers \(Cluster Nodes\) - Using Quick Start Script](#) for the usage of the “-e” option.

3. By storing the password in a file and using that file while launching the container:

```
$ docker run -env TDV_ADMIN_PASSWORD_FILE=<FILE> # The file  
should have the TDV_ADMIN_PASSWORD=<VALUE> set.
```

Note: You can also store all the docker run parameters in an environment file and use the file while launching the container:

```
$ docker run --env-file <FILE>. # <FILE> The file should have the TDV_
ADMIN_PASSWORD=<VALUE> set along with other required parameters.
```

Launching TDV Containers (Single Node) - Using Quick Start Script

Follow these steps to run the script and launch the TDV container using the quick start script:

1. Download the run_tdv_container.sh on your instance for launching the Docker container.

Note: For Docker Desktop Window users, you must run “bash run_tdv_container.sh” from a WSL shell if you want to run this script.

2. Run the following command:

```
$ ./run_tdv_container.sh [-hv] [--dry-run][--skip-wait]
[-d] [--del-vol] # deletion settings (container and volumes)
[-c <cpus>] [-i <name>:<tag>] [-m <memory>] [-n <network>] [--
name <name>] [-p <port>] [--vol <volume>] #container settings
[-e <TDV_ARG> ... ] [--env <TDV_ARG>=<TDV_VALUE> ... ] [--env-
file <file>] # TDV runtime configuration [--cluster-node] #
cluster options
```

The table below gives a description of the different parameters used with the script.

Parameter	Description	Comments
--dry-run	Show output of execution without actually executing the script.	
--skip-wait	Skip wait check for TDV container.	
-h	Help for using the script.	

Parameter	Description	Comments
-v	Enable verbose mode output.	
Deletion Settings (Container and Volume)		
-d	Delete and stop TDV container if it already exists with same <name>:<tag>.	
--del-vol	Delete and stop TDV container if it already exists with same <name>:<tag>.	
Container Settings		
	Set the number of CPUs for container.	Use decimal or positive whole number form. Default is "2.0". "1.0" CPU is the minimum. Any value below that will result in an error.
-i	Set <name>:<tag> for image to use for container.	default is "tibco/tdv:version"
-m (memory)	Set the amount of memory for container.	Default is 7GB. 4GB is the minimum. Any value below that will result in an error. Note: If this is set and -e TDV_MAX_MEMORY is not, then TDV_MAX_MEMORY defaults to the value mentioned in this option.
-n <network>	Set network for container.	Default is "tdv-bridge". If <network> doesn't exist then an error will result.
--name	Set <name> for container.	Default is "tdv<version>".

Parameter	Description	Comments
-p <port>	Set base host port for container	Default is "9400".
--vol	Set <volume> for container	default is "tdv<version>-vol"
TDV Runtime Configuration		
--env <TDV_ARG>=<TDV_VALUE>	Pass TDV parameters to container for runtime configuration.	<p>Required Settings:</p> <p>--env TDV_ADMIN_PASSWORD=<PASSWORD> # Set TDV admin password for all containers (i.e. tdv, cache and repo).</p> <p>or</p> <p>--env TDV_ADMIN_PASSWORD_FILE=<clear text password in file> # (optional) Use file to store TDV admin password. Specified file remapped to /run/secrets/tdv-admin-password in container.</p> <p>Note: only TDV_ADMIN_PASSWORD or TDV_ADMIN_PASSWORD_FILE can be specified. Setting both is not allowed.</p> <p>Optional Settings:</p> <p>--env TDV_BASE_PORT=9400 # (optional) Change TDV Server base port.</p> <p>--env TDV_MAX_MEMORY=7 # (optional) Change TDV Server memory value (GB).</p> <p>If this is set and -m <memory> is not, -m <memory> defaults to TDV_MAX_MEMORY + 1 GB.</p>
-e <TDV_ARG>	Pass TDV parameters to container for runtime	The variables used in the argument of this option are environment variables

Parameter	Description	Comments
	configuration, using defined environment variables.	defined and assigned values ahead of the usage.
--env-file <file>	Pass TDV parameters file to container for runtime configuration.	
Special Options		
--cluster-node	Enable TDV container configuration to be a TDV cluster node.	Default is to enable a standalone TDV single node.

Note: By default, the script runs the docker command with the “---restart” option set as “unless-stopped”. It means “Restart the container if it stops, except that when the container is stopped (manually or otherwise), it is not restarted even after Docker daemon restarts.”

Launching TDV Containers (Single Node) - Using Docker Compose

Docker Compose is a tool used for running multi-container Docker applications. With this tool, you make use of a YAML file to configure your application’s services.

The Docker Compose version for TDV is 1.24 or higher and the Docker Compose File Format version is 3.5 or higher.

The docker-compose YAML file is available in the [edelivery site](#). Follow these steps to run tool:

1. Download the docker-compose-tdv.yml file on your instance for building Docker images.
2. Run the following command to run the docker-compose tool:

```
$ docker-compose -f docker-compose-tdv.yml up -d
```

Launching TDV Containers (Single Node) - Using Docker Run

This section will explain how to start a TDV Docker container (Single Node). If you need to review TDV container sizing guidelines refer [Sizing Guidelines for TDV](#)

General examples for launching a single node TDV Docker container

Below is a generic example for launching a single node docker container. Note that you must execute the “docker run” command for all 3 TDV containers (repo, cache and server).

```
$ docker volume create --name tdv8.5.0clustercache-vol 2>&1 > /dev/null
```

```
$ docker volume create --name tdv8.5.0clusterrepo-vol 2>&1 > /dev/null
```

```
$ docker volume create --name tdv8.5.0cluster-vol 2>&1 > /dev/null
```

```
$ docker network create --internal tdv-bridge-internal
```

```
$ docker network create tdv-bridge
```

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge -p 9404:9404 -v tdv0.0clustercache-
vol:/var/lib/postgresql/data --env POSTGRES_USER=<user> --env POSTGRES_
PASSWORD=<tdv admin password> --env POSTGRES_DB=postgres --env POSTGRES_
INITDB_ARGS=-E UTF-8 --env POSTGRES_HOST_AUTH_METHOD=password --env
PGDATA=/var/lib/postgresql/data/tdv --name tdv0.0clustercache
tibco/tdvcache:8.5.0 postgres
```

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge -p 9408:9408 -v tdv0.0clusterrepo-
vol:/var/lib/postgresql/data --env POSTGRES_USER=<user> --env POSTGRES_
PASSWORD=<tdv admin password> --env POSTGRES_DB=postgres --env POSTGRES_
INITDB_ARGS=-E UTF-8 --env POSTGRES_HOST_AUTH_METHOD=password --env
PGDATA=/var/lib/postgresql/data/tdv --name tdv0.0clusterrepo
tibco/tdvrepo:8.5.0 postgres
```

```
$ docker run -itd --cpus 2.0 --restart unless-stopped --memory 8g --
network tdv-bridge -p 9400-9403:9400-9403 -p 9405:9405 -p 9409:9409 -p
9300-9306:9300-9306 -p 9407:9407 -v tdv0.0cluster-vol:/opt/TIBCO --name
tdv0.0cluster --env TDV_ADMIN_PASSWORD=<tdv admin password> --env TDV_
MAX_MEMORY=7000 --env TDV_CACHE_HOSTNAME=tdv0.0clustercache --env TDV_
REPO_HOSTNAME=tdv0.0clusterrepo tibco/tdv:0.0 tdv.server
```

To configure node 1 as timekeeper node on "tdv0.0cluster:9400" with cluster name "tdv0.0cluster":

```
$ docker exec -it tdv8.5.0cluster /bin/bash -c $TDV_INSTALL_
DIR/bin/cluster_util.sh -server tdv8.5.0cluster -port 9400 -user <user>
-password <password> -create -clusterName tdv8.5.0cluster
```

Note: When launching a container, you can either give a password or use a password file that has the password stored in it. Refer to the section [TDV Admin Password](#) for ways to specify the Admin Password.

References:

Refer the table below for a description of the different options used in the above **docker run** command.

Option	Docker Help Reference
--restart	Used to configure the restart policy for a container. Refer https://docs.docker.com/config/containers/start-containers-automatically/
-t	Allocate a pseudo-tty - https://docs.docker.com/engine/reference/run/
-i	Keep STDIN open even if not attached - https://docs.docker.com/engine/reference/run/
-d	Detach and run the container in background and print container ID - https://docs.docker.com/engine/reference/run/#detached--d
-v	(TDV Required) The tdv container requires a persistent storage area when running as a Docker container. See Sizing Guidelines for TDV for size

Option	Docker Help Reference
	<p>recommendations.</p> <p>https://docs.docker.com/storage/bind-mounts/</p> <p>Usage: -v <path to the file on the host machine>:<path where the file is mounted in the container>:<optional parameters></p> <p>Example: -v \$CONTAINER_CACHE_VOLUME:\$TDV_DATABASE_DIR</p> <p>Note: mount point must have a valid volume existing before starting the TDV Container.</p>
	<p>(TDV Recommended) The tdv container works best with 2 CPUs/cores in general. See Sizing Guidelines for TDV for value recommendations.</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>
-m	<p>(TDV Required) The tdv container requires a minimum of 8GB of memory. Higher tdv workloads require more. See Sizing Guidelines for TDV for value recommendations.</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>
<tdv-container-name>	<p>Container name for your TDV Docker container. Recommendation is to have tdv in the name. Examples: tdv, tdv-1, tdv-2, tdv-dev, tdv-prod, etc</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>
<repo-name>	<p>Repository name for your TDV Docker image.</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>
<image-name>	<p>Recommendation is to use tdv. You can change this to any name.</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>
<image-tag>	<p>Recommendation is to use the TDV version for this. Example: 8.4</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>

Linux

This section explains how to start a TDV Docker container on a Docker environment hosted on the Linux platform. The bridge, host and user specified bridge network options in Docker should work for a TDV Container on this platform.

TDV Docker container example

Resource Configuration: small (poc/demo) : 2 CPUs/cores, 8 GB memory, external container volume tdv-vol with 8GB persistent readable/writable storage.

TDV configuration: base port (9400), admin password (mandatory), server memory (default). Refer to the docker container files (Dockerfile.tdv, Dockerfile.tdv.repo and Dockerfile.tdv.cache) for TDV Docker image default values.

```
$ docker volume rm -f tdvcache8.6.1-vol
```

```
$ docker volume create --name "tdvcache8.6.1-vol"
```

```
$ docker volume rm -f tdvrepo8.6.1-vol
```

```
$ docker volume create --name "tdvrepo8.6.1-vol"
```

```
$ docker volume rm -f tdv8.6.1-vol
```

```
$ docker volume create --name "tdv8.6.1-vol"
```

```
$ docker network create --internal tdv-bridge-internal
```

```
$ docker network create tdv-bridge
```

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 9404:9404 -v tdvcache8.6.1-
vol:/var/lib/postgresql/data --env POSTGRES_USER=root --env POSTGRES_
PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_INITDB_ARGS="-
E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
```

```
PGDATA="/var/lib/postgresql/data/tdv" --name tdvcache8.6.1
tibco/tdvcache:8.6.1 postgres
```

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 9408:9408 -v tdvrepo8.6.1-
vol:/var/lib/postgresql/data --env POSTGRES_USER=root --env POSTGRES_
PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_INITDB_ARGS="-
E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdvrepo8.6.1
tibco/tdvrepo:8.6.1 postgres
```

```
$ docker run -itd --cpus 2.0 --restart unless-stopped --memory 8g --
network tdv-bridge-internal -p 9400-9403:9400-9403 -p 9405:9405 -p
9409:9409 -p 9300-9306:9300-9306 -v tdv8.6.1-vol:/opt/TIBCO --name
tdv8.6.1 --env TDV_ADMIN_PASSWORD=admin1 --env TDV_MAX_MEMORY=7000 --env
TDV_CACHE_HOSTNAME=tdvcache8.6.1 --env TDV_REPO_HOSTNAME=tdvrepo8.6.1
tibco/tdv:8.6.1 tdv.server
```

```
$ docker network connect tdv-bridge tdv8.6.1
```

MacOS

This section explains how to start a TDV Docker container on a Docker environment hosted on the MacOS platform. The bridge and user specified bridge network options in Docker should work for a TDV Container on this platform.

TDV Docker container example

Resource configuration: small (poc/demo): 2 CPUs/cores, 8 GB memory, external container volume tdv-vol with 8GB persistent readable/writable storage.

MacOS specific configuration: -p <host-port>:<container-port> for all DV ports exposed and --hostname=localhost

TDV configuration: base port (9400), admin password (mandatory), server memory (default). Refer to the docker container files (Dockerfile.tdv, Dockerfile.tdv.repo and Dockerfile.tdv.cache) for TDV Docker image default values.

```
$ docker volume rm -f tdvcache8.6.1-vol
```

```
$ docker volume create --name "tdvcache8.6.1-vol"
```

```
$ docker volume rm -f tdvrepo8.6.1-vol
```

```
$ docker volume create --name "tdvrepo8.6.1-vol"
```

```
$ docker volume rm -f tdv8.6.1-vol
```

```
$ docker volume create --name "tdv8.6.1-vol"
```

```
$ docker network create --internal tdv-bridge-internal
```

```
$ docker network create tdv-bridge
```

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 9404:9404 --hostname=localhost -v
tdvcache8.6.1-vol:/var/lib/postgresql/data --env POSTGRES_USER=root --
env POSTGRES_PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_
INITDB_ARGS="-E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdvcache8.6.1
tibco/tdvcache:8.6.1 postgres
```

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 9408:9408 --hostname=localhost -v
tdvrepo8.6.1-vol:/var/lib/postgresql/data --env POSTGRES_USER=root --env
POSTGRES_PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_
INITDB_ARGS="-E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdvrepo8.6.1
tibco/tdvrepo:8.6.1 postgres
```

```
$ docker run -itd --cpus 2.0 --restart unless-stopped --memory 8g --
network tdv-bridge-internal -p 9400-9403:9400-9403 -p 9405:9405 -p
9409:9409 -p 9300-9306:9300-9306 --hostname=localhost -v tdv8.6.1-
vol:/opt/TIBCO --name tdv8.6.1 --env TDV_ADMIN_PASSWORD=admin1 --env
TDV_MAX_MEMORY=7000 --env TDV_CACHE_HOSTNAME=tdvcache8.6.1 --env TDV_
REPO_HOSTNAME=tdvrepo8.6.1 tibco/tdv:8.6.1 tdv.server
```

```
$ docker network connect tdv-bridge tdv8.6.1
```

References:

For Docker Desktop for Mac, refer <https://docs.docker.com/docker-for-mac/networking/>

For larger TDV size configurations refer to [Sizing Guidelines for TDV](#).

Note: If you have issues connecting to your TDV Docker container (specifically accessing TDV via localhost and TDV port), then you may need to add "--hostname=<ip-or-hostname>" in addition to the other docker run options. The --hostname parameter sets the IP address or Hostname that the server listens to for client connections. This command may take a few seconds to execute.

Windows

This section explains how to start a TDV Docker container on a Docker environment hosted on the Windows platform. The bridge and user specified bridge network options in Docker should work for a TDV Container on this platform.

TDV Docker container example

Resource configuration: small (poc/demo): 2 CPUs/cores, 8 GB memory, external container volume tdv-vol with 8 GB persistent readable/writable storage.

Windows specific configuration: -p <host-port>:<container-port> for all DV ports exposed and --hostname=localhost or --hostname=<ip-or-hostname>

TDV configuration: base port (9400), admin password (mandatory), server memory (default). Refer to the docker container files (Dockerfile.tdv, Dockerfile.tdv.repo and Dockerfile.tdv.cache) for TDV Docker image default values.

```
$ docker volume rm -f tdvcache8.6.1-vol
```

```
$ docker volume create --name "tdvcache8.6.1-vol"
```

```
$ docker volume rm -f tdvrepo8.6.1-vol
```

```
$ docker volume create --name "tdvrepo8.6.1-vol"
```

```
$ docker volume rm -f tdv8.6.1-vol
```

```
$ docker volume create --name "tdv8.6.1-vol"
```

```
$ docker network create --internal tdv-bridge-internal
```

```
$ docker network create tdv-bridge
```

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 9404:9404 -v tdvcache8.6.1-
vol:/var/lib/postgresql/data --env POSTGRES_USER=root --env POSTGRES_
PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_INITDB_ARGS="-
E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdvcache8.6.1
tibco/tdvcache:8.6.1 postgres
```

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 9408:9408 -v tdvrepo8.6.1-
vol:/var/lib/postgresql/data --env POSTGRES_USER=root --env POSTGRES_
PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_INITDB_ARGS="-
E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdvrepo8.6.1
tibco/tdvrepo:8.6.1 postgres
```

```
$ docker run -itd --cpus 2.0 --restart unless-stopped --memory 8g --
network tdv-bridge-internal -p 9400-9403:9400-9403 -p 9405:9405 -p
9409:9409 -p 9300-9306:9300-9306 -v tdv8.6.1-vol:/opt/TIBCO --name
tdv8.6.1 --env TDV_ADMIN_PASSWORD=admin1 --env TDV_MAX_MEMORY=7000 --env
TDV_CACHE_HOSTNAME=tdvcache8.6.1 --env TDV_REPO_HOSTNAME=tdvrepo8.6.1
tibco/tdv:8.6.1 tdv.server
```

```
$ docker network connect tdv-bridge tdv8.6.1
```

References:

For Docker Desktop for Windows, refer <https://docs.docker.com/docker-for-windows/networking/>

For larger TDV size configurations refer [Sizing Guidelines for TDV](#)

Note:

The TDV Docker image uses a Linux base OS. This means if you are running Docker on Windows then you need to make sure Docker is set to use Linux containers. For more details, see <https://docs.docker.com/docker-for-windows/#switch-between-windows-and-linux-containers>.

If you have issues connecting to your TDV Docker container (specifically accessing TDV via localhost and TDV port), then you may need to add "--hostname=<ip-or-hostname>" in addition to the other docker run options. The --hostname parameter sets the IP address or Hostname that the server listens to for client connections. This command may take a few seconds to execute.

