Tivoli Application Dependency Discovery Manager
Version 7.3

Administrator's Guide

IBM
Note

Before using this information and the product it supports, read the information in "Notices" on page 283.

Edition notice

This edition applies to version 7, release 3 of IBM Tivoli Application Dependency Discovery Manager (product number 5724-N55) and to all subsequent releases and modifications until otherwise indicated in new editions.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
### Tables

1. Discovered entities with descriptions .............................. 2
2. Services default interface settings ................................. 41
3. Services default interface settings ................................. 41
4. Ping Sensor and Port Sensor default ports .......................... 42
5. Default host settings for domain server public connectivity services .................................. 44
6. Default port settings for domain server public connectivity services .................................. 44
7. Default host settings for domain server local connectivity services .................................. 45
8. Communication between the database server and the domain server ........................................ 45
9. Communication between Discovery Management Portal, API clients and Web portal and Data Management Portal clients, and the domain server .................................. 46
10. Communication between the anchor and gateway, and the domain server .................................. 46
11. Local connectivity communication configuration for a domain server .................................. 46
12. Default host settings for primary storage server, secondary storage server, and discovery server public connectivity services .................................. 47
13. Default port settings for primary storage server, secondary storage server, and discovery server public connectivity services .................................. 48
14. Default host settings for primary storage server and secondary storage server inter-server connectivity services .................................. 48
15. Default port settings for primary storage server inter-server connectivity services .................................. 48
16. Default port settings for secondary storage server inter-server connectivity services .................................. 48
17. Default host settings for primary storage server, secondary storage server, and discovery server local connectivity services .................................. 49
18. Inter-server connectivity communication configuration in the streaming server deployment .................................. 49
20. Communication between the anchor and gateway, and the discovery server .................................. 52
21. Local connectivity communication configuration in the streaming server deployment .................................. 52
22. Default host settings for domain server and synchronization server public connectivity services .................................. 53
23. Default host settings for domain server public connectivity services .................................. 53
24. Default port settings for synchronization server public connectivity services .................................. 54
25. Default host settings for domain server and synchronization server inter-server connectivity services .................................. 54
26. Default port settings for domain server inter-server connectivity services .................................. 54
27. Default port settings for synchronization server inter-server connectivity services .................................. 54
28. Default host settings for the domain server and synchronization server local connectivity services .................................. 55
29. Inter-server connectivity communication configuration in the synchronization server deployment .................................. 55
30. Communication between Discovery Management Portal, API clients and Web portal and Data Management Portal clients, and the domain and synchronization servers .................................. 56
31. Communication between the anchor and gateway, and the domain server .................................. 56
32. Local connectivity communication configuration in the synchronization server deployment .................................. 57
33. Sensor names used in the makeASDScriptPackage command .................................. 101
34. ........................................................................ 103
35. SSH keys .................................................................. 110
36. The hierarchyDomain and hierarchyType attributes values .................................. 123
37. Buffer pool size guidelines (db_cache_size) .................................. 138
38. Monitoring Coverage reports .................................. 167
39. Predefined sensor reports .................................. 168
40. Predefined snapshot reports .................................. 171
41. The supported products versions .................................. 184
42. Integrating TADDM with IBM Tivoli Monitoring via OSLC Automation .................................. 185
43. Topics that contain more information about discovery over OSLC .................................. 198
44. User tasks with corresponding integration capabilities to use .................................. 201
45. Topics that contain more information about discovery using IBM Tivoli Monitoring .................................. 202
46. Topics that contain more information about change events .................................. 203
47. Topics that contain more information about launch in context .................................. 203
48. Valid graph values and their relationship to the guid parameter .................................. 210
49. Operator names of a TADDM MQL query .................................. 213
50. State codes .................................................................. 226
51. ........................................................................ 250
52. ........................................................................ 251
53. ........................................................................ 251
54. ........................................................................ 251
55. ........................................................................ 252
56. ........................................................................ 252
57. ........................................................................ 253
About this information

The purpose of this PDF document version is to provide the related topics from the information center in a printable format.

Conventions used in this information center

In the IBM® Tivoli Application Dependency Discovery Manager (TADDM) documentation certain conventions are used. They are used to refer to the operating system-dependent variables and paths, the COLLATION_HOME directory, and the location of the collation.properties file, which is referenced throughout the TADDM documentation, including in the messages.

Operating system-dependent variables and paths

In this information center, the UNIX conventions are used for specifying environment variables and for directory notation.

When using the Windows command line, replace $variable with %variable% for environment variables, and replace each forward slash (/) with a backslash (\) in directory paths.

If you are using the bash shell on a Windows system, you can use the UNIX conventions.

**COLLATION_HOME directory**

TADDM root directory is also referred to as the COLLATION_HOME directory.

On operating systems such as AIX® or Linux, the default location for installing TADDM is the /opt/IBM/taddm directory. Therefore, in this case, the $COLLATION_HOME directory is /opt/IBM/taddm/dist.

On Windows operating systems, the default location for installing TADDM is the c:\IBM\taddm directory. Therefore, in this case, the %COLLATION_HOME% directory is c:\IBM\taddm\dist.

**Location of collation.properties file**

The collation.properties file contains TADDM server properties and includes comments about each of the properties. It is located in the $COLLATION_HOME/etc directory.

Terms and definitions

Refer to the following list of terms and definitions to learn about important concepts in the IBM Tivoli Application Dependency Discovery Manager (TADDM).

**access collection**

A collection that is used to control the access to configuration items and permissions to modify configuration items. You can create access collections only when data-level security is enabled.
asynchronous discovery
In TADDM, the running of a discovery script on a target system to discover systems that cannot be accessed directly by the TADDM server. Because this discovery is performed manually, and separately from a typical credentialed discovery, it is called “asynchronous”.

business application
A collection of components that provides a business functionality that you can use internally, externally, or with other business applications.

CI
See configuration item.

collection
In TADDM, a group of configuration items.

configuration item (CI)
A component of IT infrastructure that is under the control of configuration management and is therefore subject to formal change control. Each CI in the TADDM database has a persistent object and change history associated with it. Examples of a CI are an operating system, an L2 interface, and a database buffer pool size.

credentialed discovery
TADDM sensor scanning that discovers detailed information about the following items:
- Each operating system in the runtime environment. This scanning is also known as Level 2 discovery, and it requires operating system credentials.
- The application infrastructure, deployed software components, physical servers, network devices, virtual systems, and host data that are used in the runtime environment. This scanning is also known as Level 3 discovery, and it requires both operating system credentials and application credentials.

credential-less discovery
TADDM sensor scanning that discovers basic information about the active computer systems in the runtime environment. This scanning is also known as Level 1 discovery, and it requires no credentials.

Data Management Portal
The TADDM web-based user interface for viewing and manipulating the data in a TADDM database. This user interface is applicable to a domain server deployment, to a synchronization server deployment, and to each storage server in a streaming server deployment. The user interface is very similar in all deployments, although in a synchronization server deployment, it has a few additional functions for adding and synchronizing domains.

discover worker thread
In TADDM, a thread that runs sensors.

Discovery Management Console
The TADDM client user interface for managing discoveries. This console is also known as the Product Console. It is applicable to a domain server deployment and to discovery servers in a streaming server deployment. The function of the console is the same in both of these deployments.

discovery server
A TADDM server that runs sensors in a streaming server deployment but does not have its own database.
domain

In TADDM, a logical subset of the infrastructure of a company or other organization. Domains can delineate organizational, functional, or geographical boundaries.

domain server

A TADDM server that runs sensors in a domain server deployment and has its own database.

domain server deployment

A TADDM deployment with one domain server. A domain server deployment can be part of a synchronization server deployment.

In a domain server deployment, the following TADDM server property must be set to the following value:

com.collation.cmdbmode=domain

launch in context

The concept of moving seamlessly from one Tivoli® product UI to another Tivoli product UI (either in a different console or in the same console or portal interface) with single sign-on and with the target UI in position at the proper point for users to continue with their task.

Level 1 discovery

TADDM sensor scanning that discovers basic information about the active computer systems in the runtime environment. This scanning is also known as credential-less discovery because it requires no credentials. It uses the Stack Scan sensor and the IBM® Tivoli® Monitoring Scope sensor. Level 1 discovery is very shallow. It collects only the host name, operating system name, IP address, fully qualified domain name, and Media Access Control (MAC) address of each discovered interface. Also, the MAC address discovery is limited to Linux on System z® and Windows systems. Level 1 discovery does not discover subnets. For any discovered IP interfaces that do not belong to an existing subnet that is discovered during Level 2 or Level 3 discovery, new subnets are created based on the value of the com.collation.IpNetworkAssignmentAgent.defaultNetmask property in the collation.properties file.

Level 2 discovery

TADDM sensor scanning that discovers detailed information about each operating system in the runtime environment. This scanning is also known as credentialed discovery, and it requires operating system credentials. Level 2 discovery collects application names and the operating system names and port numbers that are associated with each running application. If an application has established a TCP/IP connection to another application, this information is collected as a dependency.

Level 3 discovery

TADDM sensor scanning that discovers detailed information about the application infrastructure, deployed software components, physical servers, network devices, virtual systems, and host data that are used in the runtime environment. This scanning is also known as credentialed discovery, and it requires both operating system credentials and application credentials.

multitenancy

In TADDM, the use by a service provider or IT vendor of one TADDM installation to discover multiple customer environments. Also, the service provider or IT vendor can see the data from all customer environments, but within each customer environment, only the data that is specific to the
respective customer can be displayed in the user interface or viewed in reports within that customer environment.

**Product Console**
See *Discovery Management Console*.

**script-based discovery**
In TADDM, the use, in a credentialed discovery, of the same sensor scripts that sensors provide in support of asynchronous discovery.

**SE**  
See *server equivalent*.

**server equivalent (SE)**
A representative unit of IT infrastructure, defined as a computer system (with standard configurations, operating systems, network interfaces, and storage interfaces) with installed server software (such as a database, a web server, or an application server). The concept of a server equivalent also includes the network, storage, and other subsystems that provide services to the optimal functioning of the server. A server equivalent depends on the operating system:

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Approximate number of CIs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>500</td>
</tr>
<tr>
<td>AIX</td>
<td>1000</td>
</tr>
<tr>
<td>Linux</td>
<td>1000</td>
</tr>
<tr>
<td>HP-UX</td>
<td>500</td>
</tr>
<tr>
<td>Network devices</td>
<td>1000</td>
</tr>
</tbody>
</table>

**storage server**
A TADDM server that processes discovery data that is received from the discovery servers and stores it in the TADDM database. The primary storage server both coordinates the discovery servers and all other storage servers and serves as a storage server. All storage servers that are not the primary are called secondary storage servers.

**streaming server deployment**
A TADDM deployment with a primary storage server and at least one discovery server. This type of deployment can also include one or more optional secondary storage servers. The primary storage server and secondary storage servers share a database. The discovery servers have no database.

In this type of deployment, discovery data flows in parallel from multiple discovery servers to the TADDM database.

In a streaming server deployment, the following TADDM server property must be set to one of the following values:

- `com.collation.taddm.mode=DiscoveryServer`
- `com.collation.taddm.mode=StorageServer`

For all servers except for the primary storage server, the following properties (for the host name and port number of the primary storage server) must also be set:

- `com.collation.PrimaryStorageServer.host`
- `com.collation.PrimaryStorageServer.port`
If the com.collation.taddm.mode property is set, the com.collation.cmdbmode property must not be set or must be commented out.

**synchronization server**
A TADDM server that synchronizes discovery data from all domain servers in the enterprise and has its own database. This server does not discover data directly.

**synchronization server deployment**
A TADDM deployment with a synchronization server and two or more domain server deployments, each of which has its own local database.

In this type of deployment, the synchronization server copies discovery data from multiple domain servers one domain at a time in a batched synchronization process.

In a synchronization server deployment, the following TADDM server property must be set to the following value:
com.collation.cmdbmode=enterprise

This type of deployment is obsolete. Therefore, in a new TADDM deployment where more than one server is needed, use the streaming server deployment. A synchronization server can be converted to become a primary storage server for a streaming server deployment.

**TADDM database**
In TADDM, the database where configuration data, dependencies, and change history are stored.

Each TADDM server, except for discovery servers and secondary storage servers, has its own database. Discovery servers have no database. Storage servers share the database of the primary storage server.

**TADDM server**
A generic term that can represent any of the following terms:
• domain server in a domain server deployment
• synchronization server in a synchronization server deployment
• discovery server in a streaming server deployment
• storage server (including the primary storage server) in a streaming server deployment

**target system**
In the TADDM discovery process, the system to be discovered.

**utilization discovery**
TADDM sensor scanning that discovers utilization information for the host system. A utilization discovery requires operating system credentials.
Administering

TADDM overview

IBM Tivoli Application Dependency Discovery Manager (TADDM) is a configuration management tool that helps IT operations personnel ensure and improve application availability in application environments. TADDM provides the details of configuration items (CIs) using automated, agentless discovery of assets and their application dependencies, and it includes a discovery library technology to help leverage data from other sources.

TADDM provides operational staff with a top-down view of applications so that they can quickly understand the structure, status, configuration, and change history of their business-critical applications. When performance and availability problems occur, this view helps the staff to immediately isolate issues and to more effectively plan for application change without disruption. The TADDM database, a configuration management database, is created and maintained without requiring custom infrastructure modeling. TADDM also provides complete cross-tier dependency maps, topological views, change tracking, event propagation, and detailed reports and analytics.

TADDM depends on the discovery of information, which is performed using sensors that are deployed as part of the TADDM product. The data that results from the discovery process is used to create cross-tier dependency maps that link the physical and logical topologies. This hierarchical directory represents your entire runtime environment.

The following steps are a high-level summary of what TADDM does:

1. Sensors determine and collect the identity, attributes, and settings of each application, system, and network component.
2. The configuration data, dependencies, and change history are stored in the TADDM database, and the topologies are stored on the TADDM server. When CIs are discovered, they are stored in the TADDM database from the following sources:
   - Sensors
   - Discovery library books, which are also known as Identity Markup Language (IdML) books, that are generated by external management software systems
   - APIs
3. The discovered data is displayed as runtime, cross-tier application topologies in the TADDM user interface. Subsequent discoveries update the topology. Also, TADDM maintains the change history of the infrastructure configuration and dependencies.
4. TADDM generates reports and additional topological views of the information that is stored in the TADDM database.

Entities that TADDM discovers

Table 1 on page 2 lists and describes the entities that TADDM discovers in your environment.
<table>
<thead>
<tr>
<th>Entity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network tier</td>
<td>The following devices are discovered with the MIB2 (RFC 1213) parameter values for each device:</td>
</tr>
<tr>
<td></td>
<td>• Routers</td>
</tr>
<tr>
<td></td>
<td>• Switches</td>
</tr>
<tr>
<td></td>
<td>• Load balancers</td>
</tr>
<tr>
<td></td>
<td>• Firewalls</td>
</tr>
<tr>
<td></td>
<td>• Generic IP devices</td>
</tr>
<tr>
<td>System tier</td>
<td>The following devices are discovered at the system tier:</td>
</tr>
<tr>
<td></td>
<td>• Server hosts and disks</td>
</tr>
<tr>
<td></td>
<td>• Host IP interfaces</td>
</tr>
<tr>
<td></td>
<td>• Database servers</td>
</tr>
<tr>
<td></td>
<td>• Load balancers or clusters</td>
</tr>
<tr>
<td>Application tier</td>
<td>The following components are discovered at the application tier. Also, for each component (except for the generic processes), version information, configuration files and properties, host information, and vendor-specific extensions are discovered.</td>
</tr>
<tr>
<td></td>
<td>• Custom servers, based on custom templates that you design</td>
</tr>
<tr>
<td></td>
<td>• Java EE application servers and configurations</td>
</tr>
<tr>
<td></td>
<td>• Java EE and Java SE components and modules</td>
</tr>
<tr>
<td></td>
<td>• Web server components</td>
</tr>
<tr>
<td></td>
<td>• Web modules, configuration files, and installation directories</td>
</tr>
<tr>
<td></td>
<td>• Generic JVM processes</td>
</tr>
<tr>
<td></td>
<td>• Databases</td>
</tr>
<tr>
<td>Infrastructure services</td>
<td>The system infrastructure services that support the application environment are discovered, and the dependency structure between these service components and the application components are discovered. The following components are in the infrastructure service:</td>
</tr>
<tr>
<td></td>
<td>• DNS and NFS services</td>
</tr>
<tr>
<td></td>
<td>• LDAP</td>
</tr>
<tr>
<td>Relationship structure</td>
<td>In addition to the discovery of components, the physical and logical connectivity at the network, system, and application tiers are discovered at the following level of support in each of the tiers:</td>
</tr>
<tr>
<td></td>
<td>• Layer 3 IP connectivity</td>
</tr>
<tr>
<td></td>
<td>• Layer 2 connectivity</td>
</tr>
<tr>
<td></td>
<td>• Application component runtime dependencies</td>
</tr>
<tr>
<td></td>
<td>• Infrastructure service dependencies</td>
</tr>
</tbody>
</table>

Configurations and interdependencies are discovered across the following entities:
• Application components, such as Web servers, application servers, and databases
• System components, such as hosts, operating systems, load balancers, and database servers
• Network components, such as routers, switches, and firewalls
• Infrastructure services, such as DNS and LDAP services

Note: Using virtual IP addresses or multiple network interface controllers might cause TADDM to report incorrect results. When planning a discovery, consider the network infrastructure.

Discovery process overview
Discovery is a multilevel process that collects configuration information about the entire application infrastructure, identifying deployed software components, physical servers, network devices, virtual systems, and host data that is used in the runtime environment. Discovery is performed using sensors that are part of TADDM product.

The job of the sensor is to discover configuration items (CIs), create model objects, and persist the model objects to the TADDM database. The sensors use protocols that are specific to the resources that they are designed to discover. The following protocols are examples:
• Cisco Discovery Protocol (CDP)
• Java™ Management Extensions (JMX)
• Secure Shell (SSH)
• Simple Network Management Protocol (SNMP)
• Structured Query Language (SQL)

When possible, a secure connection is used between the TADDM server and the target systems.

TADDM does not run discoveries over IPv6 networks, but IPv6 attributes are discovered by discoveries running on IPv4 networks.

Sensors
TADDM provides a variety of specialized sensors for discovery of almost all components in the typical data center, across the application software, host, and network tiers. You can also develop custom sensors for unique components. Sensors reside on the TADDM server and collect configuration attributes and dependencies.

Sensors are nonintrusive, meaning that they run on the TADDM server rather than on the client workstation. Therefore, by using TADDM, you can gather discovery-related information without incurring the costs of local installation and maintenance of an agent on each client workstation that you want to discover.

Because sensors use secure network connections, encrypted access credentials, and host-native utilities, they are secure, and they provide the same level of data acquisition that you have when you use software that is located on the client workstation.

A sensor has the following three configurable aspects:

Scope The discovery scope is typically a valid IP range, a subnet, or a specific address. It sets the boundary for the discovery.

Access list The access list is a collection of the credentials, such as the user names, passwords, and Simple Network Management Protocol (SNMP) community strings, that the sensor uses when accessing the configuration
items in the application infrastructure. You must configure the access list for the configuration items that you want to discover.

**Schedule**
Discovery can be run on demand, run on a schedule, or driven by externally triggered events. The schedule identifies whether sensors are run on demand or on a schedule.

**How a sensor discovers configuration items:**

These steps outline how a sensor discovers configuration items (CIs) in your environment.

1. To identify the active IP devices in the specified scope, the sensor tries a TCP connection on several ports (such as 22, 23 and 135) to detect a response. Any response is enough to inform the sensor that the device exists.
2. The sensor tries to connect to the IP device on several ports (such as 22 and 135) to determine the technology to use to discover the host.
3. If a port using the Secure Shell (SSH) protocol is open, the sensor tries to establish an SSH connection using credentials from the access list. In sequence, the sensor tries access list entries of type **computer system** or **windows computer system** until either an entry works, or the sensor reaches the end of the access list with no success.
4. If a Windows Management Instrumentation (WMI) port is open, an SSH connection is established with a gateway computer system (if one can be found for the target system). In sequence, the sensor tries access list entries of type **windows computer system** until either an entry works, or the sensor reaches the end of the access list with no success.
5. If a session cannot be established, an SNMP sensor is run. If a session is established, a computer system sensor is run.
6. A computer system sensor tries to determine the type of operating system that is installed.
7. TADDM runs a sensor that is specific to the operating system to discover the operating system in more depth.
8. During the in-depth discovery of the operating system, which is based on specific criteria (such as port number and process name), TADDM runs software-specific sensors to discover application details.

**How an application sensor is started:**

This information describes how an application sensor is started.

The GenericServerSensor runs the following commands:

**On Linux, Solaris, AIX, and Linux on System z® operating systems**
- `lssof -nP -i` to get the port information
- `ps axww` to get the command line information

**On Windows operating systems**
- `netstat.exe -nao` to get the port information
- `wmic process list` to get the command line information

The process ID (PID) is used to merge the output. The template matcher then operates on the merged data. When the logging level is set to DEBUG in the `collation.properties` file, the output of these commands is located in the following logs:
The merged data must match the criteria that is defined in the sensor template. You can find the template criteria that starts a sensor in the following sample template definition for the DB2® sensor.

Run the following command (a redirect to a file is helpful), replacing <username> and <password> with a valid user name and associated password (for example, ...
dist/sdk/bin/api.sh -u administrator -p collation find --depth=5 AppServerTemplate):

...dist/sdk/bin/api.sh -u <username> -p <password> find --depth=5 AppServerTemplate

The preceding command produces XML output that is the template definition. In the template definition, if the value for the <order> element is less than 0, the template is for a sensor. If the value for the <order> element is greater than 0, the template is for a custom server. The matching occurs starting with the lowest value for the <order> element so that is how sensors get higher matching priority than custom servers.

The following sample template definition is for the DB2 sensor. Notice the two <operand1> elements, one with a value of db2tcmpm and one with a value of db2agent. The value for the <boolExp> element indicates whether both or only one of the <operand1> values must be present. A value of 1 for the <boolExp> element indicates the logical operator OR, which means only one of the <operand1> values must be present. A value of 0 for the <boolExp> element indicates the logical operator AND, which means both of the <operand1> values must be present.

```xml
<Template array="18" guid="C1A992327AFF33409C41D5C710460B89" lastModified="117755771479"
xsi:type="coll:com.collation.platform.model.discovery.template.AppServerTemplate">
  <displayName>DB2</displayName>
  <name>DB2</name>
  <type>DatabaseServer</type>
  <internal>true</internal>
  <filterSet guid="B599AE918F436C99F0AEB6E8ED8F02" lastModified="117755771475" parent="C1A992327AFF33409C41D5C710460B89"
xsi:type="coll:com.collation.platform.model.discovery.template.FilterSet">
    <displayName>unknown</displayName>
    <filterList array="1" guid="B8E4D5165383E878FF0D532E8E" lastModified="117755771476" parent="B599AE918F436C99F0AEB6E8ED8F02"
xsi:type="coll:com.collation.platform.model.discovery.template.Filter">
      <displayName>unknown</displayName>
      <operand1>db2tcmpm</operand1>
      <operator>equals</operator>
      <part>Program Name</part>
    </filterList>
    <filterList array="2" guid="63816C902B0A317F8C38247A1EEBC17" lastModified="117755771471" parent="B599AE918F436C99F0AEB6E8ED8F02"
xsi:type="coll:com.collation.platform.model.discovery.template.Filter">
      <displayName>unknown</displayName>
      <operand1>db2agent</operand1>
      <operator>equals</operator>
      <part>Program Name</part>
    </filterList>
  </filterSet>
</Template>
```
Levels of discovery
TADDM provides four levels of discovery: Level 1 discovery, Level 2 discovery, Level 3 discovery, and utilization discovery.

Level 1 discovery
TADDM sensor scanning that discovers basic information about the active computer systems in the runtime environment. This scanning is also known as credential-less discovery because it requires no credentials. It uses the Stack Scan sensor and the IBM Tivoli Monitoring Scope sensor.

Level 1 discovery is very shallow. It collects only the host name, operating system name, IP address, fully qualified domain name, and Media Access Control (MAC) address of each discovered interface. Also, the MAC address discovery is limited to Linux on System z and Windows systems.

Level 1 discovery does not discover subnets. For any discovered IP interfaces that do not belong to an existing subnet that is discovered during Level 2 or Level 3 discovery, new subnets are created based on the value of the com.collation.IpNetworkAssignmentAgent.defaultNetmask property in the collation.properties file.

Level 2 discovery
TADDM sensor scanning that discovers detailed information about each operating system in the runtime environment. This scanning is also known as credentialed discovery, and it requires operating system credentials.

Level 2 discovery collects application names and the operating system names and port numbers that are associated with each running application. If an application has established a TCP/IP connection to another application, this information is collected as a dependency.

Level 3 discovery
TADDM sensor scanning that discovers detailed information about the application infrastructure, deployed software components, physical servers, network devices, virtual systems, and host data that are used in the runtime environment. This scanning is also known as credentialed discovery, and it requires both operating system credentials and application credentials.

utilization discovery
TADDM sensor scanning that discovers utilization information for the host system. A utilization discovery requires operating system credentials.

Level 2 and Level 3 discoveries collect more detailed information than Level 1 discoveries. If objects that are created during a Level 2 or Level 3 discovery match objects that were previously created by a Level 1 discovery, the objects that were created by the Level 1 discovery are replaced by the newly created objects, which, in turn, causes the Globally Unique Identifiers (GUIDs) for the objects to change. Therefore, in general, Level 1 data should not be used for integration with other products.
Discovery profiles
To run a discovery, you must specify a discovery profile, which defines a set of options for discovery. Using discovery profiles, you can configure individual sensors, manage multiple configurations of the same sensor, choose the appropriate configuration based on a set of criteria, and manage sets of configurations of different sensors to be applied on a single discovery run.

By selecting the appropriate discovery profile, you can control the depth of discovery, or discovery level.

By default, TADDM provides four discovery profiles. Three are for the three levels of discovery that you can choose (Level 1, Level 2, or Level 3), depending on whether you want to do a credential-less or credentialed discovery. The remaining profile is for a utilization discovery.

If no profile is specified, the Level 3 discovery profile is used by default, although you can change the default profile in the Discovery Management Console.

For additional information on discovery profiles, see A Flexible Approach to Discovery on the TADDM wiki at https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Tivoli%20Application%20Dependency%20Manager/page/A%20Flexible%20Approach%20to%20Discovery.

Enabling and disabling sensors
You can globally disable a sensor even if the sensor has been enabled by a profile. You can also globally enable a sensor and allow the setting in the profile to work.

For example, if a sensor is globally enabled, and is enabled in the profile, the sensor runs. If the sensor is globally enabled, but is disabled in the profile, the sensor does not run when the respective profile is selected for running a discovery.

For the global enabling and disabling to work for sensors that have an osgi directory ($COLLATION_HOME/osgi/plugins), you must change the AgentConfigurations in the osgi directory.

For example, for the Db2Sensor, look for these directories:

- $COLLATION_HOME/osgi/plugins/com.ibm.cdb.discover.sensor.app.db.db2_x.x.x/Db2Sensor.xml
- $COLLATION_HOME/osgi/plugins/com.ibm.cdb.discover.sensor.app.db2windows_x.x.x/Db2WindowsSensor.xml

where x.x.x is the sensor plugin version, for example 7.3.

When editing the XML files, to enable the sensor, set enabled to true. To disable the sensor, set enabled to false.

For sensors that do not use the osgi/plugins directory, the configuration information is stored in the sensor configuration XML file that is in the etc/discover-sensors directory.

Asynchronous and script-based discovery
In asynchronous and script-based discovery, instead of running individual commands, sensors provide a discovery script, which they run against the target system.
Not all sensors support asynchronous and script-based discovery. Only sensors that provide a discovery script can support these types of discovery.

For information about which sensors support asynchronous and script-based discovery, see the Sensors that support script-based and asynchronous discovery topic in the TADDM Sensor Reference.

Some differences from a nonscript-based discovery

Asynchronous discovery and script-based discovery differ from a nonscript-based discovery in the following important ways:

- In comparison to the discovery results from a nonscript-based Level 2 or Level 3 discovery, the discovery results from an asynchronous or a script-based discovery might not be as complete. Most sensors discover a greater number of model objects, attributes, and relationships in a nonscript-based discovery than in an asynchronous or a script-based discovery.

- In asynchronous or script-based discovery, application sensors are started only once for a given target system. However, if the application is listening on more than one port, each application instance is discovered. In a nonscript-based discovery, an application sensor is started for each application instance.

Asynchronous discovery:

You can run the asynchronous discovery to discover systems that cannot be accessed directly by the TADDM server. This includes systems that are in secure locations (for example, systems that are not network-accessible), systems that do not run Secure Shell (SSH), and systems with sensitive information, for which credentials cannot be obtained.

In the asynchronous discovery, users run a discovery script on a target system. The discovery script contains a main script and several sensor scripts. Each sensor script provides a discovery capability that is similar to a function that the sensor performs, when it is run in a typical discovery.

The output of the discovery script is an archive file that contains the discovery results. You must copy this file to the TADDM server. During a TADDM discovery, TADDM sensors process the discovery results from this archive file (rather than running commands).

Because this discovery is performed manually, and separately from a typical discovery where credentials are required, it is called “asynchronous”.

To run asynchronous discovery, the asynchronous discovery sensor is required. For more information, see the TADDM Sensor Reference.

For information about configuring sensors to run asynchronous discovery, see “Configuring for asynchronous discovery” on page 100.

Script-based discovery:

In a script-based discovery, you can use a discovery script in a typical discovery, where credentials are required. In this kind of discovery, the same sensor scripts are used as in the asynchronous discovery.
In script-based discovery, a sensor does not run individual commands. Rather, the sensor script is run on the target system. Application-specific credentials might not be needed.

For example, to discover the IBM WebSphere® application in a typical discovery, you must create an access list entry with credentials for the WebSphere application if security is enabled. However, using script-based discovery, the WebSphere access list entry is not needed. Script-based discovery also eliminates the use of application-specific protocols like Java Management Extensions (JMX), which can extend application discovery through IBM Tivoli Monitoring.

For information about configuring sensors to run script-based discovery, see "Configuring for script-based discovery" on page 104.

**Concurrent discovery**

You can run more than one discovery at the same time, which is called concurrent discovery. For example, because a large discovery can take several hours to complete, you might want to start smaller discoveries before the large discovery completes. Before you run concurrent discoveries, you must correctly configure them.

You can run a concurrent discovery using a different discovery profile than the one used to start the first discovery.

To manage concurrent discoveries, use the Discovery Management Console or the `api.sh` script. For more information about using the `api.sh` script, see the Command-line interface API topic in the TADDM SDK Developer’s Guide.

You can run concurrent discoveries on the same target system. If two or more discoveries are monitoring some of the same IP addresses, each discovery operates independently.

If a password change occurs while a discovery is running, and a concurrent discovery is started, the sensors in that concurrent discovery will immediately use the new credentials, assuming that those sensors did not start before the password change occurred.

TADDM does not support concurrent discovery with a profile-based access list.

If changes are made to the custom server template while a discovery is running, any concurrent discovery that is started continues to use the existing version of the custom server template. The next separate, and non-concurrent, discovery that is started uses the new version of the custom server template.

**Determining the displayed FQDN**

You can configure a preferred method of determining the fully qualified domain name (FQDN) for each discovered system.

For a Level 1 discovery, the FQDN is the result of a reverse lookup of the IP address. This lookup uses the resolver library provided by the operating system and it uses any configuration provided there. For example, if, at the operating system level, the host file is preferred over DNS, information in the hosts file is considered first.

For a Level 2 discovery, TADDM performs a reverse lookup of all discovered IP addresses using the resolver library provided by the operating system. Again,
operating system configuration dictates from where the reverse lookup gets information. If DNS is not configured, or the DNS returns unwanted FQDNs, you can use the hosts file to override it.

After the discovered IP addresses have been looked up, an attempt is made to match an FQDN to the computer system. There are a number of different ways to get an FQDN and each method is attempted, in a predefined order, until a valid FQDN is found. You can modify the order so that your preferred method has a higher priority. The following methods are available:

**Method 1**

TADDM selects the FQDN of an IP interface where the host portion of the FQDN matches the host name of the discovered system. If there are multiple matches, the selected FQDN depends on the priority of the domain name as defined in the property:

```
com.collation.platform.os.FqdnPriorities
```

This property lists the domain names in order of priority. To prioritize the domains, enter the name of the domains as a comma-separated list on one line:

```
com.collation.platform.os.FqdnPriorities=domain1.company.com, domain2.company.com, domain3.company.com
```

The FQDN with the highest priority for its domain is returned as the FQDN. This method uses information that is discovered about FQDNs of interfaces and computer system names.

If the priorities are not defined, TADDM goes through all of the IP interfaces. TADDM checks if the FQDN associated with a given IP interface equals the name of the computer system or if the host name portion of this FQDN equals the name of the computer system. The FQDN first matching the criteria is returned as the FQDN.

For example, a computer system named “myname” has two interfaces with the following FQDNs:

- interface #1 myname-domain1.com
- interface #2 myname-domain2.com

If the `com.collation.platform.os.FqdnPriorities` is not defined then the first match is returned as the FQDN name. Both names have the host portion of the FQDN matching the host name of the discovered system but the FQDN returned is “myname-domain1.com”. To prioritize which name is selected, use the property `com.collation.platform.os.FqdnPriorities`. For example, if the `com.collation.platform.os.FqdnPriorities` entry contains the following information:

```
com.collation.platform.os.FqdnPriorities=domain2.com,domain1.com
```

In this case, the FQDN returned is “myname-domain2.com” because this name has a higher priority.