Launching TDV Containers (Cluster Nodes) - Using Quick Start Script

Follow these steps to run the script and launch the TDV Cluster container using the quick start script:

1. Download the run_tdv_cluster_container.sh on your instance for launching the Docker container.
2. Run the following command:

```
$ ./run_tdv_cluster_container.sh [--dry-run] [--skip-wait] -h # -
h for help
[-hv] [--nodes <number>] [-d] [--del-vol] # deletion settings
(container and volumes)[-c <cpus>] [-i <name>:<tag>] [-m
<memory>] [-n <network>] [--name <name>] [-p <port>] [--vol
<volume>] # container settings [-env <TDV_ARG>=<TDV_VALUE> ... ]
[--env-file <file>] # TDV runtime configuration
```

The table below gives a description of the different parameters used with the script.

Parameter	Description	Comments
--dry-run	Show output of execution without actually executing the	

Parameter	Description	Comments
	script.	
--skip-wait	Skip wait check for TDV container.	
-h	Help for using the script.	
-v	Enable verbose mode output.	
--nodes <number>	Set the number of TDV cluster node containers to create	Default number of nodes is "1". Minimum number of nodes is "1".
Deletion Settings (Container and Volume)		
-d	Delete and stop TDV containers (tdv, cache, and repo) if already exists with same <name>:<tag>.	
--del-vol	Delete TDV volumes (tdv, cache, and repo) associated with container name <name>:<tag>.	
Container Settings		
	Set the number of cpus for container (decimal or positive whole number formats)	Default is "2.0". "1.0" cpu is the minimum. Any value below that will result in an error. Note: cache and repo containers are hardcoded to use 1 cpu only.
-i	Set <name>:<tag> for image to use for container.	default is "tibco/tdv:0.0" Note: cache and repo image names will be changed to be similar to <name> and <tag> accordingly.

Parameter	Description	Comments
-m (memory)	Set the amount of memory for container.	<p>Default is 7 GB. 4 GB is the minimum. Any value below that will result in an error.</p> <p>Note: If this is set and -e TDV_MAX_MEMORY is not, then TDV_MAX_MEMORY defaults to the value mentioned in this option.</p> <p>cache and repo containers are hardcoded to use 2g memory only.</p>
-n <network>	Set network for container.	<p>Default is "tdv-bridge"</p> <p>If <network> doesn't exist then an error will result.</p>
--name	Set <name> for container.	<p>Default is "tdv0.0cluster"</p> <p>Note: cache and repo container names will be changed to be similar to <name>.</p> <p>cache and repo container volume names will also be changed to be similar to <name> if --vol is not specified.</p>
-p <port>	Set base host port for container	Default is "9400".
--vol	Set <volume> for container	default is "tdv<version>-vol"
TDV Runtime Configuration		
--env <TDV_ARG>=<TDV_VALUE>	Pass TDV parameters to container for runtime configuration.	<p>Required Settings:</p> <p>--env TDV_ADMIN_PASSWORD=<PASSWORD> # Set TDV admin password for all containers (i.e. tdv, cache and repo).</p> <p>or</p>

Parameter	Description	Comments
		<p>--env TDV_ADMIN_PASSWORD_FILE=<clear text password in file> # (optional) Use file to store TDV admin password. Specified file remapped to /run/secrets/tdv-admin-password in container.</p> <p>Note: only TDV_ADMIN_PASSWORD or TDV_ADMIN_PASSWORD_FILE can be specified. Setting both is not allowed.</p> <p>Optional Settings:</p> <p>-env TDV_BASE_PORT=9400 # (optional) Change TDV Server base port.</p> <p>-env TDV_MAX_MEMORY=7 # (optional) Change TDV Server memory value (GB).</p> <p>If this is set and -m <memory> is not, -m <memory> defaults to TDV_MAX_MEMORY + 1 GB.</p>
-e <TDV_ARG>=<TDV_VALUE>	Pass TDV parameters to container for runtime configuration, using defined environment variables.	The variables used in the argument of this option are environment variables defined and assigned values ahead of the usage.
--env-file <file>	Pass TDV parameters file to container for runtime configuration.	

Launching TDV Containers (Cluster) - Using Docker Run

This section will explain how to start a TDV Docker container in a DV Cluster configuration. If you need to review TDV container sizing guidelines refer [Sizing Guidelines for TDV](#). For further information regarding the TDV Cluster, refer to the TDV Active Cluster Guide.

General example for launching two TDV Docker containers to create a TDV Cluster

Below is a generic example for launching two docker containers to create a TDV Cluster. Note that you must execute the “docker run” command for all 3 TDV containers (repo, cache and server).

For Node 1:

```
$ docker volume rm -f tdv-cluster-cache-vol
```

```
$ docker volume create --name "tdv-cluster-cache-vol"
```

```
$ docker volume rm -f tdv-cluster-repo-vol
```

```
$ docker volume create --name "tdv-cluster-repo-vol"
```

```
$ docker volume rm -f tdv-cluster-vol
```

```
$ docker volume create --name "tdv-cluster-vol"
```

```
$ docker network create --internal tdv-bridge-internal
```

```
$ docker network create tdv-bridge
```

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 9404:9404 -v tdv-cluster-cache-
vol:/var/lib/postgresql/data --env POSTGRES_USER=root --env POSTGRES_
PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_INITDB_ARGS="-
E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdv-cluster-cache
tibco/tdvcache:8.5.5 postgres
```

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 9408:9408 -v tdv-cluster-repo-
vol:/var/lib/postgresql/data --env POSTGRES_USER=root --env POSTGRES_
```

```
PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_INITDB_ARGS="-
E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdv-clusterrepo
tibco/tdvrepo:8.5.5 postgres
```

```
$ docker run -itd --cpus 2.0 --restart unless-stopped --memory 8g --
network tdv-bridge-internal -p 9400-9403:9400-9403 -p 9405:9405 -p
9409:9409 -p 9300-9306:9300-9306 -p 9407:9407 -v tdv-cluster-
vol:/opt/TIBCO --name tdv-cluster --env TDV_ADMIN_PASSWORD=admin1 --env
TDV_MAX_MEMORY=7000 --env TDV_CACHE_HOSTNAME=tdv-clustercache --env TDV_
REPO_HOSTNAME=tdv-clusterrepo tibco/tdv:8.5.5 tdv.server
```

```
$ docker network connect tdv-bridge tdv-cluster
```

To configure node 1 as timekeeper node on "tdv-cluster:9400" with cluster name "tdv-cluster":

```
$ docker exec -it tdv-cluster /bin/bash -c “\${TDV_INSTALL_
DIR/bin/cluster_util.sh -server tdv-cluster -port 9400 -user admin -
password admin1 -create -clusterName tdv-cluster”
```

For Node 2

```
$ docker volume rm -f tdv-cluster2cache-vol
```

```
$ docker volume create --name "tdv-cluster2cache-vol"
```

```
$ docker volume rm -f tdv-cluster2repo-vol
```

```
$ docker volume create --name "tdv-cluster2repo-vol"
```

```
$ docker volume rm -f tdv-cluster2-vol
```

```
$ docker volume create --name "tdv-cluster2-vol"
```

```
$ docker network create --internal tdv-bridge-internal
```

```
$ docker network create tdv-bridge
```

For Linux and Windows:

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 10404:9404 -v tdv-cluster2cache-
vol:/var/lib/postgresql/data --env POSTGRES_USER=root --env POSTGRES_
PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_INITDB_ARGS="-
E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdv-cluster2cache
tibco/tdvcache:8.5.5 postgres
```

For Mac:

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 10404:9404 --hostname=localhost -v tdv-
cluster2cache-vol:/var/lib/postgresql/data --env POSTGRES_USER=root --
env POSTGRES_PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_
INITDB_ARGS="-E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdv-cluster2cache
tibco/tdvcache:8.5.5 postgres
```

For Linux and Windows:

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 10408:9408 -v tdv-cluster2repo-
vol:/var/lib/postgresql/data --env POSTGRES_USER=root --env POSTGRES_
PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_INITDB_ARGS="-
E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdv-cluster2repo
tibco/tdvrepo:8.5.5 postgres
```

For Mac:

```
$ docker run -itd --cpus 1.0 --restart unless-stopped --memory 2g --
network tdv-bridge-internal -p 10408:9408 --hostname=localhost -v tdv-
cluster2repo-vol:/var/lib/postgresql/data --env POSTGRES_USER=root --env
POSTGRES_PASSWORD=admin1 --env POSTGRES_DB=postgres --env POSTGRES_
INITDB_ARGS="-E UTF-8" --env POSTGRES_HOST_AUTH_METHOD="password" --env
PGDATA="/var/lib/postgresql/data/tdv" --name tdv-cluster2repo
tibco/tdvrepo:8.5.5 postgres
```

For Linux and Windows:

```
$ docker run -itd --cpus 2.0 --restart unless-stopped --memory 8g --
network tdv-bridge-internal -p 10400-10403:9400-9403 -p 10405:9405 -p
10409:9409 -p 10300-10306:9300-9306 -p 10407:9407 -v tdv-cluster2-
vol:/opt/TIBCO --name tdv-cluster2 --env TDV_ADMIN_PASSWORD=admin1 --env
TDV_MAX_MEMORY=7000 --env TDV_CACHE_HOSTNAME=tdv-cluster2cache --env
TDV_REPO_HOSTNAME=tdv-cluster2repo tibco/tdv:8.5.5 tdv.server
```

For Mac:

```
$ docker run -itd --cpus 2.0 --restart unless-stopped --memory 8g --
```

```
network tdv-bridge-internal -p 10400-10403:9400-9403 -p 10405:9405 -p
10409:9409 -p 10300-10306:9300-9306 -p 10407:9407 --hostname=localhost -v
tdv-cluster2-vol:/opt/TIBCO --name tdv-cluster2 --env TDV_ADMIN_
PASSWORD=admin1 --env TDV_MAX_MEMORY=7000 --env TDV_CACHE_HOSTNAME=tdv-
cluster2cache --env TDV_REPO_HOSTNAME=tdv-cluster2repo tibco/tdv:8.5.5
tdv.server
```

```
$ docker network connect tdv-bridge tdv-cluster2
```

To configure node 2 as a non timekeeper node on "tdv-cluster2:9400" with cluster name "tdv-cluster".

```
$ docker exec -it tdv-cluster2 /bin/bash -c "\$TDV_INSTALL_
DIR/bin/cluster_util.sh -server tdv-cluster2 -port 9400 -user <user> -
password <tdv admin password> -join -memberServer tdv-cluster -
memberPort 9400 -memberUser <user> -memberPassword <tdv admin password>"
```

Note: When launching a container, you can either give a password or use a password file that has the password stored in it. Refer to the section [TDV Admin Password](#) for ways to specify the Admin Password.

References

Refer the table below for a description of the different Docker commands:

Option	Docker Help Reference
--restart	Used to configure the restart policy for a container. Refer https://docs.docker.com/config/containers/start-containers-automatically/
-t	Allocate a pseudo-tty - https://docs.docker.com/engine/reference/run/
-i	Keep STDIN open even if not attached - https://docs.docker.com/engine/reference/run/
-d	Detach and run the container in background and print container ID - https://docs.docker.com/engine/reference/run/#detached--d

Option	Docker Help Reference
-v	<p>(TDV Required) The tdv container requires a persistent storage area when running as a Docker container. See Sizing Guidelines for TDV for size recommendations.</p> <p>https://docs.docker.com/storage/bind-mounts/</p> <p>Usage: -v <path to the file on the host machine>:<path where the file is mounted in the container>:<optional parameters></p> <p>Example: -v \$CONTAINER_CACHE_VOLUME:\$TDV_DATABASE_DIR</p> <p>Note: mount point must have a valid volume existing before starting the TDV Container.</p>
	<p>(TDV Recommended) The tdv container works best with 2 CPUs/cores in general. See Sizing Guidelines for TDV for value recommendations.</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>
--m	<p>(TDV Required) The tdv container requires a minimum of 8 GB of memory. Higher tdv workloads require more. See Sizing Guidelines for TDV for value recommendations.</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>
<tdv-container-name>	<p>Container name for your TDV Docker container. Recommendation is to have tdv in the name. Examples: tdv, tdv-1, tdv-2, tdv-dev, tdv-prod, etc</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>
<repo-name>	<p>Repository name for your TDV Docker image.</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>
<image-name>	<p>Recommendation is to use tdv. Of course, you can change this to any name though</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>
<image-tag>	<p>Recommendation is to use the TDV version for this. Example: 8.4</p> <p>https://docs.docker.com/config/containers/resource_constraints/</p>

Linux

This section explains how to start two TDV Docker containers configured as a DV Cluster configuration on a Docker environment hosted on the Linux platform. Use a docker network that will allow your TDV containers to communicate with each other.

The bridge, host and user specified bridge network options in the docker should work for the TDV containers on this platform. Refer to the TDV Active Cluster Guide on how to configure TDV and create a new active cluster.

Note: Ensure that both TDV containers are running and accessible.

TDV Docker Container Example

Resource configuration: small (poc/demo) : 2 CPUs/cores, 8 GB memory, external container volume tdv-vol with 8GB persistent readable/writable storage.

TDV configuration: base port (9400), admin password (required), server memory (default). Refer to the docker container files (Dockerfile.tdv, Dockerfile.tdv.repo and Dockerfile.tdv.cache) for TDV Docker image default values.

Network configuration: user bridge docker network

Refer to [General example for launching two TDV Docker containers to create a TDV Cluster - For Node 1](#) for examples.

Verify you can access port 9400 for Node #1 from outside of your Docker environment.

Once that is done, follow the TDV configuration steps in "Creating a New Active Cluster" section in the TDV Active Cluster Guide.

That will setup a new DV cluster on Node #1.

Refer to [General example for launching two TDV Docker containers to create a TDV Cluster - For Node 2](#) to setup Node 2.

Verify you can access port 9400 for Node #2 from outside of your Docker environment.

Once that is done, following the DV configuration steps in "Adding a TDV Server to an Active Cluster" section in the TDV Active Cluster Guide.

That will setup Node #2 to join the TDV Cluster created on Node #1.

Now your DV Cluster is configured and ready for usage.

You can verify this by opening a browser client and going to `http://<IP_NODE_#1>:9400/manager`.

Select “Cluster”.

MacOS

This section explains how to start two TDV Docker containers configured as a DV Cluster configuration on a Docker environment hosted on the Mac OS platform. Use a docker network that will allow your TDV containers to communicate with each other.

The bridge, host and user specified bridge network options in the docker should work for the TDV containers on this platform. Refer to the TDV Active Cluster Guide on how to configure TDV and create a new active cluster.

Note: Ensure that both TDV containers are running and accessible.

TDV Docker container example

Resource configuration: small (poc/demo): 2 CPUs/cores, 8 GB memory, external container volume tdv-vol with 8GB persistent readable/writable storage.

MacOS specific configuration: -p <host-port>:<container-port> for all DV ports exposed and --hostname=localhost

TDV configuration: base port (9400), admin password (required), server memory (default). Refer to the docker container files (Dockerfile.tdv, Dockerfile.tdv.repo and Dockerfile.tdv.cache) for TDV Docker image default values.

Network configuration: user bridge docker network.

Refer to [General example for launching two TDV Docker containers to create a TDV Cluster](#) - [For Node 1](#):for examples.

Verify you can access port 9400 for Node #1 from outside of your Docker environment.

Once that is done, follow the TDV configuration steps in "Creating a New Active Cluster" section in the TDV Active Cluster Guide.

That will setup a new DV cluster on Node #1.

Refer to [General example for launching two TDV Docker containers to create a TDV Cluster](#) - [For Node 2](#) to setup Node 2.

Verify you can access port 9400 for Node #2 from outside of your Docker environment.

Once that is done, following the DV configuration steps in "Adding a TDV Server to an Active Cluster" section in the TDV Active Cluster Guide.

That will setup Node #2 to join the TDV Cluster created on Node #1.

Now your DV Cluster is configured and ready for usage.

You can verify this by opening a browser client and going to `http://<IP_NODE_#1>:9400/manager`.

Select "Cluster".

Windows

This section explains how to start two TDV Docker containers configured as a DV Cluster configuration on a Docker environment hosted on the Windows platform. Use a docker network that will allow your TDV containers to communicate with each other.

The bridge, host and user specified bridge network options in the docker should work for the TDV containers on this platform. Refer to the TDV Active Cluster Guide on how to configure TDV and create a new active cluster.

Note: Ensure that both TDV containers are running and accessible.

TDV Docker container example

Resource configuration: small (poc/demo): 2 CPUs/cores, 8 GB memory, external container volume `tdv-vol` with 8GB persistent readable/writable storage.

Windows specific configuration: `-p <host-port>:<container-port>` for all DV ports exposed and `--hostname=localhost` or `--hostname=<ip-or-hostname>`

TDV configuration: base port (9400), admin password (default), server memory (default). Refer to the docker container files (`Dockerfile.tdv`, `Dockerfile.tdv.repo` and `Dockerfile.tdv.cache`) for TDV Docker image default values.

Network configuration: user bridge docker network.

Refer to [General example for launching two TDV Docker containers to create a TDV Cluster - For Node 1:](#)for examples.

Verify you can access port 9400 for Node #1 from outside of your Docker environment.

Once that is done, follow the TDV configuration steps in "Creating a New Active Cluster" section in the TDV Active Cluster Guide.

That will setup a new DV cluster on Node #1.

Refer to [General example for launching two TDV Docker containers to create a TDV Cluster - For Node 2](#) to setup Node 2.

Verify you can access port 9400 for Node #2 from outside of your Docker environment.

Once that is done, following the DV configuration steps in "Adding a TDV Server to an Active Cluster" section in the TDV Active Cluster Guide.

That will setup Node #2 to join the TDV Cluster created on Node #1.

Now your DV Cluster is configured and ready for usage.

You can verify this by opening a browser client and going to `http://<IP_NODE_#1>:9400/manager`.

Select "Cluster".

Runtime TDV Container Configuration - Common Examples

The following topics are explained in this section:

[Change TDV Admin Password \(while container is running\)](#)

[Change TDV Base Port](#)

[Change TDV Server Memory Setting](#)

[Configure External Volume For Local Persisted File Data Sources](#)

[Configure Data Source With 3rd party JDBC Driver \(type 4\)](#)

[Configure Data Source With 3rd party JDBC Driver \(type 3\)](#)

[Configure JVM Settings from Docker](#)

[Configure LDAP Properties From Docker](#)

[Configure Truststore and Keystore Files From Docker](#)

Text Editor

TDV Docker distribution is packaged with the text editor Nano. For a list of editor commands refer to <https://www.nano-editor.org/dist/latest/cheatsheet.html>

Change TDV Admin Password (while container is running)

To change the TDV Docker container's admin password you will need to use the TDV Studio client.

Note: Your default TDV Docker admin user "admin" password is defined in your Dockerfile.tdv for your given TDV release.

Follow these steps to change the password:

1. Use TDV Studio (same version of TDV as your TDV Docker container) to connect to your TDV Docker container. Login to the TDV Studio using the TDV admin user "admin" and your TDV admin password along with the appropriate TDV base port.
2. Select the "File" tab in the TDV Studio client menu options and choose "Change Password".
3. In the new pop-up dialog window, provide the current TDV admin user "admin" user password and the new password. When completed, click OK.
4. TDV Studio client will not logout of the client and a new login dialog will be displayed. Proceed to login again with your new TDV admin user "admin" password to verify the update.

Change TDV Base Port

To change the TDV Docker container's base port you will need to reconfigure TDV and create a new TDV Container.

Example (change base port from 9400 to 10400):

a. Before starting the TDV Container:

To map HOST_PORT to internal TDV container port TDV_BASE_PORT (default 9400)

```
-p HOST_PORT
```

To change internal TDV container port to 10400 instead of default 9400. Host port is also 10400 if "-p" is not set.