**Method 2**

The property `com.collation.platform.os.command.fqdn` specifies an external command on the TADDM server that is used to do the reverse lookup. The following examples show how to use this property, enter the property on one line:

```
com.collation.platform.os.command.fqdn=nslookup $1 | grep Name | awk '{print $2}'
```

```
com.collation.platform.os.command.fqdn=AIX=nslookup $1 | grep Name | awk '{print $2}'
```

```
com.collation.platform.os.command.fqdn=Linux=nslookup $1
```

10 Application Dependency Discovery Manager: Administering
| grep Name | awk '{print $2}'
com.collation.platform.os.command.fqdn.SunOS=nslookup $1
| grep Name | awk '{print $2}'
com.collation.platform.os.command.fqdn.Windows=nslookup $1

Method 3
The property com.collation.platform.os.command.hostOfHostname specifies an external command on the target system that is used to provide the FQDN. The following example shows how to use this property on a UNIX system, enter the property on one line:
com.collation.platform.os.command.hostOfHostname=host `hostname`
| awk '{print $1}'

Method 4
The FQDN of the primary interface is used. The primary IP interface is specified as the lowest IP value where the IP values are sorted in ascending order.

Method 5
The IP address of the primary interface is used.

Method 6
The name of the computer system is used.

Method 7
Set to the session context IP.

Method 8
Set FQDN for CS as FQDN for the session IP.

You can define the order in which these methods are attempted by setting the com.collation.platform.os.fqdnSearchOrder property. The value of this property is a comma-separated list of the numbers of the methods. The default value is 1,2,3,4,5,6,7,8. In this case, TADDM first tries to use method 1. If it does not return a valid FQDN, it tries method 2, and so on, until it gets a valid FQDN and stops. A valid FQDN is a fully qualified domain name that conforms to the rules specified in RFC 1035.

This solution is also applicable for computer systems that are discovered through the use of SNMP sensors. You can define which solutions have a higher priority and therefore can be used to find an FQDN more quickly.

In all cases, properly configured DNS is the preferred way of setting host names. If DNS cannot be used, use the hosts file. The use of DNS or the hosts file are the standard ways of providing name resolution for IP addresses. TADDM provides ways to override these methods, but because any other methods are unique to TADDM, they might lead to names that are inconsistent with names in other management systems.

Tracing a discovery
You can trace the phases of discovery from when a discovery starts to when the change history is updated and the topology dependencies are built. Each phase of a discovery is recorded in an associated log file.

Discovery run phase and log file
After you start a discovery, each discovery is assigned a unique identifier (run ID). A time stamp of YYYY-MM-DD-hh:mm:ss:SSS identifies the discovery run, for example, 20110517225225948. The YYYY-MM-DD portion represents the year, month, and day. The hh:mm:ss.sss represents the time of day on a 24-hour clock,
carried out to the thousandths of a second. In the preceding example, the date is 2011/05/17 and the time is 22:52:25.948. You can use this identifier to create separate log files for each sensor in the $COLLATION_HOME/log/sensors directory. The time stamp is used within the log files.

During a discovery, the process flow manager monitors the state of the discovery and the state of the sensor events. The process flow manager also manages the hand off from one service to another. The process flow activity is stored in the $COLLATION_HOME/log/services/ProcessFlowManager.log file on the discovery or domain server.

The following examples show various activities that are monitored by the process flow manager and how this information is stored in the log file.

**Starting discovery:**
- 2011-05-17 22:53:01,973 ProcessFlowManager [RMI TCP Connection(42)-127.0.0.1] INFO processflowmgr.ProcessFlowManagerImpl - Discovery run, 2011051722525948 started with profile Level 2 Discovery

**Discovery done:**

**Discovery event:**

**Topology builder phase and log file**

The topology builder builds the relations and dependencies between the discovered items. The topology builder runs a list of agents that are listed in the $COLLATION_HOME/etc/TopologyBuilderConfigurationDefault.xml file. The topology agents run at specified intervals. However, events that occur during a discovery and when a discovery is completed can also trigger the topology builder. Each agent carries out a specific task, for example, consolidates, figures out dependencies, builds dependencies charts, and removes old information. The topology builder log files are stored in the $COLLATION_HOME/log/services/TopologyBuilder.log and $COLLATION_HOME/log/agents/*.log files on the domain server, synchronization server, and primary storage server.

The following examples show the various stages when building relationships and shows how this information is stored in the log file.

**Starting builder execution:**

**Topobuilder done:**
Moving to next TopoAgent:


If you find problems, for example the topology builder is hanging, check for the last started topology agent in the log file to identify the problem. If there are no entries in the TopologyBuilder.log file, check the entries in the TopologyManager.log file after the timestamp of the last started agent. If you know which agents cause the problems, you can also check the $COLLATION_HOME/log/agents/agentName.log file to identify them.

Other services and log files

The change manager processes events and updates the change history records. This processing is independent of the discovery phase; it receives events from other services, for example, the topology builder process and the bulk load program. When you open a topology view, the view manager builds the structures that are required for the GUI to render the topology efficiently. The services logs are stored in the : $COLLATION_HOME/log/services directory. Each service log has the same name as the service, for example, services/ChangeManager.log file.

The following examples show how this information is stored in the service log files.

ChangeManager:

ChangeManagerPersisterImpl - [ChangeManagerPersister.I.3] Got a create or delete event

ViewManager:


Last successful credentials caching

TADDM can cache last working access credentials. They can be reused in the next (Level 2 or script-based) discovery.

During the initial discovery of a target, TADDM server iterates through the access list and validates each item against the discovery target. When the valid credentials are found, they are stored in a cache and reused during the consecutive discoveries of the same discovery target.

A cache can store the two following values:

credentials

This value is stored in a cache when the valid credentials for a discovery target are found during the discovery. During the next discovery, they are read from the cache and checked whether they are still valid. If they are still valid, they are used for the discovery. If they are no longer valid and the fallback is disabled, the information that the last attempt failed is stored in the server and the discovery is stopped. When the fallback is enabled, the server iterates through the access list and tries to find new valid credentials. To enable the fallback, set the com.ibm.cdb.security.auth.cache.fallback.failed property to true.
information that the last attempt failed (along with the last error)

This value is stored in a cache when the valid credentials for a discovery target are not found during the discovery. If the fallback is disabled, the information that the last attempt failed is displayed and the discovery is stopped. If the fallback is enabled, the server iterates through the access list and tries to find new valid credentials. To enable the fallback, set the `com.ibm.cdb.security.auth.cache.fallback.invalid` property to `true`.

By default, fallback in both cases is enabled. You can customize the fallback behavior and credentials caching by appropriately setting the access credentials caching properties.

**Note:** Credentials are cached per IP address, location tag, credential type, and protocol that is used during connection. When access entry is removed, all associated cache entries are also removed. Credentials cache can be managed by the new utility `cachemgr`.

**Limitations**
- Credentials caching is not used in Level 3 discovery. It is used only for Level 2 computer system discovery and script-based sensors.
- A cache does not track scope access restriction changes. For example, if a discovery target is within scope access restriction and is discovered and cached, and then moved out of scoped restriction, the cached value is still used.
- The cached value has precedence over the profiled access list. For example, if you run discovery by using the main access list and valid credentials are stored, the cache value is still used, even if you specify other credentials in a profile.

You can remove a cached value by using the `cachemgr` utility. If you often use different profiles with different access entries against the same discovery target or scope, you can disable caching for them. Otherwise, wrong credentials might be used in the discovery.

**Topology building process overview**

TADDM runs the topology building process on a periodic basis. Until the topology building process has completed after a discovery or after a bulk load operation, unreconciled objects might exist in the TADDM database, and the topology relationships might be incomplete.

This process is the same regardless of which type of TADDM deployment you use.

Topology building includes the following operations:

**Cleaning up the TADDM database**

The process deletes old entities, removes dependencies that are lacking sources or targets, and removes other items that are superseded.

**Establishing dependencies between configuration items**

The process creates dependencies between communicating processes, such as between an application and the underlying database and between sending and receiving WebSphere MQ queues. It also establishes dependencies between parts of an application cluster, or simply between two computer systems.

**Creating and augmenting configuration items**

The process uses information from existing configuration items and connections to synthesize new configuration items. For example, TADDM
might create a new configuration item that is called “ApplicationServerClusters” and is based on information that is derived from earlier discoveries and bulk load operations.

**Creating information for topology views**

The process generates and stores information that can be used by the Data Management Portal to display topology views more quickly.

**Exporting data**

The process queries the TADDM database to export configuration item information to external systems. For example, integration with Registry Services is implemented as a topology agent.

**Log files and logging**

The TADDM Troubleshooting Guide and its topics describe the TADDM log files and how to set up logging for troubleshooting.

**Securing the environment**

In secure environments, TADDM enforces authentication to help protect confidential information.

You can use the Data Management Portal to configure user accounts. Each user must have a valid user account to use the Data Management Portal to access discovered information about network and infrastructure components.

When you log in to the Discovery Management Console and select the **Establish a secure (SSL) session** option, all data is encrypted (including user names and passwords) before it is sent over the network.

**Note:** Fix Pack 5 If the check box **Establish a secure (SSL) session** is selected while launching Discovery Management Console, TADDM server must be running in secure mode. To do that, the property `com.ibm.cdb.secure.server` should be set to true.

In the discovery process, the TADDM server uses Secure Shell (SSH) protocol to securely communicate with all computer hosts and other devices that support SSH.

The server supports both key-based SSH authentication, and login-based, password-based SSH authentication. When login-based, password-based SSH authentication is used, the user names and passwords that you define in the access list are used to log in to the computer hosts to be discovered.

See also “Security properties” on page 88.

**Controlling user access to configuration items**

TADDM controls user access to configuration items through the use of access collections, roles, and permissions.

Access control to configuration items is established by the following process:

1. Configuration items are aggregated into access collections.
2. Roles are defined that aggregate sets of permissions.
3. Users or user groups are defined, and roles are assigned to each user or user group to grant specific permissions (for specific access collections) to that user.
In the context of security in TADDM, a user is a person who is given access to configuration items, and a user group is several users who have the same roles or permissions.

You can create users and user groups in the Data Management Portal. The access of users and user groups to configuration items is defined by the roles and access collections that you assign to each user or user group. You can change these assignments at any time.

**Permissions**

A permission authorizes the user to perform an action or access a specific configuration item. Permissions are aggregated into roles, and users are granted permissions by assigning them roles that have those permissions.

TADDM provides four permissions, each of which is classified as either a data-level permission or a method-level permission.

**Data-level permissions**

Read and Update are data-level permissions.

**Read** The user can view information about a configuration item.

**Update** The user can change information about a configuration item.

**Method-level permissions**

Discover and Admin are method-level permissions.

**Discover** The user can start a discovery, create and update discovery scope objects, or create new objects from, for example, the Edit menu of the Discovery Management Console.

A user without the Discover permission cannot log in to the Discovery Management Console, or view the Discovery tab in the Data Management Portal.

**Admin** The user can create or update users, roles, and permissions. The user can also configure authorization policy with the authorization manager.

**Enabling data-level security**

You can enable data-level security for AIX, Linux, Linux on System z, and Windows operating systems by editing the collation.properties file.

To enable data-level security so that you can grant Read and Update permissions selectively, complete the following steps.

1. In the collation.properties file, locate the following line, and change the value of the property from false to true:
   ```
   com.collation.security.enabledatalevelsecurity=false
   ```
2. Save the file.
3. Stop the TADDM server.
4. Restart the TADDM server.
Note: In a streaming server deployment, you must update the collation.properties file on each storage server, and restart each storage server.

You can set more granular permissions by creating access collections. If data-level security is enabled, major TADDM resources can be secured by using access collections. If data-level security is enabled, users can modify only those CIs contained within access collections to which they have Update permission.

Auxiliary resources, such as physical geography resources that include the SiteInfo attribute, are not displayed when creating an access collection.

Roles
A role is a set of permissions that can be assigned to a user. Assigning a role confers specific access capabilities.

When you assign a role to a user, you must specify one or more access collections for that role. This limits the scope of the role to only those access collections that are appropriate for that user.

For example, Sarah is responsible for the NT servers and workstations of your company, and you assign her the supervisor role for an access collection that contains those systems. Jim is responsible for the Linux systems, and you assign him the supervisor role for an access collection that contains those systems. Although Sarah and Jim are assigned the same role (because they perform the same operations), they have access to different resources.

Note: If you are using a synchronization server, you must create the role for each TADDM domain, and synchronize the domain servers with the synchronization server.

Predefined roles
TADDM provides the following predefined roles:

operator
This role has Read permission.

supervisor
This role has Read, Update, and Discover permissions.

administrator
This role has Read, Update, Discover, and Admin permissions.

Additional roles that you can create
You can create additional roles to assign other combinations of permissions. The following combinations might be especially useful:

Read + Update
Permission to read and update objects in assigned access collections.

Read + Update + Admin
Permission to read and update objects in assigned access collections and to create users, roles, and permissions.
Access collections
TADDM does not manage access to configuration items on an individual basis. Instead, the configuration items are aggregated into sets called access collections. An access collection is a set of configuration items that is managed collectively for security purposes.

The security of each access collection is managed by creating roles and assigning the roles to users. The role applies only to the access collections that you specify when assigning the role to a user. Therefore, access collections are used to limit the scope of the role.

When you install TADDM, the access collection that is named DefaultAccessCollection is created, and it contains all configuration items. All users have Read and Update permissions for this access collection by default, unless you enable data-level security.

Note: Users do not have permissions to read and update access collections, they can only read and update individual configuration items. However, users have Read and Update permissions for those access collections that are members of access collections assigned to them.

Resetting security policies
If you need to reset the security policies (permissions, roles, and access collections) to their default state, you can do so by replacing two files. However, resetting security policies requires that you delete and re-create all users.

About this task
The security policies are stored in the following two files in the $COLLATION_HOME/var/policy directory, and these files are used to initialize the security policies:
- AuthorizationPolicy.xml
- AuthorizationRoles.xml

After the security policies are initialized, these files are renamed and stored in the same directory. For example, the following files have been renamed:
- AuthorizationPolicy.backup.xml
- AuthorizationRoles.backup.xml

Default versions of the files, which contain the supplied security policies, are also located in the same directory. The following files are the default versions:
- DefaultPolicy.xml
- DefaultRoles.xml

Procedure
To restore the default security policies, complete the following steps:
1. To save the current policy files, rename them, or move them to a different directory.
2. Delete any users that you created.
3. Delete the $COLLATION_HOME/var/ibmsecauthz directory.
4. Create a copy of the DefaultPolicy.xml file, and name it AuthorizationPolicy.xml.
5. Create a copy of the `DefaultRoles.xml` file, and name it `AuthorizationRoles.xml`.

6. Restart the server.

7. As needed, create users.

**Lockouts**

You can use lockouts to lock a single user, or all users, out of TADDM if the configured number of failed login attempts allowed is exceeded. Using the lockout feature provides better authentication control and helps to prevent the use of brute-force password cracking.

A local lockout is triggered if a single user exceeds the configured number of failed login attempts. As a result, the user cannot log in to TADDM for a configured length of time.

When a global lockout is triggered, no users can log in to TADDM for a configured length of time. A global lockout is triggered by one of the following two situations:

- The number of active lockouts for different users exceeds the configured number of maximum global lockouts allowed.
- The number of failed login attempts for unique user names exceeds the configured limit.

When a lockout is triggered, existing sessions are not affected.

You can specify the number of failed login attempts allowed and the length of time for which a lockout remains active by configuring properties in the `collation.properties` file. For more information about these properties, see "Lockout properties" on page 85.

When a global lockout time elapses, any local lockouts in progress are automatically cleared.

In a synchronization server deployment, the synchronization server controls the security of all TADDM domains. Any lockouts that were active on the domain server before it was connected to the synchronization server are cleared when synchronization between the domain server and the synchronization server is enabled.

The failed login attempts that count towards the total can be of any type, for example, using the CLI API, Java API, tools (scripts), SOAP, REST, Discovery Management Console, or Data Management Portal. The lockout feature applies to integrations that use the TADDM API, but it does not apply to logins using single sign-on, or database-based integrations, for example Tivoli Common Reporting.

A TADDM server administrator can clear a local or a global lockout by using the `$COLLATION_HOME/bin/lockmgr.sh` script. You can run the script from the following servers:

- Domain server, in a domain server deployment
- Synchronization server, in a synchronization server deployment
- Primary storage server, in a streaming server deployment

You can run the `lockmgr.sh` script with the following options:
lockmgr.sh -s
   Displays the lockout status.

lockmgr.sh -g
   Clears an active global lockout.

lockmgr.sh -u username
   Clears an active local lockout for a particular user.

lockmgr.sh -h
   Displays help information for the lockmgr.sh script.

Encryption

Encryption is the process of transforming data into an unintelligible form in such a way that the original data either cannot be obtained or can be obtained only by using a decryption process.

Fix Pack 5  TADDM uses the property 'com.collation.security.algo.aes.keylength' to decide the algorithm (AES 128 or AES 256) from the 'FIPS-compliant IBMJCEFIPS' security provider to encrypt the following items:
   - Passwords, including entries in the collation.properties and userdata.xml files
   - Access list entries that are stored in the database

For example:

This property defines the key length for AES
-com.collation.security.algo.aes.keylength=128.

When you install TADDM for the first time, an encryption key is generated, and passwords are encrypted using this new encryption key. The default location for the encryption key is the etc/TADDMSec.properties file.

Changing the location of the TADDM encryption key

To change the location of the encryption key, change the value of the com.collation.security.key property in the collation.properties file. You can set the property to another location that is relative to the $COLLATION_HOME directory.

To avoid data loss, store a backup copy of the encryption key in a separate location. The key can be restored if a problem occurs with the original copy.

Changing the TADDM encryption key in a domain server deployment

Note: TADDM does not support changing the encryption key after installation in a streaming server deployment and synchronization server deployment.

To change the TADDM encryption key in a domain server deployment, use the bin/changekey.sh script (or the equivalent batch script file). This script migrates encrypted entries in the collation.properties and userdata.xml files and migrates access list entries that are stored in the database. To use the bin/changekey.sh script, ensure that you are logged in as the non-root user that was defined during installation.

You must restart TADDM after successful use of this script.
Format for running the script

./changekey.sh $COLLATION_HOME admin_user admin_password

Example

./changekey.sh /opt/IBM/taddm/dist administrator taddm

FIPS compliance

You can configure TADDM to operate in a mode that uses FIPS-compliant algorithms for encryption by setting the FIPSMode property, `com.collation.security.FIPSMode` to true.

Set the `com.collation.security.FIPSMode` property to true in the following files:
- $COLLATION_HOME/dist/etc/collation.properties
- $COLLATION_HOME/dist/sdk/etc/collation.properties
- sdk/etc/collation.properties of every TADDM SDK installations that connect to the FIPS-compliant TADDM.

The default value of the `com.collation.security.FIPSMode` property is false.

When in FIPS mode, TADDM uses the following FIPS 140-2 approved cryptographic providers:
- IBMJCEFIPS (certificate 376)
- IBMJSSEFIPS (certificate 409)


FIPS mode can be used with all types of TADDM discoveries with the following exceptions:
- Level 2 SNMP discovery
- Level 2 i5/OS discovery
- Level 2 ZEnterprise discovery
- Level 2 VMware ESXi discovery
- Level 3 VMware Virtual Center discovery
- Level 3 JBoss discovery
- Level 3 Oracle Application Server discovery
- Level 3 WebLogic discovery
- Level 3 SAP CCMS and SLD discovery
- Level 3 EMC discovery
- Level 3 Sybase discovery
- Level 3 and 4 discoveries, where Windows Management Instrumentation (WMI) or PowerShell session (PowerShell session is supported in TADDM 7.3.0.2, or later) is used to discover Windows platforms, only if Windows TADDM server, Windows gateways, and Windows discovery targets do not run in FIPS-compliant mode. To configure Windows servers to run in the FIPS-compliant mode, see the Windows documentation, for example [http://support.microsoft.com/kb/811833](http://support.microsoft.com/kb/811833).

When in FIPS mode, TADDM sensors that use SSH cannot connect to the servers that support only SSHv1 protocol or only SSHv2 protocol with too weak ciphers.
TADDM is not able to verify whether SSH implementation on target servers is FIPS-compliant. You must check whether SSH implementations that you use in your environment are FIPS-compliant.

In FIPS mode, when you use TADDM SDK and Discovery Management Console in the secure mode, only IBM Java is supported.

**Related concepts:**

- "SP800-131 compliance"

You can configure TADDM to support the National Institute of Standards and Technology (NIST) SP800-131a security standard.

### SP800-131 compliance

You can configure TADDM to support the National Institute of Standards and Technology (NIST) SP800-131a security standard.


△To enable the SP800-131a mode, set the `com.ibm.jsse2.sp800-131` property to `strict` in the following files:

- `$COLLATION_HOME/dist/etc/collation.properties`
- `$COLLATION_HOME/dist/sdk/etc/collation.properties`
- `sdk/etc/collation.properties` of every TADDM SDK installation that connects to the SP800-131 compliant TADDM.

By default, the `com.ibm.jsse2.sp800-131` property is not set.

The SP800-131a compliance mode is supported for the same types of TADDM discoveries as in case of FIPS mode.

In the SP800-131 mode, TADDM uses the most secure SSL protocol (TLS v1.2) in encrypted communication. Make sure that the following requirements are met.

- When you use Data Management Portal over Web SSL port (HTTPS), you must first configure your web browser to support the TLS v1.2 protocol.
- When you use TADDM SDK and Discovery Management Console in the secure mode, you must enable the TLS v1.2 protocol in your Java Runtime Environment. Additionally, only IBM Java is supported.
- When your SSL certificate does not comply with the SP800-131a standard, it needs to be re-created. For required steps, see "Installing customized SSL certificates for use in TADDM" on page 32.

**Related concepts:**

- "FIPS compliance" on page 21

You can configure TADDM to operate in a mode that uses FIPS-compliant algorithms for encryption by setting the `FIPSMode` property, `com.collation.security.FIPSMode` to true.

### Security for a synchronization server deployment

If you use a synchronization server deployment, you must make security changes when configuring the synchronization server for your environment.
If you are using the TADDM file-based registry and a TADDM domain is added to a synchronization server, you must re-create in the synchronization server any users that already exist in a domain, including assigned roles and access that is granted to access collections. If you are using a Lightweight Directory Access Protocol (LDAP) or WebSphere federated repositories user registry, you must add to the synchronization server the authorization for any users that access TADDM.

When you add a domain to the synchronization server, authentication and authorization for the new domain is delegated to the synchronization server.

Logins to the domain are processed at the synchronization server. In addition, security manager method calls are processed by the synchronization server.

The following list summarizes other security information that you need to know to configure your synchronization server:

- For TADDM to function properly, the Data Management Portal must be running on the synchronization server. A TADDM domain delegates security operations to the Data Management Portal, and this delegation is updated every 2.5 minutes. If 5 minutes pass and this delegation is not updated, the TADDM domain no longer delegates security operations and proceeds as if no synchronization server is present. In this situation, TADDM UIs must be restarted to re-establish the sessions with the synchronization server.

- In each of the following situations, a TADDM UI must be restarted to re-establish sessions with the correct synchronization server:
  - The domain in which the UI is running is added to the Data Management Portal running on a synchronization server.
  - The UI is opened in a domain while that domain is connected to a Data Management Portal, but the synchronization server later becomes unavailable, such as during a restart of the synchronization server or when network problems occur.

- Roles, permissions, and access collections that are stored in the TADDM server are synchronized from the domain to the synchronization server. User to role mappings are not synchronized.

- Roles that you created for the domain can be used by the synchronization server after these objects are synchronized from the domain to the synchronization server.

- Users are not synchronized to the synchronization server.

- A central user registry, such as LDAP or a WebSphere federated repositories registry, is the preferred method of authentication for the synchronization server. Using a central user registry, user passwords are stored in one location.

- Access collections cannot span domains.

- Synchronization works from the domain to the synchronization server. Objects that are created in the synchronization server are not propagated to the domain.

- Create and populate access collections at the domain, and synchronize with the synchronization server.

- Create roles at the domain, and synchronize with the synchronization server.

- Authorize users at the synchronization server to provide access to access collections from multiple domains.

**Security for a streaming server deployment**

If you use a streaming server deployment, authentication and authorization are delegated to the primary storage server.
If you use the TADDM file-based registry, you must create and authorize TADDM users at the primary storage server. If you use a Lightweight Directory Access Protocol (LDAP) or WebSphere federated repositories user registry, you must authorize TADDM users at the primary storage server. The preferred registry type for TADDM authentication is one with a central user registry, such as a LDAP registry or a WebSphere federated repository registry.

Logins to the discovery servers and secondary storage servers are processed at the primary storage server. Therefore, user authentication is performed against the user registry for which the primary storage server is configured. In addition, security manager functions are processed by the primary storage server.

For TADDM to function properly, the primary storage server must be running.

If the primary storage server is stopped or restarted, a TADDM user interface must be restarted to re-establish sessions with the primary storage server.

**Configuring for LDAP**

You can configure an external LDAP server for user authentication.

**Before you begin**

If you want to authenticate to an LDAP user registry, configure an LDAP V2 or V3 registry.

**About this task**

When using LDAP and/or VMM the LDAP users and/or groups are always stored in LDAP/VMM and do not need to be created in TADDM. TADDM is used only to assign roles to the LDAP users and groups. Only these user/group to role mappings, known as permissions, need to be created and stored in TADDM. The administrator user ID is a special internal TADDM user that is always processed using file-based security regardless of what user registry is configured. This user can always be used to initially assign roles to the LDAP users and groups.

**Procedure**

To use LDAP or VMM for user authentication, complete the following steps:

1. Configure TADDM to use the LDAP registry by configuring the appropriate properties in the collation.properties file.
2. Log into the Data Management Portal using the TADDM administrator user ID.
3. Complete one of the following steps:
   - In the Users pane, use the **Search Users** field to search the LDAP registry for the appropriate user.
   - In the User Groups pane, use the **Search Groups** field to search the LDAP registry for the appropriate user group.

   **Note:** The search results list the users or group names returned by the LDAP registry search. It is not a means to create users, or copy users from LDAP into TADDM. The purpose of the list is to display what TADDM permissions need to be created for the users.
4. After the user (or group) is listed, assign the required TADDM roles to them. Only these permissions, and not the LDAP users (or groups), are stored in TADDM.
What to do next

To configure SSL for LDAP, complete the following steps:

1. In the collation.properties file, locate the following property, and change the value of the property from false to true:
   `com.collation.security.auth.ldapUseSSL`

2. Configure the following truststore and keystore properties, as appropriate:
   `com.collation.security.auth.ldapClientKeyStore`
   `com.collation.security.auth.ldapClientKeyStorePassphrase`
   `com.collation.security.auth.ldapClientTrustStore`
   `com.collation.security.auth.ldapClientTrustStorePassphrase`

3. If necessary, change the port on which the LDAP server is listening for SSL connections by configuring the following property:
   `com.collation.security.auth.ldapPortNumber`

Configuring for WebSphere federated repositories

If you have a Tivoli WebSphere application configured for a central user registry that uses WebSphere federated repositories, you can configure for WebSphere federated repositories in a federated repositories registry.

Configuring the TADDM server to use WebSphere federated repositories

WebSphere federated repositories is a flexible meta-repository within WebSphere that supports multiple types of user registries, including Microsoft Active Directory.

Before you begin

You must configure TADDM to use WebSphere federated repositories if you use other Tivoli products in your environment, and you require single sign-on between TADDM and any of the following products:

- IBM Tivoli Change and Configuration Management Database (CCMDB) or IBM SmartCloud Control Desk (SCCD)
- IBM Tivoli Business Service Manager

TADDM requires additional services not present in a standard WebSphere distribution, so when you configure TADDM for federated repositories, you must use one of the following WebSphere installations:

- WebSphere Application Server Network Deployment, as installed with CCMDB or SCCD
- WebSphere Application Server, as installed with IBM Tivoli Business Service Manager

To see supported versions of the products, go to the “Supported versions” on page 183 section.

Before beginning this procedure, you must already have configured the WebSphere federated repositories authentication service on a WebSphere Application Server Network Deployment server. For more information, refer to the IBM Tivoli Change and Configuration Management Database (CCMDB) documentation or IBM SmartCloud Control Desk (SCCD) documentation.
About this task

This configuration enables single sign-on between Tivoli applications using WebSphere Lightweight Third-Party Authentication (LTPA) tokens. For example, configuring TADDM to use the same WebSphere federated repositories used by CCMDB or SCCD supports single sign-on for launch in context between IBM Tivoli CCMDB or IBM SCCD and TADDM.

To automatically configure TADDM to use WebSphere federated repositories, install TADDM and select WebSphere Federated Repositories as your user registry during installation.

This configuration is supported on all TADDM server types, on all deployments.

Procedure

To perform the configuration manually, complete the following steps:

1. Stop the TADDM server.
2. Specify the user management module used by this TADDM server. The following values are valid:
   - file: This value is used for a file-based user registry. (This is the default value.)
   - ldap: This value is used for an LDAP user registry.
   - vmm: This value is used for a user registry that uses the federated repositories of WebSphere Application Server.

   For example, in the $COLLATION_HOME/etc/collation.properties file:
   ```
   com.collation.security.usermanagementmodule=vmm
   ```

3. Specify the WebSphere host name and port in the collation.properties file. For example:

   ```
   com.collation.security.auth.websphereHost=localhost
   com.collation.security.auth.webspherePort=2809
   ```

   When specifying the WebSphere port in the collations.properties file, use the following property: com.collation.security.auth.webspherePort. The WebSphere port should be the bootstrap port for the WebSphere server. For WebSphere Application Server and the embedded version of WebSphere Application Server, the default port is 2809. For WebSphere Application Server Network Deployment, which IBM Tivoli CCMDB or IBM SCCD uses, the default port is 9809.

4. Specify the WebSphere administrator user name and password in the collation.properties file. For example:

   ```
   com.collation.security.auth.VMMAdminUsername=administrator
   com.collation.security.auth.VMMAdminPassword=password
   ```

5. Make the following change to the authentication services configuration file:

   - For the Linux, AIX, and Linux on System z operating systems, the file is located in the following path: $COLLATION_HOME/etc/ibmessclientauthncfg.properties.
   - For the Windows operating systems, the file is located in the following path: %COLLATION_HOME%\etc\ibmessclientauthncfg.properties.
In the authnServiceURL property, substitute the fully qualified domain name of the system your WebSphere instance is installed on and the HTTP port of the WebSphere instance.

# This is the URL for the Authentication Service
authnServiceURL=http://localhost:9080/TokenService/services/Trust

6. Copy the WebSphere orb.properties and iwsorbutil.jar files into the JRE used by your TADDM installation. For example in a TADDM Linux installation, do the following:

7. Specify the WebSphere host name and port in the sas.client.props file:
   - For the Linux, AIX, and Linux on System z operating systems, file is located in the following path: $COLLATION_HOME/etc/sas.client.props.
   - For the Windows operating systems, file is located in the following path: %COLLATION_HOME%\etc\sas.client.props, for example:
     com.ibm.CORBA.securityServerHost=host1.austin.ibm.com
     com.ibm.CORBA.securityServerPort=2809

   Note: For WebSphere Application Server and the embedded version of WebSphere Application Server, the default port is 2809. For WebSphere Application Server Network Deployment, which IBM Tivoli CCMDB or IBM SCCD uses, the default port is 9809.

8. Specify the WebSphere administrator user name and password in the sas.client.props file. For example:
   # RMI/IIOP user identity
   com.ibm.CORBA.loginUserid=administrator
   com.ibm.CORBA.loginPassword=password

9. Optional: To encrypt the login password in the sas.client.props file, complete the following steps:
   a. Copy the sas.client.props file back to the TADDM server, in the $COLLATION_HOME/etc directory.
   b. Encrypt the password as follows, depending on which operating system you have installed WebSphere.
      - For Linux, AIX, and Linux on System z operating systems:
        Use the PropFilePasswordEncoder.sh command.
      - For Windows operating systems:
        Use PropFilePasswordEncoder.bat. For example,
        C:\WebSphere\profiles\AppSrv01\bin\PropFilePasswordEncoder C:\temp\sas .client.props com.ibm.CORBA.loginPassword
   c. Copy the sas.client.props file back to the TADDM server, in the etc directory.

10. Start the TADDM server.

What to do next

After the installation is finished, you can use the default administrator user defined in the local TADDM file-based repository to configure additional TADDM users, including TADDM administrators. These additional TADDM users are authenticated using WebSphere federated repositories.
There are security configurations for Tivoli CCMDB or IBM SCCD that allow groups and group memberships to be created and maintained in the Maximo® user and group applications.

When Tivoli CCMDB or IBM SCCD is configured for this, TADDM uses its own, separate repository from Tivoli CCMDB or IBM SCCD. Users must be created in both Tivoli CCMDB or IBM SCCD/Maximo and TADDM.

TADDM can be configured to use user and group definitions in external user registries through WebSphere federated repositories. However, TADDM cannot use user and group definitions that are stored in Tivoli CCMDB because these are not supported by WebSphere federated repositories.

**Updating authentication service LTPA keys**

If you are using single sign-on with WebSphere federated repositories, you must keep the authentication service Lightweight Third-Party Authentication (LTPA) keys synchronized with those used by WebSphere federated repositories.