```
--env TDV_BASE_PORT=10400
```

a. If the TDV Container is already running:

- 1) Use TDV Studio (same version of TDV as your TDV Docker container) to connect to your TDV Docker container. Login to the TDV Studio using the TDV admin user "admin" and your TDV admin password along with the appropriate TDV base port.
- 2) Goto Administration > Configuration
- 3) Search for "Port ("
- 4) From the search result, select "Port (On Server Restart): 9400".
- 5) Change your base port value to "10400". Apply your Changes.

Note: "Port (Current)" shows your current TDV Server base port configuration.

"Port (On Server Restart)" shows your future TDV Server base port configuration. This only takes affect when you restart the TDV Server though.

- 6) Now click on the "Ok" button to close the pop-up window.
- 7) Logout of the TDV Studio client.
- 8) Stop the TDV Docker container where your TDV Server is running.

```
docker stop <tdv-container-name>)
```

- 9) Start a new TDV Docker container with the new base ports and reuse the TDV Docker volume used for the container from step #8.

```
docker run -itd -p 10300:10300 -p 10301:10301 -p 10302:10302 -p
10303:10303 -p 10304:10304 -p 10305:10305 -p 10306:10306 -p 10400:10400
-p 10401:10401 -p 10402:10402 -p 10403:10403 --env TDV_ADMIN_
PASSWORD=admin1 -v tdv_new-vol:/opt/TIBCO --cpus=2.000 -m=8g --name tdv_
new myrepo/tdv:8.5 tdv.server
```

Note: Alternately you can also run the quick start script `run_tdv_container.sh` with the appropriate parameters. Refer to [Launching TDV Containers \(Single Node\) - Using Quick Start Script](#) for instructions on how to use the script.

Note: The above example expects a valid `tdv-vol`, default docker network bridge works on your Docker host, and that you already have a valid TDV Docker image that exists.

It is also expected that you reuse your TDV Container volume, otherwise your base port and any other TDV metadata changes will be lost. Basically, if you specify a new TDV volume, then the TDV Container will create a brand new, default TDV Container based on your TDV Docker image defaults (i.e. default base port and server settings).

Change TDV Server Memory Setting

To change the TDV Docker container's memory setting you will need to reconfigure TDV and create a new TDV Container.

Example (change server memory from 8192 Mbytes (8 GB) to 16,384 MBytes (16 GB):

- 1) Use TDV Studio (same version of TDV as your TDV Docker container) to connect to your TDV Docker container. Login to the TDV Studio using the TDV admin user “admin” and your TDV admin password along with the appropriate TDV base port.
- 2) Go to Administration > Configuration
- 3) Search for “Total Available Memory (“
- 4) In the search results, select “Total Available Memory (On Server Restart): 4096 Mbytes”.
- 5) Change your base port value to “8192”. Apply your changes.

Note: “Total Available Memory (Current)” shows your current TDV Server memory configuration.

“Total Available Memory (On Server Restart)” shows your future TDV Server memory configuration. This only takes affect when you restart the TDV Server though.

- 6) Click on the OK button to close the pop-up window.
- 7) Logout of the TDV Studio client.
- 8) Stop the TDV Docker container where your TDV Server is running

```
docker stop <tdv-container-name>
```

- 9) Now start a new TDV Docker container with the new server memory value and reuse the TDV Docker volume used for the container from step #8.

```
docker run -itd -p 10300:10300 -p 10301:10301 -p 10302:10302 -p
10303:10303 -p 10304:10304 -p 10305:10305 -p 10306:10306 -p 10400:10400
-p 10401:10401 -p 10402:10402 -p 10403:10403 --env TDV_ADMIN_
```

```
PASSWORD=admin1 -v tdv_new-vol:/opt/TIBCO --cpus=2.000 -m=16g --name
tdv_new myrepo/tdv:8.5 tdv.server
```

Note: The above example expects a valid tdv-vol, default docker network bridge works on your Docker host, and that you already have a valid TDV Docker image that exists.

It is also expected that you reuse your TDV Container volume, otherwise your server memory and any other TDV metadata changes will be lost. Basically, if you specify a new TDV volume, then the TDV Container will create a brand new, default TDV Container based on your TDV Docker image defaults (i.e. default "admin" user password, base port and server settings).

Configure External Volume For Local Persisted File Data Sources

To allow your TDV Docker container to introspect and query data from a locally persisted flat file data sources (For example .csv, .xml, .txt files), you will need to transfer those files into your TDV Docker container's volume.

Example (introspect a flat file csv file stored on the TDV Container's volume):

```
sudo cp <flat-file-csv> /var/lib/docker/volumes/<tdv-container-volume-
name>/_data/TDV_Server_<tdv-version>/tmp
```

Note: This example expects a valid tdv-vol, default docker network bridge that works on your Docker host, and that you already have a valid TDV Docker Server container that exists and is running.

See [References](#): for more details regarding <tdv-container-volume-name>.

1. `docker exec -it <tdv-server-container-name> ls -al TDV*/tmp/* .csv` # validate TDV container can see the new file.
2. Use TDV Studio Client to introspect and query new csv file.
 - Go to File -> New -> Data Source -> File-Delimited
 - Provide "name", select "Local File System" with "Root Path" /opt/TIBCO/TDV_Server_<tdv-version>
 - Leave all other settings with the default values.
 - Click "Create & Introspect" button.

- Open "name" data source. Click on "Show Contents" to query data in csv file data source.

Configure Data Source With 3rd party JDBC Driver (type 4)

TDV Data sources may require 3rd party JDBC drivers (type 4).

This section is to cover how to install such drivers in your TDV Docker container.

Example (install Oracle 3rd party JDBC type 4 driver for Oracle 11g):

1. Find the latest Oracle 12g JDBC driver (type 4) drivers (e.g. ojdbc10.jar and xdb.jar).

See the Oracle Adapter Guide for details on where to get this driver and how to configure it for your TDV Container.

2. Stop the TDV Docker container where your TDV Server is running

```
docker stop <tdv-container-name>
```

3. Install Oracle JDBC type 4 drivers in your TDV Server Container

```
sudo cp ojdbc10.jar /var/lib/docker/volumes/<tdv-container-volume-name>/_data/TDV_Server_<tdv-version>/conf/adapters/system/oracle_19c_thin_driver
```

```
sudo cp xdb.jar /var/lib/docker/volumes/<tdv-container-volume-name>/_data/TDV_Server_<tdv-version>/conf/adapters/system/oracle_19c_thin_driver
```

Note: The above example expects a valid tdv-vol, default docker network bridge works on your Docker host, and that you already have a valid TDV Docker Server container that exists and is not running.

See [References](#): for more details regarding <tdv-container-volume-name>.

4. Start the TDV Docker container where your TDV Server is running

```
docker start <tdv-container-name>
```

5. Validate that the TDV Docker container has the new file.


```
docker exec -it <tdv-container-name> ls -al
TDV*/conf/adapters/system/oracle_19c_thin_driver/
```

6. Check your TDV Docker container server log for acknowledgement that you have installed the JDBC driver for your "Oracle 19c (Thin Driver)" DV adapter.

```
docker exec -it <tdv-container-name> /bin/bash
```

```
$ cd TDV*/logs
```

```
$ grep -i "Oracle 19c" cs_server.log
```

7. The output of step 6 will show before and after loading of your "Oracle 19c" DV adapter. If the installation was successful, then the DV adapter will have a "loaded" message instead of the following "has not been installed" message (shown below) that was displayed before 3rd party drivers were installed for "Oracle 19c" DV Adapter

```
INFO [main] 2020-03-30 22:07:51.134 +0000 DbUtil - The adapter for
'Oracle 19c (Thin Driver)' has not been installed. For details on
adapter installation, see the Installation Guide.
```

A sample message of successful installation:

```
INFO [main] 2020-03-30 17:11:08.222 -0700 JdbcDriverClassLoaderUtil
- Adapter: Oracle 19c (Thin Driver) loaded from /opt/TIBCO/TDV_
Server_<tdv-version>/conf/adapters/system/oracle_19c_thin_driver
```

8. Once the "loaded" message is seen you can create, introspect and load data from your Oracle 19c DV Adapter.

Configure Data Source With 3rd party JDBC Driver (type 3)

TDV Data sources may require 3rd party JDBC drivers (type 3). This section describes how to install such drivers in your TDV Docker container.

Example (install SAP JCo 3rd party JDBC type 3 driver for linux x64 platforms:

1. Find latest SAP JCo JDBC type 3 driver download from SAP.

See the TDV User Guide Chapter “Configuring Advanced Adapters” section “Installing the SAP Java Connector Library” for more details on where to download SAP JCo connection library and how to install it.

2. Install SAP JCo JDBC type 3 driver (linux x64) from SAP in your TDV Container. Refer to the TDV User Guide “Installing SAP JCo on UNIX” section for more details on how to install the linux x64 version of this driver.

```
sudo cp <sap-cjo-tgz-file> /var/lib/docker/volumes/<tdv-
container-volume-name>/_data/TDV_Server_<tdv-version>/tmp
```

Note: The above example expects a valid tdv-vol, default docker network bridge works on your Docker host, and that you already have a valid TDV Docker Server container that exists and is running.

See [References](#): for more details regarding <tdv-container-volume-name>.

3. Stop your TDV container.
4. Run the following commands:

```
$ cd TDV_Server_<Version>
```

```
$ cp sapjco<version>.jar apps/dlm/app_ds_[sap|sapbw|sapbwex]/lib
```

```
$ cp libsapjco<version>.so apps/server/lib/svn/lib64
```

5. Restart your TDV container
6. Run the following commands:

```
$ docker exec -it <TDV container name> /bin/bash
```

```
$ cd TDV_Server_<version>
```

```
$ export LD_LIBRARY_PATH=apps/server/lib/svn/lib64
```

```
$ ./jdk/bin/java -jar apps/dlm/app_ds_
[sap|sapbw|sapbwbox]/lib/sapjco<version>.jar
```

This command should have no errors.

7. Now you are ready to introspect your SAP datasource.

Configure JVM Settings from Docker

This section describes how to review existing JVM settings and add custom settings for TDV Server in the docker environment.

To Review Existing Settings

1. List the contents of the files “script_env.sh”. This file exists if you added custom JVM options to your TDV Server.

```
cat conf/script_env.sh
```

2. Search the default Linux JVM arguments set for the TDV Server in the “server.properties” file.

```
cat conf/server/server.properties | grep "linux.vmargs"
```

To Add Custom JVM Settings for TDV Server

Follow these steps to add custom JVM Settings::

Navigate to<TDV_INSTALLDir>/conf

```
$ cd ./<TDV_INSTALL_DIR>/conf
```

Copy script_env.sh.sample to script_env.sh

```
$ cp script_env.sh.sample script_env.sh
```

Open script_env.sh with a text editor (nano) and uncomment the last two lines:

```
$ nano script_env.sh
```

```
# CIS_SERVER_VM_ARGS=
```

```
# export CIS_SERVER_VM_ARGS
```

Modify the script_env.sh file to have a content like this at the end of the file:

```
CIS_SERVER_VM_ARGS="-Djavax.net.debug=all -
Ddrill.java.home=/data/opt/TIBCO/<TDV_INSTALL_DIR>/jdk -server -
XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=/data/opt/TIBCO/<TDV_
INSTALL_DIR>/logs -XX:+UseG1GC -XX:MaxGCPauseMillis=200 -Xmx2048m -
Djava.library.path=/data/opt/TIBCO/<TDV_INSTALL_
DIR>/apps/server/lib/svn/lin64"
```

```
export CIS_SERVER_VM_ARGS
```

Note: You can get the content for `CIS_SERVER_VM_ARGS` from the file `<TDV_INSTALL_DIR>/conf/server/server.properties`.

Configure LDAP Properties From Docker

Query searches for retrieving user and group information are controlled by the ldap properties file. If you add LDAP domains to TDV Server, you should configure the ldap.properties file after installation and prior to adding and configuring the LDAP domain on the Studio Manager Domain Management page. In the docker environment the properties file can be accessed from the following location:

```
$ docker exec -it <CONTAINER_NAME> /bin/bash
```

```
$ cd $TDV_INSTALL_DIR/conf/server
```

Use the Nano text editor for editing the ldap.properties file:

```
$ nano ldap.properties
```

For more information on the structure of the properties file, refer to the *Administration Guide*, Chapter *LDAP Domain Administration*.

Configure Truststore and Keystore Files From Docker

Keystore and truststore files are the places where keys for secure communications are stored. Each system component participating in SSL communication requires:

- A keystore file (cis_server_keystore.jks) for its own key, which it furnishes to any other component that requests that it authenticate itself.
- A truststore file (cis_server_truststore.jks) for the keys of each other component that it trusts and needs to authenticate

The files can be accessed from the following location. Use the Nano text editor for editing the truststore and keystore files:

```
$ docker exec -it <CONTAINER_NAME> /bin/bash
```

```
$ cd $TDV_INSTALL_DIR/conf/server/security
```

```
$ nano <filename>
```

For more information on the SSL Protocol and the Keystore and Truststore files, refer to the *Administration Guide*, Chapter *TDV and SSL Authentication*.

Limitations

While running the TDV Server using the Docker, the monitor daemon is not active. Therefore you cannot use TDV Studio or the command line tool to restart TDV Server. You will have to stop the container and start it again to do so.

Upgrading to a New Version of TDV Server

In order to upgrade your TDV Server to a newer version, follow these steps:

1. Perform a Backup Export of Metadata from the previous version of TDV Server. Refer to the Administration Guide for the instructions on how to perform this using the command line utility.
2. Create a New container using the newer version provided. Refer to the sections in this chapter to build and launch TDV containers.
3. Do a Backup Import of the Metadata into your newly built TDV container instance. Refer to the Administration Guide for the instructions on how to perform this using the command line utility.

Tips from Expert

Follow these best practices tips to maximize performance of a TDV container:

- **Naming For a Container:** A docker container name should be specified when launching the container. If it is not specified a system generated name will be used and this cannot be changed later. It is easier to manage your TDV Docker container if you give it a unique name.
- **Ports:** A Docker container is a runtime instance of a Docker image. Use a new docker image when you have to use custom ports for the different TDV functionalities. When using custom ports, make sure you map it appropriately while launching the ports. If the mapping is not done appropriately, the default ports defined in the Docker image will be used.
- **Storage:** It is important to allocate a persistent storage volume for a TDV Docker container. Make sure you create the appropriate Docker volume and use the -v option (mount) when launching a TDV Docker container.

Useful Docker Commands for TDV Containers

To check details of all the Docker containers.

```
$ docker ps
```

To check the Docker TDV container system logs.

```
$ docker logs <tdv-container-name>
```

Installing the SAP Java Connector Library

```
$ docker exec -it <tdv-container-name> /bin/bash
```

To review the TDV Server logs in an interactive shell.

```
$ docker exec -it <tdv-container-name> ls -al TDV*/logs
```

To stop the container.

```
$ docker stop <tdv-container-name>
```

To start the container.