**Procedure**

If the LTPA keys used by WebSphere federated repositories are changed, use this process to resynchronize the keys used by the authentication service:

1. Export the new WebSphere LTPA keys:
   a. In the WebSphere administrative console, navigate to Secure administration, applications, and infrastructure > Authentication mechanisms and expiration.
   b. For Cross-cell single sign-on, specify a file name and password for the file to contain the exported LTPA keys.
2. At a command prompt, navigate to the bin directory of the appropriate WebSphere profile.
3. Run the following WebSphere wsadmin command:
   ```bash
   wsadmin> $AdminTask importESSLTPAKeys {-pathname pathname -password password}
   ```
   where pathname and password are the values you specified for the file name and password when exporting the LTPA keys.
4. Restart the WebSphere server.

**Securing the authentication channel**

When you configure TADDM to use WebSphere federated repositories, you can secure communications between the authentication client and the authentication service.

**About this task**

TADDM uses an authentication service that supports single sign-on. The authentication service is installed during the installation of IBM Tivoli Change and Configuration Management Database (IBM SmartCloud Control Desk (SCCD)) or IBM Tivoli Business Service Manager.

To see supported versions of the products, go to the “Supported versions” on page section.

There are two mechanisms by which you can secure communications between an authentication client and an authentication service:
• SSL
• Client authentication

**Configuring the authentication channel for SSL:**

You can secure communications by using the WebSphere signer certificates to configure SSL between the authentication client and the authentication server.

**Procedure**

To configure for SSL between the authentication client and the authentication server, complete the following steps:

1. Do one of the following:
   a. If you are using the WebSphere instance installed by Tivoli Integrated Portal, navigate to **SSL certificate and key mgmt > Manage endpoint security configurations > Node1 > Key stores and certificates > NodeDefaultTrustStore > Signer certificates**.
   b. If you are using the WebSphere instance installed by Tivoli Change and Configuration Management Database (CCMDB) or IBM SmartCloud Control Desk, navigate to **SSL certificate and key mgmt > Manage endpoint security configurations > ctgNode01 > Key stores and certificates > NodeDefaultTrustStore > Signer certificates**.

2. Export the WebSphere signer certificates to files (for example, export dummyclientsigner to signer1.cert and dummyserverSigner to signer2.cert). If you are unsure about which certificates to export, you must export all of the signer certificates.

3. Copy the .cert files to the TADDM server. Create a truststore and import the WebSphere signer certificates as follows:
   ```bash
   $COLLATION_HOME/external/jdk-Linux-i686/jre/bin/keytool -genkey -alias truststore -keystore truststore.jks
   $COLLATION_HOME/external/jdk-Linux-i686/jre/bin/keytool -import -trustcacerts -alias default -file signer1.cert -keystore truststore.jks
   $COLLATION_HOME/external/jdk-Linux-i686/jre/bin/keytool -import -trustcacerts -alias dummyserverSigner -file signer2.cert -keystore truststore.jks
   ```

4. Include the truststore password and location in the $COLLATION_HOME/etc/collation.properties entries:
   ```properties
   com.collation.security.auth.ESSClientTrustStore=/opt/IBM/taddm/dist/etc/truststore.jks
   com.collation.security.auth.ESSClientTrustPwd=password
   ```

5. Update the Tivoli Authentication Service URL in the ibmessclientauthncfg.properties file to use https and port 9443. Ensure that the WebSphere host name is correct, substituting it for localhost, and that the non-https entry is commented out.
   ```properties
   authnServiceURL=http://localhost:9080/TokenService/services/Trust
   authnServiceURL=https://localhost:9443/TokenService/services/Trust
   ```

**Configuring client authentication:**

To configure client authentication between the authentication client and the authentication server, it is recommended that you enable WebSphere application security.

**Before you begin**

After WebSphere application security is enabled, you can add the role called TrustClientRole to the WebSphere administrator user that you specified during the TADDM installation. This method provides added security for the authentication

Administration
service by restricting the users that can authenticate to the authentication service to only those users with the TrustClientRole.

**Procedure**

To add the TrustClientRole to the WebSphere administrator specified during TADDM installation, complete the following steps:

1. Log in to the WebSphere Administration Console.
2. Under the **Security** tab, click **Enterprise Applications**. The Enterprise Applications pane is displayed.
3. In the Enterprise Applications table, click the Authentication Service application (authsvc_ctges) in the Name column. The Enterprise Applications > authsvc_ctges pane is displayed.
4. In the Enterprise Applications > authsvc_ctges pane, in the Detailed Properties list, click **Security role to user/group mapping**. The Enterprise Applications > authsvc_ctges > Security role to user/group mapping pane is displayed.
5. In the table on the Enterprise Applications > authsvc_ctges > Security role to user/group mapping pane, complete the following steps:
   - In the table, select the check box next to TrustClientRole.
   - Clear the **Everyone** check box.
   - Click **Lookup Users** or **Lookup Groups**. The Enterprise Applications > authsvc_ctges > Security role to user/group mapping > Lookup users or groups pane is displayed.
   - In the Enterprise Applications > authsvc_ctges > Security role to user/group mapping > Lookup users or groups pane, complete the following steps:
     - Search for users or groups, by using the Limit and Search string input boxes. When a group or user is found, it is displayed in the Available list.
     - From the Available list, select the user or group that you want.
     - Click **Move** to add that user or group to the **Selected** list.
   - Click **OK**. The Enterprise Applications > authsvc_ctges > Security role to user/group mapping pane is displayed.
   - In the Enterprise Applications > authsvc_ctges > Security role to user/group mapping pane, clear the **Everyone** check box.
   - Click **OK**. The Enterprise Applications > authsvc_ctges pane is displayed.
   - Click **Save** to save the configuration. The Enterprise Applications pane is displayed.
   - Click **OK**. The Enterprise Applications > authsvc_ctges pane is displayed.

**Configuring for Microsoft Active Directory**

You can use Microsoft Active Directory as the authentication method for TADDM using LDAP or using WebSphere federated repositories as an intermediary. If you require single sign-on to TADDM, you should use WebSphere federated repositories.

**About this task**

You can use the users defined in an Active Directory registry, without defining new users, by configuring TADDM to use Active Directory. You can configure TADDM to use Active Directory as an LDAP registry, or you can configure
TADDM to use WebSphere federated repositories and then configure WebSphere federated repositories for Active Directory.

When you configure for Active Directory during the TADDM installation, you can configure TADDM to use any user from Active Directory as your TADDM administrator. The administrator is allowed to configure access to TADDM and grant other users access to TADDM objects and services.

This configuration is supported on all TADDM server types, on all deployments.

**Procedure**

To configure for Microsoft Active Directory, complete the following steps:

Do one of the following:

- To configure Microsoft Active Directory using LDAP:
  1. Configure TADDM for LDAP. For more information on configuring TADDM for LDAP, see "Configuring for LDAP" on page 24.
  2. Ensure that when you are using Active Directory, you set `com.collation.security.auth.ldapFollowReferrals` to `true` in the `collation.properties` file.

- To configure Microsoft Active Directory using WebSphere federated repositories:
  1. Configure TADDM for WebSphere federated repositories. For more information on configuring TADDM for WebSphere federated repositories, see "Configuring the TADDM server to use WebSphere federated repositories" on page 25.
  2. Configure WebSphere federated repositories for Microsoft Active Directory. For more information on configuring supported entity types in a federated repository configuration, see the section called **Configuring supported entity types in a federated repository configuration** in the [WebSphere Application Server Information Center](http://www-01.ibm.com/support/knowledgecenter/SSAW57_6.1.0/com.ibm.websphere.nd.doc/info/ae/aeha_twim_entitytypes.html).

**Securing TADDM web services**

You can configure TADDM to disable the HTTP port by setting the `com.ibm.cdb.secure.tomcat` property (TADDM 7.3.0) or the `com.ibm.cdb.secure.liberty` property (TADDM 7.3.0.1, and later) in `collation.properties` to `true`. Additionally you can set more secure SSL protocol by using the `com.ibm.cdb.http.ssl.protocol` flag.

The default value of `com.ibm.cdb.secure.tomcat` and `com.ibm.cdb.secure.liberty` properties is `false`. When the HTTP port is disabled, TADDM can be accessed only by the HTTPS port, for example `https://example.com:9431`.

**Limitation:** When you have TADDM installed in the streaming server deployment, and your discovery servers and secondary storage servers are up and running, you can set the `com.ibm.cdb.secure.tomcat` or `com.ibm.cdb.secure.liberty` property to `true`. In such case, the HTTP port is disabled and you can use TADDM in the secure mode. However, if you want to add a new discovery server or secondary storage server to your deployment, you must temporarily enable the HTTP port, because the TADDM installer does not support the HTTPS protocol. To temporarily disable the secure mode, complete the following steps:
1. Change the value of the com.ibm.cdb.secure.tomcat or com.ibm.cdb.secure.liberty property to false.
2. Restart the TADDM server.
3. Install a new discovery server or secondary storage server.
4. Change the value of the com.ibm.cdb.secure.tomcat or com.ibm.cdb.secure.liberty property to true.
5. Restart the TADDM server.

The default value of the com.ibm.cdb.http.ssl.protocol property is TLS. The secure values are TLS, TLSv1.1, and TLSv1.2. If you want to use the most secure protocols TLSv1.1 and TLSv1.2, you must first configure your web browser to support them.

**Installing customized SSL certificates for use in TADDM**

You can install your own customized SSL certificates and use them with TADDM.

**Procedure**

1. Create a backup copy of the following keystore files:
   - $COLLATION_HOME/etc/serverkeys
   - $COLLATION_HOME/etc/jssecacerts.cert
2. Go to the $COLLATION_HOME/etc directory, open command line and enter the keytool and TADDM sslpassphrase parameters with the values in the following manner:
   - **Linux operating system:**
     - `keytool=./external/jdk-Linux-x86_64/bin/keytool`  
     - `pass=XXXXXXXX30374`
   - **Windows operating system:**
     - `set keytool=./external/jdk-Windows-i386-64\bin\keytool.exe`  
     - `set pass=XXXXXXXX30374`

   The value of the pass parameter is the value of the com.collation.sslpassphrase property that is specified in the collation.properties file.
3. Remove self-signed certificate and key from TADDM by running the following commands:
   - **Linux operating system:**
     - `$keytool -delete -alias collation -noprompt -keystore jssecacerts.cert -storepass $pass`
     - `$keytool -delete -alias collation -noprompt -keystore serverkeys -storepass $pass`
   - **Windows operating system:**
     - `%keytool% -delete -alias collation -noprompt -keystore jssecacerts.cert -storepass %pass%`
     - `%keytool% -delete -alias collation -noprompt -keystore serverkeys -storepass %pass%`
4. Generate SSL key with the required CN, validity, algorithm, and other parameters, and save it to the serverkeys file. For example, you can run the following command:
   - **Linux operating system:**
     - `$keytool -genkey -alias collation -keystore serverkeys -validity 3650 -keyAlg RSA -sigAlg SHA256WithRSA -keypass $pass -storepass $pass -dname "CN=John Public, OU=Engineering, OU=NA, o=Company, L=Manhattan, S=New York, c=US"`
- Windows operating system:
  
  ```bash
  %keytool% -genkey -alias collation -keystore serverkeys -validity 3650
  -keyAlg RSA -sigalg SHA256WithRSA
  -keypass %pass% -storepass %pass%
  -dname "CN=John Public, O=Company, L=Manhattan, S=New York, C=US"
  
  5. Create another backup copy of the serverkeys file, where you saved the generated SSL key.
  
  6. Generate the certificate signing request (CSR file) by running the following command:
     - Linux operating system:
       ```bash
       $keytool -certreq -alias collation -storepass $pass
       /tmp/certreq.csr -keystore serverkeys
       
       - Windows operating system:
       ```bash
       %keytool% -certreq -alias collation -storepass %pass%
       -file C:\temp\certreq.csr -keystore serverkeys
     
  7. Use the CSR file to get the SSL certificate from official certificate authority. Save the certificate on your TADDM server, for example in the tmp directory on Linux operating system, or in the C:\temp directory on Windows operating system.

  Note: There are two types of certificates: 'Individual certificate' and 'Full chain of certificate'.
  
  8. To import the received certificate ('Individual certificate' or 'Full chain of certificate') to both serverkeys and jssecacerts.cert files on TADDM, run the below commands:

  Important: For the -file parameter, specify the path to the file where you saved the SSL certificate in the previous step, for example: /tmp/cert.crt on Linux operating system.

  Individual certificate
     - Linux operating system:
       ```bash
       $keytool -import -trustcacerts -alias root -noprompt
       -keystore serverkeys -storepass $pass -keypass $pass
       /tmp/CAcert.cer
       $keytool -import -trustcacerts -alias intermediate
       -keystore serverkeys -storepass $pass -keypass $pass
       /tmp/IntermediateCAcert.cer
       $keytool -import -trustcacerts -alias server
       -keystore serverkeys -storepass $pass -keypass $pass
       /tmp/serverCAcert.cer
       
       $keytool -import -trustcacerts -alias root -noprompt
       -keystore jssecacerts.cert -storepass $pass -keypass $pass
       /tmp/CAcert.cer
       $keytool -import -trustcacerts -alias intermediate
       -keystore jssecacerts.cert -storepass $pass -keypass $pass
       /tmp/IntermediateCAcert.cer
       $keytool -import -trustcacerts -alias server
       -keystore jssecacerts.cert -storepass $pass -keypass $pass
       /tmp/serverCAcert.cer
     
     - Windows operating system:
       ```bash
       %keytool% -import -trustcacerts -alias root -noprompt
       -keystore serverkeys -storepass %pass% -keypass %pass%
       /tmp\CAcert.cer
       %keytool% -import -trustcacerts -alias intermediate
       -keystore serverkeys -storepass %pass% -keypass %pass%
       /tmp\IntermediateCAcert.cer
       %keytool% -import -trustcacerts -alias server
       -keystore serverkeys -storepass %pass% -keypass %pass%
       /tmp\serverCAcert.cer
%keytool% -import -trustcacerts -alias root -noprompt -keystore jssecacerts.cert -storepass %pass% -keypass %pass% -file C:\temp\CAcert.cer
%keytool% -import -trustcacerts -alias intermediate -noprompt -keystore jssecacerts.cert -storepass %pass% -keypass %pass% -file C:\temp\IntermediateCAcert.cer
%keytool% -import -trustcacerts -alias server -noprompt -keystore jssecacerts.cert -storepass %pass% -keypass %pass% -file C:\temp\serverCAcert.cer

Full chain of certificates
- Linux operating system:
  $keytool --import --trustcacerts --alias collection --noprompt --keystore serverkeys --storepass $pass --keypass $pass --file /tmp/cert_chain.crt
  $keytool --import --trustcacerts --alias collection --noprompt --keystore jssecacerts.cert --storepass $pass --keypass $pass --file /tmp/cert_chain.crt
- Windows operating system:
  %keytool% -import -trustcacerts -alias collection -noprompt -keystore serverkeys -storepass %pass% -keypass %pass% -file C:\temp\cert_chain.crt
  %keytool% -import -trustcacerts -alias collection -noprompt -keystore jssecacerts.cert -storepass %pass% -keypass %pass% -file C:\temp\cert_chain.crt

9. Restart the TADDM server.

What to do next

Keep the backup copies of the serverkeys file that you generated in step 4, and the file where you saved the SSL certificate in step 7. If you must replace or renew the certificate, these files are necessary. To replace, or renew the certificate, complete the following steps:
1. Repeat the steps 2 and 3.
2. Restore the serverkeys file.
3. Repeat the steps 8 and 9.

Managing the TADDM servers

Before you configure TADDM for discovery, you must understand how to manage the TADDM servers, which includes many tasks.

Checking TADDM server status

You can use the Administrator Console or the control command to check the status of the TADDM server.

Using the Administrator Console to check status

To use the Administrator Console to check the status, open a web browser, and enter the URL and port number of the system where you installed the TADDM server. The following URL is an example:

http://system.company.com:9430

The Administrator Console is then shown, and it lists the components of the TADDM server and their status.
Using the control command to check status

To use the control command to check the status, complete the following steps:

1. Log in as the non-root user that was defined during the installation process.
2. From a command prompt, go to the directory where you installed the TADDM server.
3. Run one of the following commands:
   • For AIX, Linux, and Linux on System z operating systems:
     $COLLATION_HOME/bin/control status
   • For Windows operating systems:
     %COLLATION_HOME%\bin\control.bat status

The following output is shown, depending on the deployment that you have and on the type of server that TADDM is running on in the respective deployment:

synchronization server deployment

synchronization server

• TADDM 7.3.0:
  DbInit: Started
  Tomcat: Started
  EcmdbCore: Started

  TADDM: Running

• TADDM 7.3.0.1, and later:
  DbInit: Started
  Liberty: Started
  EcmdbCore: Started

  TADDM: Running

domain server

• TADDM 7.3.0:
  Discover: Started
  DbInit: Started
  Tomcat: Started
  Topology: Started
  DiscoverAdmin: Started
  Proxy: Started
  EventsCore: Started

  TADDM: Running

• TADDM 7.3.0.1, and later:
  Discover: Started
  DbInit: Started
  Liberty: Started
  Topology: Started
  DiscoverAdmin: Started
  Proxy: Started
  EventsCore: Started

  TADDM: Running

streaming server deployment

storage server

• TADDM 7.3.0:
Starting the TADDM server

If you chose the Start at Boot option at installation, the TADDM server automatically starts during every system boot.

About this task

Important: A local or remote database server must be started and running before the TADDM server is started. The TADDM server cannot initialize or run properly if the database is not available.

Procedure

To manually start the TADDM server, complete the following steps:

1. Log in as the non-root user that was defined during the installation process.
2. Open a command prompt window.

   Note: On a Windows Server 2008 system with User Account Control turned on, open the command prompt window with administrator privileges. You can do this by right-clicking on the Command Prompt icon and then clicking Run as administrator.

3. Go to the directory where you installed the TADDM server.
4. Use one of the following commands to run the start script:
   - For Linux, AIX, and Linux on System z operating systems:
     $COLLATION_HOME/bin/control start
   - For Windows operating systems:
When starting the server on a Windows system, you might see the following timeout error message: Error 1053: The service did not respond to the start or control request in a timely fashion. This error occurs because the TADDM server can take longer than the allowed time to start. You can disregard this message; the startup process continues until it completes.

If you installed the TADDM server with root privileges, you can manually start the TADDM server by running the following script:

/etc/init.d/collation start

**Stopping the TADDM server**

You can manually stop the TADDM server and related discovery processes.

**Procedure**

To manually stop the TADDM server, complete the following steps:

1. Log in as the non-root user that was defined during the installation process.
2. Open a command prompt window.

   **Note:** On a Windows Server 2008 system with User Account Control turned on, open the command prompt window with administrator privileges. You can do this by right-clicking on the Command Prompt icon and then clicking **Run as administrator**.

3. Go to the directory where you installed the TADDM server.
4. Use one of the following commands to run the stop script:
   - For Linux, AIX, and Linux on System z operating systems:
     
     $COLLATION_HOME/bin/control stop

   - For Windows operating systems:
     
     %COLLATION_HOME%/bin/stopServer.bat

   If you installed the TADDM server with root privileges, you can manually stop the TADDM server by running the following script:

   /etc/init.d/collation stop

**What to do next**

Some sensors run in their own special Java Virtual Machine (JVM). When running a discovery, if you use the control script (./control stop) to stop TADDM, you might need to manually stop these additional JVMs, which are called local anchors. If you do not stop the local anchors, unexpected behavior can result. For example, there might be degraded performance of certain discoveries.

To verify that the process for the local anchor is no longer running, enter the following command:

```bash
% ps -ef | grep -i anchor
```

This command identifies any local anchor processes that are running. The output looks like the following code example:

```
coll 23751 0.0 0.0 6136 428 ? S Jun02 0:00 /bin/sh
local-anchor.sh 8494 <more information here>
```

If a process is running, stop the process by running the following command:

```
% kill -9 23751
```
After running the command, verify that the process stopped by running the following command:

```
% ps -ef |grep -i anchor
```

### Backing up data

Back up your data on a regular basis so you can recover from a system failure.

#### Before you begin

Before you back up data, stop the TADDM server.

#### Procedure

To back up files for the TADDM server, complete the following tasks:

1. Save all the files in the directory where you installed the TADDM server.
   - For Linux, AIX, and Linux on System z operating systems, the default path to the directory is `/opt/IBM`.
   - For Windows operating systems, the default path to the directory is `C:\opt\IBM`.

#### What to do next

To backup the database files, use the documentation provided by the database vendor.

### Restoring data

After a system failure, you can restore the configuration, data, and database files. As a result, you can resume operation from the point of the last backup prior to the failure.

#### Procedure

To restore data from backup media, complete the following steps:

1. Do one of the following:
   - Restore the `/opt/IBM` directory, and restart TADDM.
   - Restore the `C:\opt\IBM` directory, and restart TADDM.
2. Locate the backup copy of the data files.
3. Open a command prompt window.
4. Navigate to the directory where you installed the TADDM server.
5. Copy the backup copy of the data files to the installation directory.
6. Close the command prompt window.
7. Start the TADDM server.

#### What to do next

If the database is affected by the system failure, restore the database files using the documentation from the database vendor.
Copying discovery scopes, profiles, and custom server templates between TADDM servers

You can use the `datamover.sh|bat` command to copy discovery scopes, discovery profiles, and custom server templates between TADDM servers.

You can export discovery scopes, profiles, and custom server templates (all entities) or specify which entity to export from a server. Then on the destination server you can import the entity or entities.

**Restriction:** To maintain data integrity, you must move the data between the same versions of TADDM servers.

To copy the entities between TADDM servers, complete the following steps:

1. Run the following command on the source server to export the required entity or entities to a file:
   ```sh
   datamover.sh|bat -u user -p password -a action [-t type ] [-f filename]
   ```
   where:
   - **user**
     The TADDM user name.
   - **password**
     The TADDM user password.
   - **action**
     Specify one of the following actions: import, export, or help.
   - **Optional: type**
     Specify one of the following actions: all, scope, profile, template. The default value is all.
   - **Optional: filename**
     Specify a file name. The default value is `datamover.xml`.

   Default discovery profiles are not exported, all custom server templates, user created profiles, and scopes can be exported.

   After running the command, the information about exported entities is displayed. For example, if the output file is `exporthost.xml`, the following information is provided:
   ```
   Exported 6 scopes
   Exported 1 profiles
   Exported 57 templates
   ```

2. Copy the file or files to the destination server and run the `datamover.sh|bat` and import the entity or entities.
   The following rules are applied when importing entities:
   - If a scope or profile with the same name exists on the server then the imported scope or profile is renamed. The file is renamed to `name_TADDM`.
   - If a template exists with the same name on the server then the template is merged to the existing template.

**Deploying the Discovery Management Console**

After you confirm that the TADDM server is available, you can deploy the Discovery Management Console.
**Procedure**

To deploy the Discovery Management Console, complete the following steps:

1. Provide users with the URL (including the port number) of the system where you installed the TADDM server.
   
   For example, you can provide users with something similar to the following URL:
   
   `http://system.company.com:9430`

2. Provide users with their user name and password.

3. Specify whether users should use Secure Sockets Layer (SSL).
   
   In cases where SSL is being used, instruct users to save a truststore for the TADDM server by following the instructions on the Discovery Management Console Installation and Start page. For more information, see the TADDM Installation Guide.

   **Important:** You should use SSL for all communication between the Discovery Management Console and the TADDM server.

4. Users need to have a supported Java runtime environment installed on the system used to view the Discovery Management Console. For more information about client prerequisites, see the TADDM Installation Guide.

5. Refer users to the TADDM User’s Guide for information about how to start the Discovery Management Console.

**Configuring TADDM communication**

To establish TADDM communication, you must configure all necessary services, connections, and firewalls.

**TADDM services**

TADDM connectivity can be divided into three areas:

**Public connectivity**

Public connectivity covers network connectivity that is done from outside of the TADDM infrastructure. For example, Data Management Portal, Discovery Management Console, or API clients, which connect to the TADDM server. It is the highest level of connectivity.

**Inter-server connectivity**

Inter-server connectivity covers network connectivity between elements of the TADDM core infrastructure, that is, discovery servers and storage servers. It is the middle level of connectivity.

**Local connectivity**

Local connectivity covers network connectivity between local services on one machine. It is the lowest level of connectivity.

You can configure connectivity for each service during the installation phase, or later by changing configuration properties in the `collation.properties` configuration file.

**Services default interface**

To configure services default listen interface, change the `com.ibm.cdb.global.hostname` property in the `collation.properties` file.
Table 2. Services default interface settings

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Default interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global services host</td>
<td>com.ibm.cdb.global.hostname</td>
<td>0.0.0.0</td>
</tr>
</tbody>
</table>

Listen interface that depends on the type of communication

To configure listening interfaces separately for services for each connectivity area, change the appropriate property in the collation.properties file.

Table 3. Services default interface settings

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Default interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public connectivity services host</td>
<td>com.ibm.cdb.public.hostname</td>
<td>Defined by com.ibm.cdb.global.hostname</td>
</tr>
<tr>
<td>Inter-server connectivity services host</td>
<td>com.ibm.cdb.interserver.hostname</td>
<td>Defined by com.ibm.cdb.global.hostname</td>
</tr>
<tr>
<td>Local connectivity services host</td>
<td>com.ibm.cdb.local.hostname</td>
<td>127.0.0.1</td>
</tr>
</tbody>
</table>

Note: If no interface is specified or an interface has the 0.0.0.0 value, one local external network interface must be open to communication with itself. If an interface is specified, it must be open to communication with itself.

Listen interface for specific services

You can configure separate TCP port for each service during the installation phase, or later by changing respective property in the collation.properties file.

Service interface configuration

To configure a specific listen interface for each service, change the appropriate property with the host suffix in the collation.properties file.

Example for the TopologyManager service:
com.ibm.cdb.service.TopologyManager.host=192.168.1.5

Note: This naming convention does not apply to public or inter-server service registries.

Service port configuration

To configure a specific listen port for each service, change the appropriate property with the port suffix in the collation.properties file.

The following example is for the TopologyManager service:
com.ibm.cdb.service.TopologyManager.port=9550

SSL service configuration

To configure a specific listen interface or port for each SSL service, change the appropriate property with the secure infix in the collation.properties file.

The following example is for the SecureApiServer service:
- com.ibm.cdb.service.SecureApiServer.secure.host=192.168.1.5
• com.ibm.cdb.service.SecureApiServer.secure.port=9531

Web portal interface (HTTP and HTTPS) configuration

To configure a listen interface for a web portal (HTTP and HTTPS), change the com.ibm.cdb.service.web.host property in the collation.properties file.

Note: The HTTP and HTTPS host is configured by changing one property in contrast to other services.

Database connections

To configure a specific database connection, change the com.collation.db.port and com.collation.db.server properties in the collation.properties file.

For example:
• com.collation.db.port=65432
• com.collation.db.server=9.156.47.156

DNS connections

If you want to use fully qualified domain names (FQDN) for communication, ensure that the host that participates in communication can resolve the FQDN from DNS service.

Sensor connections

Configuration of the ports that are used by the ping sensor and the port sensor to make connections is included in the documentation of the ping sensor and the port sensor. Ensure that the ports to the services that you want to discover are opened.

Table 4. Ping Sensor and Port Sensor default ports

<table>
<thead>
<tr>
<th>Port name</th>
<th>Default port</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSH</td>
<td>22</td>
<td>TCP</td>
</tr>
<tr>
<td>Telnet</td>
<td>23</td>
<td>TCP</td>
</tr>
<tr>
<td>DNS</td>
<td>53</td>
<td>TCP</td>
</tr>
<tr>
<td>WMI</td>
<td>135</td>
<td>TCP</td>
</tr>
<tr>
<td>PowerShell</td>
<td>5985, 5986</td>
<td>TCP</td>
</tr>
<tr>
<td>LDAP</td>
<td>389</td>
<td>TCP</td>
</tr>
<tr>
<td>SMB</td>
<td>445</td>
<td>TCP</td>
</tr>
<tr>
<td>Oracle</td>
<td>1521</td>
<td>TCP</td>
</tr>
<tr>
<td>CiscoWorks</td>
<td>1741</td>
<td>TCP</td>
</tr>
</tbody>
</table>

Anchor connections

TADDM can connect to an anchor server by using one of the following connection types: ssh or direct. To configure a specific anchor connection type, change the value of the com.collation.discover.anchor.connectType property in the collation.properties file to either ssh and direct.
To configure a specific anchor connection type for a particular address, change the `com.collation.discover.anchor.connectType` property with the IP address as a suffix, in the `collation.properties` file, for example:

```
com.collation.discover.anchor.connectType.1.2.3.4=direct
```

Additionally, port 8497 is defined as the default port for connection to an anchor server. You can configure this port by using the Discovery Management Console.

- In `ssh` mode, open ports for SSH communication on a public interface that is accessed from the TADDM server and from the anchor connection port on a loopback interface on the machine that is hosting the anchor server.
- In `direct` mode, open ports for SSH communication and anchor connection on a public interface that is accessed from the TADDM server.

**Gateway connections**

TADDM can connect to a gateway server using a SSH connection.

On the gateway, the host SSH port must be open for communication on a public interface that is accessed from the TADDM server.

**Resolving a server's host name to a fully qualified domain name**

To ensure successful communication between servers, the host server must be able to resolve its host name to a fully qualified domain name (FQDN) by using the operating system’s resolver library. One of the following conditions must be fulfilled:

- In the search order of the operating system’s host resolution, DNS must precede local files. To configure this setting, refer to the operating system documentation.
- In the host file, the TADDM server’s FQDN must precede the short name.

If neither of these conditions can be fulfilled, you can set the `com.collation.serverID` property in the `collation.properties` file to the IP or host name of the TADDM server. Also, ensure that the `ServerID` in Synchronization Server / Enterprise Server > Data Management Portal > Domain Management > Domain Host Name is set to the same value.

**Ephemeral ports**

TADDM communication includes the usage of ephemeral ports. These ports are temporary and are specific to an operating system. Each operating system has a defined range of port numbers from which specific ports are chosen randomly. TADDM does not define these ports. For information about the range of ports, necessary configuration and more details, see documentation of the operating system that you use.

**Configuring firewalls**

To establish TADDM communication, you must configure the necessary firewalls. The details of this task vary depending on whether you have configured a domain server deployment, streaming server deployment or synchronization server deployment.

The firewall configuration information is presented in tables. Each table includes the direction of the communication. On the target machine, the target service port must be open on the firewall as the source of the outgoing connections and as the destination of the incoming connections. On the source machine, the target service
port must be open on the firewall as the destination of the outgoing connections and as the source of the incoming connections.

**Important:** Higher level services must also be available from low-level clients. For example, public services must also be opened for inter-server connectivity.

When the direction specified in the table is described as being loopback, all communication must be opened on this interface. When you change any of the default ports configuration, ensure that you open the ports that are appropriate to the configuration of your environment.

**Configuring firewalls in a domain server deployment**

You must configure the firewalls in a domain server deployment so that specific ports are open for communication.

The following figure shows TADDM communication in a domain server deployment.

![Figure 1. TADDM communication in a domain server deployment](image)

**Connectivity services:**

For a domain server deployment, you can configure public, inter-server, and local connectivity services.

**Public connectivity services**

The following table shows the default host settings for domain server public connectivity services.

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Default interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public service host</td>
<td>com.ibm.cdb.public.hostname</td>
<td>Defined by com.ibm.cdb.global.hostname</td>
</tr>
</tbody>
</table>

The following table shows the default port settings for domain server public connectivity services.

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Protocol</th>
<th>Default port</th>
</tr>
</thead>
<tbody>
<tr>
<td>API server port</td>
<td>com.ibm.cdb.service.ApiServer.port</td>
<td>TCP</td>
<td>9530</td>
</tr>
<tr>
<td>Secure API server port</td>
<td>com.ibm.cdb.service.SecureApiServer.secure.port</td>
<td>TCP</td>
<td>9531</td>
</tr>
</tbody>
</table>
Table 6. Default port settings for domain server public connectivity services (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Protocol</th>
<th>Default port</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP port (without SSL)</td>
<td>com.ibm.cdb.service.web.port</td>
<td>TCP</td>
<td>9430</td>
</tr>
<tr>
<td>HTTPS port (with SSL)</td>
<td>com.ibm.cdb.service.web.secure.port</td>
<td>TCP</td>
<td>9431</td>
</tr>
<tr>
<td>GUI server communication port</td>
<td>com.ibm.cdb.service.ClientProxyServer.port</td>
<td>TCP</td>
<td>9435</td>
</tr>
<tr>
<td>GUI server SSL communication port</td>
<td>com.ibm.cdb.service.SecureClientProxyServer.secure.port</td>
<td>TCP</td>
<td>9434</td>
</tr>
<tr>
<td>Public service registry port</td>
<td>com.ibm.cdb.service.registry.public.port</td>
<td>TCP</td>
<td>9433</td>
</tr>
</tbody>
</table>

Local connectivity services

Local services ports are not explicitly set. All ports must be open on the interface that is defined for local services. The default interface is the loopback.

The following table shows the default host settings for domain server local connectivity services.

Table 7. Default host settings for domain server local connectivity services

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Default interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local service host</td>
<td>com.ibm.cdb.local.hostname</td>
<td>127.0.0.1</td>
</tr>
</tbody>
</table>

Configuring communication in the domain server deployment:

To establish successful communication in the domain server deployment, configure public, and local connectivity services.

The following tables show the elements that you can connect in the domain server deployment and the ports that you must open for the communication to be successful.

Communication between the database server and the domain server

Table 8. Communication between the database server and the domain server.

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database server</td>
<td>5000</td>
<td></td>
<td>Domain server</td>
<td></td>
</tr>
</tbody>
</table>

Communication between Discovery Management Portal, API clients and Web portal and Data Management Portal clients, and the domain server

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discovery Management Portal</td>
<td>9433</td>
<td>→</td>
<td>Domain server - Public service registry</td>
<td>com.ibm.cdb.service.registry.public.port</td>
</tr>
<tr>
<td></td>
<td>9435</td>
<td>→</td>
<td>Domain server - ClientProxyServer</td>
<td>com.ibm.cdb.service.ClientProxyServer.port</td>
</tr>
<tr>
<td></td>
<td>9434</td>
<td>→</td>
<td>Domain server - SecureClientProxyServer</td>
<td>com.ibm.cdb.service.SecureClientProxyServer.secure.port</td>
</tr>
<tr>
<td>API clients</td>
<td>9433</td>
<td>→</td>
<td>Domain server - Public service registry</td>
<td>com.ibm.cdb.service.registry.public.port</td>
</tr>
<tr>
<td></td>
<td>9530</td>
<td>→</td>
<td>Domain server - API server</td>
<td>com.ibm.cdb.service.ApiServer.port</td>
</tr>
<tr>
<td></td>
<td>9531</td>
<td>→</td>
<td>Domain server - Secure API server</td>
<td>com.ibm.cdb.service.SecureApiServer.secure.port</td>
</tr>
<tr>
<td>Web portal and Data Management Portal clients</td>
<td>9430</td>
<td>→</td>
<td>Domain server - Web</td>
<td>com.ibm.cdb.service.web.port</td>
</tr>
<tr>
<td></td>
<td>9431</td>
<td>→</td>
<td>Domain server - Secure web</td>
<td>com.ibm.cdb.service.web.secure.port</td>
</tr>
</tbody>
</table>

Communication between the anchor and gateway, and the domain server

Table 10. Communication between the anchor and gateway, and the domain server.

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anchor (in ssh mode) - SSH</td>
<td>22</td>
<td>←</td>
<td>Domain server (in ssh mode)</td>
<td></td>
</tr>
<tr>
<td>Anchor (in direct mode) - SSH</td>
<td></td>
<td>←</td>
<td>Domain server (in direct mode)</td>
<td></td>
</tr>
<tr>
<td>Anchor (in ssh mode) - SSH tunnel forwarding</td>
<td>8497</td>
<td>←</td>
<td>Domain server (in ssh mode)</td>
<td></td>
</tr>
<tr>
<td>Anchor (in direct mode) - direct</td>
<td></td>
<td>←</td>
<td>Domain server (in direct mode)</td>
<td></td>
</tr>
<tr>
<td>Gateway - SSH</td>
<td>22</td>
<td>←</td>
<td>Domain server</td>
<td></td>
</tr>
</tbody>
</table>

Local communication

Table 11. Local connectivity communication configuration for a domain server.

<table>
<thead>
<tr>
<th>Local communication</th>
<th>Direction</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain server - Local service registry</td>
<td>←</td>
<td>com.ibm.cdb.local.hostname</td>
</tr>
<tr>
<td>Domain server - Local services</td>
<td>←</td>
<td></td>
</tr>
<tr>
<td>Domain server - 127.0.0.1</td>
<td>←</td>
<td></td>
</tr>
</tbody>
</table>

Configuring firewalls in a streaming server deployment

You must configure the firewalls in a streaming server deployment so that specific ports are open for communication.
The following figure shows TADDM communication in a streaming server deployment.

Figure 2. TADDM communication in a streaming server deployment

**Connectivity services:**

For a streaming server deployment, you can configure public, inter-server, and local connectivity services.

**Important:** The default ports for the properties provided later in this section apply only to those properties that are listed in the `collation.properties` file. If a property is not coded or is commented out in the `collation.properties` file, it defaults to a random port. Especially, make sure that the `com.ibm.cdb.service.RegistriesURLProvider.port` property is listed in the `collation.properties` file so that the startup is successful.

**Public connectivity services**

The following table shows the default host settings for primary storage server, secondary storage server, and discovery server public connectivity services.

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Default interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public service host</td>
<td><code>com.ibm.cdb.public.hostname</code></td>
<td>Defined by <code>com.ibm.cdb.global.hostname</code></td>
</tr>
</tbody>
</table>

The following table shows the default port settings for primary storage server, secondary storage server, and discovery server public connectivity services.
### Table 13. Default port settings for primary storage server, secondary storage server, and discovery server public connectivity services

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Protocol</th>
<th>Default port</th>
</tr>
</thead>
<tbody>
<tr>
<td>API server port</td>
<td>com.ibm.cdb.service.ApiServer.port</td>
<td>TCP</td>
<td>9530</td>
</tr>
<tr>
<td>Secure API server port</td>
<td>com.ibm.cdb.service.SecureApiServer.secure.port</td>
<td>TCP</td>
<td>9531</td>
</tr>
<tr>
<td>HTTP port (without SSL)</td>
<td>com.ibm.cdb.service.web.port</td>
<td>TCP</td>
<td>9430</td>
</tr>
<tr>
<td>HTTPS port (with SSL)</td>
<td>com.ibm.cdb.service.web.secure.port</td>
<td>TCP</td>
<td>9431</td>
</tr>
<tr>
<td>GUI server communication port</td>
<td>com.ibm.cdb.service.ClientProxyServer.port</td>
<td>TCP</td>
<td>9435</td>
</tr>
<tr>
<td>GUI server SSL communication port</td>
<td>com.ibm.cdb.service.SecureClientProxyServer.secure.port</td>
<td>TCP</td>
<td>9434</td>
</tr>
<tr>
<td>Public service registry port</td>
<td>com.ibm.cdb.service.registry.public.port</td>
<td>TCP</td>
<td>9433</td>
</tr>
</tbody>
</table>

### Inter-server connectivity services

The following table shows the default host settings for primary storage server and secondary storage server inter-server connectivity services.

#### Table 14. Default host settings for primary storage server and secondary storage server inter-server connectivity services

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Default interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inter-server service host</td>
<td>com.ibm.cdb.interserver.hostname</td>
<td>Defined by com.ibm.cdb.global.hostname</td>
</tr>
</tbody>
</table>

The following table shows the default port settings for primary storage server inter-server connectivity services.

#### Table 15. Default port settings for primary storage server inter-server connectivity services

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Protocol</th>
<th>Default port</th>
</tr>
</thead>
<tbody>
<tr>
<td>TopologyManager port</td>
<td>com.ibm.cdb.service.TopologyManager.port</td>
<td>TCP</td>
<td>9550</td>
</tr>
<tr>
<td>SecurityManager port</td>
<td>com.ibm.cdb.service.SecurityManager.port</td>
<td>TCP</td>
<td>9540</td>
</tr>
<tr>
<td>RegistriesURLProvider port</td>
<td>com.ibm.cdb.service.RegistriesURLProvider.port</td>
<td>TCP</td>
<td>9560</td>
</tr>
<tr>
<td>Inter-server service registry port</td>
<td>com.ibm.cdb.service.registry.interserver.port</td>
<td>TCP</td>
<td>4160</td>
</tr>
</tbody>
</table>

The following table shows the default port settings for secondary storage server inter-server connectivity services.

#### Table 16. Default port settings for secondary storage server inter-server connectivity services

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Protocol</th>
<th>Default port</th>
</tr>
</thead>
<tbody>
<tr>
<td>TopologyManager port</td>
<td>com.ibm.cdb.service.TopologyManager.port</td>
<td>TCP</td>
<td>9550</td>
</tr>
<tr>
<td>RegistriesURLProvider port</td>
<td>com.ibm.cdb.service.RegistriesURLProvider.port</td>
<td>TCP</td>
<td>9560</td>
</tr>
<tr>
<td>Inter-server service registry port</td>
<td>com.ibm.cdb.service.registry.interserver.port</td>
<td>TCP</td>
<td>4160</td>
</tr>
</tbody>
</table>
Local connectivity services

Local services ports are not explicitly set. All ports must be open on the interface that is defined for local services. The default interface is the loopback.

The following table shows the default host settings for primary storage server, secondary storage server, and discovery server local connectivity services.

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Default interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local connectivity area service host</td>
<td>com.ibm.cdb.local.hostname</td>
<td>127.0.0.1</td>
</tr>
</tbody>
</table>

Configuring communication in the streaming server deployment:

To establish successful communication in the streaming server deployment, configure public, inter-server, and local connectivity services.

The following tables show the elements that you can connect in the streaming server deployment and the ports that you must open for the communication to be successful.

Inter-server communication

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
<th>TLS Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discovery server</td>
<td></td>
<td></td>
<td>Primary storage server</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9433</td>
<td></td>
<td>Primary storage server</td>
<td>com.ibm.cdb.service.registry.public.port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>4160</td>
<td></td>
<td>Primary storage server - Inter-server service registry</td>
<td>com.ibm.cdb.service.registry.interserver.port</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>9560</td>
<td></td>
<td>Primary storage server - Registries URLProvider</td>
<td>com.ibm.cdb.service.Registries.URLProvider.port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>9540</td>
<td></td>
<td>Primary storage server - SecurityManager</td>
<td>com.ibm.cdb.service.SecurityManager.port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>9550</td>
<td></td>
<td>Primary storage server - TopologyManager</td>
<td>com.ibm.cdb.service.TopologyManager.port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>9430</td>
<td></td>
<td>Primary storage server - web</td>
<td>com.ibm.cdb.service.web.port</td>
<td>No</td>
</tr>
</tbody>
</table>
Table 18. Inter-server connectivity communication configuration in the streaming server deployment. (continued)

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
<th>TLS Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discovery server</td>
<td></td>
<td></td>
<td>Secondary storage server</td>
<td>com.ibm.cdb. service.registry. interserver:port</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>4160</td>
<td></td>
<td>Secondary storage server - Inter-server service registry</td>
<td>com.ibm.cdb. service.registry. interserver:port</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>9560</td>
<td></td>
<td>Secondary storage server - Registries URLProvider</td>
<td>com.ibm.cdb. service.Registries URLProvider:port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>9550</td>
<td></td>
<td>Secondary storage server - TopologyManager</td>
<td>com.ibm.cdb. service.TopologyManager:port</td>
<td>Yes</td>
</tr>
<tr>
<td>Secondary storage</td>
<td></td>
<td></td>
<td>Primary storage server</td>
<td>com.ibm.cdb. service.registry. registry.public:port</td>
<td>No</td>
</tr>
<tr>
<td>server</td>
<td>4160</td>
<td></td>
<td>Primary storage server - Inter-server service registry</td>
<td>com.ibm.cdb. service.registry. interserver:port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>9560</td>
<td></td>
<td>Primary storage server - Registries URLProvider</td>
<td>com.ibm.cdb. service.Registries URLProvider:port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>9540</td>
<td></td>
<td>Primary storage server - SecurityManager</td>
<td>com.ibm.cdb. service.SecurityManager:port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>9550</td>
<td></td>
<td>Primary storage server - TopologyManager</td>
<td>com.ibm.cdb. service.TopologyManager:port</td>
<td>Yes</td>
</tr>
<tr>
<td>Database server</td>
<td>5000</td>
<td></td>
<td>Primary storage server</td>
<td>com.ibm.cdb. service.registry. registry.public:port</td>
<td>No</td>
</tr>
<tr>
<td>Database server</td>
<td>5000</td>
<td></td>
<td>Secondary storage server</td>
<td>com.ibm.cdb. service.registry. registry.public:port</td>
<td>No</td>
</tr>
</tbody>
</table>

Communication between Discovery Management Portal, API clients and Web portal and Data Management Portal clients, and the TADDM servers


<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
<th>TLS Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discovery Management Portal</td>
<td>9433</td>
<td></td>
<td>Discovery server - Public service registry</td>
<td>com.ibm.cdb. service. registry.public:port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>9435</td>
<td></td>
<td>Discovery server - ClientProxyServer</td>
<td>com.ibm.cdb. service. ClientProxyServer:port</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>9434</td>
<td></td>
<td>Discovery server - SecureClientProxyServer</td>
<td>com.ibm.cdb. service. SecureClientProxyServer:port</td>
<td>Yes</td>
</tr>
<tr>
<td>Element A</td>
<td>Port</td>
<td>Direction</td>
<td>Element B</td>
<td>Configuration property</td>
<td>TLS Support</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>API clients</td>
<td>9433</td>
<td></td>
<td>• Discovery server - Public service registry</td>
<td>com.ibm.cdb.service.registry.public.port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Primary storage server - Public service registry</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Secondary storage server - Public service registry</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9530</td>
<td></td>
<td>• Discovery server - API server</td>
<td>com.ibm.cdb.service.ApiServer.port</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Primary storage server - API server</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Secondary storage server - API server</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9531</td>
<td></td>
<td>• Discovery server - Secure API server</td>
<td>com.ibm.cdb.service.SecureApiServer.security.port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Primary storage server - Secure API server</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Secondary storage server - Secure API server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web portal and Data Management Portal</td>
<td>9430</td>
<td></td>
<td>• Discovery server - Web</td>
<td>com.ibm.cdb.service.web.port</td>
<td>No</td>
</tr>
<tr>
<td>Portal clients</td>
<td></td>
<td></td>
<td>• Primary storage server - Web</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Secondary storage server - Web</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9431</td>
<td></td>
<td>• Discovery server - Secure web</td>
<td>com.ibm.cdb.service.web.security.port</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Primary storage server - Secure web</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Secondary storage server - Secure web</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Communication between the anchor and gateway, and the discovery server
Table 20. Communication between the anchor and gateway, and the discovery server.

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anchor (in ssh mode) - SSH</td>
<td>22</td>
<td></td>
<td>Discovery server (in ssh mode)</td>
<td></td>
</tr>
<tr>
<td>Anchor (in direct mode) - SSH</td>
<td></td>
<td></td>
<td>Discovery server (in direct mode)</td>
<td></td>
</tr>
<tr>
<td>Anchor (in ssh mode) - SSH tunnel forwarding</td>
<td>8497</td>
<td></td>
<td>Discovery server (in ssh mode)</td>
<td></td>
</tr>
<tr>
<td>Anchor (in direct mode) - direct</td>
<td></td>
<td></td>
<td>Discovery server (in direct mode)</td>
<td></td>
</tr>
<tr>
<td>Gateway - SSH</td>
<td>22</td>
<td></td>
<td>Discovery server</td>
<td></td>
</tr>
</tbody>
</table>

Local communication

Table 21. Local connectivity communication configuration in the streaming server deployment.

<table>
<thead>
<tr>
<th>Local communication</th>
<th>Direction</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discovery server</td>
<td></td>
<td>com.ibm.cdb.local.hostname</td>
</tr>
<tr>
<td>Discovery server - Local service registry</td>
<td></td>
<td>com.ibm.cdb.local.hostname</td>
</tr>
<tr>
<td>Discovery server - Local services</td>
<td></td>
<td>com.ibm.cdb.local.hostname</td>
</tr>
<tr>
<td>Discovery server - 127.0.0.1</td>
<td></td>
<td>com.ibm.cdb.local.hostname</td>
</tr>
</tbody>
</table>

Primary storage server

| Primary storage server - Local service registry |           | com.ibm.cdb.local.hostname |
| Primary storage server - Local services |           | com.ibm.cdb.local.hostname |
| Primary storage server- 127.0.0.1 |           | com.ibm.cdb.local.hostname |

Secondary storage server

| Secondary storage server - Local service registry |           | com.ibm.cdb.local.hostname |
| Secondary storage server - Local services |           | com.ibm.cdb.local.hostname |
| Secondary storage server- 127.0.0.1 |           | com.ibm.cdb.local.hostname |

Configuring firewalls in a synchronization server deployment

You must configure the firewalls in a synchronization server deployment so that specific ports are open for communication.

The following figure shows TADDM communication in a synchronization server deployment.
Connectivity services configuration:

For a synchronization server deployment, you can configure public, inter-server, and local connectivity services.

**Important:** The default ports for the properties provided later in this section apply only to those properties that are listed in the `collation.properties` file. If a property is not coded or is commented out in the `collation.properties` file, it defaults to a random port. Especially, make sure that the `com.ibm.cdb.service.RegistriesURLProvider.port` property is listed in the `collation.properties` file so that the startup is successful.

Public connectivity services

The following table shows the default host settings for the domain server and synchronization server public connectivity services.

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Default interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public service host</td>
<td><code>com.ibm.cdb.public.hostname</code></td>
<td>Defined by <code>com.ibm.cdb.global.hostname</code></td>
</tr>
</tbody>
</table>

The following table shows the default port settings for the domain server public connectivity services.

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Protocol</th>
<th>Default port</th>
</tr>
</thead>
<tbody>
<tr>
<td>API server port</td>
<td><code>com.ibm.cdb.service.ApiServer.port</code></td>
<td>TCP</td>
<td>9530</td>
</tr>
<tr>
<td>Secure API server port</td>
<td><code>com.ibm.cdb.service.SecureApiServer.secure.port</code></td>
<td>TCP</td>
<td>9531</td>
</tr>
<tr>
<td>HTTP port (without SSL)</td>
<td><code>com.ibm.cdb.service.web.port</code></td>
<td>TCP</td>
<td>9430</td>
</tr>
<tr>
<td>HTTPS port (with SSL)</td>
<td><code>com.ibm.cdb.service.web.secure.port</code></td>
<td>TCP</td>
<td>9431</td>
</tr>
<tr>
<td>GUI server communication port</td>
<td><code>com.ibm.cdb.service.ClientProxyServer.port</code></td>
<td>TCP</td>
<td>9435</td>
</tr>
<tr>
<td>GUI server SSL communication port</td>
<td><code>com.ibm.cdb.service.SecureClientProxyServer.secure.port</code></td>
<td>TCP</td>
<td>9434</td>
</tr>
<tr>
<td>Public service registry port</td>
<td><code>com.ibm.cdb.service.registry.public.port</code></td>
<td>TCP</td>
<td>9433</td>
</tr>
</tbody>
</table>
The following table shows the default port settings for the synchronization server public connectivity services.

**Table 24. Default port settings for synchronization server public connectivity services**

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Protocol</th>
<th>Default port</th>
</tr>
</thead>
<tbody>
<tr>
<td>API server port</td>
<td>com.ibm.cdb.service.ApiServer.port</td>
<td>TCP</td>
<td>9530</td>
</tr>
<tr>
<td>Secure API server port</td>
<td>com.ibm.cdb.service.SecureApiServer.secure.port</td>
<td>TCP</td>
<td>9531</td>
</tr>
<tr>
<td>HTTP port (without SSL)</td>
<td>com.ibm.cdb.service.web.port</td>
<td>TCP</td>
<td>9430</td>
</tr>
<tr>
<td>HTTPS port (with SSL)</td>
<td>com.ibm.cdb.service.web.secure.port</td>
<td>TCP</td>
<td>9431</td>
</tr>
<tr>
<td>Public service registry port</td>
<td>com.ibm.cdb.service.registry.public.port</td>
<td>TCP</td>
<td>9433</td>
</tr>
</tbody>
</table>

**Inter-server connectivity services**

The following table shows the default host settings for domain server and synchronization server inter-server connectivity services.

**Table 25. Default host settings for domain server and synchronization server inter-server connectivity services**

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Default interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inter-server service host</td>
<td>com.ibm.cdb.interserver.hostname</td>
<td>Defined by com.ibm.cdb.global.hostname</td>
</tr>
</tbody>
</table>

The following table shows the default port settings for the domain server inter-server connectivity services.

**Table 26. Default port settings for domain server inter-server connectivity services**

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Protocol</th>
<th>Default port</th>
</tr>
</thead>
<tbody>
<tr>
<td>TopologyManager port</td>
<td>com.ibm.cdb.service.TopologyManager.port</td>
<td>TCP</td>
<td>9550</td>
</tr>
<tr>
<td>SecurityManager port</td>
<td>com.ibm.cdb.service.SecurityManager.port</td>
<td>TCP</td>
<td>9540</td>
</tr>
<tr>
<td>RegistriesURLProvider port</td>
<td>com.ibm.cdb.service.RegistriesURLProvider.port</td>
<td>TCP</td>
<td>9560</td>
</tr>
<tr>
<td>Inter-server service registry port</td>
<td>com.ibm.cdb.service.registry.interserver.port</td>
<td>TCP</td>
<td>4160</td>
</tr>
</tbody>
</table>

The following table shows the default port settings for the synchronization server inter-server connectivity services.

**Table 27. Default port settings for synchronization server inter-server connectivity services**

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Protocol</th>
<th>Default port</th>
</tr>
</thead>
<tbody>
<tr>
<td>RegistriesURLProvider port</td>
<td>com.ibm.cdb.service.RegistriesURLProvider.port</td>
<td>TCP</td>
<td>9560</td>
</tr>
<tr>
<td>EnterpriseSecurityManager port</td>
<td>com.ibm.cdb.service.EnterpriseSecurityManager.port</td>
<td>TCP</td>
<td>9570</td>
</tr>
<tr>
<td>Inter-server service registry port</td>
<td>com.ibm.cdb.service.registry.interserver.port</td>
<td>TCP</td>
<td>4160</td>
</tr>
</tbody>
</table>
Local connectivity services

Local services ports are not explicitly set. All ports must be open on the interface that is defined for local services. The default interface is the loopback.

The following table shows the default host settings for the domain server and synchronization server local connectivity services.

**Table 28. Default host settings for the domain server and synchronization server local connectivity services**

<table>
<thead>
<tr>
<th>Name</th>
<th>Configuration property</th>
<th>Default interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local service host</td>
<td>com.ibm.cdb.local.hostname</td>
<td>127.0.0.1</td>
</tr>
</tbody>
</table>

Configuring communication in the synchronization server deployment:

To establish successful communication in the synchronization server deployment, configure public, inter-server, and local connectivity services.

The following tables show the elements that you can connect in the synchronization server deployment and the ports that you must open for the communication to be successful.

**Inter-server communication**

**Table 29. Inter-server connectivity communication configuration in the synchronization server deployment.**

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain server</td>
<td></td>
<td></td>
<td>Synchronization server</td>
<td>com.ibm.cdb.service.registry.interserver. port</td>
</tr>
<tr>
<td></td>
<td>4160</td>
<td></td>
<td>Synchronization server - Inter-server service registry</td>
<td>com.ibm.cdb.service.RegistryURL.Provider.port</td>
</tr>
<tr>
<td></td>
<td>9560</td>
<td></td>
<td>Synchronization server - RegistriesURL.Provider</td>
<td>com.ibm.cdb.service.RegistryURL.Provider.port</td>
</tr>
<tr>
<td></td>
<td>9570</td>
<td></td>
<td>Synchronization server - EnterpriseSecurityManager</td>
<td>com.ibm.cdb.service.EnterpriseSecurityManager.port</td>
</tr>
<tr>
<td>Database server</td>
<td>5000</td>
<td></td>
<td>Domain server</td>
<td></td>
</tr>
</tbody>
</table>
### Table 29. Inter-server connectivity communication configuration in the synchronization server deployment. (continued)

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database server</td>
<td>5000</td>
<td>←</td>
<td>Synchronization server</td>
<td></td>
</tr>
</tbody>
</table>

**Communication between Discovery Management Portal, API clients and Web portal and Data Management Portal clients, and the domain and synchronization servers**

### Table 30. Communication between Discovery Management Portal, API clients and Web portal and Data Management Portal clients, and the domain and synchronization servers.

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discovery Management Portal</td>
<td>9433</td>
<td>←</td>
<td>Domain server - Public service registry</td>
<td>com.ibm.cdb.service.registry.public.port</td>
</tr>
<tr>
<td></td>
<td>9435</td>
<td>←</td>
<td>Domain server - ClientProxyServer</td>
<td>com.ibm.cdb.service.ClientProxyServer.port</td>
</tr>
</tbody>
</table>
| API clients                       | 9433 | ←         | • Domain server - Public service registry  
|                                   |      |           | • Synchronization server - Public service registry | com.ibm.cdb.service.registry.public.port |
|                                   | 9530 | ←         | • Domain server - API server        | com.ibm.cdb.service.ApiServer.port        |
|                                   | 9531 | ←         | • Domain server - Secure API server | com.ibm.cdb.service.SecureApiServer.secure.port |
| Web portal and Data Management Portal clients | 9430 | ←         | • Domain server - Web               | com.ibm.cdb.service.web.port              |
|                                   | 9431 | ←         | • Domain server - Secure web        | com.ibm.cdb.service.web.secure.port       |

**Communication between the anchor and gateway, and the domain server**

### Table 31. Communication between the anchor and gateway, and the domain server.

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anchor (in ssh mode) - SSH</td>
<td>22</td>
<td>←</td>
<td>Domain server (in ssh mode)</td>
<td></td>
</tr>
<tr>
<td>Anchor (in direct mode) - SSH</td>
<td></td>
<td>←</td>
<td>Domain server (in direct mode)</td>
<td></td>
</tr>
</tbody>
</table>
Table 31. Communication between the anchor and gateway, and the domain server. (continued)

<table>
<thead>
<tr>
<th>Element A</th>
<th>Port</th>
<th>Direction</th>
<th>Element B</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anchor (in ssh mode) - SSH tunnel forwarding</td>
<td>8497</td>
<td></td>
<td>Domain server (in ssh mode)</td>
<td></td>
</tr>
<tr>
<td>Anchor (in direct mode) - direct</td>
<td></td>
<td></td>
<td>Domain server (in direct mode)</td>
<td></td>
</tr>
<tr>
<td>Gateway - SSH</td>
<td>22</td>
<td></td>
<td>Domain server</td>
<td></td>
</tr>
</tbody>
</table>

Local communication

Table 32. Local connectivity communication configuration in the synchronization server deployment.

<table>
<thead>
<tr>
<th>Local communication</th>
<th>Direction</th>
<th>Configuration property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Domain server - Local service registry</td>
<td>←→</td>
<td>com.ibm.cdb.local.hostname</td>
</tr>
<tr>
<td>Domain server - Local services</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Domain server - 127.0.0.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Synchronization server</td>
<td>←→</td>
<td>com.ibm.cdb.local.hostname</td>
</tr>
<tr>
<td>Synchronization server - Local service registry</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Synchronization server - Local services</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Synchronization server - 127.0.0.1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TADDM server properties reference

The collation.properties file contains properties for the TADDM server. You can edit some of these properties.

The collation.properties file is located in the $COLLATION_HOME/etc directory. The file contains comments about each of the properties.

If you update the collation.properties file, you must save the file and restart the server for the change to take effect.

Scoped and non-scoped properties

The collation.properties file contains two types of properties: scoped and non-scoped.

scoped property
A property to which you can append either an IP address or the name of a scope set. The IP address or the scope set name makes the property dependent on the host that is being discovered. You can use only scope set names that do not contain spaces, apostrophes (’), periods (.), and forward slashes (/).

non-scoped property
A property that you cannot restrict to be specific to an object.
For example, the following properties are non-scoped properties:
• `com.collation.log.filesize`
• `com.collation.discover.agent.command.1sof.Linux`

However, the `com.collation.discover.agent.command.1sof.Linux` property can be a scoped property if you append either an IP address or a scope set name to the property, as shown in the following examples:
• Example of appending the IP address 129.42.56.212:
  `com.collation.discover.agent.command.1sof.Linux.129.42.56.212=sudo 1sof`
• Example of appending a scope set named “scope1”:
  `com.collation.discover.agent.command.1sof.Linux.scope1=sudo 1sof`

**Properties that you must not change**
Changing some properties in the `collation.properties` file can make your system inoperative.

You must not change the following properties:

- **com.collation.version**
  Identifies the product version.

- **com.collation.branch**
  Identifies the branch of code.

- **com.collation.buildnumber**
  Identifies the build number. This number is set by the build process.

- **com.collation.oalbuildnumber**
  Identifies the build number for another build process.

- **com.collation.SshWeirdReauthErrorList=Permission denied**
  The value of this property must be `Permission denied`.
  The property is needed because Windows systems can randomly deny valid login attempts. You can try the user name and password pairs that have previously worked during discovery runs.

**Access credentials caching properties**
These properties apply to access credentials caching.

- **com.ibm.cdb.security.auth.cache.disabled=false**
  The default value is `false`.
  This property determines whether credentials caching is disabled.
  This property is a scoped and profiled property. You can append an IP address, the name of a scope set or a profile name. You can also set it in the profile configuration in Discovery Management Console.

- **com.ibm.cdb.security.auth.cache.fallback.failed=true**
  The default value is `true`.
  This property turns on fallback, when a cache contains valid credentials, but upon retrieval, it fails to validate. If fallback is enabled and cached credentials are no longer valid, cache iterates over all available access entry types until a match is found.
  This property is a scoped and profiled property. You can append an IP address, the name of a scope set or profile name.
  The following entries are the examples of entries that can be found in the `collation.properties` file:
You can also set this property in Discovery Management Console in the profile configuration on the **Platform Properties** tab.

**com.ibm.cdb.security.auth.cache.fallback.invalid=true**

The default value is true.

This property turns on fallback, when entry read from cache contains invalid attempt (the last access failed, there was no valid credentials). If fallback is enabled, cache iterates over all available access entry types until a match is found.

This property is a scoped and profiled property, you can append an IP address, the name of a scope set or profile name.

The following entries are the examples of entries that can be found in the collation.properties file:

- `com.ibm.cdb.security.auth.cache.fallback.invalid=false`
- `com.ibm.cdb.security.auth.cache.fallback.invalid.10.160.160.11=true`
- `com.ibm.cdb.security.auth.cache.fallback.invalid.ScopeA=true`
- `com.ibm.cdb.security.auth.cache.fallback.invalid.GroupA=true`
- `com.ibm.cdb.security.auth.cache.fallback.invalid.Level_2_Discovery=false`

You can also set this property in Discovery Management Console in the profile configuration on the **Platform Properties** tab.

**Fix Pack 5 com.ibm.cdb.security.auth.cache.itm.disabled=true**

The default value is true.

This property determines whether credentials caching is disabled for OSLC discovery.

This property is a scoped and profiled property. You can append an IP address, the name of a scope set or a profile name. You can also set it in the profile configuration in Discovery Management Console.

**Related concepts:**

- "Last successful credentials caching" on page 13

  TADDM can cache last working access credentials. They can be reused in the next (Level 2 or script-based) discovery.

**API port properties**

These properties apply to API ports.

**com.ibm.cdb.service.ApiServer.port=9530**

The default value is 9530. The value must be an integer.

This property specifies the port that the API server listens on for non-SSL requests. The value can be set to any available port on the server. Any client that uses the API for connection must specify this port for a non-SSL connection.

**com.ibm.cdb.service.SecureApiServer.secure.port=9531**

The default value is 9531. The value must be an integer.

This property specifies the port that the API server listens on for SSL requests. The value can be set to any available port on the server. Any client that uses the API for connection must specify this port for an SSL connection.
Cleanup agents properties
The cleanup agents remove the orphaned aliases and configuration items, or fix the missing rows in the tables. Most of them read properties that are defined in the collation.properties file.

AliasesCleanupAgent
The agent removes aliases from the ALIASES table that no longer match the CI naming attributes. It also removes aliases and rows in the PERSOBJ table which do not have any corresponding CIs. The agent reads the following properties from the collation.properties file:

- **Fix Pack 2**  
  com.ibm.cdb.topomgr.topobuilder.deleteAliasesWithoutMaster
  The default value is true.
  The property specifies whether the aliases that do not have the corresponding master alias are deleted from the ALIASES table. By default, the deletion is enabled.

- com.ibm.cdb.topomgr.topobuilder.max.row.fetch
  The default value is 1000.
  The property configures the batch size used to fetch aliases from the ALIASES table.
  If you set the property to -1, the agent does not verify the aliases.

- com.ibm.cdb.topomgr.topobuilder.max.row.delete
  The default value is 5000.
  The property configures the batch size used to delete aliases.
  If you set the property to -1, the agent does not remove aliases but only reports the corrupted ones.

- com.ibm.cdb.topomgr.topobuilder.agents.AliasesCleanupAgent.maxNumberOfMastersToScan
  The default value is 1000.
  The property configures the number of CIs that require aliases verification during a single run of the agent.

- com.ibm.cdb.topomgr.topobuilder.cleanupOrphanedAliasesAndPersobj
  The default value is true. The agent runs the cleanup.
  The property enables or disables the cleanup of those aliases in the ALIASES table and GUIDs in the PERSOBJ table which do not have any corresponding CIs.

- com.ibm.cdb.topomgr.topobuilder.DelayToRemoveAliases
  The default value is 12 (hours). The orphaned aliases that are older than 12 hours are removed by the agent.
  The property defines the time in hours after which the aliases without a corresponding CI are removed by the agent. It protects new aliases that might not have a corresponding CI because the CI storing is not complete.
  Use this property with caution. Do not set it to a smaller value.
**AliasesInTableCleanupAgent**

This agent removes old rows from the ALIASES_JN table. This table contains the history of changes to the ALIASES table. It is used to find potential over merges of configuration items in the database. The agent reads the following properties from the *collation.properties* file:

- **com.ibm.cdb.topmgr.topobuilder.agents.AliasesInTableCleanupAgent.maxRow**
  The default value is 5000. It is advised not to change the default value.
  This property specifies the maximum number of rows that are deleted at a time by the agent.

- **com.ibm.cdb.topmgr.topobuilder.agents.AliasesInTableCleanupAgent.removeOlderThanDays**
  The default value is 30 (days).
  This property removes rows that are older than the specified time. By default, it removes rows that are older than 30 days.
  If you set this property to 0 or a lower value, the agent is disabled.

- **com.ibm.cdb.topmgr.topobuilder.agents.AliasesInTableCleanupAgent.timeout**
  The default value is 1800 (seconds).
  This property specifies time, after which the agent times out. If the specified time is not enough to delete all old rows, the agent attempts to delete them the next time it runs.

**DependencyCleanupAgent**

The agent removes the dormant Relationship objects. The agent reads the following properties from the *collation.properties* file:

- **com.ibm.cdb.topmgr.topobuilder.agents.DependencyCleanupAgent.timeout**
  The default value in seconds is 600. After this time, the agent stops removing the objects, even if there are still some left.

- **com.ibm.cdb.topmgr.topobuilder.agents.DependencyCleanupAgent.removeOlderThanDays**
  The default value in days is 90. The Relationship objects, which are older than the specified value are treated as dormant objects.

**ObjectsWithoutAliasesCleanupAgent**

The agent removes the CIs that do not have the aliases in the ALIASES table. The agent reads the following property from the *collation.properties* file:

- **com.ibm.cdb.topmgr.topobuilder.agents.ObjectsWithoutAliasesCleanupAgent.maxToRemove**
  The default value is 1000
  The property limits the number of CIs that the agent removes during one run. If you set the property to -1, the agent exits without performing any cleanup, and shows the **ObjectsWithoutAliasesCleanupAgent is disabled** message.
**PersobjCleanupAgent**

The agent fixes all missing rows in the PERSOBJ table. It does not use any configuration in the `collation.properties` file. The agent shows the summary of how many rows were fixed, like in the following example:

```
2012-08-22 18:12:21,500 TopologyBuilder [TopologyBuilderEngineThread$Cleanup@4.0] INFO agents.PersobjCleanupAgent - Fixed 10 rows in PERSOBJ table
```

**StorageExtentCleanupAgent**

The agent removes the dormant StorageExtent objects. The agent reads the following properties from the `collation.properties` file:

- `com.ibm.cdb.topomgr.topobuilder.agents.StorageExtentCleanupAgent.timeout`
  - The default value in seconds is 1800. After this time, the agent stops removing the objects, even if there are still some left.

- `com.ibm.cdb.topomgr.topobuilder.agents.StorageExtentCleanupAgent.removeOlderThanDays`
  - The default value in days is 1. The StorageExtent objects, which are more than 1 day older than their parent ComputerSystems are treated as dormant objects.

**VlanInterfaceCleanupAgent**

The agent removes the dormant VlanInterface objects. The agent reads the following properties from the `collation.properties` file:

- `com.ibm.cdb.topomgr.topobuilder.agents.VlanInterfaceCleanupAgent.timeout`
  - The default value in seconds is 1800. After this time, the agent stops removing the objects, even if there are still some left.

- `com.ibm.cdb.topomgr.topobuilder.agents.VlanInterfaceCleanupAgent.removeOlderThanDays`
  - The default value in days is 1. The VlanInterface objects which are more than 1 day older than their parent Vlans are treated as dormant objects.

**Commands that might require elevated privilege**

These properties specify the operating system commands that TADDM uses that might require elevated privilege, such as root (or superuser) authority, to run on the target system.

Typically, sudo is used on UNIX and Linux systems to provide privilege escalation. The following alternatives can be used instead of sudo:

- Enable the setuid access right on the target executable program
- Add the discovery service account to the group associated with the target executable program
- Use root for the discovery service account (not preferred)

For each property, sudo can be configured globally, meaning to run the command with sudo on every operating system target, or restricted to a specific IP address or scope set.

**Important:** On each target system for which privilege escalation is needed, sudo must be configured with the NOPASSWD option. Otherwise, your discovery hangs until sudo times out.
com.collation.discover.agent.command.hastatus.Linux=\texttt{sudo} /opt/VRTSvcs/bin/hastatus
com.collation.discover.agent.command.haclus.Linux=\texttt{sudo} /opt/VRTSvcs/bin/haclus
com.collation.discover.agent.command.hasys.Linux=\texttt{sudo} /opt/VRTSvcs/bin/hasys
com.collation.discover.agent.command.hares.Linux=\texttt{sudo} /opt/VRTSvcs/bin/hares
com.collation.discover.agent.command.hagrp.Linux=\texttt{sudo} /opt/VRTSvcs/bin/hagrp
com.collation.discover.agent.command.hatype.Linux=\texttt{sudo} /opt/VRTSvcs/bin/hatype
com.collation.discover.agent.command.hauser.Linux=\texttt{sudo} /opt/VRTSvcs/bin/hauser

- These properties are required to discover Veritas Cluster components.
- To execute these commands without \texttt{sudo}, the TADDM service account must be a member of the Veritas Admin Group on the target.

com.collation.discover.agent.command.vxdisk=vxdisk
com.collation.discover.agent.command.vxdg=vxdg
com.collation.discover.agent.command.vxprint=vxprint
com.collation.discover.agent.command.vxlicrep=vxlicrep
com.collation.discover.agent.command.vxupgrade=vxupgrade

- These properties discover Veritas standard storage information plus additional Veritas specific information like disk group, Veritas volumes,plexes, and subdisks.

\textbf{Fix Pack 6} \hspace{1em} com.collation.discover.agent.command.zlogin=\texttt{sudo} zlogin

com.collation.platform.os.command.ps.SunOS=/usr/ucb/ps axww
com.collation.platform.os.command.psEnv.SunOS=/usr/ucb/ps axwweee
com.collation.platform.os.command.psParent.SunOS=ps -elf -o ruser,pid,ppid,comm
com.collation.platform.os.command.psUsers.SunOS=/usr/ucb/ps auxw

- These properties are required to discover process information about Solaris systems.

You can specify a particular Solaris version by appending the SunOS version number to the property name. For example, the following property is specific to Solaris 10:

\texttt{com.collation.platform.os.command.ps.SunOS5.10=sudo /usr/ucb/ps axww}

com.collation.platform.os.command.ps.Linux=ps axww
com.collation.platform.os.command.psEnv.Linux=ps axwweee
com.collation.platform.os.command.psParent.Linux=ps -ax -o ruser,pid,ppid,comm
com.collation.platform.os.command.psUsers.Linux=ps auxw

- These properties are required to discover process information about Linux systems.

com.collation.platform.os.command.ps.AIX=ps axww
com.collation.platform.os.command.psEnv.AIX=ps axwweee
com.collation.platform.os.command.psParent.AIX=ps -elf -o ruser,pid,ppid,comm
com.collation.platform.os.command.psUsers.AIX=ps auxw

- These properties are required to discover process information about AIX systems.

com.collation.platform.os.command.ps.HP-UX=sh UNIX95= ps -elfx -o pid,ttystate,time,args
com.collation.platform.os.command.psEnv.HP-UX=ps -elfx
com.collation.platform.os.command.psParent.HP-UX=sh UNIX95= ps -elfx -o
ruser,pid,ppid,comm
com.collation.platform.os.command.psUsers.HP-UX=ps -efx
  • These properties are required to discover process information about HP-UX systems.
com.collation.discover.agent.command.lsof.Vmnix=lsof
com.collation.discover.agent.command.lsof.Linux=lsof
com.collation.discover.agent.command.lsof.SunOS.1.2.3.4=sudo lsof
com.collation.discover.agent.command.lsof.Linux.1.2.3.4=sudo lsof
com.collation.discover.agent.command.lsof.HP-UX=lsof
com.collation.discover.agent.command.lsof.AIX=lsof
  • These properties are required to discover process or port information.
  You can specify a particular Solaris version by appending the SunOS version number to the property name. For example, the following property is specific to Solaris 10:
com.collation.discover.agent.command.lsof.SunOS5.10= /usr/local/bin/lsof

com.collation.discover.agent.command.dmidecode.Linux=dmidecode
com.collation.discover.agent.command.dmidecode.Linux.1.2.3.4= sudo dmidecode
  • These properties are required to discover UUID, manufacturer, model, and serial number on Linux systems.
com.collation.discover.agent.command.vmcp.Linux=
  This property can be used to discover a guest user ID on a target Linux virtual system running on a z/VM® operating system.
com.collation.discover.agent.command.cat.SunOS=cat
com.collation.discover.agent.command.cat.SunOS.1.2.3.4= sudo cat
  • This property is required to discover configuration information for a Check Point firewall on Solaris systems.
com.collation.discover.agent.command.interfacesettings.SunOS=sudo ndd
com.collation.discover.agent.command.interfacesettings.Linux= sudo mii-tool
com.collation.discover.agent.command.interfacesettings.SunOS.1.2.3.4= sudo ndd
com.collation.discover.agent.command.interfacesettings.Linux.1.2.3.5= sudo mii-tool
com.collation.discover.agent.command.interfacesettings.HP-UX=lanadmin
com.collation.discover.agent.command.interfacesettings.AIX=netstat
  • These properties are required to discover advanced network interface information (interface speed, for example).
com.collation.discover.agent.command.adb.HP-UX=adb
com.collation.discover.agent.command.adb.HP-UX.1.2.3.4= sudo adb
  • This property is required to discover processor information about HP systems.
com.collation.discover.agent.command.kmadmin.HP-UX=kmadmin
com.collation.discover.agent.command.kmadmin.HP-UX.1.2.3.4= sudo /usr/sbin/kmadmin
  • This property is required to discover kernel modules on HP systems.
com.collation.platform.os.command.partitionTableListing.SunOS=prtvtoc
  • This property is required to discover partition table information about Solaris systems.
com.collation.platform.os.command.lvm.lvdisplay.1.2.3.4= sudo lvdisplay -c
com.collation.platform.os.command.lvm.vgdisplay.1.2.3.4= sudo vgdisplay -c
com.collation.platform.os.command.lvm.pvdisplay.1.2.3.4=sudo pvdisplay -c
  • These properties are required to discover storage volume information.

com.collation.platform.os.command.Iputil.SunOS.1.2.3.4=sudo /usr/sbin/lpfc/Iputil
  • This property is required to discover Emulex fibre channel HBA information on Solaris systems.

com.collation.platform.os.command.crontabEntriesCommand.SunOS=crontab -l
com.collation.platform.os.command.crontabEntriesCommand.Linux=crontab -l
com.collation.platform.os.command.crontabEntriesCommand.AIX=crontab -l
com.collation.platform.os.command.crontabEntriesCommand.HP-UX=crontab -l
  • These properties are required to discover crontab entries. You can specify these properties as a scoped property by appending an IP address or a scope set name to the property. The following example uses an appended IP address:

  com.collation.platform.os.command.crontabEntriesCommand.AIX.1.2.3.4=crontab -l

com.collation.platform.os.command.filesystems.Linux=df -kTP
com.collation.platform.os.command.filesystems.SunOS=df -k | grep -v 'No such file or directory' | grep -v 'Input/output error' | awk '{print $1, $2, $4, $6}'
com.collation.platform.os.command.filesystems.AIX=df -k | grep -v 'No such file or directory' | grep -v 'Input/output error' | awk '{print $1, $2, $3, $7}'
com.collation.platform.os.command.filesystems.HP-UX=df -kP | grep -v 'No such file or directory' | grep -v 'Input/output error' | grep -v FileSystem
  • These properties are required to discover file systems.

com.collation.platform.os.command.fileinfo.ls=sudo ls
com.collation.platform.os.command.fileinfo.ls.1.2.3.4=sudo ls
com.collation.platform.os.command.fileinfo.cksum=sudo cksum
com.collation.platform.os.command.fileinfo.cksum.1.2.3.4=sudo cksum
com.collation.platform.os.command.fileinfo.dd=sudo dd
com.collation.platform.os.command.fileinfo.dd.1.2.3.4=sudo dd
  • These properties are required for privileged file capture.
  • Privileged file capture is used in situations where the discovery service account does not have read access to application configuration files that are required for discovery.

com.collation.discover.agent.WebSphereVersionAgent.versionscript=sudo
  This property can be enabled for access to the WebSphere versionInfo.sh file if the discovery user does not have access on the target WebSphere Application Server system.

com.collation.platform.os.command.fileinfo.OnlyDirectoryRecursive
  This flag changes the way the configuration files are discovered. The default value is False.

If you change this flag to True, this mechanism does not use the find command to recursively discover the content of a directory.

When you set the flag to False, this mechanism uses the find command to discover a file recursively, without the exact location of the file being specified.
**Context Menu Service and Data Integration Service properties**

These properties apply to the Context Menu Service (CMS) and Data Integration Service (DIS).

```java
com.ibm.cdb.DisCmsIntegration.enabled=true
```

The default value is **true**.

This property specifies whether to enable the CMSDISAgent topology builder agent for periodic updates of TADDM data that is registered with the Context Menu Service and Data Integration Service database.

```java
com.ibm.cdb.DisCmsIntegration.dbUser=user
```

This property specifies the database user ID for the Context Menu Service and Data Integration Service database.

```java
com.ibm.cdb.DisCmsIntegration.dbPassword=password
```

This property specifies the password for the Context Menu Service and Data Integration Service database user.

```java
com.ibm.cdb.DisCmsIntegration.dbUrl=url
```

This property specifies the database URL for the Context Menu Service and Data Integration Service database.

```java
com.ibm.cdb.DisCmsIntegration.dbDriver=driver
```

This property specifies the database driver for the Context Menu Service and Data Integration Service.

```java
com.ibm.cdb.DisCmsIntegration.changehistory.days_previous=30
```

The default value is **30**.

This property specifies the number of days of change history to be displayed in change reports for the Context Menu Service and Data Integration Service.

**Database properties**

These properties apply to the TADDM database.

```java
com.collation.db.password=password
```

This property specifies the database password, which is stored on the TADDM server, for the database user.

```java
com.collation.db.archive.password=password
```

This property specifies the database password, which is stored on the TADDM server, for the database archive user.

```java
com.ibm.cdb.db.max.retries
```

This property specifies the number of retry attempts to establish the connection to the database.

```java
com.ibm.cdb.db.timeout
```

This property specifies the sleep time (in milliseconds) between the retry attempts.

```java
com.ibm.cdb.db.connection.ssl.enable=false
```

This property specifies whether the connection to the database is established in the SSL mode for the database user.

The default value is **false**.

```java
com.ibm.cdb.db.connection.ssl.truststore.file=filename
```

This property specifies a truststore file that is used for the SSL connection to the database for the database user. The truststore file must be in the $COLLATION_HOME/etc/ directory.
com.ibm.cdb.db.connection.ssl.truststore.password=password
This property specifies a truststore password that is used for the SSL connection to the database for the database user.

com.ibm.cdb.db.archive.connection.ssl.enable=false
This property specifies whether the connection to the database is established in the SSL mode for the archive database user.

The default value is false.

com.ibm.cdb.db.archive.connection.ssl.truststore.file=filename
This property specifies a truststore file that is used for the SSL connection to the database for the archive database user. The truststore file must be in the $COLLATION_HOME/etc/ directory.

com.ibm.cdb.db.archive.connection.ssl.truststore.password=password
This property specifies a truststore password that is used for the SSL connection to the database for the archive database user.

To encrypt the database passwords in the collation.properties file, complete the following steps:
1. Edit the database user, or archive the user password using clear text, or both.
2. Stop the TADDM server.
3. Run either the encryptprops.sh file or the encryptprops.bat file (located in the $COLLATION_HOME/bin directory). This script encrypts the passwords.
4. Restart the TADDM server.

Discovery properties
These properties apply to discovery generally. The TADDM server properties that affect a specific sensor are documented in the TADDM Sensor Reference for the respective sensor.

**com.discover.anchor.maxChannelNumber**
This property specifies the maximum number of channels opened simultaneously in the SSH session between the TADDM server and the anchor. If the number of opened channels is too high, the discovery on such anchor might hang, and the sensors included in such scope might time out. In such case, use this property to control the number of opened channels.

The default value is 50.

**com.collation.platform.os.copyToLocal.preferScpCommand**
This property specifies whether the external scp command is used to copy files from remote hosts, usually discovery targets, to the TADDM server. The external scp command is defined in the com.collation.platform.os.scp.command property. To enable the usage of external scp command, set this property to true.

The default value of this property is false.

**Note:** This property applies only to the SSH sessions that are established with a key-based login (see ["Configuring for discovery using the Secure Shell (SSH)" on page 110](#)). In case of authentication with a user name and a password, the internal scp command is used regardless of the com.collation.platform.os.copyToLocal.preferScpCommand property value.
This property is a scoped property. You can append an IP address or the name of a scope set to the property. For example:
com.collation.platform.os.copyToLocal.preferScpCommand.12.234.255.4=true

**com.collation.platform.os.scp.command**

This property specifies the path to the Operating System `scp` command. It can be used when an internal SSH client fails to send files between the TADDM server and remote hosts, usually discovery targets. You can also use an alternative command but it must have the same syntax as the `scp` command.

The example value: `/usr/local/bin/scp`.

**Fix Pack 3**

**com.collation.platform.session.ssh.winAuth**

This property specifies whether logging in with Windows credentials is attempted when SSH session is used. The default value is true.

You can set the value to false, if there is a risk that during the discovery there are attempts to log in to non-Windows servers with Windows credentials. It can prevent locking out Windows Active Directory accounts.

**Fix Pack 3**

**com.collation.platform.os.ignoreL2InterfaceDescription**

This property specifies the descriptions of discovered L2Interfaces that you want to be ignored during the computer system signature calculation. For example, if you do not want Microsoft Load Balancer Interface to be used to calculate the signature of a computer system, specify the following value:

```
com.collation.platform.os.ignoreL2InterfaceDescription=Microsoft Load Balancer Interface
```

The value of this property is treated as a regular expression. It means that you can add more than one interface description, and you do not need to use any separator such as comma.

**Fix Pack 3**

**com.ibm.cdb.topomgr.topobuilder.agents.ConnectionDependencyAgent2.dependencyPlaceholders**

This property, if set to true, creates placeholder App Servers for undiscovered dependencies.

**Note:** This property is not included in the `collation.properties` file by default. You must add it there.

When you set the value to true for the first time, you must restart TADDM to enable extended attributes for LogicalConnection and SSoftwareServer classes. These extended attributes are necessary for proper functioning of this feature.

For more information about placeholders, see "Configuring for discovery of placeholders" on page 123.

**com.collation.platform.session.EncodingOverRide**

This property specifies the type of encoding used during a discovery session. It is especially useful when your target servers use different encoding than the one in the TADDM server.

The value of this property is the name of the encoding, for example `UTF-8`. It is not included in the `collation.properties` file by default, you must add it there.

You can also add a scope, or an IP address to this property. For example:
**com.collation.discover.anchor.forceDeployment=true**

The default value is true.

This property specifies whether the anchors for the discovered scope are to be deployed during discovery startup.

If you set the value to false, the anchors are deployed only if either of the following conditions are met:

- If any IP address from the scope cannot be pinged
- If port 22 cannot be reached on any of the discovered IP addresses

If chained anchors exist, this condition applies to all anchors in the chain. If an anchor in the chain is restricted with a condition, the prior anchors must meet the condition before all the anchors can be deployed.

**com.collation.discover.anchor.lazyDeployment=false**

The default value is false.

This property specifies whether files that a sensor requires are copied when an anchor is deployed (a value of false) or when the sensor that requires the files is about to start (a value of true).

For example, the IBM WebSphere sensor has dependencies in the dist/lib/websphere directory. The size of the directory is 130 MB. If the value of this property is false, the dependency data is copied to the target host when the anchor is deployed. If the value is true, the data is copied when the WebSphere sensor is about to run on the anchor. If no WebSphere sensor is run through the anchor, 130 MB is not sent to the remote host.

**com.collation.discover.DefaultAgentTimeout=600000**

The value is 600000 (in milliseconds), which is 10 minutes.

This property specifies the timeout for sensors in milliseconds. The default timeout should not be changed. Instead, you can specify the timeout for individual sensors.

To override the timeout for a particular sensor, add the following line to the collation.properties file:

```java
com.collation.discover.agent.sensorNameSensor.timeout=timeInMilliseconds
```

Here is an example:

```java
com.collation.discover.agent.OracleSensor.timeout=1800000
```

**com.collation.IpNetworkAssignmentAgent.defaultNetmask=ip_start-ip_end/netmask[, ...]**

This property defines how IP addresses discovered during a Level 1 discovery are assigned to generated subnets. A Level 1 discovery does not discover subnets. Instead, IpNetwork objects are generated to contain any interfaces that are not associated with an existing subnet discovered during a Level 2 or Level 3 discovery. This configuration property defines which IpNetwork objects are created, and how many nodes each subnet contains. (It also applies to any interface discovered during a Level 2 or Level 3 discovery that for any reason cannot be assigned to a discovered subnet.)

The value for this property consists of a single line containing one or more entries separated by commas. Each entry describes an IP address range in IPv4 dotted decimal format, with a subnet mask specified as an integer in
the range of 8 - 31. Discovered interfaces in the specified range are then
placed in created subnets that are no larger than the size that is specified
by the subnet mask.

For example, the following value defines two subnet address ranges with
different subnet masks:
9.0.0.0-9.127.255.255/23, 9.128.0.0-9.255.255.255/24

The specified address ranges can overlap. If a discovered IP address
matches more than one defined range, it is assigned to the first matching
subnet as they are listed in the property value.

After you create or change this configuration property and restart the
TADDM server, any subsequent Level 1 discoveries use the defined
subnets. To reassign existing IpInterface objects in the TADDM database,
go to the $COLLATION_HOME/bin directory and run one of the following
commands:

- adjustL1Networks.sh (Linux and UNIX systems)
- adjustL1Networks.bat (Windows systems)

If the value is not specified correctly then the appropriate messages are
displayed only when running the command-line utility
adjustL1Networks.sh (Linux and UNIX systems) or adjustL1Networks.bat
(Windows systems). Otherwise the messages are placed in the
TopologyBuilder.log file in the $COLLATION_HOME/log/services directory,
and in the IpNetworkAssignmentAgent.log file in the $COLLATION_HOME/log/
agents directory.

This script reassigns all IpInterface objects discovered during Level 1
discoveries to the appropriate subnets as described in the configuration
property. Any generated IpNetwork object that contains no interfaces is
then deleted from the database. After the script is completed, the TADDM
interface might show multiple notifications of changed components
because of the modified objects. You can clear these notifications by
refreshing the window.

Note: Before you use this command, make sure that the TADDM server is
running, and that no discovery or bulk load operation is currently in
progress. This script is not supported on the synchronization server.

com.collation.number.persist.discovery.run=30
The default value is 30.

Specifies the number of discoveries for which information is saved in the
discovery history in the Data Management Portal and the Discovery
Management Console.

To change the default value in a streaming server deployment, enter the
new value on the primary storage server.

com.collation.platform.os.hostappdescriptorfiles.dir="path"

Specifies the fully qualified path to the directory where component
application descriptor files for computer systems (hosts) are deployed. This
property is required if you want to add computer systems to business
applications using application descriptors. You can scope this property to a
specific host name or IP address in order to specify a different location for
each host. The following examples show how to specify the host
application descriptor path:

- Linux and UNIX systems: /home/taddm/hostappdescriptors
Windows systems: c://taddm//hostappdescriptors

com.collation.platform.session.GatewayForceSsh
Specifies whether to force the gateway to act independently of the anchor. Valid values are true and false. Set the value to true to resolve Cygwin issues when both the gateway and anchor are on the same system. When the value is set to true, an SSH session is used to transfer traffic between the gateway and anchor rather than a local session.

com.collation.rediscoveryEnabled=false
The default value is false.
This property applies to the rediscovery of a configuration item that has already been discovered. The rediscovery functionality is available in the Data Management Portal.

Restriction: Rediscovery cannot use the credentials from a custom profile, it uses the credentials from the global list.

Note:
To enable rediscovery in a domain server deployment, set the value to true on the domain server.
To enable rediscovery in a streaming server deployment, set the value to true on the discovery server and the storage server.

Rediscovery in a streaming server deployment
When rediscovery is used in a streaming server deployment, a configuration item can be discovered by different discovery servers, but only the last discovery server to discover the configuration item can rediscover that configuration item. Because there are multiple discovery servers, rediscovery information for a configuration item is overwritten by each discovery server.

When you enable rediscovery on the discovery server, for each object discovered, additional information about the rediscovery is created.

When you enable rediscovery on the storage server, each object that is discovered is stored with additional information about the rediscovery.

If rediscovery is enabled on the discovery server but disabled on the storage server, information about rediscovery will not be available in the TADDM database. In addition, you must ensure that the same credentials are used for both the discovery server and the storage server.

com.ibm.cdb.discover.sensor.sys.utilization.workingdir=/tmp/taddm
The default value is /tmp/taddm.
This property specifies the root path for the IBM Tivoli Utilization sensor scripts to run on the target system. If this value is not specified the path defined by the com.ibm.cdb.taddm.script.path property is used.

com.ibm.cdb.locationTag
Specifies the location tag attribute for each configuration item (CI) created on the TADDM server. The location tag attribute, which identifies the location of a CI, is used to support static location tags. Before specifying this tag, you must set the com.ibm.cdb.locationTaggingEnabled value to true.
**com.ibm.cdb.locationTaggingEnabled=false**

The default value is false.

Specifies whether location tagging functionality is enabled. Set the value of this property to true to:

- Specify a location tag attribute for each configuration item (CI) created on the TADDM server (static location tags). See the `com.ibm.cdb.locationTag` property for details.
- Specify a dynamic location tag for configuration items (CIs) created during a single discovery, using the command-line interface (CLI). Dynamic location tags override location tags that already exist (static location tags).
- Specify a dynamic location tag for configuration items (CIs) created when loading data using the bulkload program.
- Specify a location tag value when running a BIRT report to filter the data and report information only about that specified location.
- Create a location tag value for configuration items created during a typology build process.

**com.ibm.cdb.taddm.host**

Specifies the TADDM server host alias. If this value is not specified the system hostname is used. If the TADDM server cannot resolve the system hostname or it resolves to the localhost then you must specify this property manually.

**com.ibm.cdb.taddm.script.path=/tmp/taddm**

The default value is /tmp/taddm.

This property specifies the root path for the sensor scripts to run on the target system. In this location, a subdirectory tree is created using the format: `host_alias/discovery_number/sensor_name`. The `host_alias` name is retrieved from the `com.ibm.cdb.taddm.host` property. If this property is not specified the system hostname is used. To differentiate between concurrent discoveries on the same discovery server, a number is assigned to the `discovery_number` directory. The discovery scripts and discovery results are stored using this directory structure.

**com.collation.discover.agent.signature.ignore.1.2.3.4=true**

This property is used to skip an IP address during the signature calculation.

In case of some configurations, the signature for a computer system might result as not unique, which causes problems during the reconciliation with the existing entries in the TADDM database. For example, it can happen when you use virtual machines with virtual network cards that have a valid hardware address and IP address. In such cases, you must exclude the signature calculation and use other naming rules, for example `ProductModel`, `Manufacturer` or `Serial Number`.

For each IP address that you want to be ignored, add the `com.collation.discover.agent.signature.ignore.1.2.3.4=true` property, where 1.2.3.4 is the IP address to be ignored.

If you want many IP addresses to be ignored, you can create a discovery scope. Add the `com.collation.discover.agent.signature.ignore.blacklist=true` property to the `collation.properties` file, where `blacklist` is the discovery scope with all IP addresses to be ignored.
Advanced discovery properties:

The advanced discovery properties specify the buffer capacity for storing work items, the number of restarts of particular discovery elements, or the time value for printing the statistics to a log. Do not change these properties unless you must closely tune the discovery process.

com.ibm.cdb.discover.buffers.workitem.capacity=64
The default value is 64. However, this value is always twice the value of com.collation.discover.dwcount, which is 32 by default.

This property specifies the buffer capacity for storing discovery work items. It is used to limit the memory requirements of the discovery process and thus avoid the OutOfMemory errors. For each discovery, a new sensor is started.

Do not set the value to be smaller than the number of the discovery workers that is specified in com.collation.discover.dwcount, because otherwise some of them are left in the idle state.

com.ibm.cdb.discover.buffers.workitem.maxresets=10
The default value is 10.

This property specifies the number of times a sensor can restart in case of an unexpected failure, such as a failure of a TADDM JVM that is responsible for the discovery.

Also, the number of restarts for an element of the discovery process is limited by com.ibm.cdb.discover.runrestartlimit that specifies the number of discovery restarts.

com.ibm.cdb.discover.buffers.seed.capacity=100
The default value is 100.

This property specifies the buffer capacity for storing seed work items. It is used to limit the memory requirements of the discovery process and thus avoid the OutOfMemory errors.

com.ibm.cdb.discover.buffers.result.capacity=100
The default value is 100.

This property specifies the buffer capacity for storing result work items. It is used to limit the memory requirements of the discovery process and thus avoid the OutOfMemory errors. For each result work item, a new sensor can start.

Set the value to the same size as com.ibm.cdb.discover.buffers.discovered.capacity.

com.ibm.cdb.discover.buffers.result.maxresets=10
The default value is 10.

This property specifies the number of times a discovery process can start a new sensor for a result work item in case of an unexpected failure, such as a failure of a TADDM JVM that is responsible for the discovery.

Also, the number of restarts for an element of the discovery process is limited by com.ibm.cdb.discover.runrestartlimit that specifies the number of discovery restarts.

com.ibm.cdb.discover.buffers.discovered.capacity=100
The default value is 100.
This property specifies the buffer capacity for storing discovered work items. Each discovered work item represents a discovery result that is stored in the database.

Do not specify this value to be smaller than the number of the database writing threads that is specified in com.collation.discover.observer.topopumpcount.

```
com.ibm.cdb.discover.buffers.statistics.interval.seconds=60
```

The default value is 60. Specify the value in seconds.

This property specifies the time value for saving the discovery buffer statistics to a log. The log is at /log/services/DiscoveryState.log.

```
com.ibm.cdb.discover.buffers.timeout.interval.seconds=600
```

The default value is 600, which is 10 minutes. Specify the value in seconds.

This property specifies the time value for checking the work items for timeout.

```
com.ibm.cdb.discover.runcontroller.statistics.interval.seconds=60
```

The default value is 60. Specify the value in seconds.

This property specifies the time value for saving the discovery run statistics to a log. The log is at /log/services/DiscoveryRunController.log.

```
com.ibm.cdb.discover.runrestartlimit=11
```

The default value is 11.

This property specifies the number of times an uninitialized discovery can be restarted after a failure. The discovery is in the uninitialized state when the process is not yet started for all of the elements of the discovery scope.

```
com.collation.discovery.oracle.tablelimit=1000
```

The default value is 1000. The property supports only positive values.

This property controls the quantity of tables that are discovered by Oracle sensor.

**Concurrent discovery properties:**

These properties apply to concurrent discovery.

```
com.collation.discover.concurrent.discovery=true
```

The default value is true.

This property is used to enable concurrent discovery.

```
com.collation.discover.max.concurrent.discoveries=10
```

The default value is 10.

This property defines the maximum number of concurrent discoveries.

**Asynchronous discovery properties:**

These properties apply to asynchronous discovery.