```
$ docker start <tdv-container-name>
```

To see a detailed information on Docker objects. such as docker images, containers,

```
docker inspect [OPTIONS] NAME|ID [NAME|ID...]
```

Refer <https://docs.docker.com/engine/reference/commandline/inspect/> for more information.

networks,
volumes,
plugins, etc.

A quick
reference for
using Nano
text editor.

<https://www.nano-editor.org/dist/latest/cheatsheet.html>

TDV Container Orchestration Using Kubernetes

Introduction to Container Orchestration

Container orchestration is a procedure used to manage the lifecycles of containers, especially in large, dynamic environments. Many organizations use container orchestration to control and automate tasks such as:

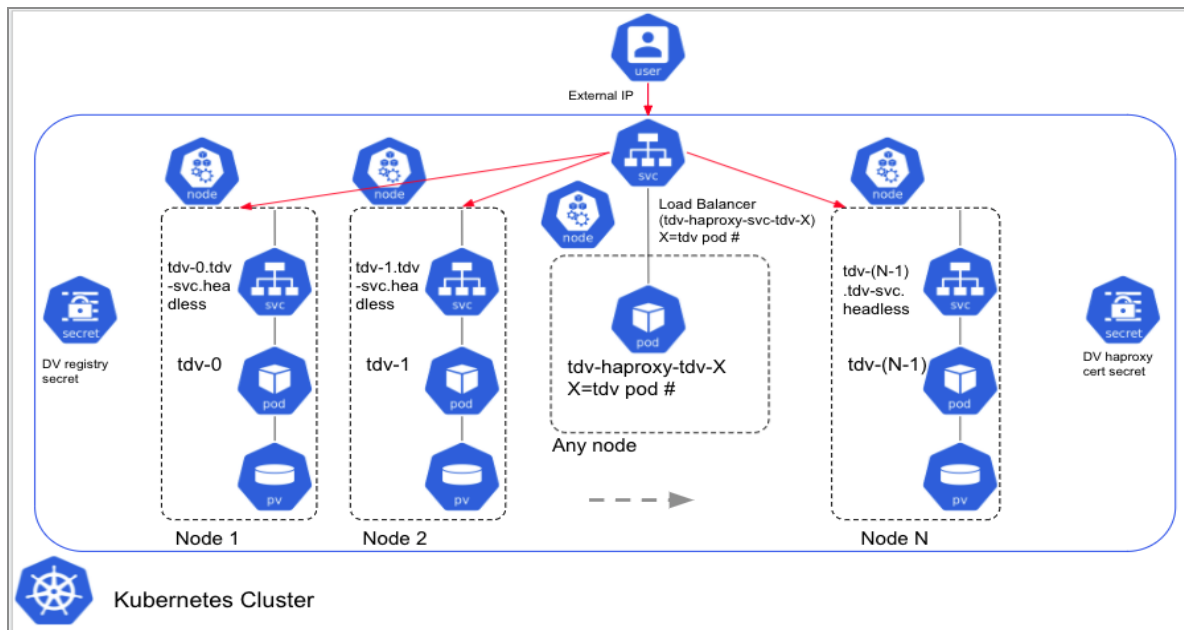
- Provisioning and deployment of containers
- Allocation of resources between containers
- Load balancing of service discovery between containers
- Configuration of an application in relation to the containers running it
- Scaling up or removing containers to spread application load evenly across host infrastructure

TIBCO Data Virtualization Container distribution is based on an orchestration framework and is scalable, lightweight, and supports widely accepted industry standards. You can deploy, monitor, and manage the application using Kubernetes-based orchestration framework provided by TDV.

TDV Container Orchestration Architecture

There are two types of TDV Server deployment in a Kubernetes environment:

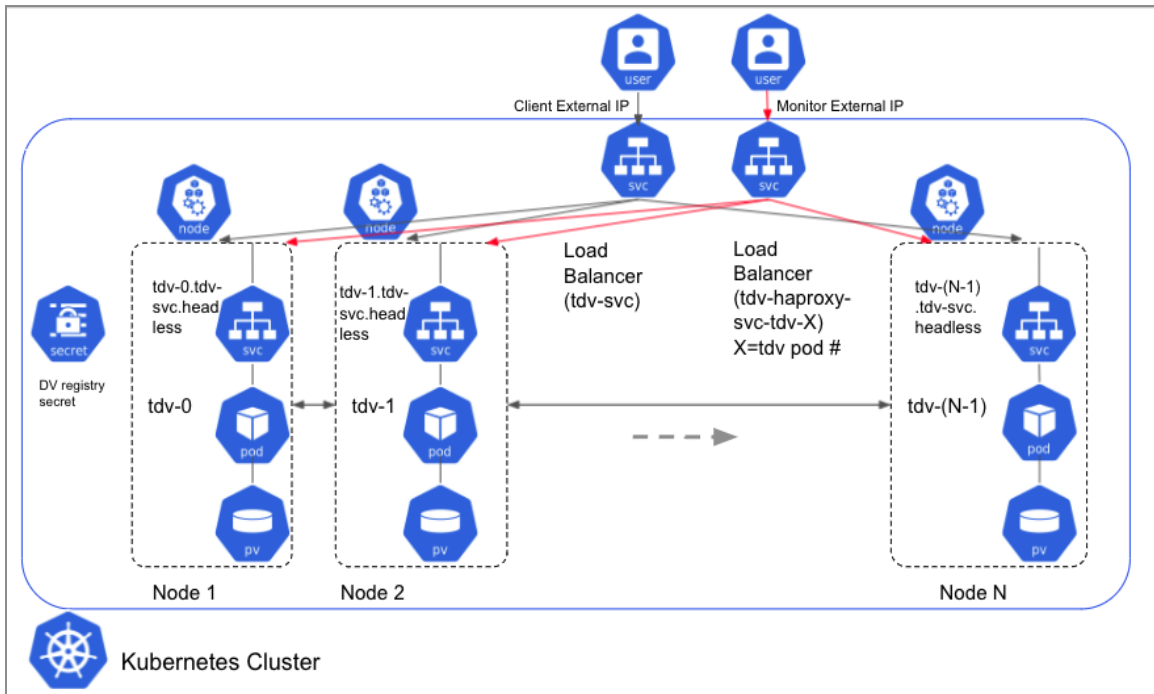
Single TDV Deployment Mode:



In the single TDV deployment mode, each TDV pod consists of a single TDV container that has the TDV repo, cache and the Server.

There is one external IP exposed to provide access to your single TDV deployment mode pod. That IP can be used to access your TDV pod via port 9400 with any TDV client.

ClusterTDV Deployment Mode:



In the cluster TDV deployment mode, each TDV pod consists of a single TDV container that has the TDV repo, cache and the Server.

There are two external IPs exposed to provide TDV client access and direct TDV pod access for monitoring purposes.

Client External IP

This IP is for TDV clients that need to access the TDV services and do not care which TDV pod that handles the request.

It is connected to a Kubernetes Load Balancer service which means requests will get sent to a random TDV pod.

Monitor External IP

This IP is for TDV clients that need to monitor one specific TDV pod for administrative purposes (e.g. TDV Web Manager, TDV Drill console, etc).

It is connected to a Kubernetes Load Balancer service and only routes requests to a specific TDV pod. One uses the TDV HAProxy service to configure this IP to map to a specific TDV pod.

Prerequisites

The TIBCO Data Virtualization Container distribution comes with a Helm chart to simplify the deployment of TDV. This section describes all the prerequisites you need before deploying a TDV application.

1. Ensure that your Kubernetes cluster version (i.e. server version) is 1.21 (or later), helm client is version 3.4 (or later), and any OCI compliant container runtime supported by Kubernetes:
<https://kubernetes.io/docs/setup/production-environment/container-runtimes/>
TDV uses Docker for development/testing of the TDV container images.
2. Refer to [Sizing Guidelines for TDV](#) for sizing recommendations and accordingly choose your environment.
3. You will need a container registry url, username and password that can be accessible by your Kubernetes cluster to retrieve the TDV container image. It is required to pre-build the TDV container image for this distribution before you deploy any TDV applications.

Pre-Configuration of the Runtime Environment (On-Premises/Private Cloud only):

Follow these pre-configuration steps to setup your runtime environment.

1. Download the following files from edelivery and unzip the contents to your instance where you intend to build the image:
 - TIB_tdv_<version>_kubernetes.zip (contains all the files needed for the TIBCO Data Virtualization Container distribution - one archive file for the TDV image creation (TIB_tdv_<version>_docker_k8s.zip) and one for the TDV Helm charts (TIB_tdv_<version>_helmcharts.tar.gz)).
 - TIB_tdv_<version>.md5
2. Run the md5 checksum in order to validate the integrity of the downloaded files:

```
For example:  
$ md5sum TIB_tdv_8.6_kubernetes.zip
```

```
072c7444a4bbf858cd627f2ccc00aa91 TIB_tdv_8.6_kubernetes.zip
```

```
$ grep 072c7444a4bbf858cd627f2ccc00aa91 TIB_tdv_8.6.md5
```

```
072c7444a4bbf858cd627f2ccc00aa91 TIB_tdv_8.6_kubernetes.zip
```

Building the Container Image

Follow these steps to build your container image:

1. Build the container image using the TDV Docker file and tar.gz file from the TIB_tdv_<version>_docker_k8s.zip archive. That archive file is contained in TIB_tdv_<version>_kubernetes.zip. **The edelivery has separate archive files for each distribution. When building the container image for the Kubernetes distribution, it is important that you choose this archive.**

Note: The default password in the Dockerfile is *tdvk8s*. If you build the image, without changing the password for the docker_k8s Dockerfile at build time, then your image and the future containers will use this default TDV admin password. The best practice is to change the default TDV admin password. It will need to be done at build time as given below:

```
unzip TIB_TDV_<version>_docker_k8s.zip
```

```
chmod 755 *.sh
```

```
./build_tdv_image.sh --build-arg tdv_admin_password=<NEW PASSWORD>
```

For the Kubernetes distribution, the default image and tag name is "tibco/tdvk8s:<version>" where <version> is the TDV version. If you want to change the image tag, use the following command-line argument:

```
-n: Set name:tag for image (default is "tibco/tdvk8s:8.6").
```

When the image is successfully built, you will see a message similar to:

```
Successfully built <image>
```

```
Successfully tagged tibco/tdvk8s:8.6
```

2. To verify that the image is built successfully, use the following command:

```
docker images -a |grep <image tag>
```

If the image was built successfully, it will be listed.

3. As a best practice, before transitioning to AKS or other container deployment tools, verify that you can run the image in Docker. To do this, run the script **run_tdv_container.sh**:

```
$ chmod 777 run_tdv_container.sh
```

```
$ ./run_tdv_container.sh
```

The script runs Docker commands to verify that the image runs in Docker. If the image runs successfully in Docker, it will also run in AKS.

To complete the Kubernetes configuration in your instance using the quick start script, refer [Configuring Kubernetes Using Quick Start Script](#).

You can also manually complete the configuration if you are familiar with the Helm and Kubernetes command line client tools (i.e. helm and kubectl).

For a list of helm commands, refer to:

<https://helm.sh/docs/helm/>

For a list of kubectl commands, refer:

<https://kubernetes.io/docs/reference/kubectl/cheatsheet/>

Useful command examples of helm and kubectl can be found at: [Useful Kubernetes Commands](#)

Configuring Kubernetes Using Quick Start Script

The Quick Start script shipped with TDV can be used for configuring Kubernetes for Private/Public cloud.

Follow these steps to run the TDV script to launch a TDV application (i.e. Single or Cluster mode) using the quick start script:

1. TDV is packaged with quick start shell scripts and YAML files that you will need to configure Kubernetes. Extract the TIB_tdv_<version>_kubernetes.zip file which contains the quick start scripts and TDV Helm charts. For more details refer to the section [TIBCO Data Virtualization\(R\) Container Distribution](#).
2. Run the script run_tdv_helmchart.sh (which is part of the TIB_tdv_<version>_kubernetes.zip file) on your Kubernetes master node. Run the following command:

```
./run_tdv_helmchart.sh [-hv] [--dry-run] [--skip-wait]
```

```
[-d] [--del-pvc] [--del-reg-secret]
```

```
[--docker-server <host or url>] [--docker-username <user>] [--docker-password <password>] [--docker-email <email>]
```

```
[--name <name>] [--set <key1>=<val1>,<key2>=<val2>,...] [-f <yaml file>] [--values <yaml file>]
```

The table below gives a description of the different parameters used with the script.

Parameter	Description	Comments
--dry-run	Show output of execution without actually executing the script.	
--skip-wait	Skip wait check for TDV pods.	
-h	Help for using the script.	
-v	Enable verbose mode output.	
Deletion Settings		
-d	Delete and uninstall TDV helmchart if already exists	

Parameter	Description	Comments
	with same --name <name>.	
--del-pvc	Delete TDV Dynamic PVC storage.	
--del-reg-secret	Delete TDV registry secret (default name "tdv-registry-secret")	
Docker Registry Settings		
--docker-server	Server location for Docker registry.	This is required. If not provided, an error will be displayed.
--docker-username	Username for Docker registry authentication.	This is required. If not provided, an error will be displayed.
--docker-password	Password for Docker registry authentication.	This is required. If not provided, an error will be displayed.
--docker-email	Email for Docker registry.	This is required. If not provided, an error will be displayed.
Helm Chart Settings		
--name	Set <name> for helmchart (default is "tdv").	
	Pass TDV helm chart parameters from a yaml file.	
	Pass TDV helm chart parameters from a yaml file.	

Parameter	Description	Comments
--set <key1>=<val1>,<key2>=<val2>,...	Pass TDV helm chart parameters to helm chart for runtime configuration.	<p>This option is used to configure several runtime settings such as setting the tdv image, arbiter password, specifying list of external IP source ranges, etc.</p> <p>It is recommended that you run "run_tdv_helmchart.sh -h" for a complete list of the various configuration settings that can be tuned using this option.</p>

You will see the following messages upon successful execution of the script **run_tdv_container.sh**:

```
Waiting for TDV Server <version> to start inside container <container name>
```

```
..... (30s elapsed)
```

```
..... (60s elapsed)
```

```
..... (90s elapsed)
```

```
.....
```

```
TDV Server <version> took <time in seconds> to start.
```

Sample Values YAML file

TDV is pre-packaged with a yaml file that contains the parameters that is needed to install a helm chart successfully. When running the quick start script the default values defined in this file are used. To override the default values, you can use the `--set` flag when the running the script `run_tdv_container.sh`.

Deployment Environment Setting

The `deploymentEnvironment` parameter in the `values.yaml` file indicates whether the deployment is for a private (on-premises) Kubernetes installation or for a public cloud environment (example: "aks" for Azure AKS). Please review the TDV Helm chart's `values.yaml` for more details on default, required and optional settings. The default `deploymentEnvironment` is "private".

Deployment Mode Setting

The `deploymentMode` parameter in the `values.yaml` file indicates whether the TDV application will run as a single stand alone instance or in a TDV cluster configuration. The valid values are "single" and "cluster" and the default is "cluster".

Application Memory Setting

You can change the resource request memory setting for the TDV Pod by tuning the following options:

Use the `tdv.resources.requests.memory` setting to set the memory for the TDV Pod container. Use the `tdv.serverHeapMax` to set the memory for TDV processes inside the pod. By default, 8GB is the memory set for the TDV Pod container and 7000 (7GB) for the TDV processes inside the Pod.

Deploying TDV as a Single Mode instance

TDV application can be deployed as a single mode or a cluster mode. This section describes how to deploy TDV as a single mode using the quick start script.

Refer to the section [TDV Container Orchestration Architecture](#) to understand the implementation of TDV Container Orchestration. As illustrated, each pod contains a TDV container (which consists of the TDV Repo, Cache and the Server).

Example to run TDV as a Single Mode

Run the following command to run a TDV instance as a Single Mode:

```
$. /run_tdv_helmchart.sh -set tdv.deploymentMode=single,tdv.image=
"<REPO_URL>/tdvk8s:8.5"
```

Note: By default, the no. of replicas is set to "1" and the application deploymentMode is set to "cluster". To change these settings, you can use the -set flag. For example, to deploy 2 pods, run the following command:

```
$ ./run_tdv_helmchart.sh --set
tdv.replicas=2,tdv.deploymentMode=single,tdv.image="<REPO_
URL>/tdvk8s:8.5"
```

You will see a log like the one below, indicating the number of pods that will be starting, the number of worker nodes in the cluster, the name of the worker node, and a series of elapsed times. When the pod is ready you will see a message indicating the which pod has started to run.

```
Request to start "1" TDV server pod(s).
```

```
Available worker nodes: "3"
```

```
Worker node names: "tdv-k8s-1 tdv-k8s-2 tdv-k8s-3"
```

```
Waiting for TDV Server pod "tdv-0" to start.
```

```
..... (30s elapsed)
```

```
TDV pod "tdv-0" is now running after 30s
```

You will also see the service ports that the TDV server is running on:

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORTS(S)	AGE	SELECTOR
------	------	------------	-------------	----------	-----	----------

tdv-svc	NodePort	x.x.x.x	<None>	9400:31400/TCP,9401:31401/TCP,9402:31402/TCP,9403:31403/TCP,9407:31407/TCP,9393:31303/TCP	11s	app=tdv
---------	----------	---------	--------	---	-----	---------

Run the following command to check the status of a specific pod:

```
$ kubectl describe pod <pod name> -n tdv
```

After you see the output of `run_tdv_helmchart.sh` indicating that the pod and services are ready, then you need to run `"run_tdv_haproxy.sh"` in order to create a service that exposes one of your TDV pods for access with an external IP. Note that you can only setup one TDV HAProxy service at a time because the service is tied to the TDV pod and requires a unique external IP for its service.

Deploying TDV as a Cluster Mode Instance

This section describes how to deploy TDV as a cluster mode using the quick start script.

Refer to the section [TDV Container Orchestration Architecture](#) to understand the implementation of TDV Container Orchestration.

Example to run a TDV instance as a Cluster Mode

Run the following command to run TDV as a Cluster Mode:

```
$ ./run_tdv_helmchart.sh --set tdv.replicas=2,tdv.image="<REPO_
URL>/tdvk8s:8.5,tdv.cluster.externalIPs="
{100.20.30.20}",tdv.cluster.loadBalancerSourceRanges="{100.20.30.0/12}"
```

You will see a log like the one below, indicating the number of pods that will be starting, the number of worker nodes in the cluster, the name of the worker node, and a series of elapsed times. When the pod is ready you will see a message indicating the which pod has started to run.

```
Request to start "2" TDV server pod(s).
```

```
Available worker nodes: "3"
```

```
Worker node names: "tdv-k8s-1 tdv-k8s-2 tdv-k8s-3"
```

```
Waiting for TDV Server pod "tdv-0" to start.
```

```
..... (30s elapsed)
```

```
..... (60s elapsed)
```

```
..... (90s elapsed)
```

```
..... (120s elapsed)
```

```
..... (150s elapsed)
```

```
..... (180s elapsed)
```

```
..... (210s elapsed)
```

```
..... (240s elapsed)
```

```
..... (270s elapsed)
```

```
.....
```

```
TDV pod "tdv-0" is now running after 300s
```

Run the following command to check the status of a specific pod:

```
$ kubectl describe pod <pod name> -n tdv
```

After you see the output of `run_tdv_helmchart.sh` indicating that the pods and services are ready, then you can use the external IP to access your TDV service. If you want to setup a monitor external IP for one of the TDV pods in your TDV cluster then you have to run "`run_tdv_haproxy.sh`" to create a service that exposes your TDV pod for access with an external IP. Note that you can only setup one TDV HAProxy service at a time because the service is tied to the TDV pod and requires a unique external IP for its service. Refer to section [Installing the TDV HAProxy Service using the Quick Start Script](#) for instructions on using this script "`run_tdv_haproxy.sh`".

Installing the TDV HAProxy Service using the Quick Start Script

The Quick Start script shipped with the TDV Helm Charts archive can be used for installing a TDV HAProxy service.

Run the script `run_tdv_haproxy.sh` (which is part of the `TIB_tdv_<version>_kubernetes.zip` file) on your Kubernetes master node. Refer to the section [TIBCO Data Virtualization\(R\) Container Distribution](#) for a list of files that are pre-packaged with TDV.

Run the following command:

```
./run_tdv_haproxy.sh [-hv] [--dry-run] [--skip-wait]
```

```
[-d] [--del-secret]
```

```
 [--name <name>] [--set <key1>=<val1>,<key2>=<val2>,...] [-f  
<yaml file>] [--values <yaml file>]
```

The table below gives a description of the different parameters used with the script.

Parameter	Description	Comments
<code>--dry-run</code>	Show output of execution	

Parameter	Description	Comments
	without actually executing the script.	
--skip-wait	Skip wait check for TDV pods.	
-h	Help for using the script.	
-v	Enable verbose mode output.	
Deletion Settings		
-d	Delete and uninstall TDV HAProxy helmchart if already exists with same --name <name>.	
--del-secret	Delete gateway secret for TDV HAProxy service (default name "tdv-haproxy-gateway-certs-secret")	
Helm Chart Settings		
--name	Set <name> for helmchart (default is "tdv-haproxy").	
	Pass TDV helm chart parameters from a yaml file.	
	Pass TDV helm chart parameters from a yaml file.	
--set <key1>=<val1>,<key2>=<val2>,...	Pass TDV helm chart parameters to helm chart for runtime configuration.	It is recommended that you run "run_tdv_haproxy.sh -h" for a complete list of the various configuration

Parameter	Description	Comments
		settings that can be tuned using this option.

Note: You can only run one TDV HAProxy service at a time.

Removing a TDV application via Helm

To remove a TDV application completely, you must do the following:

1. Uninstall the HAProxy Service
2. Uninstall TDV application
3. Uninstall Persistent Volume Claims
4. Delete the TDV registry Secret

Uninstall HAProxy Service

Run the following command to uninstall the TDV HAProxy service:

```
helm uninstall tdv-haproxy-tdv-0 -n tdv # uninstalls TDV HAProxy (pod
tdv-0) application in Kubernetes
```

Run this command only if you deployed the TDV HAProxy application via `run_tdv_haproxy.sh`.

Uninstall Application

Run the following command to remove the TDV application(s) you have deployed:

```
helm uninstall tdv -n tdv
```

Uninstall the Persistent Volume Claim

The dynamical volumes will not be removed automatically. You will need to manually remove the dynamically provisioned volumes. The example below deletes the 3 dynamic

volumes created:

```
kubectl delete pvc tdv-data-tdv-X -n tdv. Where X=0,1,2
```

Kubernetes cleans up unused images on the Kubernetes nodes so there is no need to manually clean up the image. Refer to the following documentation on this topic:

<https://kubernetes.io/docs/concepts/architecture/garbage-collection/#containers-images>

Delete the TDV Registry Secret

You will finally need to also clean up the TDV Registry Secret. Run the following command to do this:

```
kubectl delete secret tdv-haproxy-gateway-certs-secret -n tdv # delete the TDV registry secret
```

Running TDV on a Public Cloud - Microsoft Azure

The TIBCO Data Virtualization Container distribution provides a distributed platform for containerized applications. In this section you will find detailed instructions on how to run TDV on the Microsoft Azure Public Cloud platform.

For a basic understanding of the Azure Kubernetes Service (AKS) see:

<https://docs.microsoft.com/en-us/azure/aks/concepts-clusters-workloads>

In order to interact with Azure APIs an AKS cluster requires either an Azure Active Directory service principal or a managed identity. You will need this to manage Azure resources such as the load balancer, container registry, etc. You can access the Azure portal using your account information at

<https://portal.azure.com>

Setting up the Resource Group

After you login to the Azure portal, follow these steps to set up a Resource Group. A resource group is an area for all your AKS resources such as Network Security Group,

Virtual Network, Storage account (used to deploy helm charts), Container Registry (where the TDV image is stored) and a Kubernetes Cluster.

Creating a Container Registry

1. Login to the Azure portal using your account credentials.
2. Choose your Resource Group.
3. Click on the "Create" button from your Resource Group Page.
4. Choose Container Registry.
5. In the New Container Registry dialog, choose the default settings for Basics, Networking, encryption and Tags. Finally, review the chosen options and click on Create to create your Container Registry.
6. Once the Container Registry is created, push your TDV images to this location. Refer to the chapter [TDV Docker Container](#) for more information on how to build your TDV image and push it to the registry. You may also refer to the following Azure documentation for more details about how to push an image to a container registry.

<https://docs.microsoft.com/en-us/azure/aks/tutorial-kubernetes-prepare-acr?tabs=azure-cli>

Creating a Cluster

Follow these steps to create a Kubernetes cluster in your resource group:

1. From your Resource Group, click on Create and choose the option Kubernetes Service to create a Kubernetes Cluster.
2. Choose your subscription and your resource group if it is not selected already.
3. Provide a name for your Kubernetes cluster.
4. Choose a region that is appropriate for your organization.
5. Choose a Kubernetes version. Currently TDV supports version 1.21.0 or a higher version in 1.21.X.
6. Use the defaults for the worker node settings such as Node size, Scale method and Node count range.
7. Use the default settings for the rest of the Kubernetes Create Cluster dialogue - Node pools, Authentication, Tags.
8. In the Networking tab, choose your Container Registry. Refer to the section [Creating](#)

[a Container Registry](#) for more information on creating a registry.

9. Finally click on Create to create the cluster.

Deploying the TDV Helm Chart

The TDV Helm chart provisions the necessary storage for the TDV application on the Kubernetes Cluster. TIBCO Data Virtualization Container distribution includes the helm chart with certain default settings. You can run the helm chart using a pre-built script provided.

You can use an Azure CLI or a Cloud-shell to connect to Azure and execute administrative commands on the Azure resources. These tools allow the execution of commands through interactive command-line prompts or a script. Refer to the following documentation on the Azure CLI and Azure Cloud Shell for more information about using these tools:

<https://docs.microsoft.com/en-us/cli/azure/install-azure-cli>

<https://docs.microsoft.com/en-us/azure/cloud-shell/overview>

To edit/run the helm chart and scripts using one of these tools:

1. Open the Cloud Shell interface or the Azure CLI.
 2. Connect to your cluster:
- Click on the option Connect in the top menu bar of the Cluster. A window is displayed with all the connection details and the associated kubernetes commands to use in your shell.
 - Use the commands in your shell interface to connect to the cluster. (for example, run these commands:

```
$ az account set --subscription <subscription id>
```

```
$ az aks get credentials --resource-group <resource group name> --name  
<user id>
```

- Once the connection to cluster is established, you can then issue any kubernetes command.

For example, to get information on all the pods running, type the following command:

```
$ kubectl get pods --all-namespaces
```

7. Upload your TDV Helm charts archive file (i.e. TIB_tdv_<version>_helmchart.tar.gz) to Azure using the Upload option in your Cloud shell. If you are using the Azure CLI, make sure that has access to the files extracted from the TIB_tdv_<version>_helmchart.tar.gz.
8. Once you have connected to the cluster and have all the files you need, you can use the TDV scripts to install your helm chart. Refer to the section [Configuring Kubernetes Using Quick Start Script](#).
9. Azure AKS uses a load balancer service to route all client requests across all your TDV applications that are ready to fulfill those requests. Azure AKS will automatically assign an external IP for your TDV service endpoint. To get the external IP of that load balancer service run:

```
$ kubectl get svc -o wide -n tdv
```

Note: If you also deployed the TDV HAProxy application, then you will see two TDV services with external IPs. One is for the Client External IP and one for the Monitor External IP (Refer to the illustration [ClusterTDV Deployment Mode](#): for the Cluster TDV Deployment Mode architecture).

The screen below is a sample output of the Azure Cloud Shell displaying the load balancer service and the IP to access.

```
kubectl get service -o wide -n tdv | grep tdv
tdv-svc          LoadBalancer  10.0.124.134  20.69.84.64  9400:31400/TCP,9401:31401/TCP,9402:31402/TCP,9403:31403/TCP,
9303:31303/TCP  20m  app=tdv
tdv-svc-headless ClusterIP      None          <none>       9300/TCP,9301/TCP,9302/TCP,9303/TCP,9304/TCP,9305/TCP,9306/T
CP,9400/TCP,9401/TCP,9402/TCP,9403/TCP,9407/TCP  20m  app=tdv
tdv-svc-http     ClusterIP      10.0.207.49  <none>       9400/TCP,9402/TCP,9303/TCP
20m  app=tdv
tdv-svc-tcp      ClusterIP      10.0.95.109  <none>       9401/TCP,9403/TCP
20m  app=tdv
stephen@Azure:~/85/k8s$
```

Running TDV on a Public Cloud - Amazon Web Service

The TIBCO Data Virtualization Container distribution provides a distributed platform for containerized applications.

In this section you will find detailed instructions on how to run TDV on the Amazon Web Service (AWS) Public Cloud platform.

For a basic understanding of the AWS Elastic Kubernetes Service (EKS) see:

<https://aws.amazon.com/eks/>

In order to interact with AWS APIs with an EKS cluster requires an AWS account. You will need that account and appropriate Identity and Access Management (IAM) roles/permissions to manage AWS resources such as the load balancer, container registry, etc. You can access the AWS console using your account information at:

<https://aws.amazon.com/>

Creating a Container Registry

The following link shows how to setup an AWS ECR registry:

<https://docs.aws.amazon.com/AmazonECR/latest/userguide/repository-create.html>

Once the container registry is created, push your TDV images to that location. Refer to the chapter [TDV Docker Container](#) for more information on how to build your TDV image and push it to the registry. You may also refer to the following AWS ECR documentation for more details about how to push an image to a container registry:

<https://docs.aws.amazon.com/AmazonECR/latest/userguide/docker-push-ecr-image.html>

Example:

1. Login to AWS management console with appropriate IAM role/permissions to create an AWS ECR resource.
2. Create a private registry `<ID>.dkr.ecr.<REGION>.amazonaws.com` in region `us-west-2` with repository name `"tibco/tdvk8s"`.
3. Push TDV `docker_k8s` image to private registry.

Login to the console on the Docker Server that has the `"tibco/tdvk8s:8.6.0"` image locally.

- a. Configure local AWS CLI config files (i.e. `$HOME/.aws/config` and `$HOME/.aws/credentials`) with AWS Access Key ID and AWS Secret Access Key.
- b. Run the following commands:

```
$docker login --username AWS -p $(aws ecr get-login-password --
region us-west-2) <ID>.dkr.ecr.<REGION>.amazonaws.com
```

```
$docker tag tibco/tdvk8s:8.6.0
<ID>.dkr.ecr.<REGION>.amazonaws.com/tibco/tdvk8s:8.6.0
```

```
$docker push <ID>.dkr.ecr.<REGION>.amazonaws.com/tibco/tdvk8s:8.6.0
```

4. Validate "tibco/tdvk8s:8.6.0" image now exists on

```
https://<REGION>.console.aws.amazon.com/ecr/repositories/private/
<ID>/tibco/tdvk8s?region=<REGION>
```

Creating a Cluster

Follow these steps to create a Kubernetes cluster in your AWS account.

<https://docs.aws.amazon.com/eks/latest/userguide/getting-started.html>

Example:

1. Login to AWS management console with appropriate IAM role/permissions to create an AWS EKS resource.
2. Select "region".
3. Goto:

<https://<REGION>.console.aws.amazon.com/eks/home?region=<REGION>#/clusters>

4. Click "Add Cluster" -> "Create"
5. Configure the "Configure cluster" info. Provide the required fields: EKS cluster name, Kubernetes version, and Cluster service role.
Note: TDV 8.5+ requires Kubernetes 1.21 or higher.
6. Configure the "Specify networking" info. Provide the required fields: VPC, subnets, security group.
7. Configure the "Configure logging" info (optional).

8. Review and click "create" button.
9. Wait for new cluster to have Status="Active".
10. Cluster name you specified in step 5 will now be "<NAME>_eks_cluster". Click on "<NAME>_eks_cluster".
11. Use the same Cluster service role you provided in step 5.
12. Select "Compute" tab.
13. Click on "Add Node Group".

Node group compute configuration

Select the "Instance type" and Disk size appropriate for each node to run one or more TDV pods. See [Sizing Guidelines for TDV](#) for sizing guidelines for a TDV deployment (In this case a TDV pod that contains a TDV container). For example, Suppose you have 2 nodes in the node group where each node is a 4 cpu, 16GB type with 16GB of storage. This TDV Medium configuration will allow you to run 2 TDV pods per node.

14. Wait for new node group to be in the operational status
15. Add internet gateway to "Route table" for each subnet in VPC.

Note: Without this you won't be able to access your Kubernetes cluster from outside of AWS.

```
0.0.0.0/0    igw-<NUMBER>  igw-<NUMBER>
```

16. Add two tags to each subnets in the VPC

```
tag name "kubernetes.io/role/elb", tag value "1"
```

```
tag name "kubernetes.io/cluster/<EKS cluster name>" "shared"
```

Deploying the TDV Helm Chart

To deploy the TDV Helm chart on the AWS Public cloud, follow these steps:

1. Console login to an instance that has AWS CLI, Helm, and Kubernetes clients. See [Prerequisites](#) for details on the versions of Helm and Kubernetes needed for these clients.

2. Configure local AWS CLI config files (i.e. \$HOME/.aws/config and \$HOME/.aws/credentials) with AWS Access Key ID and AWS Secret Access Key.

```
$ aws ecr get-login-password --region <REGION>
```

```
$ aws eks update-kubeconfig --region <REGION> --name "<NAME>_eks_
cluster"
```

```
$ kubectl config current-context
```

```
arn:aws:eks:<REGION>:<ID>:cluster/<NAME>_eks_cluster
```

```
$ kubectl get nodes -o wide
```

NAME	STATUS	ROLES	AGE	VERSION
INTERNAL-IP		EXTERNAL-IP		OS-IMAGE
KERNEL-VERSION				CONTAINER-RUNTIME

ip-172-31-24-119.<REGION>.compute.internal	Ready			
<none>	47m	v1.21.5-eks-9017834	172.31.24.119	
34.221.117.5		Amazon Linux 2	5.4.181-	
99.354.amzn2.x86_64		docker://20.10.7		

ip-172-31-55-142.<REGION>.compute.internal	Ready			
<none>	47m	v1.21.5-eks-9017834	172.31.55.142	
35.81.76.37		Amazon Linux 2	5.4.181-	
99.354.amzn2.x86_64		docker://20.10.7		

```
$ tar xvpfz TIB_tdv_<TDV_VERSION>_helmchart.tar.gz . # This comes
from eDelivery (TIB_tdv_<tdv_version>_kubernetes.zip).
```

Example to run 2 TDV pod cluster in EKS with default cpu, memory, storage specified by values.yaml for TDV 8.6.0.

(sample bash script):

```
#!/bin/bash
```

```
TDV_VERSION=8.6.0
```

```
REPLICAS=2
```

```
REPO="<ID>.dkr.ecr.<REGION>.amazonaws.com/tibco/tdvk8s"
```

```
TDV_IMAGE_TAG=${TDV_VERSION}
```

```
DOCKER_SERVER=<ID>.dkr.ecr.<REGION>.amazonaws.com
```

```
DOCKER_USER=aws
```

```
DOCKER_PASSWORD=`aws ecr get-login-password --region us-west-2`
```

```
DOCKER_EMAIL=xyz@abc.com
```

```
./run_tdv_helmchart.sh -v -d --del-pvc --docker-server $DOCKER_SERVER --  
docker-username $DOCKER_USER --docker-password $DOCKER_PASSWORD --  
docker-email $DOCKER_EMAIL --set \
```

```
tdv.replicas=$REPLICAS,\
```

```
tdv.deploymentEnvironment="eks",\
```

```
tdv.resources.dynamic.storageClass="csi-sc-standard",\
```

```
tdv.image="$REPO:$TDV_IMAGE_TAG"
```

TIBCO Data Virtualization(R) Container Distribution

This following table lists the files that are part of the TIBCO Data Virtualization Container distribution.

File Name	Description
TIB_tdv_<version>_helmchart.tar.gz	<p>The compressed file consists of:</p> <p>csi-sc-standard - Sample configuration of an Azure storage class.</p> <p>local-path-storage - Sample configuration for a local storage class that utilizes local storage on your Kubernetes nodes.</p> <p>tdv - Helm chart for the TDV application.</p> <p>tdv-haproxy - Helm chart for TDV HAProxy application.</p> <p>run_tdv_helmchart.sh - Quick start script for the TDV deployment in Kubernetes.</p> <p>run_tdv_haproxy.sh - Quick start script for the TDV HAProxy deployment in Kubernetes.</p>
TIB_tdv_<version>_docker_k8s.zip	<p>This compressed file contains:</p> <p>TIB_tdv_<version>_docker.tar.gz, Dockerfile.tdv - TDV docker container files.</p> <p>build_tdv_image.sh - Quick start script to build the TDV image.</p> <p>run_tdv_container.sh - Quick start script to launch TDV container.</p>

Useful Kubernetes Commands

The table below lists some commonly used Kubernetes commands

Generic Kubernetes Commands

To list all services running	<code>\$ kubectl get services</code>
To list all nodes with more details such as Internal IP.	<code>\$ kubectl get nodes -o wide</code>
To list images	<code>\$ kubectl get pods --all-namespaces -o jsonpath="{.items[*].spec.containers[*].image}" tr -s '[:space:]' '\n' sort uniq -c</code>
To list all Pods running	<code>\$ kubectl get pods</code>
View details about a particular node	<code>\$ kubectl describe node [node-name]</code>
View details about a particular pod	<code>\$ kubectl describe pod [pod-name]</code>
To print logs from containers in a pod	<code>\$ kubectl logs [pod-name]</code>
To install/uninstall helm chart	<code>helm install/uninstall <chart></code>
A list of all installed helm charts	<code>helm list</code>

Useful TDV-Specific Commands

To list the TDV Pods	<code>\$ kubectl get pods -n tdv</code>
To describe TDV Pod.	<code>\$ kubectl describe pod tdv-0 -n tdv</code>
To list TDV volumes	<code>\$ kubectl get pvc -n tdv</code>

To display the log of the init container in the specified pod.

```
$ kubectl logs tdv-0 init -n tdv
```

To invoke bash command prompt for the specified pod.