```
com.ibm.cdb.discover.asd.AsyncDiscoveryResultsDirectory=var/asdd
```

The default value is var/asdd, which is relative to the com.collation.home directory.
This property defines the location of the root directory for the archive files on the TADDM server that contain asynchronous discovery results. The location can be a relative or absolute path. A relative path is relative to the com.collation.home directory.

**com.ibm.cdb.discover.asd.ProcessUnreachableIPs=false**

The default value is false.

This property is used to enable the processing of unreachable IP addresses, which are used in asynchronous discovery. To enable the processing of these addresses, set the value to true.

**com.ibm.cdb.tarpath=**

The default value is tar.

This property specifies the path of the **tar** command on the TADDM server in asynchronous discovery.

On operating systems such as AIX or Linux, this property is typically not needed because the **tar** command is already installed and available. However, to generate an asynchronous discovery script package or to process discovery archive files on a TADDM server that is running the Windows operating system, you must install a third party tar program and specify the full path name for that program.

The following example shows how to specify the path of the **tar** command on the TADDM server for the AIX operating system:

```bash
com.ibm.cdb.tarpath=tar
```

**com.ibm.cdb.targettarpath=**

The default value is tar.

This property specifies the path of the **tar** command on the target system in asynchronous discovery.

On target operating systems such as AIX or Linux, this property is typically not needed because the **tar** command is already installed and available. However, to generate discovery archive files on Solaris operating systems, because of a limitation in the length of file names, you must use the gtar archive utility, and you must specify the path to the utility.

The following examples show how to specify the path of the **tar** command on the target system, depending on the operating system:

**For AIX**

```bash
com.ibm.cdb.targettarpath.AIX=tar
```

**For Solaris**

```bash
com.ibm.cdb.targettarpath.SunOS=/usr/sfw/bin/gtar
```

### Script-based discovery properties:

These properties apply to script-based discovery.

**com.ibm.cdb.discover.enableOutputFileSplittingProcess=true**

The default value is true.

This property specifies whether the main output file that is created during a script-based discovery is split into smaller files. By default, the file is split. This setting prevents performance issues when the output file is large. See also the

**com.ibm.cdb.discover.numberOfLinesForOutputFileSplittingProcess** property.

---

Administering 75
**Fix Pack 4**

`com.ibm.cdb.discover.numberOfLinesForOutputFileSplittingProcess=10000`

The default value is 10000.

This property is enabled only when the `com.ibm.cdb.discover.enableOutputFileSplittingProcess` property is set to true.

This property specifies the approximate number of lines allowed in the smaller output files, which were created by splitting the main output file. The exact number of lines is determined by the file format. After the specified number of lines, the file is split only when the end of the meaningful set of data is reached to ensure that the whole file format is correct. It means that when the value is set to 10000, the smaller files can have, for example, 10200 lines.

`com.ibm.cdb.taddm.asd.prefix=sh`

The default value is `sh`.

This property specifies a prefix to be added to script that is run during a discovery, for example `prefix script.sh`. This property is a scoped property, you can append either an IP address or the name of a scope set.

`com.ibm.cdb.discover.DeleteScriptDiscoveryOutputs=true`

The default value is true.

This property specifies whether to delete the script output that, during script-based discovery, is copied to the TADDM server for processing by the sensors. This output might be useful for troubleshooting, but by default, it is deleted after discovery completes. If you set the value of this property to false, the script output will not be deleted.

`com.ibm.cdb.discover.DeleteRemoteBeforeScriptsRun=false`

The default value is false.

This property specifies whether TADDM removes any outputs that are left by previous discovery from the remote directory before attempting to run a new discovery.

`com.ibm.cdb.discover.PreferScriptDiscovery=false`

The default value is false.

This property is used to enable script-based discovery, and it affects only the sensors that support script-based discovery. Setting the value to true enables script-based discovery.

`com.ibm.cdb.discover.smallFileSizeLimit=1048576`

The default value is 1048576 (1024*1024 - 1 MB).

This property defines the limit of file size expressed in bytes for copy operations that triggers the checksum usage. The files, the size of which is below this limit, are copied without the checksum calculations. The files, the size of which equals the limit, or is greater than this limit, are copied only if they are not present in the target directory and their checksum does not match the local (source) file.

You can disable the limit by using the following values:

- 0 - copy operation always uses the checksum.
- -1 - copy operation always avoids using the checksum.
Properties for discovery using IBM Tivoli Monitoring (old method):

These properties apply to discovery using IBM Tivoli Monitoring (old method).

The old integration method

This section applies to a deprecated method of TADDM integration with IBM Tivoli Monitoring. Starting with the TADDM version 7.3.0, it is advised to integrate with IBM Tivoli Monitoring 6.3 using OSLC Automation. The old method of integration with the use of IBM Tivoli Monitoring Scope sensor is deprecated and will be removed from future releases. For more information about properties that are used to configure the discovery process by using OSLC Automation, see “Integrating TADDM with IBM Tivoli Monitoring via OSLC Automation” on page 189 and “Properties for discovery using OSLC Automation Session” on page 79.

Properties that affect how TADDM discovers Tivoli Monitoring endpoints

TADDM Level 2 and Level 3 discovery normally requires a domain server (in a domain or synchronization server deployment) or a discovery server (in a streaming server deployment) to connect directly to a target system using one of the following methods:

- Secure Shell (SSH) for UNIX-based target systems
- Windows Management Instrumentation (WMI) for Windows systems

To use these methods, the domain or discovery server must know the user credentials (account and password).

Discovery using IBM Tivoli Monitoring allows TADDM to discover Level 2 (and some Level 3) information about target systems for which no user credentials are available. Sensors run through the Tivoli Monitoring infrastructure, and only the credentials of the Tivoli Enterprise Portal Server are required. After the IBM Tivoli Monitoring Scope sensor has been configured and run, future Level 2 discoveries use Tivoli Monitoring for discovery by default. Because you might not want this default behavior in your environment, TADDM provides the following server properties to control whether Tivoli Monitoring or direct connection (SSH or WMI) is used for discovery. These properties can be set at a global level or for a specific scope or discovery profile.

com.ibm.cdb.session.allow.ITM=true

The default value is true, which means that TADDM can use IBM Tivoli Monitoring to discover Tivoli Monitoring endpoints.

This property specifies whether TADDM can use IBM Tivoli Monitoring to discover Tivoli Monitoring endpoints.

To connect directly to a Tivoli Monitoring endpoint, set the value to false.

You can also use this property to specify a custom discovery scope, as indicated in the following example:

com.ibm.cdb.session.allow.ITM.ip_address=false

The following example specifies that TADDM uses the discovery scope 10.20.30.40 and connects directly to the endpoint even if it is monitored by Tivoli Monitoring:

com.ibm.cdb.session.allow.ITM.10.20.30.40=false

com.ibm.cdb.session.prefer.ITM=true

The default value is true, which means that TADDM uses IBM Tivoli Monitoring to discover Tivoli Monitoring endpoints.
This property specifies whether TADDM uses IBM Tivoli Monitoring as the preferred method for discovering Tivoli Monitoring endpoints, assuming that discovery using IBM Tivoli Monitoring is allowed for the endpoints. If TADDM uses IBM Tivoli Monitoring for the discovery and the discovery is not successful, TADDM then uses a direct connection to the endpoints. Similarly, if discovery using IBM Tivoli Monitoring is not preferred and direct connection to the endpoint is not successful, TADDM tries to connect to the endpoints using IBM Tivoli Monitoring, again assuming that discovery using IBM Tivoli Monitoring is allowed for the endpoints.

You can also use this property to specify a custom discovery scope, as indicated in the following example:

```plaintext
com.ibm.cdb.session.prefer.ITM.ip_address=false
```

The following example specifies that TADDM uses the discovery scope 10.20.30.40 and connects directly to the Tivoli Monitoring endpoints:

```plaintext
com.ibm.cdb.session.prefer.ITM.10.20.30.40=false
com.ibm.cdb.session.prefer.ITM.Level_3_Discovery=false
```

The default value is `false`, which means that TADDM connects directly to the Tivoli Monitoring endpoints if you use a Level 3 discovery profile, but for all other discovery levels, TADDM uses IBM Tivoli Monitoring to discover the Tivoli Monitoring endpoints, depending on the values of the following properties:

- `com.ibm.cdb.session.allow.ITM`
- `com.ibm.cdb.session.prefer.ITM`

This property specifies whether TADDM uses IBM Tivoli Monitoring to discover Tivoli Monitoring endpoints if you use a Level 3 discovery profile.

If you set the value to `true`, TADDM can use IBM Tivoli Monitoring to discover the Tivoli Monitoring endpoints from a Level 3 discovery profile.

Properties for tuning the connection between the TADDM server and the portal server

For an IBM Tivoli Monitoring Level 2 discovery, TADDM uses the following TADDM server properties to tune connection recovery behavior if the connection between the TADDM server and the Tivoli Enterprise Portal Server lags:

- `com.collation.discover.agent.ITM.CmdWrapperSelectionPattern`  
  This property specifies the commands that must be wrapped by a script when running a discovery through an IBM Tivoli Monitoring environment.

- `com.collation.platform.session.ITMSessionConnectionCooldownPeriod=60000`
  This property specifies the time interval in milliseconds to wait before the connection to the Tivoli Enterprise Portal Server is reinitialized after a failure has been detected.

- `com.collation.platform.session.ITMSessionConnectionRetryLimit=5`
  This property specifies the number of times to try accessing a connection if the initial connection fails before reporting an error.

- `com.collation.platform.session.ITMSessionNumProgressChecks=600`
  This property specifies the number of times that a connection is checked for progress before the connection fails.
com.collation.platform.session.ITMSessionProgressCheckInterval=1000
This property specifies the time interval in milliseconds between each connection progress check.

Properties for discovery using OSLC Automation Session:
These properties apply to discovery using OSLC Automation Session.

Properties related to integration over OSLC
com.ibm.cdb.topobuilder.integration.oslc.automationprovider
This property specifies direct URL addresses of OSLC Execute Automation Service Providers that are not registered in Jazz SM Registry Services.

The following example shows URL addresses of OSLC Execute Automation Service Provider for ITM:

The following example shows how to specify URL addresses for several OSLC Execute Automation Service Providers:
com.ibm.cdb.topobuilder.integration.oslc.automationprovider.2=http://9.2.2.2:15210/itmautomationprovider

com.ibm.cdb.topobuilder.integration.oslc.automation.scope.alwaysrefresh=false
The default value is false.

This property is a global property that specifies whether OSLCAutomationAgent rebuilds scope sets during every run. Rebuilding the scope sets requires connection to Jazz SM Registry Services or OSLC Execute Automation Service Providers, or both.

If the property is set to true, the agent rebuilds scope sets, even if the automation plan provided by OSLC Execute Automation Service Provider did not change since the last agent’s run.

com.ibm.cdb.topobuilder.integration.oslc.frsurl
This property specifies Jazz SM Registry Services (FRS) IP address that is used in integration with other products over OSLC. Jazz SM Registry Services address must have the following format:
protocol://ip_or_hostname:port

This property is also used by OSLCAgent.

com.ibm.cdb.topobuilder.integration.oslc.automation.frsurl
This property specifies the IP address in the form of a full path to the registration collection of Jazz SM Registry Services (FRS). It may be used when Jazz SM Registry Services uses other services path than the default /oslc.

Properties related to discovery using Automation Session
com.ibm.cdb.session.oslcautomation.pluginId=com.ibm.cdb.session.oslcautomation_1.0.0
The default value is com.ibm.cdb.session.oslcautomation_1.0.0.

This property specifies OSGI bundle id of the OSLC Automation Session plug-in.
com.ibm.cdb.session.itm.endpointClass=com.collation.platform.session.oslcautomation.OSLCAutomationEndpoint
   The default value is com.collation.platform.session.oslcautomation.OSLCAutomationEndpoint.
   This property specifies the endpoint class to be used.

com.ibm.cdb.session.allow.OSLCAutomation=true
   The default value is true.
   This property is a scoped property that specifies whether TADDM can use the OSLC Automation Session during the discovery.
   Example of usage:
   com.ibm.cdb.session.allow.OSLCAutomation=true
   com.ibm.cdb.session.allow.OSLCAutomation.9.100.1.0=true
   com.ibm.cdb.session.allow.OSLCAutomation.scope_set2=true

com.ibm.cdb.session.prefer.OSLCAutomation=true
   The default value is true.
   This property is a scoped property that specifies whether OSLC Automation Session is a preferred session for a discovery. The value of this property takes precedence over any other preferred values, for example, a standard ITM session.
   Example of usage:
   com.ibm.cdb.session.prefer.OSLCAutomation=true
   com.ibm.cdb.session.prefer.OSLCAutomation.9.100.100.200=true
   com.ibm.cdb.session.prefer.OSLCAutomation.scope_name1=true

com.ibm.cdb.session.oslcautomation.timeout.httpconnect=60000
   The default value is 60000 (60 seconds). The value is expressed in milliseconds.
   This property is a global property that specifies timeout for connection to OSLC Execute Automation Service Provider.

com.ibm.cdb.session.oslcautomation.timeout.httpread=240000
   The default value is 240000 (4 minutes). The value is expressed in milliseconds.
   This property is a global property that specifies timeout for reading data from OSLC Execute Automation Service Provider.

com.ibm.cdb.session.oslcautomation.request.async.maxretries=60
   The default value is 60.
   This property is a global property that specifies the maximum number of consecutive requests for asynchronously generated AutomationResults.

com.ibm.cdb.session.oslcautomation.request.async.delay=10000
   The default value is 10000 (10 seconds). The value is expressed in milliseconds.
   This property is a global property that specifies the delay time between consecutive requests for asynchronously generated AutomationResults.

Note: In case the SSH Session to server fails due to timeout issues, try configuring an optimum value for the below property:

com.collation.mindterm.Ssh2Preferences=hello-timeout=30;alive=25;compression=9
com.collation.discover.agent.app.packagedapp.mysap.SLDServerPortList = 51200
This property allows to change the SLD port and the specified port shall be
added in the sensor configuration.

com.ibm.cdb.security.auth.cache.itm.disabled=true
The default value is true.
This property determines whether credentials caching is disabled for OSLC
discovery.
This property is a scoped and profiled property. You can append an IP
address, the name of a scope set or a profile name. You can also set it in
the profile configuration in Discovery Management Console.

DNS lookup customization properties
These properties apply to DNS lookup customization.
com.collation.platform.os.disableDNSLookups=false
The default value is false.
Valid values are true or false. If you change the property to true, DNS
lookups are disabled for the TADDM server.
com.collation.platform.os.disableRemoteHostDNSLookups=false
The default value is false.
Valid values are true or false. If you change the property to true, name
lookups (DNS only) are disabled on remote discovered hosts. This property
forces all name lookups to occur on the TADDM server.
com.collation.platform.os.command.fqdn=nslookup $1 | grep Name | awk
'{print $2}'
The default value is nslookup $1 | grep Name | awk '{print $2}'
This command is used to find the fully-qualified domain name (fqdn). In
most situations, this property is not needed because the default fully
qualified domain name (FQDN) algorithm works in most production
environments. If this property is not needed, you must comment it out.
However, in environments where the fully-qualified domain name is to be
derived from the host name, you might enable this property. For example,
allow this property if the host names are configured as aliases in DNS.
If this property is used, ensure that DNS is available and properly
configured. Otherwise, the nslookup command is likely to fail or have a
slow response time.
If enabled, this property is only used on the TADDM server. Currently,
only AIX, and Linux operating systems are supported. This property is not
supported on a Windows TADDM server.

GUI properties
These properties apply to the TADDM GUI.
Fix Pack 3
com.ibm.cdb.gui.supportedJRE.warning=true
This property specifies whether the warning message CTJTG0034E is
displayed when the Discovery Management Console is started. This
message warns you that you have an unsupported version of Java runtime
environment. If you want to use TADDM with the unsupported version of
Java runtime environment, and you do not want this message to be
displayed, set this property to false.
The default value of this property is true.
GUI JVM memory properties:

These properties apply to GUI JVM memory.

**com.collation.gui.initial.heap.size=128m**

The default value is 128m. Initial heap size for the TADDM user interface.

**com.collation.gui.max.heap.size=512m**

The default value is 512m. Maximum heap size for the TADDM user interface.

These properties are appropriate for a small TADDM domain. For the purposes of sizing, the following categories of TADDM servers are used (based on server equivalents):

- Small: up to 1000 server equivalents
- Medium: 1000 - 2500 server equivalents
- Large: 2500 - 5000 server equivalents

Increasing these values for medium and large environments improve performance for some GUI operations. Some views do not complete properly if there is not sufficient memory available to TADDM at the time of the action.

For a medium environment:

**com.collation.gui.initial.heap.size=256m**

The default value is 256m.

**com.collation.gui.max.heap.size=768m**

The default value is 768m.

For a large environment:

**com.collation.gui.initial.heap.size=512m**

The default value is 512m.

**com.collation.gui.max.heap.size=1024m**

The default value is 1024m.

**GUI port properties:**

These properties apply to GUI ports.

**com.collation.tomcatshutdownport=9436 (TADDM 7.3.0 only)**

The default value is 9436. This port is used for the Tomcat shutdown command.

**com.ibm.cdb.service.web.port=9430**

The default value is 9430. The HTTP port is used without SSL.

**com.ibm.cdb.service.web.secure.port=9431**

The default value is 9431. The HTTPS port is used with SSL.
The default value is 9435.
The RMI data port to use without SSL.

The default value is 9434.
The RMI data port to use with SSL.

The default value is 9433.
The public service registry port.

**LDAP properties**
These properties apply to LDAP.

An external LDAP server can be used for user authentication. Both anonymous authentication and password-based authentication are supported with an external LDAP server.

The LDAP server host name, port number, base distinguished name, bind distinguished name, and password (required for password-based authentication) are configurable in the collation.properties file. You can also configure the specific naming attribute that can be searched for to match the user ID (UID).

LDAP configuration is recommended in synchronization server and domain server deployments. In an enterprise environment, configure the domain server and the synchronization server to use the same user registry. When you log in to a domain server that is connected to a synchronization server, the login is processed at the synchronization server. If a network connection problem occurs between the synchronization server and a domain server, you can successfully log in to the domain server without reconfiguration if the domain server is configured to use the same user registry as the synchronization server.

**com.collation.security.auth.ldapAuthenticationEnabled=true**
The default value is true.

This property is used to enable LDAP authentication.

**com.collation.security.auth.ldapBaseDN=ou=People,dc=ibm,dc=com**
The default value is ou=People,dc=ibm,dc=com.

This property defines the LDAP Base Distinguished Name (DN). The LDAP Base Distinguished Name is the starting point for all LDAP searches.

**com.collation.security.auth.ldapBaseGroupDN**
In the collation.properties file, this property is commented out by default.

This property defines the LDAP root branch for searching groups, which can be different from the root branch for all LDAP queries. To specify more than one LDAP root branch for searching for groups, separate the branch names by using the “;” character.

If you do not specify a value for this property, the default value is the value of the com.collation.security.auth.ldapBaseDN property.

**com.collation.security.auth.ldapBindDN=uid=ruser,dc=ibm,dc=com**
The default value is uid=ruser,dc=ibm,dc=com.
If simple authentication is used, this property defines the user ID that is used to authenticate to LDAP.

**Important:**

- If a value for `com.collation.security.ldapBindDN` is not supplied or if the property is commented out, an anonymous connection to LDAP is attempted. The following example shows how the property can be commented out with the number sign (#):
  ```
  #com.collation.security.auth.ldapBindDN=uid=ruser, dc=ibm,dc=com
  ```
- If a value is specified for `com.collation.security.auth.ldapBindDN`, simple authentication is used and
- a value for `com.collation.security.auth.ldapBindPassword` must also be specified.

```
com.collation.security.auth.ldapBindPassword=ruser
```

The default value is `ruser`.

If simple authentication is used, this property defines the user password that is used to authenticate to LDAP.

```
com.collation.security.auth.ldapClientKeyStore=ks_path
```

The property defines the location of the keystore that contains the certificates on the TADDM server. The store must contain the client certificate to authenticate the TADDM server with the LDAP server.

```
com.collation.security.auth.ldapClientKeyStorePassphrase=ks_passphrase
```

Optional: This property defines the password to the keystore.

```
com.collation.security.auth.ldapClientTrustStore=ts_path
```

The property defines the location of the truststore that contains the certificates on the TADDM server. The store must contain the LDAP server certificate.

```
com.collation.security.auth.ldapClientTrustStorePassphrase=ts_passphrase
```

Optional: This property defines the password to the truststore.

```
com.collation.security.auth.ldapGroupMemberAttribute=member
```

The default value is `member`.

This property defines the name of the attribute used to contain the members of a group in LDAP.

```
com.collation.security.auth.ldapGroupNamingAttribute=cn
```

The default value is `cn`.

This property defines the name of the attribute used for naming groups in LDAP.

```
com.collation.security.auth.ldapGroupObjectClass=groupofnames
```

The default value is `groupofnames`.

This property defines the class used to represent user groups in LDAP.

```
com.collation.security.auth.ldapHostName=ldap.ibm.com
```

The default value is `ldap.ibm.com`.

This property defines the host name for the LDAP server.

```
com.collation.security.auth.ldapPortNumber=389
```

The default value is 389.

This property defines the port for the LDAP server.
com.collation.security.auth.ldapUIDNamingAttribute=uid
   The default value is uid.
   This property defines the name of the attribute used for naming users in LDAP.

com.collation.security.auth.ldapUserObjectClass=person
   The default value is person.
   This property defines the name of the class used to represent users in LDAP.

com.collation.security.auth.ldapUseSSL=false
   The default value is false.
   The property is used to enable authentication to an LDAP user registry with an SSL connection.

com.collation.security.usermanagementmodule=ldap
   The default value is ldap.
   This property defines the user management module used by the TADDM server. The valid values are:
   • file for a file-based user registry. The default value is true.
   • ldap for an LDAP user registry
   • vmm for a user registry that uses the federated repositories of WebSphere Application Server

**Lockout properties**
These properties apply to lockouts.

com.collation.security.lockout.treshold=3
   The default value is 3.
   This property specifies the number of failed login attempts, for a particular user, that triggers a local lockout for that user.

com.collation.security.lockout.timeout=30
   The default value is 30.
   This property specifies the time, in minutes, for which the user who triggered the local lockout, is locked out of TADDM when a local lockout is triggered.

com.collation.security.lockout.globalthreshold=100
   The default value is 100.
   This property specifies the number of simultaneous single-user lockouts that triggers a global lockout.

com.collation.security.lockout.globaltimeout=30
   The default value is 30.
   This property specifies the time, in minutes, for which all users are locked out of TADDM when a global lockout is triggered.

com.collation.security.lockout.failedloginthreshold=1000
   The default value is 1000.
   This property specifies the total number of failed login attempts for unique users that triggers a global lockout.

**Logging properties**
These properties apply to logging.
**com.collation.logfilesize=20MB**  
The default value is 20MB.

The maximum size for the log file. When the file reaches this size limit, a new log file is created. The current log file is saved with the \$N file extension. \$N is the number 1 through the value set in the **com.collation.log.filecount** property. You set how many log files can be created and kept before the files are rotated with the **com.collation.log.filecount** property.

You can enter the number of bytes directly, or by specifying the number of kilobytes or megabytes using KB and MB, respectively.

The following examples are valid log file size values:
- 1000000
- 512 KB
- 10 MB

**com.collation.log.filecount=5**  
The default value is 5.

The number of log files that you maintain.

**com.collation.log.level.vm.vmName=INFO**  
The default value is INFO

Sets the log level for each virtual system.

\$vmName is a Java virtual system associated with a TADDM service name. The following list identifies other valid options:
- Topology
- DiscoverAdmin
- EventsCore
- Proxy
- Discover
- EcmbdCore
- StorageService
- DiscoveryService

The following list identifies other valid options:
- FATAL
- ERROR
- WARNING
- INFO
- DEBUG (Setting the DEBUG option decreases system performance.)
- TRACE (Setting the TRACE option causes passwords to be logged.)

**Performance properties**
These properties apply to TADDM performance.

**com.collation.discover.dwcount=32**  
The default value is 32. The value must be an integer.

This property influences the discovery rate. A discover worker thread is a thread that runs sensors. This property specifies how many discover
worker threads can run simultaneously, and it applies only for a discovery server in a streaming server deployment or a domain server in a domain server deployment.

For discovery using IBM Tivoli Monitoring (old method that uses IBM Tivoli Monitoring Scope sensor), the value must be set to 16. For all other types of discovery, the valid range of values is 32 - 160.

**com.collation.discover.observer.topopumpcount=16**

The default value is 16. The value must be an integer.

This property influences the rate at which discovery results are stored in the TADDM database. It specifies the number of writer threads that are created for communicating with the TADDM database.

For a discovery server in a streaming server deployment, this property controls the number of threads that the discovery server uses to send discovery results to the storage server pool.

For a storage server in a streaming server deployment, this property controls the number of threads that receive discovery results from the discovery servers.

For a domain server in a domain server deployment, this property controls the number of threads that receive discovery results from the discover worker threads.

The threads then use a database connection from the connection pool to communicate with the TADDM database (for example, to store results and to retrieve data). If no more pooled JDBC connections exist, the thread creates a non-pooled connection.

**com.ibm.cdb.discover.observer.topopump.threshold=0.7**

**com.ibm.cdb.discover.observer.topopump.threshold.topo_agent_grp_name=0.7**

The default value is 0.7. The value must be a floating constant.

This property specifies the fraction of database writer threads that you can start when the topology agents are running. You can specify the threshold value separately for a particular agent group or all of them at once. If the value is not defined for agent group, the general threshold value is used. This value allows limitation of threads that store the discovery results in the TADDM database when the topology agents are running.

**com.ibm.cdb.typesServiceRefreshInterval=120**

The default value is 120. The minimum value is 30, and the maximum value is 1800.

This property specifies, in seconds, the refresh interval for updating component types when you create a custom query, display a change history, or display component comparison information.

**com.ibm.cdb.ea.metaRefreshFrequency=20**

The default value is 20. The value must be an integer.

This property specifies, in seconds, the refresh interval for updating information about the defined extended attributes, for example in the storage servers.

### Secure Shell (SSH) properties

These properties apply to the Secure Shell (SSH).
com.ibm.cdb.platform.SshVersionSessionSkipList
This property specifies the versions of SSH servers for which the session is not established. In case of such servers, the session sensor is finished without failure.

The value of this property is a comma-separated list, for example Cisco, Data ONTAP, SSH-2.0-OpenSSH_5.9 PKIX FIPS, OpenSSH_0A.

com.collation.SshLogInput=false
The default value is false.

Valid values are true or false. If you set the value to true, SSH input is logged.

com.collation.SshPort=22
The default value is 22. The value must be an integer.

This property indicates the port that the server uses for all SSH connections.

com.collation.SshSessionCommandTimeout=120000
The default value is 120000. The value must be an integer.

This value indicates the time (in milliseconds) that is allowed for the SSH command to run. If this property is used from an agent, the value for this property must be a lesser value than the value for the AgentRunnerTimeout property to be effective.

com.collation.SshWeirdReauthErrorList=Permission denied
This property allows for the retry of the user name and password pairs that previously worked during discovery runs. The property is needed because Windows systems randomly deny valid login attempts. The property needs to have the Permission denied setting. Do not change this property.

com.collation.WmiInstallProviderTimeout=240000
The default value is 240000. The value must be an integer.

This value indicates the time (in milliseconds) that is allowed to wait for the WMI InstallProvider script to run.

com.collation.SshSessionReuseSuppressList
Some versions of the SSH server do not support the reuse of connections as implemented by TADDM. The SSH server versions that are not supported for reuse must be added to this property so that TADDM successfully discovers targets that run those SSH server versions.

The value of this property is a comma-separated list. It is sufficient to specify only the beginning of the SSH server version, for example SSH-2.0-BoKS_SSH_6.

You can find the SSH server version in the session sensor log file.

Security properties
These properties apply to security.

com.ibm.cdb.secure.server=false
The default value is false.

This property specifies whether all TADDM services from the public and external RMI registries are secure. If set to true, all public services that are not secure (ClientProxyServer and API Server) are moved to the internal
RMI registry. Also, the SSL protocol is enforced on external services, for example, RegistriesURLProvider, SecurityManager, and TopologyManager.

If you set this property to true, set also the
com.collation.security.enablesslforconsole and
com.collation.security.enforceSSL properties to true.

This property might affect the integration with other products that connect to TADDM with unsecured connection.

If you modify the default value of this property, set it in the following locations:

• $COLLATION_HOME/dist/etc/collation.properties
• $COLLATION_HOME/dist/sdk/etc/collation.properties
• sdk/etc/collation.properties of every TADDM SDK installation.

**Fix Pack 5**  
com.ibm.cdb.rmi.registry.secure=false

The default value is false.

Valid values are true or false. To enable secure registry mode set this flag to true.

If the server is running in Secure Registry mode
(com.ibm.cdb.rmi.registry.secure=true), the following port will be secured with SSL protocol:
com.ibm.cdb.service.registry.public.port (Default Value: 9433)

If server is running in Secure Registry mode
(com.ibm.cdb.rmi.registry.secure=true), then while launching the Data Management Console, check box 'Establish a secure (SSL) session' must be checked.

**Fix Pack 1**  
com.ibm.cdb.secure.liberty=false

The default value is false.

Valid values are true or false. To disable the non-secure HTTP port, set this flag to true.

com.collation.security.privatetruststore=true

The default value is true.

Valid values are true or false. The value must be true when SSL is enabled.

com.collation.security.enablesslforconsole=true

The default value is true.

Valid values are true or false.

com.collation.security.enabledatalevelsecurity=false

The default value is false.

Valid values are true or false. To restrict access to collections of TADDM objects by user or user group, set this value to true.

com.collation.security.enforceSSL=false

The default value is false.

Valid values are true or false. To disable non-secure connections and force the use of SSL connections, set this flag to true.

com.collation.security.usermanagementmodule=file

The default value is file.
There are three options for this property:

- file for a TADDM file-based user registry
- ldap for an LDAP user registry
- vmm for a user registry that uses the federated repositories of WebSphere Application Server

**com.collation.security.auth.sessionTimeout=240**

The default value is 240. The value must be an integer.

**com.collation.security.auth.searchResultLimit=100**

The default value is 100. The value must be an integer.

Use this property if you have many users.

**Important:** If you have more than 100 users in an LDAP or WebSphere Federated repository, increase this value to support the expected number of users. For example, com.collation.security.auth.searchResultLimit=150

**com.collation.security.auth.websphereHost=localhost**

The default value is localhost.

Type the fully qualified domain name of the system that hosts the federated repositories functionality of the WebSphere Application Server.

**com.collation.security.auth.webspherePort=2809**

The default value is 2809.

It must be an integer value. This value indicates the WebSphere system port.

**com.ibm.cdb.service.SecurityManager.port=9540**

For servers other than a synchronization server:

The default value is 9540.

Specifies the firewall port that is used by the security manager.

For a synchronization server:

The default value is not set.

Domains communicate with a synchronization server by using a port that is specified in the **com.collation.EnterpriseSecurityManager.port** parameter. The default value for this property is 19433.

**com.collation.cdm.analytics.authorizedRole=**

The Analytics pane can be restricted to a specific role. By default, this property is not defined in the collation.properties file and the Analytics pane is available for everyone. The value of the property must be the name of the role that is allowed to access the pane.

The access to the following areas of the Analytics pane can be subject to the specified role:

- Fix Pack 2 Grouping Patterns
- Inventory Summary
- Application Summary
- Service Summary
- System Inventory
- Software Server Inventory
- BIRT Reports
com.collation.security.discoverOutsideScope=true
The default value is true.
Valid values are true or false. To disable discovering elements which are not inside the scope, set this flag to false.

com.ibm.cdb.security.tomcat=false (TADDM 7.3.0 only)
The default value is false.
Valid values are true or false. To disable the non-secure HTTP port, set this flag to true.

com.ibm.cdb.http.ssl.protocol=TLS
The default value is TLS.
This property modifies the SSL protocol that is used by Web SSL port (HTTPS port), by default 9431. You can set the port by using the com.ibm.cdb.service.web.secure.port property.
For the list of supported values, see IBM Java 7 documentation at http://www-01.ibm.com/support/knowledgecenter/SSYKE2_7.0.0/com.ibm.java.security.component.70.doc/security-component/jsse2Docs/protocols.html If you use the most secure protocols, for example TLS v1.1 or TLS v1.2, you must first configure your web browser to support them. Additionally, too strong protocols might affect integration with other products that connect to TADDM through Web SSL port.

Fix Pack 5 When com.ibm.cdb.http.ssl.protocol=TLSv1.2 and JAVA7 is being used at the client side, the following settings need to be updated:
<jAVA_HOME>/jre/lib/security/java.security
jdk.tls.disabledAlgorithms=SSLv2, SSLv3, TLSv1, TLSv1.1
Also TLSv1 and TLSv1.1 should be disabled in the browser.

com.ibm.cdb.ssl.protocol=TLS
This property is not added to the collation.properties file by default. If it is not added, the default value is TLS. To modify it, add this property to the collation.properties file manually with the new value.
This property modifies the SSL protocol that is used by the following ports:
• The port that the API server listens on for SSL requests, by default 9531. You can set the port by using the com.ibm.cdb.service.SecureApiServer.secure.port property.
• The RMI data port to use with SSL, by default 9434. You can set the port by using the com.ibm.cdb.service.SecureClientProxyServer.secure.port property.
For the list of supported values, see IBM Java 7 documentation at http://www-01.ibm.com/support/knowledgecenter/SSYKE2_7.0.0/com.ibm.java.security.component.70.doc/security-component/jsse2Docs/protocols.html If you use the most secure protocols, for example TLS v1.1 or TLS v1.2, you must first configure your web browser to support them. Additionally, too strong protocols might affect integration with other products that connect to TADDM through the listed ports.

com.ibm.cdb.http.ssl.ciphers=
Ciphers are getting set to the LibertyServer and communication will be done on the given ciphers only. Otherwise it will pick the default ciphers which could be the weak algorithms.
**com.ibm.cdb.rmi.ssl.protocol**

This property `com.ibm.cdb.rmi.ssl.protocol` helps to enable specific protocol on SSL Connection which was created on `com.ibm.cdb.ssl.protocol`.

`com.ibm.cdb.rmi.ssl.protocol` must be from supported protocol list on `com.ibm.cdb.ssl.protocol`.

**com.ibm.cdb.rmi.ssl.ciphers**

With this property you can set the ciphers algorithms for RMI data port and port on that API server listens.

### Temporary directory properties

These properties apply to the use of temporary directories.

Temporary directories are used by TADDM to store temporary files under certain conditions. For example, anchor log files, discovery scripts, discovery results, and information required by some sensors when running a discovery can be stored in temporary directories. TADDM uses three temporary directories: **ANCHOR_DIR**, **ASD_TEMP_DIR**, and **TADDM_TEMP_ROOT**.

**com.ibm.cdb.taddm.anchor.root**=

The default value is `.\`

This entry specifies the location of the **ANCHOR_DIR** directory where the anchor server is deployed. This property is a scoped property, and you can append the IP address, name of the scope, or operating system to this property. For example, `com.ibm.cdb.taddm.anchor.root.SunOS=`.

For a Windows system, the following property name and default value are used:

```
com.ibm.cdb.taddm.anchor.root.Windows=%windir%\temp\taddmdirs\anchor
```

The property value uses variables that are resolved on target hosts. The Linux, AIX, and SunOS variables must be prefixed with a dollar sign ($). Variables for Windows must be enclosed in percent signs (%). For example, `com.ibm.cdb.taddm.anchor.root=$TMP/taddmdirs/anchor` and `com.ibm.cdb.taddm.anchor.root.Windows=%TEMP%\taddmdirs\anchor`.

If the resolved property value is a relative directory path, it is prefixed with:

- `%windir%\temp\` - for Windows
- Home directory - for AIX, Linux, and SunOS systems

The path is suffixed with the `taddmversion/anchor` directory. For example, `/home/taddmusr/taddm7.2.1/anchor` and `c:\Windows\Temp\taddm7.2.1\anchor`.

**com.ibm.cdb.taddm.asd.temp**

This entry specifies the location of the **ASD_TEMP_DIR** directory and this directory stores discovery scripts and discovery results. This property is a scoped property, you can customize the property by appending the IP address or operating system to this property.

In the specified location, `taddmversion/asd/` directory is created. For example, `/tmp/taddm7.2.1/asd/`. If you specify a new location, all users must have all access rights to the new location.

**com.ibm.cdb.taddm.file.temp**=

The default value is `.\`

---

92 Application Dependency Discovery Manager: Administering
This entry specifies the location of the TADDM_TEMP_ROOT and this directory is used by various sensors to store temporary data that is necessary for running a discovery. Examples of sensors that store temporary data are DB2® and WebLogic sensors.

The TADDM_TEMP_ROOT directory is created in the home directory in taddmversion/temp/. For example, /home/taddmusr/taddm7.2.1/temp/.

**Topology builder properties**

These properties apply to the topology builder.

**com.collation.topobuilder.RuntimeGcUnknownServerRetentionSpan=5**

The default value is 5.

This property specifies how long (in days) to keep unknown processes. The maximum value is 14. Unknown processes determine when custom server templates are needed, however, without regular clean up, the number of unknown processes can build up over time. This might cause topology performance issues. The zOS Address Spaces item is not removed by this processing.

**com.collation.topobuilder.RuntimeGcThreadCount=**

The default value is 4.

This property adds parallelism to the RuntimeGC agent which can improve performance

**com.collation.topobuilder.agent.DerivedAppToAppDependencyAgent.ServiceDependency.enabled**

The default value is false.

This property specifies whether the topology agent DerivedAppToAppDependency creates a dependency between business applications when their members are in service dependency.

To enable the agent to create such dependency, set the property to true.

**Topology manager properties**

These properties apply to the topology manager.

**com.ibm.JdoQuery.FetchBatchSize=500**

The default value is 500.

The batch size is a configurable property and corresponds to the kodo.FetchBatchSize property. This property represents the number of rows to fetch at a time when scrolling through a result set of a query run.

**com.ibm.cdb.service.TopologyManager.port=9550**

The default value is 9550.

Specifies the firewall port used by the topology manager.

**View manager properties**

These properties apply to the view manager.

**Fix Pack 2**

**com.ibm.taddm.hideNetworkConnectionUnusedColumns.enabled**

The default value is false.

This property specifies whether the following columns of the Network Connections tab are displayed in Data Management Portal:

- Flows
- Packets
Verifying data integrity

You can run the `verify-data` command to verify the data integrity of configuration items in the TADDM database. You can verify relationships, inheritance mapping, duplicates, and over merges.

Before you begin

Do not run a discovery, bulkload, or synchronization with the repair option enabled. The data integrity tool analyzes a great amount of data and it might take some time to finish the process, especially with the repair option enabled. The TADDM server must be up and running but make sure that it does not perform any tasks.

About this task

The Data integrity verification tool reports and repairs configuration items data integrity problems in the TADDM database. The executable script is in the `$COLLATION_HOME/bin` directory. The tool reports and logs to the `verify-data.log` file. You can stop the tool and run it again at any time.

Verifying relationships

Verification for relationships queries and verifies foreign keys in all model and intersection tables.

About this task

With the repair option enabled, the verification for relationships removes child objects if a parent object does not exist in the database, and clears invalid foreign key values for relationships that are defined as being not contained. It might also delete a significant number of low-level configuration items. However, if the items do not have a parent object, they can be safely removed.

Procedure

To verify relationships, run one of the following commands:

- `verify-data.sh -v ro [-a repair]`
- `verify-data.bat -v ro [-a repair]`

Verifying inheritance mapping

Verification for inheritance mapping queries all tables that map a configuration item class, and verifies that all tables contain an entry for each row.
About this task

When the repair option is enabled, the records are re-created.

Procedure

To verify inheritance mapping, run one of the following commands:

- `verify-data.sh -v io [-a repair]`
- `verify-data.bat -v io [-a repair]`

Verifying duplicates

Verification for duplicates searches for duplicated configuration items based on naming rule field values in the database.

About this task

With the repair option enabled, the duplicated objects are merged. After the merge, the durable object remains in the database, and the transient object is deleted.

The merge is performed by several threads in parallel. The default number of threads is 5. You can change the number of threads in the `collation.properties` file by setting the `com.ibm.cdb.topomgr.dataverification.generator.ThreadCount` flag to an appropriate number, as in the following example:

- `com.ibm.cdb.topomgr.dataverification.generator.ThreadCount=10`

You must restart the TADDM server after you change the number of threads.

Some errors might occur during the merge of the objects. The cause of the errors is included in a log file.

- `ERROR_INVALID_DURABLE_GUID`
- `ERROR_INVALID_TRANSIENT_GUID`

The cause of the errors are missing aliases in the aliases table or an invalid object. You must wait for the cleanup agents to delete the invalid objects.

Procedure

To verify duplicates, run one of the following options:

- `verify-data.sh -v dup [-a repair]`
- `verify-data.bat -v dup [-a repair]`

Verifying over merges

Verification for over merges uses the data that is gathered in the ALIASES_JN table to find and report GUIDs with large numbers of master alias changes.

About this task

The ALIASES_JN table contains history of changes to the ALIASES table. Over merge is a situation when a few objects change their parent to the same model object. Child objects are then clustered around some number of parent objects. Over merges that occurred before the TADDM 7.2.1 Fix Pack 3 was installed cannot be found because there is no required data in the ALIASES_JN table. Verification does not have the repair option because it might find and report false positive results.
By default, the detailed tracking is enabled for the ComputerSystem, AppServer, and Operating System classes, and all other classes that inherit from them. If you want to enable tracking for different classes, you can edit the following property in the collation.properties file:
com.ibm.tivoli.namereconciliation.service.overmergeClasses

The following is the example of the property that is specified to look for the ComputerSystem, AppServer, and Operating System classes:
com.ibm.tivoli.namereconciliation.service.overmergeClasses=ComputerSystem,AppServer,OperatingSystem

Meaning of the actions that are used to run the command:
• s1s2s1 - Verification looks for CIs that change their naming attributes values in a loop. For example, a computer system with a signature A, then signature B, and then again signature A would be detected.
• s1s2s3 - Verification looks for CIs that contain a number of changes for given naming attributes.
• m1m2m1 - Verification looks for CIs which GUIDs changed their master GUID many times. For example, an alias A with a master GUID B that was later reassigned to master GUID C, and then again reassigned back to master GUID B would be detected.
• m1m2m3 - Verification looks for CIs which GUIDs changed their master GUIDs a few times.
• WinCSLinCSWinCS - Verification looks for CIs that changed their type a few times. For example, a computer system that was initially stored as WindowsComputerSystem, and later updated to LinuxUnitaryComputerSystem, and then again updated to WindowsComputerSystem would be detected.

Procedure

To verify over merges, run one of the following commands:
• verify-data.sh -v om [-a <action>] [-p <class>] [-from <time stamp>] [-to <time stamp>]
• verify-data.bat -v om [-a <action>] [-p <class>] [-from <time stamp>] [-to <time stamp>]

where:
– <action>: s1s2s1, s1s2s3, m1m2m1, m1m2m3, WinCSLinCSWinCS
– <class>: any class from the TADDM model, for example, ComputerSystem.
– <time stamp>: time stamp in the YYYY-MM-DD HH24:MI:SS format.

Example

verify-data.sh -v om -a s1s2s1 m1m2m1 WinCSLinCSWinCS
-p ComputerSystem -from 2012-11-13 14:50:00 -to 2012-11-14 14:50:01

This command looks for over merges of type s1s2s1, m1m2m1, and WinCSLinCSWinCS for the ComputerSystem class, and all classes that inherit from it, created between 2012-11-13 14:50:00 and 2012-11-14 14:50:01.

Solving the over merge issue:

An over merge occurs when a few objects change their parent to the same model object. Child objects are then clustered around some number of parent objects.
Procedure
1. Run the over merges verification.
2. Check the reported configuration items. Verification might incorrectly report them as over merges.
3. Correct the configuration in environments that might be the cause of over merge. The configuration problems might include the same signature, serial number, VMID, and other CIs naming attributes.
4. Remove the over merged objects from the TADDM database.
5. Run a discovery on deleted objects, and validate the results.
6. Remove all records from the ALIASES_JN table after solving any over merge problems.

Managing credential cache - cachemgr utility
You can use the cachemgr.sh or cachemgr.bat command to list and delete the content of credentials cache.

Command syntax

```
cachemgr -h | -u user -p password (-l|-r) valid|invalid|all [[ -s IP|scope|scope group|range|subnet ] [ -a addressSpace ] [ -n accessCredentialName ] [ -c type ] [ -d yyyy/mm/dd ] [ -k key ] [ -t locationTag ]
```

Parameters

- `-a`, `--addressSpace addressSpace`
  Is the address space name.

- `-c`, `--class type`
  Is the type of a selected access entry that is described by the name of the specific class that implements access entry.

- `-d`, `--date yyyy/mm/dd`
  Is the date threshold that is used to select entries not modified until specified time. The format is yyyy/mm/dd.

- `-h`, `--help`
  Shows help.

- `-k`, `--key key`
  Is the key of a selected cache entry.

- `-l`, `--list valid|invalid|all`
  Is the listing operation that is controlled by the following arguments:
  - `valid` - lists only valid authentication attempts held in a cache.
  - `invalid` - lists only invalid authentication attempts held in a cache.
  - `all` - lists both valid and invalid authentication attempts held in a cache.

- `-n`, `--name accessCredentialName`
  Is the name of access credentials, the same as in the access list.

- `-p`, `--password password`
  Is the password for the user that logs in to the TADDM server.

- `-r`, `--remove valid|invalid|all`
  Is the remove operation that is controlled by the following arguments:
  - `valid` - removes only valid authentication attempts held in a cache.
  - `invalid` - removes only invalid authentication attempts held in a cache.
- all - removes both valid and invalid authentication attempts held in a cache.

-s, --scope IP|scope|scope group|range|subnet
Is the scope of an access entry. It is controlled by the following arguments:

- IP
- scope
- scope group
- range
- subnet

-t, --locationTag locationTag
Is the location tag of a selected access entry.

-u, --username username
Is the user that logs in to the TADDM server.

Examples

- The following command lists all invalid authentication attempts for computers in the scope "ScopeSet":
  ```
cachemgr.sh -u user -p password -l invalid -s ScopeSet
  ```

This command generates the following output:

Following entries are matching provided criteria:

CachedAuthEntry
guid: 3B954CE4CFB346C8DF538F09F1F7FFD
lastModified: Thursday, 5 September 2013 11:00:38
Authorization: invalid. Error message: CTJTP1190E The server did not complete the authorization process.
CachedAuthEntry
guid: ACC2F35A66D3379BAC13FC6C606C5A08A3
lastModified: Thursday, 5 September 2013 11:00:38
Authorization: invalid. Error message: CTJTP1190E The server did not complete the authorization process

- The following command deletes invalid authentication attempts in IP range 9.123.149.10 - 9.123.149.12 and the access entry com.collation.platform.security.auth.HostAuth:
  ```
  ```

This command generates the following output:

AuthEntries removed from cache successfully (2).

Cachemgr utility return codes

If you write a cron script or some other script that calls the cachemgr utility, the following return codes indicate how the program exited.

0 Program completed successfully.

1 An invalid command line parameter was supplied. Either the parameter itself or the data that is supplied with the parameter is incorrect. Correct the command and try again.

2 A date command line parameter was not in the expected format.

3 Either provided scope definition does not resolve to any IP address or provided access entry is not valid.
Preparing for discovery

To optimize the information that TADDM gathers from your environment during discoveries, you must complete configuration tasks to prepare your environment for discovery.

About this task

The specific configuration tasks are dependent on the type and level of discovery that you must support in your environment.

What to do next

In addition to configuring your environment for discovery, you must configure TADDM sensors as appropriate. For information about how to do this, see the TADDM Sensor Reference.

For information about how to run a discovery, including defining a scope and setting a schedule, see the TADDM User’s Guide.

Configuring user logon ID

TADDM requires an interactive user to successfully run discoveries. You must configure the user logon ID.

An interactive user logon ID is used in a non-interactive mode for all discovery sessions, including a server to gateway session and a gateway to target session. The user must be interactive to be able to run commands. However, the commands are run in a non-interactive way, which means that a user runs the command and waits for the results.

In /etc/passwd, set the user in the following way:

taddmusr:x:100:100::/export/home/taddmusr:/bin/sh

where taddmusr is the TADDM user name.

Configuring for alternative methods of discovery

You might want to use alternative methods of discovery such as asynchronous discovery, script-based discovery, or discovery using IBM Tivoli Monitoring.

Notes:

1. Asynchronous and script-based discovery is supported only if the target computer system is running the AIX, FreeBSD, HP NonStop, Linux (on x86 systems only), Solaris, or Windows operating system.
2. If the target computer system is running the Solaris operating system, script-based discovery might not work if SunSSH 1.0 is used.
Configuring for asynchronous discovery
To run an asynchronous discovery, you must first configure the discovery.

About this task
To configure for asynchronous discovery, you must generate a discovery script package, copy the package to the target system, and run the script on the target system. The output of the discovery script is an archive file that contains the discovery results. You must then move this archive file to the TADDM server.

Note: If you configured the discovery to run in the asynchronous mode and then upgraded TADDM, you must generate a discovery script package again because the sensor plug-in ID might change.

Procedure
1. To generate a discovery script package, enter one of the following commands from the $COLLATION_HOME/bin directory:
   
   Regular method
   ```
   makeASDScriptPackage OUTPUT_DIR UNAME [IPADDRESS] [PACKING_METHOD]
   ```

   **OUTPUT_DIR**
   The directory path for the script package.

   **UNAME**
   The operating system of the target system, on which the script is to be run. The valid values are AIX, Linux, SunOS, FreeBSD, Windows, or NONSTOP_KERNEL.

   **IPADDRESS (optional)**
   The IP address of the target system on which the script is to be run.

   The scripts that are used for asynchronous discovery use information from TADDM server properties that are defined in the collation.properties file, and some of these properties might be scoped.

   **scoped property**
   A property to which you can append either an IP address or the name of a scope set. The IP address or the scope set name makes the property dependent on the host that is being discovered. You can use only scope set names that do not contain spaces, apostrophes (’), periods (.), and forward slashes (/).

   If you have customized any of the TADDM server properties so that they are scoped, you should include the IPADDRESS option in the makeASDScriptPackage command.

   **PACKING_METHOD (optional)**
   The method to use to package the files. The valid values are tar or zip.

   If no method is specified, the method is determined by the operating system. For example, for operating systems such as Linux, the tar method is used.

   By default, the system path is searched for the archive utility. If necessary, add the com.ibm.cdb.tarpath property to the collation.properties file, and specify an alternative path for the archive utility.
On Solaris operating systems, because of a limitation in the length of file names, you must use the gtar archive utility, and you must specify the path to the utility.

The following example shows how to specify the path of the tar command on the TADDM server for the AIX operating system:

```bash
com.ibm.cdb.tarpath=tar
```

The following examples show how to specify the path of the tar command on the target system, depending on the operating system:

For AIX

```bash
com.ibm.cdb.targettarpath.AIX=tar
```

For Solaris

```bash
com.ibm.cdb.targettarpath.SunOS=/usr/sfw/bin/gtar
```

For example, to generate a discovery script package for the AIX operating system, enter the following command:

```bash
./makeASDScriptPackage /tmp AIX
```

This command creates the following AIX script package in the tmp directory:

```bash
/tmp/taddm_AIX.tar
```

**Extended method**

```bash
makeASDScriptPackage --outputDir OUTPUT_DIR --uname UNAME
[--ipAddress IP_ADDRESS] [--packingMethod PACKING_METHOD] [--sensors SENSOR]
```

---

**--outputDir OUTPUT_DIR**

See the description of the OUTPUT_DIR parameter of the regular method.

**--uname UNAME**

See the description of the UNAME parameter of the regular method.

**[--ipAddress IP_ADDRESS] (optional)**

See the description of the IPADDRESS parameter of the regular method.

**[--packingMethod PACKING_METHOD] (optional)**

See the description of the PACKING_METHOD parameter of the regular method.

**[--sensors SENSOR] (optional)**

The name of the sensor that you want to include in your package. The following table contains the sensor names that must be used in this command.

**Table 33. Sensor names used in the makeASDScriptPackage command.**

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Name used in the command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache sensor</td>
<td>apacheserver</td>
</tr>
<tr>
<td>Citrix XenServer sensor</td>
<td>xenserver</td>
</tr>
<tr>
<td>FreeBSD computer system sensor</td>
<td>computersystem</td>
</tr>
<tr>
<td>Generic server sensor</td>
<td>genericserver</td>
</tr>
<tr>
<td>HP NonStop computer system sensor</td>
<td>computersystem</td>
</tr>
<tr>
<td>IBM AIX computer system sensor</td>
<td>computersystem</td>
</tr>
<tr>
<td>IBM DB2 sensor</td>
<td>db2</td>
</tr>
<tr>
<td>IBM Lotus® Domino® server sensor</td>
<td>dominoserverinitial</td>
</tr>
</tbody>
</table>

**Administering**

101
Table 33. Sensor names used in the `makeASDScriptPackage` command. (continued)

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Name used in the command</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Tivoli Utilization sensor</td>
<td>utilization</td>
</tr>
<tr>
<td>IBM WebSphere MQ Server sensor</td>
<td>mqserver</td>
</tr>
<tr>
<td>IBM WebSphere sensor</td>
<td>webserverscript</td>
</tr>
<tr>
<td>JBoss Application Server 7 sensor</td>
<td>jboss7</td>
</tr>
<tr>
<td>KVM sensor</td>
<td>kvm</td>
</tr>
<tr>
<td>Linux computer system sensor</td>
<td>computersystem</td>
</tr>
<tr>
<td>Microsoft Exchange sensor</td>
<td>exchange</td>
</tr>
<tr>
<td>Microsoft IIS Web server sensor</td>
<td>iisserver</td>
</tr>
<tr>
<td>Oracle sensor</td>
<td>oracle</td>
</tr>
<tr>
<td>Solaris computer system sensor</td>
<td>computersystem</td>
</tr>
<tr>
<td>WebLogic SSH sensor</td>
<td>weblogiclaunchersensor</td>
</tr>
<tr>
<td>Windows computer system sensor</td>
<td>computersystem</td>
</tr>
</tbody>
</table>

The Asynchronous discovery sensor is added to every package by default. All operating system sensors have `computersystem` name. They are differentiated on the basis of the `--uname` parameter. For example, if you specify the following parameters:

```
[...] --uname Linux --sensors computersystem
```

Linux computer system sensor is added to the package.

For example, to generate a discovery script package for the AIX operating system, enter the following command:

```
./makeASDScriptPackage --outputDir /tmp --uname AIX --sensors computersystem
```

This command creates the following AIX script package in the `tmp` directory:

```
/tmp/taddm_AIX.tar
```

2. Copy the script package from the `OUTPUT_DIR` to the target system, and extract the script package.

3. As a root user on UNIX systems, or administrator on Windows system, grant execute privileges to all script files. If the discovery script is run as a non-root, or non-administrator user, some sensor scripts might not complete a successful discovery, or the data that the sensor discovers might be limited.

4. Run the `scriptsRunner.sh` script for the UNIX targets, or the `scriptsRunner.bat` for the Windows target.

5. Move the resulting archive file (for example, `/tmp/taddm${version}/asd/taddmasd-${hostname}-${execution timestamp}.tar`) to the TADDM server in the location that is defined by the `com.ibm.cdb.discover.asd.AsyncDiscoveryResultsDirectory` property in the `collation.properties` file.


7. Ensure that the asynchronous discovery sensors (ASDPingSensor and ASDSensor) are enabled in your discovery profile. By default, only the ASDSensor is enabled in the Level 2 and Level 3 discovery profiles.

8. Create a scope with the IP address of the target system.
What to do next

Run the discovery. You do not need root authority to run this discovery.

During the discovery, if the ping, port, or session sensor cannot access the target system, the target system is determined to be unreachable. If the value of the `com.ibm.cdb.discover.asd.ProcessUnreachableIPs` property is set to true, the asynchronous discovery sensor is run to process the discovery archive file for the target system. The archive file is processed only if the IP address from the discovery scope matches the IP address of the system that produced the archive file. Based on the contents of the archive file, sensors will be scheduled to process their script output. After the archive file is processed, it is renamed to `tarfilename.tar_DONE` so that it is not processed again.

The discovery archive file is processed only once. If a sensor is not enabled to process its script output at the time that the archive file is processed, running a second discovery with the sensor enabled does not process a previously processed archive file, unless you complete the following steps:

1. Rename the archive file to its original name. For example, remove _DONE from the file name.
2. The .processed file in the $COLLATION_HOME/var/asdd directory contains a list of the processed archive files. Remove the name of the archive file from the .processed file.

Multiple archive files from different systems can be processed in a single discovery run, but only one archive file per target system is processed during a single discovery run. If one target system has multiple archive files, only the one with latest time stamp is processed.

To discover multiple archive files from different systems in a single discovery run, copy each archive file to the location that is defined by the `com.ibm.cdb.discover.asd.AsyncDiscoveryResultsDirectory` property. Include the IP address of each target system in the discovery scope.

Because the discovery script uses the `tar` command to create the discovery archive file, if you are using a TADDM server that is running the Windows operating system, you must install a third party tar program for TADDM to use to extract the files from the archive file. The location of the tar program is defined by the `com.ibm.cdb.tarpath` property.

For reference, 'bsdtar' third party tar implementation is found to be supported when configured through above mentioned property for TADDM server running on Windows operating system.

**Restriction:** Your tar program must support long file paths. GNU Tar 1.13 is not supported because it might truncate long file names.

The process of manually starting discovery can be automated with following properties:

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.ibm.cdb.discover.asd.autodiscovery.enabled</td>
<td>True/false</td>
<td>If true, it will enable the option to process the stored ASD results files. Default value is false.</td>
</tr>
<tr>
<td>Property Name</td>
<td>Possible Values</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.autodiscovery.asdScope</td>
<td>&lt;scope name&gt; Default = ASD</td>
<td>The thread will pick the target mentioned in this scope to process result file. If this property is not mentioned, default ASD scope is processed.</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.autodiscovery.asdProfile</td>
<td>&lt;profile name&gt; Default = ASD</td>
<td>The thread will pick the sensors mentioned in this profile to process result file. If this property is not mentioned, default ASD profile is processed.</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.autodiscovery.filesThreshold</td>
<td>&lt;file threshold&gt; Default=20</td>
<td>Minimum Number of files required by thread, to start processing them. The thread will process the result if either the File threshold or Time threshold is met.</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.autodiscovery.timeThreshold</td>
<td>&lt;time threshold&gt; Default=60 (seconds)</td>
<td>Time threshold (in seconds) after which thread will process the result files, even if File Threshold is not met.</td>
</tr>
</tbody>
</table>

**Configuring for script-based discovery**

To run a script-based discovery, you must first configure the discovery.

**About this task**

In comparison to regular sensors, sensors that are run in the script-based mode are more apparent, which means that all commands that the sensor uses are in one script, which you can view. For the list of the sensors that support script-based mode, and the restrictions that apply to some of the sensors, see the *Sensors that support script-based and asynchronous discovery* topic in the TADDM Sensor Reference.

**Procedure**

Configure the sensor in one of the following ways:

- **Enabling all sensors that support script-based discovery**
  
  To globally enable all sensors that support script-based discovery, open the *collation.properties* file and set the value of the `com.ibm.cdb.discover.PreferScriptDiscovery` property to true.

- **Enabling all sensors that support script-based discovery in a specific discovery profile**
  
  For a specific discovery profile, to enable all sensors that support script-based discovery, complete the following steps:

  1. In the Discovery Management Portal, select the discovery profile for which you want to enable script-based mode.
  2. In the *Platform Properties* tab, set the value of the `com.ibm.cdb.discover.PreferScriptDiscovery` property to true.
Enabling a sensor that supports script-based discovery in a discovery profile

In a discovery profile, to enable a specific sensor that supports script-based discovery, update the configuration of that sensor in the respective discovery profile. Complete the following steps:

1. In Discovery Management Portal, go to the discovery profile that contains the sensor that you want to enable.
2. In the Sensor Configuration tab, select the sensor and click New.
3. In the Create Configuration window, specify the name of the configuration and select the Perform Script Based Discovery option.
4. Click OK to save the configuration.

What to do next

Configuring TADDM to select non-default users for the discovery

By default, only the user that is requested by the script is used for the discovery. If you have problems with running a discovery with the default user and you have another user that has all required permissions, you can configure TADDM to select this user for the discovery.

Note: Use the following configuration with caution. If a user that does not have all required permissions is used for a discovery, the discovery might fail, or some of the targets might not be discovered.

In the plugin.xml file that you can find in a package for each sensor in the COLLATION_HOME/osgi/plugins directory, edit the script node definition, for example, like in the plugin.xml snippet of IBM WebSphere MQ Server sensor:

```xml
<scriptset>
  <ostype>AIX</ostype>
  <mainScript name="sensorCommon.sh" />
  <script name="script.sh" authClassName="com.collation.platform.security.MQServerAuth" authMode="preferred" hostAuthFallback="true"/>
</scriptset>
```

The following properties can be defined:

**authMode**

Defines how TADDM approaches entries in the Access List for the type that is specified by authClassName. The following values are available:

- **single** - only a user that is requested by the script is used. This is the default value.
- **preferred** - first a user who is preferred by the script is used, but if it is not available or fails, the remaining Access List entries of the defined type are used.
- **regular** - Access List entries are used in their specified order without checking the user preference.

**hostAuthFallback**

Defines whether, in case of problems with establishing connection to the target for a specific authClassName or the preferred user, or both, TADDM falls back to the session that is established by the generic user that is used for connecting to the target. The following values are available:

- **false** - the default value.
- **true**.
Configuring for discovery using IBM Tivoli Monitoring (old method)

TADDM can perform Level 1, Level 2, and some Level 3 discoveries using IBM Tivoli Monitoring 6.2.1 or later infrastructure.

The old integration method

This section applies to a deprecated method of TADDM integration with IBM Tivoli Monitoring. Starting with the TADDM version 7.3.0, it is advised to integrate with IBM Tivoli Monitoring 6.3 using OSLC Automation. The old method of integration with the use of IBM Tivoli Monitoring Scope sensor is deprecated and will be removed from future releases. For more information about configuring for discovery by using OSLC Automation, see “Configuring for discovery over OSLC Automation Session” on page 107.

If you are using IBM Tivoli Monitoring 6.2.1-TIV-ITM-FP0001, 6.2.2-TIV-ITM-FP0002, or a later level, you can discover Tivoli Monitoring endpoints through the Tivoli Enterprise Portal Server. These fix packs resolve APAR IZ63983, which improves Tivoli Monitoring performance during TADDM discoveries. Using earlier releases or levels of IBM Tivoli Monitoring to perform TADDM discoveries through the Tivoli Enterprise Portal Server might cause excessive processor and network load, especially on Tivoli Monitoring components.

Note: Discovery using IBM Tivoli Monitoring is only possible when the Tivoli Enterprise Portal Server database is on Microsoft SQL Server and DB2. It is not possible when Apache Derby database is used as the Tivoli Enterprise Portal Server database.

TADDM server properties that are specific to discovery using Tivoli Monitoring

For information about the TADDM server properties that are specific to discovery using IBM Tivoli Monitoring, including properties that affect how TADDM discovers Tivoli Monitoring endpoints, see “Properties for discovery using IBM Tivoli Monitoring (old method)” on page 77.

In a discovery profile, you can configure the TADDM server properties that affect how TADDM discovers Tivoli Monitoring endpoints. To do this, complete the following steps, depending on whether you are using a custom profile or the default profile:

Configuring the properties for a custom profile

1. Start the Discovery Management Console.
2. Open Discovery Profiles.
3. Click the discovery profile that you want to configure.
4. Click the Platform Properties tab.
5. Change the value of the property that you want to update, and select the Included check box for this property.
6. Save the changes.

Configuring the properties for the default profile

In the $COLLATION_HOME/etc/collation.properties file, add (or edit) the respective property, as indicated in the following example, where discovery_profile represents the profile name:

```plaintext
com.ibm.cdb.session.allow.ITM.discovery_profile=true
```
For example, the following property specifies that TADDM uses the discovery profile “Utilization Discovery” and uses IBM Tivoli Monitoring to discover Tivoli Monitoring endpoints:
com.ibm.cdb.session.allow.ITM.Utilization_Discovery=true

Note: In the collation.properties file, you must replace the space character between “Utilization” and “Discovery” in the profile name with an underscore character.

Additional TADDM server properties that you might need to configure

The following configuration tips describe additional TADDM server properties that you might need to configure:

- The value of the following property, which is specific to Windows systems, must be set to true (which is the default value) to enable discovery of Windows target systems in discovery using IBM Tivoli Monitoring. If the value is set to false, TADDM cannot establish an IBM Tivoli Monitoring session to Windows target systems.
  com.collation.AllowPrivateGateways=true

- High processor usage might occur on the Tivoli Enterprise Portal Server during the discovery. To minimize this, you can limit the number of discover worker threads that run during the discovery. On the TADDM server, set the following server property:
  com.collation.discover.dwcount=16

- In a large IBM Tivoli Monitoring environment, the IBM Tivoli Monitoring Scope sensor might time out before completing. To allow for a longer processing time, set the following server properties:
  com.collation.platform.session.ITMSessionNumProgressChecks=3600
  com.collation.discover.agent.ITMScopeSensor.timeout=3600000

Configuring for discovery over OSLC Automation Session
TADDM can run Level 2, and some Level 3 discoveries by using OSLC.

Before you begin

To configure the discovery on scope sets provided by OSLC Execute Automation Service Providers, you must meet the following requirements:

- You must have at least one installed and working OSLC Execute Automation Service Provider.
- TADDM must be connected to OSLC Execute Automation Service Provider.

Procedure

To run a discovery over OSLC Automation Session, complete the following steps:

1. Add access credentials of the product that you are integrating with to the access list. To do so, create a new access list entry of type "Integration">"OSLC Automation". If you are integrating TADDM with ITM, provide ITM TEPS credentials. During the discovery, OSLC Automation access list entries and ITM access list entry type are used to ensure compatibility with earlier versions.

2. Check the discovery scope. The scope sets are created periodically by OSLCAutomationAgent. The new scope sets are listed in the Scope Sets tab. If you are integrating TADDM with ITM, one scope set per each ITM TEMS is created. You can run OSLCAutomationAgent manually by using the following command:
/taddm/dist/support/bin/runTopoBuild.sh

3. Configure discovery properties that allow using OSLC Automation Session. You can set the properties in the collation.properties file or in a new custom discovery profile.

- The collation.properties file:
  com.ibm.cdb.session.prefer.OSLCAutomation=true
  com.ibm.cdb.session.allow.OSLCAutomation=true

The examples of scoped properties:
com.ibm.cdb.session.prefer.OSLCAutomation.9.222.222.124=false
com.ibm.cdb.session.prefer.OSLCAutomation.Level_3_Discovery=false

- Custom discovery profile. In Discovery Management Console, create a new discovery profile and configure the Platform Properties tab in the following way:
  com.ibm.cdb.session.allow.OSLCAutomation=true
  com.ibm.cdb.session.prefer.OSLCAutomation=true

4. Run a regular discovery of scopes created by OSLCAutomationAgent by choosing either of the following methods:

- The default L2, or L3 profile when the collation.properties file is configured to support OSLCAutomation Session.
- The new discovery profile with properly configured Platform Properties tab.

Related reference:

"Properties for discovery using OSLC Automation Session” on page 79
These properties apply to discovery using OSLC Automation Session.

"Command line interface for OSLCAutomationAgent” on page 199
OSLCAutomationAgent is used to collect data from OSLC Execute Automation Service Providers. You can use commands to run the agent manually, and to refresh or update the scope sets that it creates.

Configuring the level of discovery

You must configure the level of discovery.

Configuring for Level 1 discovery

Some minimal configuration is required for Level 1 discovery (credential-less discovery), which scans the TCP/IP stack to gather basic information about active computer systems.

About this task

For Level 1 discovery, you must configure the network devices in your environment that you want the TADDM server to discover.

Procedure

To do this, complete the following steps:

1. Depending on your SNMP version, record the following information for use with the TADDM server:
   - For SNMP V1 and V2, record the SNMP MIB2 GET COMMUNITY string.
   - For SNMP V3, record the SNMP user name and password.
2. Assign permission for MIB2 System, IP, Interfaces, and Extended Interfaces.
Configuring for Level 2 discovery
In addition to the requirements for Level 1 discovery, Level 2 discovery requires configuration to support discovery of detailed host configuration information.

Before you begin
If the target systems are IBM Tivoli Monitoring endpoints that are discovered by the IBM Tivoli Monitoring Scope sensor, the credentials for those target systems are not required for Level 2 discovery. For more information, see the following sources:

- “Integrating TADDM with IBM Tivoli Monitoring (old method)” on page 200
- “Configuring for discovery using IBM Tivoli Monitoring (old method)” on page 106
- TADDM Sensor Reference for information about the IBM Tivoli Monitoring Scope sensor

About this task
On the target operating systems (computer systems) that you want TADDM to discover, at a minimum, you must configure the following software:

Secure Shell (SSH)
You can use either OpenSSH or the vendor-supplied version of SSH that is provided with the operating system. For more information about Windows operating systems, see “Windows Management Instrumentation (WMI) dependency” on page 121.

SUNWscpu (Solaris environment only)
To provide complete information about processes, install the SUNWscpu (Source Compatibility) package.

LiSt Open Files (lsof)
To provide complete information on dependencies, install the LiSt Open Files (lsof) program according to the requirements in lsof requirements in the TADDM Wiki at https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Tivoli%20Application%20Dependency%20Manager/page/TADDM%20lsof%20requirements

Creating the service account:
You must create a service account on all computer systems that are discovered using SSH key-based and password-based connections. This is the primary method for discovering the computer systems (servers) in your network.

About this task
To simplify the discovery setup, create the same service account on each target computer system that you want to discover. The service account must allow access to all resources on the target computer system that TADDM must discover. The service account must have write access privileges to its home directory on each target computer system. This directory requires approximately 20 MB of free space. During a discovery, scripts and temporary result files can be stored in this directory. After the discovery is run, the files are deleted.

A service account with non-root privilege can be used. However, to run on the target computer system, some operating system commands that are used during discovery might require elevated privilege, such as root (or superuser) authority.
Procedure

Complete one of the following procedures to create a service account on the target computer system:

1. For a Linux, Solaris, AIX, and Linux for System z operating system, assume that the service account name is coll and use the following commands to create the service account:

```
# mkdir -p /export/home/coll
# useradd -d /export/home/coll -s /bin/sh -c "Service Account" -m coll
# chown -R coll /export/home/coll
```

2. For a Windows computer system, create a service account that is a member of the local administrator's group. This account can be a local account or a domain account. Because TADDM relies on WMI for discovery, the account must have access to all WMI objects on the local computer. The service account must be created on the Windows Gateway and all target Windows computer systems.

   Note: The service account must have read/write access to the \WINDOWS\system32 or \WINDOWS\system64 directory and its subdirectories. On Windows Server 2008 systems, new users do not have the required access by default, so you must explicitly grant it for the service account.

Configuring for discovery using the Secure Shell (SSH):

The TADDM server can connect to either OpenSSH (version 1 or 2) or to the vendor-supplied version of SSH that is provided with the operating system.

The TADDM server supports the following authentication methods:
- SSH2 key-based login (RSA or DSA key) and SSH1 key-based login (RSA only)
- User name and password using SSH2, and user name and password using SSH1

Although you can use any of the authentication methods, the SSH2 key-based login is preferred. The server automatically tries each method in the order listed previously and uses the first method that works successfully. The TADDM server then uses the same method with that host for the entire discovery run.

   Note: For SSH2 key-based login, the TADDM server attempts login only with one key, RSA or DSA, whichever is found on the TADDM server. If both keys exist, only RSA is used.

Creating key pairs for key-based login with the TADDM server:

You can create a public/private key pair using the Secure Shell protocol (SSH) for key-based login with the TADDM server.

About this task

Depending on the version of SSH that you are using, SSH key-based login uses the keys shown in Table 35.