```
$ kubectl exec -it tdv-0 -c tdv -n tdv -- /bin/bash
```

To install Docker Registry Secret

```
$ kubectl create secret tdv-registry-secret -n tdv --docker-server=<DOCKER_SERVER> --docker-username=<DOCKER_USERNAME> --docker-password=<DOCKER_PASSWORD> --docker-email=<DOCKER_EMAIL>
```

Note: Name of the secret must be "tdv-registry-secret" otherwise TDV deployment in Kubernetes will not work with the TDV Helm chart.

To Manually Deploy the Local Path Storage class

```
$ kubectl apply -f local-path-storage/local-path-storage.yaml
```

```
$ kubectl get deployment -n local-path-storage --no-headers
```

To Manually Deploy the Azure csi-standard-cs Storage class

```
$ kubectl apply -f csi-sc-standard/sc-azure-disk.yaml
```

```
$ kubectl get storageclass
```

Note: This storage class is only used for `tdv.deploymentEnvironment="aks"`.

To install HAProxy Certificate Secret

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -out gateway.crt -keyout gateway.key -subj "/CN=tdv.haproxy/0=tdv-haproxy-gateway-certs-secret"
```

```
cat gateway.crt gateway.key | tee
gateway.pem
```

```
kubectl create secret generic $tdv-
haproxy-gateway-certs-secret --from-
file=gateway.pem=./gateway.pem -n tdv
```

Note: Name of the secret must be "tdv-haproxy-gateway-certs-secret" otherwise TDV deployment in Kubernetes will not work with the TDV Helm chart.

TDV Deployment Examples

Private Cloud (On-Premises) Deployment

Private cloud

```
tdv.deploymentEnvironment="private"
```

TDV Cluster
example
using custom
TDV
password.

```
$/run_tdv_helmchart.sh --set
tdv.cluster.arbiterPassword=random1Password,tdv.cluster.
externalIPs="{100.20.30.20}",tdv.cluster.loadBalancerSourceRanges="{
100.20.30.0/12}",tdv.image="<HOSTNAME>/tibco/tdvk8s:8.5"
```

Note: One
must set a
custom TDV
password
during the
build phase
of the tdkv8s
image.

tdv.cluster.ar
biterPasswor
d must match

the password set during the build phase.

TDV Cluster example with 3 replicas, custom memory, cpu, and storage settings.

```
$/run_tdv_helmchart.sh --set
tdv.replicas=3,tdv.cluster.externalIPs="
{100.20.30.20}",tdv.cluster.loadBalancerSourceRanges="
{100.20.30.0/12}",tdv.image="<HOSTNAME>/tibco/tdvk8s:8.5
",tdv.resources.dynamic.requests.cpu="2000m",tdv.resourc
es.
dynamic.requests.memory="16G",tdv.resources.dynamic.stor
age="16G"
```

TDV Single example with default replicas=1 and default memory and cpu.

```
./run_tdv_helmchart.sh --set
tdv.deploymentMode=single,tdv.image="<REPO_
URL>/tdvk8s:8.5"
```

TDV Single example with default replicas=1 and default memory and cpu - Using Helm command.

```
helm install --create-namespace -n tdv ./tdv --set
tdv.deploymentMode=single,tdv.image="<REPO_
URL>/tdvk8s:8.5"
```

TDV Cluster example with default values and loadBalancer source range settings.

```
./run_tdv_helmchart.sh --set tdv.externalIPs="
{100.20.30.20}",tdv.load BalancerSourceRanges="
{100.20.30.0/12}",tdv.image="<REPO_URL>/tdvk8s:8.5"
```

TDV Cluster
example with
default values
- Using Helm
command.

```
helm install --create-namespace -n tdv ./tdv --set
tdv.image="<REPO_URL>/tdvk8s:8.5"
```

TDV HAProxy
example (tdv-
0 default pod
monitoring)

```
$/run_tdv_haproxy.sh -d --set tdv.externalIPs="
{100.20.30.20}",tdv.loadBalancerSourceRanges="
{100.20.30.0/12}"
```

TDV HAProxy
example (tdv-
1 pod
monitoring)

```
$/run_tdv_haproxy.sh -d --set tdv.podName="tdv-
1",tdv.externalIPs="
{100.20.30.20}",tdv.loadBalancerSourceRanges="
{100.20.30.0/12}"
```

Public Cloud Deployment

**Public
cloud
(Azur
e)**

```
tdv.deploymentEnvironment="aks"
```

TDV
Cluster
example
with 3
replicas,
custom
memor
y, cpu,
and
storage
settings.

```
$ ./run_tdv_helmchart.sh --set
tdv.replicas=3,tdv.deploymentEnvironment="aks",tdv.resources.dyna
mic.storageClass="csi-sc-
standard",tdv.image="<NAME>.azurecr.io/tibco/tdvk8s:8.5",tdv.reso
urces.dynamic.requests.cpu="2000m",
tdv.resources.dynamic.requests.memory="16G"
,tdv.resources.dynamic.storage="16Gi,tdv.serverHeapMax=15000"
```

TDV
Single
example
with
default
replicas
=1 and
default
memory
and cpu.

```
./run_tdv_helmchart.sh --set  
tdv.deploymentMode=single,tdv.deploymentEnvironment="aks",  
tdv.resources.dynamic.storageClass="csi-sc-  
standard",tdv.image="<REPO_URL>/tdvk8s:8.5"
```

TDV
Single
example
with
default
replicas
=1 and
default
memory
and cpu
- Using
Helm
comma
nd.

```
helm install --create-namespace -n tdv tdv ./tdv --set  
tdv.deploymentMode=single,tdv.deploymentEnvironment="aks",tdv.res  
ources.dynamic.storageClass=  
"csi-sc-standard",tdv.image="<REPO_URL>/tdvk8s:8.5"
```

TDV
Cluster
example
with
default
replicas
=1 and
default
memory
and cpu.

```
./run_tdv_helmchart.sh --set  
tdv.deploymentEnvironment="aks",tdv.resources.dynamic.storageClas  
s="csi-sc-standard",tdv.image="<REPO_URL>/tdvk8s:8.5"
```

TDV

```
helm install --create-namespace -n tdv tdv ./tdv --set
```

Cluster example with default replicas =1 and default memory and cpu
- Using Helm command.

```
tdv.deploymentEnvironment="aks",  
tdv.resources.dynamic.storageClass="csi-sc-  
standard",tdv.image="<REPO_URL>/tdvk8s:8.5"
```

TDV HAProxy example (tdv-0 default pod monitoring)

```
$/run_tdv_haproxy.sh --set tdv.deploymentEnvironment="aks"
```

TDV HAProxy example (tdv-1 pod monitoring)

```
$/run_tdv_haproxy.sh --set tdv.podName="tdv-  
1",tdv.deploymentEnvironment="aks"
```

**Public
cloud
(Amazon
Web
Service)**

```
tdv.deploymentEnvironment="eks"
```

TDV
Cluster
example
with
3
replicas,
custom
memory,
cpu,
and
storage
settings.

```
$ ./run_tdv_helmchart.sh --set  
tdv.replicas=3,tdv.deploymentEnvironment="eks",tdv.resources.dynamic  
ic.storageClass="csi-sc-standard",tdv.image="<NAME>.azure  
cr.io/tibco/tdvk8s:8.5",tdv.resources.dynamic.requests.cpu="2000m"  
,tdv.resources.dynamic.requests.memory="16G",  
tdv.resources.dynamic.storage="16Gi,tdv.serverHeapMax=15000"
```

TDV
Single
example
with
default
replicas=1 and
default
memory
and
cpu.

```
./run_tdv_helmchart.sh --set  
tdv.deploymentMode=single,tdv.deploymentEnvironment="eks",tdv.reso  
urces.dynamic.storageClass="csi-sc-standard",tdv.image="<REPO_  
URL>/tdvk8s:8.5"
```

TDV
Single
exampl
e with
default
replica
s=1 and
default
memor
y and
cpu -
Using
Helm
comma
nd.

```
helm install --create-namespace -n tdv tdv ./tdv --set  
tdv.deploymentMode=single,tdv.deploymentEnvironment="eks",tdv.reso  
urces.dynamic.storageClass="csi-sc-standard",tdv.image="<REPO_  
URL>/tdvk8s:8.5"
```

TDV
Cluster
exampl
e with
default
replica
s=1 and
default
memor
y and
cpu.

```
./run_tdv_helmchart.sh --set  
tdv.deploymentEnvironment="eks",tdv.resources.dynamic.storageClass  
="csi-sc-standard",tdv.image="<REPO_URL>/tdvk8s:8.5"
```

TDV
Cluster
exampl
e with
default
replica
s=1 and
default
memor
y and
cpu -

```
helm install --create-namespace -n tdv tdv ./tdv --set  
tdv.deploymentEnvironment="eks",tdv.resources.dynamic.storageClass  
="csi-sc-standard",tdv.image="<REPO_URL>/tdvk8s:8.5"
```

Using
Helm
comma
nd.

TDV
HAProx
y
exampl
e (tdv-0
default
pod
monito
ring)

```
$. /run_tdv_haproxy.sh --set tdv.deploymentEnvironment="eks"
```

TDV
HAProx
y
exampl
e (tdv-1
pod
monito
ring)

```
$. /run_tdv_haproxy.sh --set tdv.podName="tdv-1",tdv.deploymentEnvironment="eks"
```

TDV for AWS Marketplace

This chapter will cover the TIBCO Data Virtualization (TDV) for AWS Marketplace Users.

The following sections are described in this chapter:

[Prerequisites](#)

[Launching TDV Server on AWS Marketplace](#)

[TDV Server Configuration](#)

[TDV Updates and Bundled TDV Software](#)

Prerequisites

You will need a valid AWS account to access TDV on the AWS Marketplace. Refer to the *TDV Release notes* for information about the *TIBCO Data Virtualization AWS Infrastructure License considerations*.

<https://docs.tibco.com/products/tibco-data-virtualization>

The pre-installed version of TDV Server on the AWS marketplace is available on the following Operating systems:

- Windows Server 2019 - Version 1809 (OS Build 17763.2268)
- Red Hat Enterprise Linux Server release 7.9 (Maipo)

Note: You will need to install one of the TDV supported web browser clients. Refer to [Web Browser Support](#) for a list of supported web browsers.

Launching TDV Server on AWS Marketplace

In order to launch TDV Server on AWS, you will need access to the AWS portal. This section describes the procedure to launch the TDV Server.

Launching a TDV Windows Image on AWS Marketplace

Follow these steps to launch a Windows image of TDV:

1. Login to the AWS portal <https://aws.amazon.com/marketplace>

Note: You will need a valid AWS account to access TDV in the marketplace.

2. Go to the “Search” area at the top of the page and search for “TIBCO® Data Virtualization” to find the latest version of the product.
3. Select the appropriate TDV offering, based on the license type and operating system.
4. Click on “Continue to Subscribe”.
5. Review EULA and price information (click “Show Details”) and then click on “Continue to Configuration”.
6. Review the fulfillment options and choose the appropriate “Region”. Then click on “Continue to Launch”.
7. “EC2 Instance Type” uses a default TDV medium resource configuration value. Choose another one if you are running a larger TDV workload. Refer to [Sizing Guidelines for TDV](#) for more details.

Note: It is recommended that the EC2 Instance Type should be at least 2 CPUs and 8GB of memory.

8. Choose the Network settings appropriate for your AWS account. (i.e. VPC settings and Subnet settings).
9. Choose the recommended Security Group settings.
 - a. Click on “Create New Based On Seller Settings” button in the “Security Group Settings” section.
 - b. Provide a name and description for this new TDV specific security group. A reference to the TDV ports exposed in this recommended Security Group can be found in [Default TDV Security Group Configuration](#).
 - c. Click on “Save”.
10. Select an existing key pair or create a new one.
11. Click on Launch.

Connecting to the TDV Windows Instance

Connecting to the virtual machine using the Remote Desktop is an essential step of launching the TDV Windows server on AWS. Follow these steps to connect to the virtual machine:

1. Login to the AWS Console.
2. Click on the “EC2 dashboard” located in the left vertical menu.
3. Click on the “Running instances” link under the “Resources” area of the EC2 Dashboard.
4. Select the check box for your TDV instance.
5. Click on the “Connect” button.
6. In the new popup window displayed,
 - a. Click on the Download the Remote Desktop File.
 - b. Click on Get Password. Save this password in a secure location.
7. Connect to AWS launched server using the downloaded RDP file and password obtained.

Note: Once connected to the virtual machine, a command window opens and the TDV installation starts. Wait for the installation to complete before closing the window. It takes approximately 15-20 minutes for the installation to complete.

Connecting to the TDV Server Application

Follow these steps to Launch the TDV server using the Windows image:

1. Login to AWS Console.
2. Click on the “EC2 Dashboard” displayed on the left vertical menu.
3. Click on the “Running instances” link under the “Resources” area of the EC2 Dashboard.
4. Select the check box for your TDV instance.
5. In the Description area, find IPv4 Public IP and copy the value.
6. Now you can access your TDV Server in the AWS environment via the TDV service ports defined in your TDV Security group (Refer [Default TDV Security Group](#)

[Configuration](#))

Examples

1. Browser client: `http://<PUBLIC_IP>:9400` to access the TDV Web Manager client.
2. TDV Studio client:

`host=<PUBLIC_IP>`

`port=9400`

3. JDBC client: `host=<PUBLIC_IP>, port=9401`

Note: The default TDV Admin password can be found at [TDV Admin Password](#)

Launching a TDV Linux Image on AWS Marketplace

Follow these steps to launch a Linux image of TDV:

1. Login to the AWS portal [Lhttps://aws.amazon.com/marketplace](https://aws.amazon.com/marketplace)

Note: You will need a valid AWS account to access TDV in the marketplace.

2. Go to the “Search” area at the top of the page and search for “TIBCO® Data Virtualization” to find the latest version of the product.
3. Select the appropriate TDV offering, based on the license type and operating system.
4. Click on “Continue to Subscribe”.
5. Review EULA and price information (click “Show Details”) and then click on “Continue to Configuration”.
6. Review the fulfillment options and choose the appropriate “Region”. Then click on “Continue to Launch”.
7. “EC2 Instance Type” uses a default TDV medium resource configuration value. Choose another one if you are running a larger TDV workload. Refer to [Sizing Guidelines for TDV](#) for more details.

Note: It is recommended that the EC2 Instance Type should be at least 2 CPUs and 8GB of memory.

8. Choose the Network settings appropriate for your AWS account. (i.e. VPC Settings and Subnet Settings).

9. Choose the recommended Security Group settings.
 - a. Click on “Create New Based On Seller Settings” button in the “Security Group Settings” section.
 - b. Provide a name and description for this new TDV specific security group. A reference to the TDV ports exposed in this recommended Security Group can be found in [Default TDV Security Group Configuration](#).
 - c. Click on “Save”.
10. Select an existing key pair or create a new one.
11. Click on Launch.

Connecting to the TDV Server Application

Follow these steps to connect to the TDV server using the Linux image:

1. Login to AWS Console.
2. Click on the “EC2 Dashboard” displayed on the left vertical menu.
3. Click on the “Running instances” link under the “Resources” area of the EC2 Dashboard.
4. Select the check box for your TDV instance.
5. In the “Description” area below the check box, find "IPv4 Public IP" and copy the value.
6. Now you can access your TDV Server in the AWS environment via the TDV service ports defined in your TDV Security group (Refer [Default TDV Security Group Configuration](#))

Examples

1. Browser client: `http://<PUBLIC_IP>:9400` to access the TDV Web Manager client.
2. TDV Studio client:

`host=<PUBLIC_IP>`

`port=9400`

3. JDBC client: `host=<PUBLIC_IP>, port=9401`

Note: The default TDV Admin password can be found at [TDV Admin Password](#)

TDV Server Configuration

This section describes the different port settings you will need while launching the TDV image.

TDV Instance Id

Follow these steps to get the TDV Instance Id:

1. Login to AWS Console.
2. Click on the “EC2 Dashboard” displayed on the left vertical menu.
3. Click on the “Running instances” link under the “Resources” area of the EC2 Dashboard.
4. Select the check box for your TDV instance.
5. Copy the “Instance ID” value.

TDV Admin Password

The TDV Server admin password is set to the AWS Instance ID for all TDV AWS Marketplace offerings.

See [TDV Instance Id](#) to get the Instance Id.

Default TDV Security Group Configuration

As a standalone installation TDV only requires ports 9400 - 9403, 9405 and 9409. If you want to run TDV in a cluster configuration then you need to additionally open up port 9407 for communication to work between TDV cluster nodes. By default, the TDV Server Security Group will have these inbound ports exposed in the AWS environment to the outside world.

For TDV Linux installations only, there will be additional inbound ports 9300-9306 open for the MPP Engine feature to work. Refer to [Port Requirements](#) for a list of all TDV ports and their descriptions.

Note: The Source (IP or Group) for your TDV Security Group's inbound ports will default to 0.0.0.0/0. You will want to change this by limiting access to only known IP addresses that you want accessing your TDV installation. Refer to the section [Review Security Group \(TDV Ports for inbound/outbound traffic\)](#) for further information.

TDV Security Group

This section covers how to review and make additional changes to your TDV Security Group.

Review Security Group (TDV Ports for inbound/outbound traffic)

1. Login to AWS Console.
2. Click on the "EC2 Dashboard" displayed on the left vertical menu.
3. Click on the "Security groups" link under the "Resources" area of the EC2 Dashboard.
4. Select the check box for your TDV Security Group.
5. At the bottom of the page there will be an overview of your security group.
6. Click on the "Inbound rules" or "Outbound rules" tab to see a specific rule set.
7. After modifying a rule set, click on "Save".

Notes:

- a. In the windows instance, modifying the inbound ports require OS level firewall changes. Refer to [Additional Firewall changes](#) for more information.
- b. All outbound traffic is allowed by default. No restrictions are in place.
- c. The TDV Security group for Linux has more inbound ports open than the Windows. This is because the MPP Engine feature is only available on Linux TDV Server installations.

Additional Port Configuration

If you need additional inbound or outbound ports (for example, data source) to your TDV instance, then you will need to modify your TDV Security Group.

Refer [Review Security Group \(TDV Ports for inbound/outbound traffic\)](#) section for steps on how to do this.

For data sources that do not reside in your AWS instance that you want to access with your TDV AWS instance, it is a good idea to verify network connectivity as follows:

1. Make sure your data source IP and port are open to connections from the AWS network.
2. Test connectivity from your TDV AWS instance using the following command:

```
openssl s_client -connect <DATA_SOURCE_IP>:<DATA_SOURCE_PORT>
```

Additional TDV Security Configuration

The section describes the additional security configuration you can do, if you require your TDV Server to only provide secure client connections.

To review TDV Port definitions, refer to [Port Requirements](#).

Disable Unsecured Ports

Follow these steps to disable HTTP port (i.e. Web services port):

1. Connect to your TDV Server using the TDV Studio. Login as the “admin” user.
2. Select Administration > Configuration.
3. In the search window, search for the option "Disable HTTP".
4. Click on “Disable HTTP (On Server Restart)”
5. Choose “True”.
6. click "OK".
7. Restart your TDV Server.
8. From outside of the Amazon environment, run a network port check to verify only secure ports are open.

For example:

- openssl s_client -connect <PUBLIC_IP>:9400 # should NOT be open
- openssl s_client -connect <PUBLIC_IP>:9402 # should be open
- TDV Studio test -> connect to your TDV Server with port=9400 and click on the “Encrypt” check box. This should allow a secure connection via port 9402 to the TDV Server. Connecting without the “Encrypt” check box enabled (i.e. port 9400) should not be allowed anymore.

9. Change your TDV Security Group to remove port 9400.

Follow these steps to disable cluster port:

1. If you are using a standalone TDV without being in a TDV Cluster then you can remove port 9407 from your security group.
2. Change your TDV Security Group to remove port 9407.

Note: Changing the security group ports requires additional OS configuration of the firewall settings if you are on the Windows platform. Refer to [Additional Firewall changes](#) for more information.

Additional Inbound Port Security

To further secure your TDV Security Group, change your Source IPs to match only known IPs.

This will allow you to ensure that only specific IPs access your TDV instances.

Refer [Review Security Group \(TDV Ports for inbound/outbound traffic\)](#) for how to access your security group to make that type of modification.

Additional Firewall changes

This section is to be used when you need to make port changes after already starting your DV instance. After you make the appropriate port changes in your security group, follow the steps given below to activate those changes.

Note: Review your respective security group in your Cloud console before performing the steps below. Your security group inbound ports must always match (i.e. a one to one mapping) with the underlying OS firewall inbound port rules.

Windows Server 2019

Open a Remote Desktop Connection to your instance and follow the steps below:

1. Launch “Control Panel”
2. Select “System and Security”
3. Select “Windows Defender Firewall”
4. Click “Advanced settings” link on left hand side vertical menu area
5. Select “Inbound Rules” on left hand side vertical menu area
6. In the “Name” column, select “TDV Ports” and double click it.
7. Click on “Protocols and Ports” tab
8. Modify the “Local Port” area accordingly.
9. Click on “OK” button to save changes.

Data Source Driver Management

Refer to [Supported Data Sources](#) for a list of data sources supported by TDV. The Adapter guides describe the driver management for each of the adapters. Refer to the data source specific Adapter guide for more information.

TDV Updates and Bundled TDV Software

The TDV Client installers and patches can be found in the following directories, depending upon your platform:

- **Windows:** C:\Program Files\TIBCO\downloads
- **UNIX:** /opt/TIBCO/downloads

Studio installers, ODBC clients and ADO.NET client are also available under the downloads folder.

Refer to the chapter [Installing TDV, Studio, and Drivers](#) for instruction on client and driver installations.

Note: When a new version of TDV is released, it will be made available in the AWS Marketplace, but there is no automatic process to update existing customer AMIs to newer ones. TDV software offerings in AWS Marketplace are not maintained by TIBCO or AWS. After launching in AWS, you will need to manually apply the patch/hotfix updates as well as manage the security of the instance.

Installing Optional TDV Products

This topic describes the installation of optional TDV products. These topics are covered:

- [Version Support](#)
- [Installation Requirements](#)
- [Installing an Optional TDV Product](#)
- [Installing the TDV Client Drivers that are Distributed with TDV](#)
- [Importing Resources Defined in an Earlier Release](#)
- [Manage Active Cluster Security](#)

Version Support

TDV supports the versions listed in [Supported Add-On Adapters](#) and [Supported Advanced Data Source Adapters](#).

Installation Requirements

- [Add-On Adapter Installation Requirements](#)
- [Active Cluster Installation Requirements](#)
- [Installing the Advanced Adapters](#)

Add-On Adapter Installation Requirements

Individual adapters have these requirements:

- SAP BW BEx Adapter should be installed on a separate machine from the SAP GUI, to avoid possible conflict between JCo versions. See [Installing the SAP Java Connector Library](#), in the TDV SAP BW Adapter Guide.

- For Salesforce.com and SAP adapter installations, disable User Account Control.
- SAP BW can cause TDV errors similar to:

```
com.compositesw.cdms.webapi.WebapiException: Error [sapbw-
29000000]: BAPI_ODSO_READ_DATA_UC failed: Key figure 0CMPYPTAMT
unknown in InfoProvider 0BP_REL
```

To avoid this error, locate and install the patch listed in SAP “Note 1243987 - Extraction from DataStore object fails.” Install this patch, and use the program SAP_RSADMIN_MAINTAIN to set the parameter RSDRI_DS_NEW in the table RSADMIN to ' ' (empty or space).

Note: To upgrade from an earlier version of an adapter, install the new version and then see [Importing Resources Defined in an Earlier Release](#) .

Active Cluster Installation Requirements

This section lists the software and hardware requirements for Active Cluster. All data sources and databases that are supported with this release of TDV are supported by Active Cluster.

- [TDV File Customizations](#)
- [Digital Certificates](#)
- [Supported Platforms](#)
- [Disk Space and Physical Memory](#)
- [Load Balancer Requirements](#)

TDV File Customizations

The data source capability files and LDAP properties file are not automatically synchronized with other machines in the cluster. Therefore, if you customized the ldap.properties file or data source capability files on a TDV Server that will be in a cluster, you need to copy these files manually to all computers that are members of the cluster.

For example, if you modified the external domain configuration file and the data source capability file for DB2, you would need to copy the following files to all computers that are or will become members of the cluster:

```
<TDV_install_dir>/conf/server/ldap.properties
```

```
<TDV_install_dir>/apps/server/apps/dlm/cis_ds_db2/conf/db2.capabilities
```

Digital Certificates

A digital certificate ensures the identity of a particular computer and the data it transmits to another computer. Every server in an Active Cluster must have a digital certificate set up on the computer. A trial digital certificate is shipped with TDV Server but must be changed to ensure full security. See [Updating the Digital Certificate to Secure Cluster Communication](#) for how to do this.

Supported Platforms

See [Installation Requirements and Support Information](#) for a list of the platforms and protocols supported by Active Cluster.

Disk Space and Physical Memory

Active Cluster requires an additional 4 MB of disk space.

Load Balancer Requirements

Although a load balancer is not required to be used with Active Cluster, it is highly recommended to achieve the maximum benefits of using Active Cluster.

Installing an Optional TDV Product

All optional TDV products are installed for you when you perform the TDV Server install.

Before you create or join a cluster, make sure that all cluster members use the same level of encryption.

Installing the Advanced Adapters

To deploy a new adapter, you will need to have a running TDV Server and the new adapter jar file.

Auto Deployment

Follow these steps to deploy the adapter automatically when TDV is restarted:

1. Obtain new adapter (e.g. tdv.<adapter_name>.zip)
2. Copy tdv.<adapter_name>.zip to <TDV_install_dir>/tmp
3. Unzip the tdv.<adapter_name>.zip under <TDV_install_dir>/tmp
4. Copy the <adapter name>.jar file to the folder <TDV_Install_Dir>/packages/autodeploy_ds_adapters
5. Restart TDV and the adapter is deployed automatically
6. You can verify the deployment using Studio. Navigate to host/packages to verify that the adapter is deployed successfully.

Manual Deployment

Follow these instructions to install the advanced adapters:

1. Obtain new adapter (e.g. tdv.<adapter_name>.zip)
2. Copy tdv.<adapter_name>.zip to <TDV_install_dir>/tmp
3. Unzip the tdv.<adapter_name>.zip under <TDV_install_dir>/tmp
4. Open a shell window and go to the <TDV_Install_dir>

UNIX - /bin/sh

cd <TDV_Install_Dir>

Windows - cmd.exe with "Admin Privileges"

cd <TDV_Install_Dir>

5. Check if you have already deployed tdv.<adapter-name>

UNIX: ls -al ./packages

Windows: `dir .\packages`

Example: `tdv.googlebigquery.jar` is represented as `<TDV_install_dir>/packages/GoogleBigQuery_1.jar` when TDV Server has already deployed it.

6. If you find your `<adapter-name>` in the previous step, you must undeploy it first. You can undeploy the adapter using the command below:

UNIX: `./bin/server_util.sh -server <hostname> [-port <port>] -user <user> -password <password> -undeploy -name <adapter-name> -version 1`

Windows: `.\bin\server_util.bat -server <hostname> [-port <port>] -user <user> -password <password> -undeploy -name <adapter-name> -version 1`

Note: `<adapter-name>` for undeploy must match the adapter name under `<TDV_install_dir>/packages` that you are trying to undeploy.

Example:

- **Unix:** `./bin/server_util.sh -server <hostname> [-port <port>] -user <user> -password <password> -undeploy -name GoogleBigQuery -version 1`
- **Windows:** `.\bin\server_util.bat -server <hostname> [-port <port>] -user <user> -password <password> -undeploy -name GoogleBigQuery -version 1`

7. To deploy the adapter using the command below:

UNIX: `./bin/server_util.sh -server <hostname> [-port <port>] -user <user> -password <password> -deploy -package ./tmp/tdv.<adapter-name>/tdv.<adapter-name>.jar`

Windows: `.\bin\server_util.bat -server <hostname> [-port <port>] -user <user> -password <password> -deploy -package .\tmp\tdv.<adapter-name>\tdv.<adapter-name>.jar`

Example:

Unix: `./bin/server_util.sh -server <hostname> [-port <port>] -user <user> -password <password> -deploy -package ./tmp/tdv.googlebigquery/tdv.googlebigquery.jar`

Windows: `.\bin\server_util.bat -server <hostname> [-port <port>] -user <user> -password <password> -deploy -package .\tmp\tdv.googlebigquery\tdv.googlebigquery.jar`

8. To verify the new adapter was undeployed and deployed, check `<TDV_install_dir>/logs/cs_server.log`.

You will see messages about undeployment and deployment of your adapter.

Example: log snippet from `<TDV_install_dir>/logs/cs_server.log` when undeploying and deploying `tdv.googlebigquery.jar` (aka name=GoogleBigQuery, version=1)

```
INFO [jetty thread pool-413] 2020-04-29 19:17:30.749 -0700  
ExtensionManager - Undeployment of Extension Package GoogleBigQuery:1  
is successful!
```

```
INFO [jetty thread pool-420] 2020-04-29 19:19:16.672 -0700  
ExtensionManager - Deployment of Extension Package GoogleBigQuery:1 is  
successful!
```

Installing the TDV Client Drivers that are Distributed with TDV

This client distribution (driver zip) file includes the following components:

- ODBC
- ADO.NET
- JDBC
- SSIS
- Power BI

This zip file can be unpacked on each machine that has client application that needs access to the TDV Server.

To install the drivers distributed with TDV

1. Read any README files included with or associated with the download file.
2. Locate and extract the drivers zip file.
3. When installing the ODBC Win 64-bit driver on Windows 10, make sure to select Run as Administrator. Select the client EXE file, right click and select Run as Administrator. When prompted, select Yes and allow the installation to run to completion.
4. Follow the instructions in the TDV Administration Guide for details on how to complete configuration of each driver.
5. When the installation is complete, click Done to exit the installation program.

Importing Resources Defined in an Earlier Release

If you used a previous release of the adapter and defined resources, you can use them with this release. Follow the instructions for exporting and importing the resources in the *TDV User Guide*.

Manage Active Cluster Security

Users who create and manage an Active Cluster must have administrative privileges. SSL is used for inter-node communications and each server in an Active Cluster must have a valid digital certificate for authentication.

All cluster members must use the same level of encryption.

- [Updating the Digital Certificate to Secure Cluster Communication](#)
- [Set Access Privileges](#)

Updating the Digital Certificate to Secure Cluster Communication

Every TDV Server ships with a trial digital certificate so SSL works right out of the box. However, the security is poor. To secure cluster communication, you must update the digital certificate on each TDV Server node in the cluster.

Updating the digital certificate entails getting a signed digital certificate from a Certificate Authority (CA) and installing it in the keystore on each TDV Server. CAs are independent vendors (such as VeriSign) that have instructions on their websites for how to generate public key/private key pairs that accompany certificate requests. The CA then returns the digital certificate back to you. After you have this information, you need to install the digital certificate on the TDV Server.

To install a digital certificate on TDV Server

1. Open Studio, and select Administration > Launch Manager (Web) from the menu to open the Manager Web interface.

2. Click the CONFIGURATION tab and select SSL.
Manager displays the SSL Management page.
3. Enter new values as appropriate for your digital certificate, and click APPLY.

Set Access Privileges

You must have administrative privileges for Active Cluster management. Refer to the *Active Cluster Guide* for the specific rights needed for various cluster operations. Refer to the *Administration Guide* for more information about setting TDV access rights.

TDV and Business Directory Product Maintenance

This chapter explains the procedures involved in installing the latest service pack and the precautions that must be taken. If you are installing a base version of the products, you can skip this chapter.

- [About Hotfix Maintenance](#)
- [Downgrade/Rollback](#)
- [Applying the Service Pack to TDV Server, Studio, and Business Directory](#)
- [Applying the Service Pack to TDV Server, Studio, and Business Directory](#)
- [Applying the Service Pack or Hotfix to Active Cluster](#)
- [Upgrading from an Earlier Release and Migrating The Metadata](#)
- [Downgrade/Rollback](#)
- [Maintaining TDV-to-Client Application Connections](#)

Upgrade, Downgrade, and Rollback

Throughout this chapter, the following terminologies will be used:

- Upgrade - This is the process to upgrade to a higher release (for example, from 8.0.x to 8.1.x) by running the upgrade script that comes with the service pack.
- Downgrade - This is the process to revert to a previous release (for example, 8.1.0 to 8.0.1, or 8.0.3 to 8.0.1) by running the downgrade script that comes with the service pack. Refer to [Downgrade/Rollback](#) for more details.
- Rollback - This is the process to remove a hotfix/service pack within a release by running a rollback script, which is included with every service pack/hotfix. Refer to [Downgrade/Rollback](#) for more details.

About Hotfix Maintenance

If you are sent a patch to address a specific urgent issue, it is referred to as a hotfix patch. Hotfix patches have been quickly created by our engineering group for timeliness and are not for general use.

Hotfix patches are not automatically carried over through export and import processes during an upgrade or migration. They need to be reapplied if you migrate to another instance that is the same version. Hotfixes are also built for specific versions of TDV, so they should not be used for later versions of TDV than the version for which they were built.

For hotfixes, run the update process for TDV and Business Directory.

Applying the Hotfix to TDV Server, Studio, and Business Directory

Hot fixes include bug fixes to any TDV major, minor or service pack version.

The latest hotfix, must be applied to the same TDV major, minor and service pack level. For example,

- hotfix TDV 8.3.0.005 can only be applied to TDV 8.3.0
- hotfix TDV 8.3.1.002 can only be applied to TDV 8.3.1

The latest hotfix, requires a clean update of your TDV Server and Business Directory environment. The optimal way to achieve the clean environment is to apply the hotfix as if it was a Service Pack.

To apply a hotfix

1. Optionally, perform a full TDV and Business Directory backup. For details about backup export, see the TDV Administration Guide or the Business Directory Guide.
2. If installing on Windows, be sure to use Run as Administrator.
3. For Windows, make sure to close any open windows to the repository/jdk folder.
4. Stop the TDV Server.
5. Copy the hotfix zip file to <TDV_install_dir>.

6. Unzip the file.

To Rollback a hotfix, refer to [To rollback a hotfix for TDV Server, Studio, or Business Directory](#).

About Service Pack Maintenance

A service pack is a zipped package of files that fixes known issues and often provides enhanced functionality. All TDV Server, Studio and Business Directory software service packs are cumulative and supersede previously released service packs. A service pack should be applied on all computers where TDV products are installed, keeping them all at the same revision level. Installation of a service pack does not change configuration settings and custom functionality.

It is recommended that you keep your TDV Server and Business Directory Server instances at the same service pack level.

Applying the Service Pack to TDV Server, Studio, and Business Directory

Important bug fixes and additional functionality are added to TDV with each service pack.

Service packs typically involve careful ordering of procedural steps to make sure appropriate scripts are generated and available, and backup files saved.

Notes:

- Customers using advanced data source adapters that require OAuth need to first run TDV as a stand-alone server (no Monitor Daemon), set up OAuth once, and then run TDV with the Monitor Daemon.
- Sometimes when running Manager after a service pack install an error message pops up (Unable to finish loading...). You can safely ignore this message, close, and reopen Manager.

To apply a service pack

1. Optionally, perform a full TDV and Business Directory backup. For details about backup export, see the TDV Administration Guide or the Business Directory Guide.

2. If installing on Windows, be sure to use Run as Administrator.
3. For Windows, make sure to close any open windows to the repository/jdk folder.
4. Stop all TDV processes, instances, and databases.
5. Copy the service pack zip file to <TDV_install_dir>.
6. Unzip the file.
7. When asked whether to replace the existing files, reply Yes.

If you are not asked whether to replace files, the service pack is being extracted to some directory other than <TDV_install_dir>. The service pack must be extracted to the appropriate installation directory, and must overwrite files in that directory.

8. Run one or more of these scripts to upgrade your products.

Note: When running on a Liinux/AIX environment, you will have to grant execute permission on the upgrade scripts. Run the following command to do this:

```
chmod 755 *.sh
```

9. Enter the repo and cache passwords when prompted.

Product	Directory	Script
TDV Studio	bin	studio_upgrade_patch.bat
TDV Server	bin	cis_upgrade_patch.<sh bat>
TDV Business Directory	bin	bd_upgrade_patch.<sh bat>

10. After you run the scripts successfully, restart the TDV Server to fully activate any new libraries included in the upgrade.

To Rollback a service pack, refer to [Downgrade/rollback a service pack for TDV Server, Studio, or Business Directory](#).

Applying the Service Pack or Hotfix to Active Cluster

When updating Active Cluster, it is critical that clusterid remain unchanged. Changing it can adversely affect cached data.

These instructions are guidelines developed from testing with a two node Active Cluster environment.

To upgrade with scheduled system downtime

1. Determine and note the clusterid, so that it can remain unchanged.
2. Make sure that the cluster is in sync. That is, make sure both nodes are in the cluster and are functional.
3. Disable any cache refreshes on both of the nodes.
4. Configure the load balancer to stop sending traffic to node 1.
Shutting down node 1 would cause in-flight requests to fail.
5. Remove node 1 from the cluster.
6. If your TDV products are running, stop them, including all processes and databases used as repositories or caches.
7. Apply the service pack or hot fix to node1.
8. Start up node 1, but do not re-join the cluster.
9. Configure the load balancer to send traffic to node 1 instead of node 2.
10. Remove node 2 from the cluster.
11. Apply the service pack or hot fix to node 2.
12. Join node 1 and node 2 to the cluster.
13. Enable cache refreshes on both of the nodes.
14. Verify the cluster status.

Upgrading from an Earlier Release and Migrating The Metadata

This section is a guide for customers who are upgrading to TDV from a previous version and want to migrate metadata from that version to the new version.

Note: This process is different from many other software vendor upgrade procedures, which typically modify the existing instance.

The metadata upgrade process requires installing a new TDV instance in parallel with the existing TDV instance, exporting the metadata from the old instance, and importing the old instance's backup CAR file into the new TDV instance.

TDV recommends that you keep the older TDV instance until you are sure the new installation is stable. However, be aware that:

- If you are running two versions of TDV simultaneously, their port numbers must be different.
- If you are using Active Cluster, all servers in the cluster must be running at the same version and patch level.
- New instances of TDV can use the repository database of older instances.

Make sure that you have administrator privileges and perform all of the steps below as that user. In addition, it is advised that the installation and the upgrade steps be performed by the same user to avoid any permission issues that may arise.

To upgrade and migrate your existing installation, follow the steps (in the given order) in these sections

1. [Documenting the Existing TDV Instance](#)
2. [Considerations for Upgrading to TDV 8.X](#)
3. [Exporting Metadata from the Existing TDV Instance](#)
4. [Installing the New Version of TDV](#)
5. [Importing Metadata into the New TDV Instance](#)
6. [Verifying a Successful Installation](#)

Documenting the Existing TDV Instance

Before making a backup of the existing TDV instance, document the key features of the instance. These settings are later applied to the new TDV instance to ensure the consistency of results returned from published resources.

Note: If you cannot upgrade directly from the existing TDV instance to the new version, multiple versions of TDV and multiple export and import processes might be required, so that database schemas remain compatible.

Make note of the settings in the following table.

Setting	What to Record
Ports	The port numbers for the existing instance, because after the installation of the new server is complete, the port numbers of the new instance might need to be changed.
Authentication mechanism	The authentication mechanism. If LDAP or another dynamic authentication is used, the same settings need to be applied to the new server. This setting determines various authentication mechanisms enabled within Server.
Users/groups	The groups created in Server and the users that belong to these groups. If LDAP authentication is used, note the LDAP groups that were imported into Server.
Metadata repository	The full path of the repository location and the administrator user ID and password.
Custom data sources	Custom data sources that were introspected and any custom drivers that were used to introspect these sources.
External libraries	Any external libraries that were referenced from the instance.
Customized settings, including Java flags, managed and unmanaged memory	Configuration parameter settings for the existing Server instance. From the Administration menu, select Configuration and check all relevant parameter settings. The new TDV instance's settings should match the old instance settings if you want similar results and performance.

Considerations for Upgrading to TDV 8.X

In order to keep your database schemas compatible, you will need to preserve your metadata. You may need to perform an export and import of metadata.

During the export you perform in [Exporting Metadata from the Existing TDV Instance](#) all of the relevant information from your old repository is captured.

During the import you perform in [Importing Metadata into the New TDV Instance](#) all of the relevant information from your old repository is transferred into the new TDV PostgreSQL repository database.

Upgrading from a Pre-8.5 TDV Server Installation

On the windows platform only, if you are upgrading from a pre-8.5 TDV Server installation to TDV Server 8.5 or higher, note that you need to upgrade your TDV Studio to the same version you upgraded your TDV Server with.

Exporting Metadata from the Existing TDV Instance

The first step for upgrade or migration is to export the existing metadata information from the repository. This process writes out a CAR file that includes six files containing metadata, scheduling, settings, and user information.

To run the export

1. Verify that you have administrator privileges.
2. Open a command prompt window.
3. Navigate to <TDV_install_dir>/bin.
4. Perform a full backup with the options that you need:
 - Using Studio. See “Using Studio for a Full Server Backup” in the *TDV User Guide*.
 - Using the TDV backup_export utility. For more information, see the *TDV Administration Guide*.
5. Locate and copy the resulting CAR file to a safe and easily accessible location for use later.

Installing the New Version of TDV

To install a new TDV when you are upgrading from an earlier release

1. Review the new features and bug fixes as documented in the *TDV and Business Directory Release Notes*.
2. Install the new version of Server as described in [Installing TDV, Studio, and Drivers](#) or [Silent Mode Installation](#).
3. Install the latest versions of all other TDV software that you use.

4. Set up any external libraries, including JDBC drivers, and then shut down and restart the server.
5. Deliver the upgraded drivers (the TDV ODBC driver and the TDV JDBC driver) to the dependent clients.
6. To ensure consistency in results and performance, make the configuration of the new server instance similar to the old instance.
7. Perform the instructions in [Importing Metadata into the New TDV Instance](#) .

Perform the instructions in [Verifying a Successful Installation](#) .

Importing Metadata into the New TDV Instance

If you are upgrading your version of TDV from an earlier version and you have completed the instructions in [Exporting Metadata from the Existing TDV Instance](#) , then follow the instructions in this section. If you are performing a new installation, you can skip these instructions.

After the new TDV instance is successfully installed, the metadata from old TDV instance needs to be imported into the new instance. After the import is successfully completed, settings such as Java configurations, managed memory setting and ports can be updated on the new instance.

To run the import

1. Verify that you have administrator privileges.
2. Locate the CAR file that you produced from [Exporting Metadata from the Existing TDV Instance](#) .
3. Perform a full backup import with the options that you need:
 - Using the Studio Import dialog window. For more information, see the *TDV User Guide*.
 - Using the TDV backup_import utility. For more information, see the *TDV Administration Guide*.
4. Validate that the TDV resources or other settings are as you expect in the new version of Studio.

Verifying a Successful Installation

To verify that your installation of the TDV software was successful, follow the steps in this section.

To verify a successful installation

1. Start Studio. Follow instructions in the TDV Getting Started Guide or in the TDV User Guide.
2. If you have just completed an upgrade from one version of TDV to a new one, then we suggest that you complete these instructions:
 - a. Determine a set of tests that will touch all published resources and all introspected data sources, and then apply the tests against:
 - The existing instance of TDV, as a sanity check.
 - The new instance of TDV, to ensure the same results are produced.
 - b. Configure and use the PubTest tool to test all your published resources.

The PubTest program can be configured to test all published resources using JDBC, ODBC, and Web services. Additional configuration might be required to test the ODBC and Web services. Starting with TDV 4.0, an end-to-end testing program referred to as PubTest (pubtest.java) is included with the TDV installation. This program is located in the <TDV_install_dir>\apps\jdbc directory. A PubTest.doc file in this directory provides additional documentation about using this tool.

Downgrade/Rollback

If you installed a service pack or hotfix and you later decide that you want to revert to the previous installation, you can do that.

Precautions

It is important to understand the requirements and limitations of downgrade/rollback procedures:

- You can use this process to downgrade to an earlier release (for example, 8.7 to 8.4). But, you cannot use this process to downgrade to a version prior to the current version (for example, 8.x to 7.x).

- The rollback scripts are only for rolling back to a previous service pack. If a release has only one service pack or hotfix applied to it, rollback is not guaranteed to work, in an attempt to rollback to the base GA version. For example, 8.0.1 release refers to the hotfix release 1 of a 8.0.0 base GA release. You cannot rollback to 8.0.0 in this case. If you do this, starting the TDV server may result in error messages such as these:
 - In the cs_repository.log: “Could not receive data from client: No connection could be made because the target machine actively refused it.”
 - In the cs_monitor.log: “MONITOR STOP. The metadata repository was created with a newer version of the server. The server cannot continue.”

Downgrade/rollback a service pack for TDV Server, Studio, or Business Directory

If downgrading on Windows, any open windows to the repository or jdk folder must be closed.

If your TDV products are running, stop them, including all processes and databases used as repositories or caches.

Downgrade from 8.7 to 8.6/8.5/8.4/8.3 Server

To downgrade TDV Server from version 8.6 to 8.4/8.3, follow these steps:

1. Stop monitor daemon:

```
composite.<bat|sh> monitor stop
```

Note: To check if the Monitor daemon has stopped completely, check for the “Bye” message in cs_server.log.

2. Start server process without monitor daemon:

```
composite_server.sh(bat) run
```

3. Run the following command:

```
server_util.bat/sh -server <id> -port <port> -user <uid>
```

```
-password <password> -rollbackRepository -toVersion 8.4.0
```

(or 8.3.0). For BD bd_server_util.bat/sh

Example:

```
bin/server_util.bat -server localhost -port 9400 -user
```

```
admin -password admin1 -rollbackRepository -toVersion 8.4.0
```

(or 8.3.0)

4. Kill all java processes.
5. Stop repo and cache.
6. Run <install_dir>/install/rollback_patch.sh(bat) script (for Server).
7. Run the command:

```
chmod 733 /bin/cis_downgrade_patch.sh
```

8. Run the command

```
cis_<product>_<version>_downgrade_patch.bat (CIS - windows) /
cis_<product>_<version>_downgrade_patch.sh (CIS - UNIX)
```

9. Apply 8.3 or 8.4 patch (downgraded version).
10. Restart the service.
11. Import backup CAR files as needed.
12. Optionally, redefine all your VCS roots and connections.
13. To check the metadata version after a downgrade, run the following command. A list of metadata versions for each TDV release is given in [Repository Versions of TDV Server/BD](#)

```
./repository/bin/psql -Uroot -hlocalhost -p9408 -d cisrepo -c "select *
from cisrepo.metadata_version"
```

Downgrade from 8.7 to 8.6/8.5/8.4/8.3 BD/Studio

To downgrade TDV Studio/BD from version 8.7 to 8.6/8.5/8.4/8.3, follow these steps:

1. Stop monitor daemon:

```
composite.<bat|sh> monitor stop
```

Note: To check if the Monitor daemon has stopped completely, check for the “Bye” message in cs_server.log.

2. Run <install_dir>/install/rollback_patch.sh(bat) script (for Studio) and <install_dir>/bd/install/rollback_patch.sh(bat) script (for BD)
3. Stop the repo and cache (For BD only).
4. Run bd/studio_downgrade_patch.bat (sh)
5. Apply downgraded version of BD/Studio patch
6. Restart the service (For BD only).

To rollback a hotfix for TDV Server, Studio, or Business Directory

1. If working on Windows, any open windows to the repository or jdk folder must be closed.
2. Make sure that the TDV Server is stopped.

```
composite.<bat|sh> monitor stop
```

Note: To check if the Monitor daemon has stopped completely, check for the “Bye” message in cs_server.log.

3. Run the rollback script located in

```
<TDV_Install_Dir>/install/rollback_TIB_tdv_<product><version>_HF-
<hotfix_number>_all.<bat|sh>
```

4. Navigate to <install_dir> and reapply the last service pack or hotfix. After downgrade it is necessary to apply the downgraded version of service pack/hotfix before starting the server.
5. Start the Monitor Daemon.

```
For TDV: composite.<bat|sh> monitor start
```

```
For BD: bd.<bat|sh> monitor start
```

In the server.log, verify that the version is the target version you intended.

Repository Versions of TDV Server/BD

Release Version	TDV Server	BD
8.3	32	30
8.4	33	30
8.5	34	30
8.6	34	30
8.7	35	30

Maintaining TDV-to-Client Application Connections

This section includes instructions for how to update connections between TDV Server and your client applications in the following sections:

- [Updating an ODBC Client Application](#)
- [Updating a JDBC Client Application](#)

Updating an ODBC Client Application

To patch ODBC client applications

1. Install the ODBC patch as you would for a Studio installation, but apply the ODBC patch only to where you installed a TDV ODBC client:
 - 32-bit driver: <TDV_install_dir>/apps/odbc
 - 64-bit driver: <TDV_install_dir>/apps/odbc64
2. See the *TDV Client Interfaces Guide* for more information.

To patch ODBC client users on Windows 7

1. Locate your Windows 7 DSN entry file.
2. Recreate any custom system DSNs, using DSN tools.
3. See the *TDV Client Interfaces Guide* for more information.

Updating a JDBC Client Application

To patch JDBC client applications

1. If you only need to obtain the updated TDV JDBC driver for your JDBC client, you can do one of the following:
 - Obtain the updated csjdbc.jar file from the server's <TDV_install_dir>/apps/jdbc/lib directory after the Studio patch is installed.
 - Extract any one of the csjdbc.jar files directly from the patch zip file.
2. See the *TDV Client Interfaces Guide* for more information.

Uninstalling TDV

This topic describes the process of uninstalling TIBCO® Data Virtualization (TDV) and related products for Windows and UNIX. The uninstall process is similar, regardless of whether you performed a silent or interactive installation.

- [Uninstalling TDV on Windows](#)
- [Uninstalling TDV on UNIX](#)

Uninstalling TDV on Windows

When you uninstall TDV, everything stored in the metadata repository is deleted along with the TDV software.

To uninstall TDV on Windows

1. Stop the Server and Repository if they are running.
2. Start the uninstallation process:
... > Uninstall TDV
For a silent uninstall, TDV is uninstalled without further interaction. For an interactive uninstall, go to step 3.
3. Click OK to confirm the uninstall.
4. Click Done when the uninstallation process is completed.

Uninstalling TDV on UNIX

The following tasks are described here:

- [Preparing for Uninstalling on UNIX](#)
- [Uninstalling TDV On UNIX](#)

Preparing for Uninstalling on UNIX

Before you uninstall TDV, remove the TDV service files from the installation machine, because the uninstaller does not remove these files automatically.

To remove the TDV service files `cis.repository` and `cis.server`

1. Log into the installation machine as root.
2. Change the working directory to `<TDV_install_dir>/bin`.
3. Run the following command:

```
cis_remove_services.sh
```

Uninstalling TDV On UNIX

During the uninstallation process, all the components from the previous installation are removed. You cannot uninstall the components individually.

To uninstall TDV on UNIX

1. Log into the installation machine as the user that installed the software.
2. Run the following command:

```
<TDV_install_dir>/uninstall
```

For a silent uninstall, run

```
./uninstall --mode "unattended"
```

For an interactive uninstall, go to step 3.

3. Press the Enter key.

You will see a warning about loss of data.

4. Press the Enter key to complete the uninstallation process and leave the uninstaller.

TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the [Product Documentation website](#), mainly in HTML and PDF formats.

The [Product Documentation website](#) is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The following documentation for this product is available on the [TIBCO Data Virtualization](#) page.

Users

- TDV Getting Started Guide
- TDV User Guide
- TDV Web UI User Guide
- TDV Client Interfaces Guide
- TDV Tutorial Guide
- TDV Northbay Example

Administration

- TDV Installation and Upgrade Guide
- TDV Administration Guide
- TDV Active Cluster Guide
- TDV Security Features Guide

Data Sources

- TDV Adapter Guides

TDV Data Source Toolkit Guide (Formerly Extensibility Guide)

References

TDV Reference Guide

TDV Application Programming Interface Guide

Other

TDV Business Directory Guide

TDV Discovery Guide

TIBCO TDV and Business Directory Release Notes Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Release Version Support

TDV 8.5 is designated as a Long Term Support (LTS) version. Some release versions of TIBCO Data Virtualization products are selected to be long-term support (LTS) versions. Defect corrections will typically be delivered in a new release version and as hotfixes or service packs to one or more LTS versions. See also [Long Term Support](#).

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the [TIBCO Ideas Portal](#). For a free registration, go to [TIBCO Community](#).

Legal and Third-Party Notices

SOME CLOUD SOFTWARE GROUP, INC. (“CLOUD SG”) SOFTWARE AND CLOUD SERVICES EMBED, BUNDLE, OR OTHERWISE INCLUDE OTHER SOFTWARE, INCLUDING OTHER CLOUD SG SOFTWARE (COLLECTIVELY, “INCLUDED SOFTWARE”). USE OF INCLUDED SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED CLOUD SG SOFTWARE AND/OR CLOUD SERVICES. THE INCLUDED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER CLOUD SG SOFTWARE AND/OR CLOUD SERVICES OR FOR ANY OTHER PURPOSE.

USE OF CLOUD SG SOFTWARE AND CLOUD SERVICES IS SUBJECT TO THE TERMS AND CONDITIONS OF AN AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER AGREEMENT WHICH IS DISPLAYED WHEN ACCESSING, DOWNLOADING, OR INSTALLING THE SOFTWARE OR CLOUD SERVICES (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH LICENSE AGREEMENT OR CLICKWRAP END USER AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE SAME TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of Cloud Software Group, Inc.

TIBCO, TIBCO logo, TIBCO O logo, ActiveSpaces, Enterprise Messaging Service, Spotfire, TERR, S-PLUS, and S+ are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only. You acknowledge that all rights to these third party marks are the exclusive property of their respective owners. Please refer to Cloud SG’s Third Party Trademark Notices (<https://www.cloud.com/legal>) for more information.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: <https://scripts.sil.org/OFL>

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

Cloud SG software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the “readme” file

for the availability of a specific version of Cloud SG software on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. CLOUD SG MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S), THE PROGRAM(S), AND/OR THE SERVICES DESCRIBED IN THIS DOCUMENT AT ANY TIME WITHOUT NOTICE.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "README" FILES.

This and other products of Cloud SG may be covered by registered patents. For details, please refer to the Virtual Patent Marking document located at <https://www.tibco.com/patents>.

Copyright © 2002-2023. Cloud Software Group, Inc. All Rights Reserved.