```
<table>
<thead>
<tr>
<th>SSH Version/Algorithm</th>
<th>Private Key</th>
<th>Public Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>openssh/SSH2/RSA</td>
<td>$HOME/.ssh/id_rsa</td>
<td>$HOME/.ssh/id_rsa.pub</td>
</tr>
<tr>
<td>openssh/SSH2/DSA</td>
<td>$HOME/.ssh/id_dsa</td>
<td>$HOME/.ssh/id_dsa.pub</td>
</tr>
</tbody>
</table>
```

Table 35. SSH keys
Table 35. SSH keys (continued)

<table>
<thead>
<tr>
<th>SSH Version/Algorithm</th>
<th>Private Key</th>
<th>Public Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Openssh/SSH1/RSA</td>
<td>$HOME/.ssh/identity</td>
<td>$HOME/.ssh/identity.pub</td>
</tr>
<tr>
<td>Commercial/SSH2/RSA</td>
<td>$HOME/.ssh2/id_dss_1024_a</td>
<td>$HOME/.ssh2/id_dss_1024_a.pub</td>
</tr>
</tbody>
</table>

You can also generate a public/private key pair using OpenSSH, version 2. To generate a public/private key pair using an SSH program other than OpenSSH or another version of OpenSSH, refer to the SSH documentation.

**Procedure**

To generate a public/private key pair using OpenSSH, version 2, complete the following steps:

1. Log in as the owner of the TADDM server.
2. To generate the SSH key, enter the following command:
   
   $ ssh-keygen -t rsa
   
   Accept the command defaults. TADDM supports key pairs with or without a passphrase.
3. On each target computer system where you want to allow for a key-based login, insert the contents of the `id_rsa.pub` file into the `$HOME/.ssh/authorized_keys` file for the service account. Certain SSH2 implementations generate the keys in a directory other than `$HOME/.ssh`. If your SSH implementation generates the keys in a different directory or with a different name, copy, link, or move the private key file to the `$HOME/.ssh/id_rsa` or `$HOME/.ssh/id_dsa` directory, depending on the algorithm.

Adding an access list entry for the computer system service account:

To configure password authentication with the Secure Shell (SSH), you must add an access list entry for the computer system service account that you created on the target system.

To add an access list entry for the computer system service account, complete the following steps:

1. From the TADDM start page, ensure that all services in the Administrator Console have been started.
2. Start the Discovery Management Console.
3. Select the Establish a secure (SSL) session check box so that you use the SSL security option. This option encrypts all data, including access list user names and passwords, before the data is transmitted between the Discovery Management Console and the TADDM server.
4. Add a computer system access list entry for the service account, and specify the login name and password.

Configuring System p and System i:

Discovery of an IBM Power5 technology-based system (System p or System i®) and its logical partitions is done through a management console. TADDM supports two types of management consoles: the Hardware Management Console (HMC) and the Integrated Virtualization Manager (IVM).
TADDM discovers the management console using SSH. The discovery scope must include the IP address of the management console and the Access List must include an entry of type Computer System with the proper credentials (user name and password) specified.

In addition to the user credentials, the discovery user must be defined on the management console with the following minimal permissions:

- **Hardware Management Console (HMC)**
  - For an HMC management console, a user based on the hmcoperator role is needed. For example, create a new role called taddmViewOnly based on the hmcoperator role. In addition, the following command line tasks must be assigned to the new role:
    
    **Managed System**
    - Needed to use the lshwres and lssyscfg commands
    
    **Logical Partition**
    - Needed to use the lshwres, lssyscfg, and viosvrcmd commands.
    
    **HMC Configuration**
    - Needed to use the lshmc command.

- **Integrated Virtualization Manager (IVM).**
  - For an IVM management console, a user with the View Only role is needed.

**Configuring for Level 3 discovery**

In addition to the requirements for Level 2 discovery, Level 3 discovery requires configuration to support discovery of application configuration and host data.

**Configuring Web and application servers for discovery:**

You must configure the Web servers and application servers in your environment that you want the TADDM server to discover.

This section provides the steps for configuring Web and application servers.

The Microsoft IIS server does not require configuration. There are no special access requirements. The user account that is already established on the host is sufficient.

For the Apache Web server, the TADDM service account for the host system must have read permissions to the Apache configuration files, such as the httpd.conf file.

For the Oracle iPlanet Web server, the TADDM service account for the host system must have read permissions to the iPlanet configuration files.

For Lotus Domino servers, ensure that you meet the prerequisites that are in the IBM Lotus Domino server sensor topic in the TADDM Sensor Reference.

**Configuring an Oracle Application server:**

The discovery of an Oracle Application server uses JAR files that are included with the Oracle Application server. These JAR files are not included in the TADDM server installation.
About this task

There is a property in the $COLLATION_HOME/etc/collation.properties file for pointing to an existing installation of the Oracle Application server. The following text is in the $COLLATION_HOME/etc/collation.properties file:

```
# Location of the root directory for Oracle Application Server on
# the Tivoli Application Dependency Discovery Manager
# 1. An example is /home/oracle/product/10.1.3/OracleAS_1
# 2. A relative directory is relative to com.collation.home
# 3. This directory (and its subdirectories) must be accessible
# for the user under which the server runs, usually the collation user.
# 4. Ignore if you do not intend to discover an Oracle Application server.
```

To point to an existing installation of the Oracle Application server, edit the following line in the $COLLATION_HOME /etc/collation.properties file:

```
com.collation.oracleapp.root.dir=lib/oracleapp
```

In an Oracle Application server installation, the directories that contain the required JAR files are owned by the oracle user with permissions: rwx------. This means no user other than from the owner (usually, an Oracle application) can access these directories. If the TADDM server is run using the oracle user, these directories are accessible. However, if this is not the case, you must change the directory permissions of the following directories to 711 so that all users can access the them:

- OracleAppServerHome
- OracleAppServerHome/j2ee
- OracleAppServerHome/j2ee/home
- OracleAppServerHome/opmn
- OracleAppServerHome/opmn/lib, where an example of OracleAppServerHome is /home/oracle/product/10.1.3/OracleAS_1

For discovery of an Oracle Application server, the com.collation.platform.os.ignoreLoopbackProcesses property in the $COLLATION_HOME/etc/collation.properties file must be set to true:

```
com.collation.platform.os.ignoreLoopbackProcesses=true
```

Procedure

To configure the access list, complete the following steps:

1. From the Discovery Management Console, create a discovery scope set that contains your Oracle Application server, or use an existing scope that contains your Oracle Application Server.
2. To create an access list, click the Access List icon.
3. In the Access List window, click Add.
4. In the Component Type field of the Access Details window, click Application Servers.
5. In the Vendor field, click Oracle Application Server.
6. Type the credentials for the Oracle Application server.

Configuring VMware servers:

When properly configured, the TADDM discovery process returns information about VMware servers.
About this task

To configure VMware servers for discovery, set the read-only permissions for the non-root TADDM service account in the VMware ESX console. As an alternative, you can use the root user for discovery. For more information about VMware servers, you can search the topics on the VMware community at [https://communities.vmware.com/welcome](https://communities.vmware.com/welcome)

Database set up for discovery:

To support discovery of your databases, you must create DB2, Oracle, or Sybase database users for the TADDM server. The TADDM server uses these database users to collect information about the databases that are running on remote hosts.

Creating a DB2 user:

To more completely discover DB2 instances on remote computer hosts, create a DB2 user.

Procedure

To create a DB2 user, complete the following steps:
1. Create a user with access to the following items:
   - The DB2 database TADDM server
   - All the instances in the DB2 database TADDM server that need to be discovered
2. Configure this DB2 user to have SSH access to the system that hosts the DB2 database server.
3. In the TADDM server access list, complete the following steps to add the user name and password for the DB2 user:
   a. In the Discovery Management Console toolbar, click Discovery > Access List. The Access List pane is displayed.
   b. Click Add. The Access Details window is displayed.
   c. In the Access Details window, complete the following information:
      1) In the Component Type list, select Database.
      2) In the Vendor list, select DB2.
      3) Enter the Name, User Name, and Password for the DB2 user.
   d. Click OK to save your information. The Access List pane is displayed with the new information.

Creating a Microsoft SQL Server user:

To more completely discover Microsoft SQL Server instances on remote computer hosts, create a Microsoft SQL server user.

Procedure

To create a Microsoft SQL server user, complete the following steps:
1. Create a Microsoft SQL server user with db_datareader role privileges and VIEW_ANY_DEFINITION permission. This might need to be completed by the Microsoft SQL server administrator.
2. In the Discovery Management Console, complete the following steps to add the user name and password for the Microsoft SQL server user in the TADDM server access list:
   a. In the toolbar, click **Discovery > Access List**. The Access List pane is displayed.
   b. Click **Add**. The Access Details window is displayed.
   c. In the Access Details window, enter the following information:
      1) In the **Component Type** list, select **Database**.
      2) In the **Vendor** list, select **Microsoft SQL server**.
      3) Enter the **Name**, **User Name**, and **Password**.
   d. Click **OK** to save your information. The Access List pane is displayed with the new information.

Creating an Oracle user:

To more completely discover Oracle instances on remote computer hosts, create an Oracle user.

**Procedure**

To create an Oracle user, complete the following steps:

1. Create an Oracle user with SELECT_CATALOG_ROLE privileges. This might need to be completed by the Oracle administrator.
   For example, use the following command to create the IBM Oracle user:
   ```sql
   create user collation identified by collpassword;
   grant connect, select_catalog_role to collation;
   ```

2. In the Discovery Management Console, complete the following steps to add the user name and password for the Oracle user in the TADDM server access list:
   a. In the toolbar, click **Discovery > Access List**. The Access List pane is displayed.
   b. Click **Add**. The Access Details window is displayed.
   c. In the Access Details window, complete the following information:
      1) In the **Component Type** list, select **Database**.
      2) In the **Vendor** list, select **Oracle**.
      3) Enter the **Name**, **User Name**, and **Password** for the computer.
   d. Click **OK** to save your information. The Access List pane is displayed with the new information.

Creating a Sybase user:

To completely discover Sybase ASE on remote computer hosts, create a Sybase user assigned to an appropriate role.

**Procedure**

To create a Sybase user, complete the following steps:

1. Use the following command to create a Sybase user that is a member of sa-role.
   ```sql
   sp_role "grant",sa_role,IBM
   ```
   Ensure that the Sybase IQ user is a member of DBA. If the Sybase IQ user is not a member of DBA, the Sybase IQ database-specific information cannot be found.
2. In the Discovery Management Console, complete the following steps to add the user name and password for the Sybase user in the TADDM server access list:
   a. To create an access list, click the Access List icon.
   b. In the Access List window, click Add.
   c. In the Component Type field of the Access Details window, click Database.
   d. In the Vendor field, click Database.
   e. Type the credentials (user name and password) to establish Java Database Connectivity (JDBC) to the Sybase server.

Configuring for discovery of Windows systems

For discovery of Windows computer systems, TADDM supports both gateway-based discovery and SSH-based discovery, as well as asynchronous and script-based discovery.

For details about asynchronous discovery, see "Configuring for asynchronous discovery" on page 100. For details about script-based discovery, see "Configuring for script-based discovery" on page 104.

Gateway-based discovery requires a dedicated Windows computer system, which is accessible through SSH, to serve as the gateway. All discovery requests go through the gateway. The gateway uses Windows Management Instrumentation (WMI) to discover the target Windows computer systems.

Fix Pack 2  If you use TADDM 7.3.0.2, or later, instead of WMI, you can also use PowerShell session to discover the target Windows computer systems. You can configure TADDM to allow communication via the PowerShell session only. For details, see the Configuring for discovery through a firewall without an anchor topic in the TADDM User’s Guide.

SSH-based discovery does not require a dedicated gateway computer system. Instead, discovery uses a direct SSH connection to the target Windows computer system.

Typically, gateway-based discovery is preferred over SSH-based discovery because configuring the gateway and WMI, or PowerShell, are easier than configuring SSH. WMI is available by default on all Windows target systems that are supported by TADDM. PowerShell is supported only for targets that run Windows Server 2008, and later. You must have PowerShell version 2, or later, installed on the gateway and the target systems. Other than the gateway computer, which requires an SSH server, no special software requirements exist for the Windows targets. However, discovery using SSH can be faster because a gateway is not involved in the discovery flow, and no WMI Provider is deployed.

Doing a direct discovery requires an SSH server on each Windows target system. In addition, for direct discovery using SSH, you must install the Microsoft .NET Framework version 2, or 3 on each Windows target system. .NET Framework is not installed by default on Windows Server 2000.

Note: Fix Pack 2  If you use TADDM 7.3.0.2, or later, you can also install .NET Framework versions 4, or 4.5.

For both types of discovery, the TADDM Windows discovery program, TaddmTool.exe file, is used to perform the discovery. For discovery using a gateway, the TaddmTool program is deployed to the gateway during discovery.
initialization. For discovery using SSH, the TaddmTool program is deployed to each Windows target computer system. The TaddmTool program is a .NET application.

By default, TADDM is configured to use only gateway-based discovery. This configuration is controlled by the following two TADDM server properties, which are described in the TADDM Sensor Reference for the Windows computer system sensor:
- com.collation.AllowPrivateGateways=true
- com.collation.PreferWindowsSshOverGateway=false

By default, TADDM is configured to use the WMI session. To learn when to use the PowerShell session and how to enable it, see “PowerShell session” on page 122.

Whether you use a Windows gateway with WMI or directly connect with SSH, the information that is retrieved is identical. The following list identifies the prerequisites for gateway-based and SSH-based discovery:

**Prerequisites for gateway-based discovery with WMI**

1. A dedicated Windows Server computer system is required to serve as the gateway. Operating system requirements for gateway servers are the same as Windows operating system requirements for TADDM servers. For details about supported Windows operating systems, see the TADDM server software requirement topic in the TADDM Installation Guide.
2. The gateway must be in the same firewall zone as the Windows computers to be discovered.
3. You must install a supported version of an SSH server on the gateway computer system.
4. The gateway uses remote WMI to discover each Windows target. In addition, a WMI Provider is automatically deployed to each Windows target computer system during the discovery initialization. The WMI Provider is used to discover data not included in the core WMI. Enable WMI on the Windows target computer system to be discovered. By default, on most Windows 2000 and later systems, WMI is enabled.

**Prerequisites for gateway-based discovery with PowerShell**

1. A dedicated Windows Server computer system is required to serve as the gateway. Operating system requirements for gateway servers are the same as Windows operating system requirements for TADDM servers. For details about supported Windows operating systems, see the TADDM server software requirement topic in the TADDM Installation Guide.
2. You must install PowerShell version 2, or later, on the gateway and the target systems. Only targets that run Windows Server 2008, or later, are supported.
3. You must configure the gateway by running the following command:
   ```powershell
   Set-Item WSMan:\localhost\Client\TrustedHosts * -Force
   ```
   This command sets the trustedHosts list. By default, the list exists but it is empty, so it must be set before the remote session is opened. With the -Force parameter, PowerShell executes the command without prompting you for each step.
4. You must configure the target systems by running the following command:

   Enable-PSRemoting -Force

   This command starts the WinRM service, sets it to start automatically with your system, and creates a firewall rule to allow the incoming connections. With the -Force parameter, PowerShell executes these actions without prompting you for each step.

Prerequisites for SSH-based discovery
1. You must install a supported version of an SSH server on each Windows target system.
2. You must install the Microsoft .NET Framework version 2, or 3 on each Windows Server target system.

   Note: If you use TADDM 7.3.0.2, or later, you can also install .NET Framework versions 4, or 4.5.

See also the Configuring for a non-admin Windows discovery topic in the TADDM Sensor Reference.

Configuring Bitvise WinSSHD
You can use Bitvise WinSSHD to provide SSH access to Windows systems.

Before you begin
For gateway-based discovery, Bitvise WinSSHD must be installed on the gateway system. For direct SSH discovery, Bitvise WinSSHD must be installed on each Windows system.

For more information about the supported Bitvise WinSSHD versions, see the Windows gateways topic in the TADDM Installation Guide.

Bitvise WinSSHD is available from [http://www.bitvise.com/](http://www.bitvise.com/)

About this task
The following steps describe how to configure Bitvise WinSSHD 5.22. The specific steps can differ depending on the release of Bitvise WinSSHD that you have.

Procedure
1. To restrict SSH host access to the TADDM server, complete the following steps:
   a. In the WinSSHD Control Panel, click Open easy settings.
   b. On the Server settings tab, for the Open Windows Firewall field, select As set in Advanced WinSSHD settings.
   c. Click Save Changes.
   d. In the WinSSHD Control Panel, click Edit advanced settings. The Advanced WinSSDH Settings window is displayed.
   e. Click Settings > Session.
   f. Set the value of the following items to 0:
      - IP blocking - window duration
      - IP blocking - lockout time
   g. Click OK.
h. In the WinSSHD Control Panel, click **Edit advanced settings**. The Advanced WinSSDH Settings window is displayed.

i. Click **Settings > Access Control**.

j. In the right pane, click **IP rules**.

k. Click **Add**.

l. Type the IP address of the TADDM server.

m. In the **Number of significant bits** field, type 32.

n. In the **Description** field, type TADDM server.

o. Ensure that the **Allow connect** check box is selected.

p. Click **OK**.

q. Remove the **0.0.0.0/0** entry from the list.

2. To create and configure a virtual group and users, complete the following steps:

a. In the WinSSHD Control Panel, click **Edit advanced settings**. The Advanced WinSSDH Settings window is displayed.

b. Click **Settings > Virtual Groups**.

c. To add a new group, click **Add**.

d. In the **Group** and **Windows Account Name** fields, type a name.

e. Click **OK**.

f. Click **Settings > Virtual Accounts**.

g. To add a new account, click **Add**.

h. In the **Virtual account name** field, type a name.

i. Set a password using the link for the virtual account password.

j. From the drop-down list, select the virtual group that you created in a previous step, and ensure that the **Use group default Windows account** check box is selected.

k. Click **OK**.

3. In the WinSSHD Control Panel, click **Start WinSSHD**.

**What to do next**

If you are discovering multiple Windows servers, you might experience the following message:

*A working gateway cannot be found*

For more information about additional configuration that might help, see the **Gateway problems** topic in the TADDM **Troubleshooting Guide**.

**Configuring the Cygwin SSH daemon**

You can use the Cygwin SSH daemon (sshd) to provide SSH access to Windows systems.

**About this task**

For gateway-based discovery, the Cygwin SSH daemon must be installed on the gateway system; for direct SSH discovery, the daemon must be installed on each Windows system.

For more information about the supported Cygwin SSH daemon versions, see the **Windows gateways** topic in the TADDM **Installation Guide**.
Important: For successful discovery by using Cygwin SSH, the following requirements must be fulfilled:

- Anchors and gateways are supported on Cygwin 64-bit edition on Windows Server 2012 x64 and Windows Server 2008 x64.
- The discovery user and the user that starts the service must be the same. The discovery user must be a member of the Administrators group.

Cygwin is available from [http://www.cygwin.com/](http://www.cygwin.com/)

Procedure

To configure the Cygwin SSH daemon:

1. Start the cygwin bash shell.
2. From your system information, use the `cygwin mkpasswd` utility to create an initial `/etc/passwd`. You can also use the `mkgroup` utility to create an initial `/etc/group`. See the Cygwin User’s Guide for more details.

   For example, the following command sets up the password file, `passwd`, from the local accounts on your system:

   ```bash
   mkpasswd -l > /etc/passwd
   ```

3. Run the `ssh-host-config` program setup.
4. Configure SSH. Answer Yes to all questions.
5. Start the SSH server by running the following command:

   ```bash
   net start sshd
   ```

What to do next

The Cygwin (sshd) service must use an administrative domain user account when accessing the gateway server. This user account is required for some sensors for example, the Microsoft Exchange sensor. Complete the following steps:

1. Configure the domain user account by running the following commands:

   ```bash
   mkpasswd -u [domain_user] -d [domain] >> /etc/passwd
   mkgroup -d [domain] >> /etc/group
   ```

2. Start the services.msc program. Check the log on properties for the Cygwin (sshd) service that was created. Verify that the service is set up to be run by an administrative domain user account.

3. Cygwin (sshd) configuration and log files must be owned by the same domain user account that the Cygwin (sshd) service uses to accesses the gateway. Run the following commands:

   ```bash
   $ chown [domain_user] /var/log/sshd.log
   $ chown -R [domain_user] /var/empty
   $ chown [domain_user] /etc/ssh*
   ```

4. The domain user account must have the following permissions on the gateway server:

   - Adjust memory quotas for a process
   - Create a token object
   - Log on as a service
   - Replace a process level token

   If you are discovering multiple Windows servers, you might experience the following message:

   A Working gateway cannot be found
For more information about additional configuration that might help, see the Gateway problems topic in the TADDM Troubleshooting Guide.

**Configuring Remotely Anywhere**
You can use Remotely Anywhere to provide SSH access to Windows systems.

**About this task**

For more information about the supported Remotely Anywhere versions, see the Windows gateways topic in the TADDM Installation Guide.

For gateway-based discovery, Remotely Anywhere must be installed on the gateway system.

For direct SSH discovery, Remotely Anywhere must be installed on each Windows system.

You can use the default configuration values in Remotely Anywhere. For more information, go to [http://remotelyanywhere.com/](http://remotelyanywhere.com/)

**Configuring Tectia SSH Server**
You can use Tectia SSH Server to provide SSH access to the Windows systems.

**About this task**

For more information about the supported Tectia SSH Server versions, see the Windows gateways topic in the TADDM Installation Guide.

For the gateway-based discovery, Tectia SSH Server must be installed on the gateway system.

For the direct SSH discovery, Tectia SSH Server must be installed on each Windows system.

You can use the default configuration values in Tectia SSH Server. For more information, go to [http://www.ssh.com](http://www.ssh.com)

**Windows Management Instrumentation (WMI) dependency**
TADDM relies on Windows Management Instrumentation (WMI) to discover Windows computer systems. TADDM can be configured to restart the WMI service if a problem occurs with WMI. If the WMI service is restarted, all WMI-dependent services that were running before the restart are also restarted.

The following TADDM server properties control the restarting of WMI.

**Note:** The default value for WMI restart is false. Setting the values of the following properties to true might provide more reliable Windows discovery, but you must also consider the potential negative impact of the WMI service being temporarily stopped and restarted.

- com.collation.RestartWmiOnAutoDeploy=false
- com.collation.RestartWmiOnAutoDeploy.1.2.3.4=false
- com.collation.RestartWmiOnFailure=false
- com.collation.RestartWmiOnFailure.1.2.3.4=false
For more information about the TADDM server properties that the Windows computer system sensor uses, see the Configuring the collation.properties file topic in the Windows computer system sensor section of the TADDM Sensor Reference.

**PowerShell session**

To discover Windows computer systems, you can use either the WMI or the PowerShell session. In comparison to the WMI session, with the PowerShell session TADDM sends fewer requests to access target systems, which reduces the number of events that are logged. The PowerShell session can be used only with the script-based sensors. If you want to start using the PowerShell session, you must enable it because it is disabled by default.

You can use both sessions at the same time. If you are running regular and script-based discoveries, you cannot disable the WMI session because the regular discovery fails without it. However, you can prioritize the usage of the PowerShell session.

**Important:** If you run only regular discoveries, PowerShell session is not supported.

You can control the usage and prioritization of the PowerShell session by using the following properties:

- `com.collation.PowerShellAccessEnabled=false`
- `com.collation.WmiAccessEnabled=true`
- `com.collation.PreferPowerShellOverWMI=true`
- `com.collation.PowerShellPorts=5985,5986`
- `com.ibm.cdb.session.ps.useSSL=false`
- `com.ibm.cdb.session.ps.allowDNS=true`
- `com.ibm.cdb.session.ps.fallbackToIP=true`
- `com.collation.PowerShellTimeoutFudge=10000`
- `com.ibm.cdb.session.ps.urlPrefix=wsman`

To enable the PowerShell session, set the `com.collation.PowerShellAccessEnabled` property to `true`. The PowerShell session is preferred over the WMI session by default.

To learn more about these properties, see the Configuring the collation.properties file entries for the Windows computer system sensor in the TADDM Sensor Reference.

**Note:** In a very specific case, when you configured your firewall to allow communication via PowerShell session only, you must open PowerShell ports and configure the Ping sensor property. For details, see the Configuring for discovery through a firewall without an anchor topic in the TADDM User’s Guide.

**Example scenarios**

Depending on how you discover your Windows target systems, you can configure the preceding properties in the following ways.

- You are using only the sensors that support script-based discovery. In such case, you can enable PowerShell session by setting the `com.collation.PowerShellAccessEnabled` property to `true` and disable the WMI
session by setting the `com.collation.WmiAccessEnabled` property to `false`. However, when PowerShell is not available, the session and discovery fails.

- You are using sensors that support script-based and regular discovery. In such case, do not disable the WMI session, because it results in the failure of the regular discovery. Enable the PowerShell session by setting the `com.collation.PowerShellAccessEnabled` property to `true`. To establish the PowerShell session whenever possible, do not change the default value of the `com.collation.PreferPowerShellOverWMI` property. In such case, TADDM creates a hybrid session that is capable to use both PowerShell and WMI functions. The WMI session is used only when the PowerShell session is not capable to execute tasks that are requested by the regular sensors.

## Configuring for discovery of placeholders

You can configure TADDM to create placeholders for undiscovered dependencies in your infrastructure.

Placeholder is an object that is a part of your infrastructure but is not represented in TADDM with the default settings. The reasons why it is not represented might be that one side of the connection is not discovered, no sensor supports such type of the object, or no custom server template is created for it.

Placeholders are of SSoftwareServer class. They have `hierarchyDomain` and `hierarchyType` attributes set. The following table specifies the values of the attributes:

<table>
<thead>
<tr>
<th>Connection side</th>
<th>hierarchyDomain attribute value</th>
<th>hierarchyType attribute value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
<td>app.placeholder.client.local</td>
<td>Name of the command which originates the connection, for example Java</td>
</tr>
<tr>
<td>Remote</td>
<td>app.placeholder.server.remote</td>
<td>Unknown</td>
</tr>
</tbody>
</table>

By using these values, you can filter unwanted relations in the traversal configuration of business applications. For details, see the **Traversal configuration** topic in the TADDM User’s Guide.

When a placeholder is created, and then the equivalent App Server is created by a sensor, or a custom server template, the PlaceholderCleanupAgent merges the placeholder with the discovered App Server.

**Note:** You can create placeholders in TADDM 7.3.0.2, but it is limited. Therefore, it is advised to use placeholders in TADDM 7.3.0.3, and later. Migration of placeholders created in FP2 to FP3 is not supported.

## Enabling creation of placeholders

To enable the creation of placeholders, add the following property to the `collation.properties` file:

```java
com.ibm.cdb.topomgr.topobuilder.agents.ConnectionDependencyAgent2
dependencyPlaceholders=true
```

The default value is `false`. 

---

Administering 123
When you set this property to true for the first time, you must restart TADDM to enable extended attributes for LogicalConnection and SoftwareServer classes. These extended attributes are necessary for proper functioning of this feature.

In case, above property is set to true then there is no need to set the below properties in `collation.properties` explicitly, rather their default coded values will be used.

```
com.ibm.taddm.dependencyPlaceholders.create.localClient.to.remoteServer = true

The default value is true.
com.ibm.taddm.dependencyPlaceholders.create.remoteClient.to.localServer = false

The default value is false.
```

**Note:** Placeholders behaviour can be changed by setting these properties in `collation.properties`.

**Important:** When you enable the creation of placeholders, your business applications might grow significantly and the building process might be longer. To prevent it, you can filter unwanted relations in the traversal configuration of business applications.

**Viewing placeholders**

You can view the placeholders in the Inventory Summary pane after you set the filter to Placeholders. Placeholders for undiscovered dependencies are in the **Software Servers** tab.

**Creating custom server templates**

You can use placeholders to create custom server templates in the following ways:

- By using information about placeholders that is generated by the `bizappscli` tool. For details, see the *Actions for analyzing the content of business applications* topic in the TADDM User’s Guide.
- By using the command line information that is displayed on the **Runtime** tab in the Details pane for placeholders of type `app.placeholder.*.local`.

For more information about custom server templates, see the *Creating and managing custom server templates* topic in the TADDM User’s Guide.

**Creating Level 3 application servers with no credentials**

If you want to discover basic Level 3 information about your infrastructure elements, you do not need to provide credentials in the access list. You can create application servers by using sensor internal templates. These templates can be processed by CustomAppServerTopoAgent, or during a discovery run by custom server template sensor.

**About this task**

By creating application servers with no credentials you can discover only basic information about your infrastructure, for example what kind of software is
installed. Choose this mode if you do not want to provide credentials for Level 3
discovery, but want to discover basic information about your infrastructure.

There are two methods to create Level 3 application servers. You can run custom
server template sensor, or enable CustomAppServerTopoAgent.

Procedure

- Run a discovery with custom server template sensor
  Complete the following steps:
  1. In collation.properties file, set the
     com.collation.internaltemplatesenabled property to true. This property
     enables internal templates of Level 3 sensors. The default value is false.
  2. Run the discovery by using a profile that does not contain sensor that would
     normally discover the information that you want to discover by using
     custom server template sensor. For example, if you want to discover basic
     information for DB2 server, choose Level 2 profile discovery, or your own
     profile that does not contain IBM DB2 sensor. If the profile contains IBM DB2
     sensor, then this sensor is run instead of the custom server template sensor.

- Run the CustomAppServerTopoAgent
  CustomAppServerTopoAgent uses runtime processes previously discovered by
generic server sensor. You can run the agent manually, or set it to run
automatically. Complete the following steps:
  1. For both manual and automatic modes of the agent, in collation.properties
     file, set the com.collation.internaltemplatesenabled property to true. This
     property enables internal templates of Level 3 sensors. The default value is
     false.
  2. To manually start the CustomAppServerTopoAgent, run the following
     command:
     COLLATION_HOME/support/bin/runtopobuild.sh -a CustomAppServerTopoAgent
  3. To set up automatic runs of the agent, set the
     com.ibm.cdb.topobuilder.groupinterval.discovery= property in the
     collation.properties file.
     This property specifies how often the agent runs. By default, no value is
     provided, which means that the agent is disabled. To enable it, specify the
     value in hours, for example

- Optional: Select templates to exclude from processing
  If you want to enable only some of internal templates of Level 3 sensors, you
can control it by using the following property:
  com.collation.discovery.ignoreTemplateList

  This property specifies a list of internal templates that you do not want to
process. The value of this property is a semicolon-separated list of template
names, for example
  com.collation.discovery.ignoreTemplateList=DB2Unix;MSSQL. You can find the
  name of an internal template in Data Management Portal in the Object Name
  field, which is in the General tab in the Details pane of Other Database
  Servers.

Configuring location tagging

Location tagging indicates where each Configuration Item (CI) was created. It
enables location-based filtering of configuration items in BIRT reports and API
queries.
If you enable location tagging, each discovered object that is stored in the discovery database includes the `locationTag` attribute (string). Objects such as relationships, aggregation objects, and inheriting objects that are created by topology agents, include location tag data under certain conditions:

- A one-to-one relationship (such as Dependency or NetworkConnection) includes a location tag if the location is the same for both connected objects.
- An aggregation object (such as a cluster) includes a location tag if the location is the same for all aggregated objects.

**Note:** In case of custom collections, the `locationTag` attribute is set only when the *value* of location tag of all custom collection's core CIs is the same. When the custom collection is extended with a core CI that has a different location tag, the `locationTag` attribute for such custom collection is cleared.

- A simple object includes the location tag from the object it is based on.

In all other cases, objects that are created by topology agents do not include a location tag value.

To enable location tagging, set the following property in the `collation.properties` file:

```plaintext
com.ibm.cdb.locationTaggingEnabled=true
```

Location tag values can be either static (specified for a particular server or anchor) or dynamic (specified for a specific discovery or IdML book import). A location tag value is limited to 192 characters. If the specified location tag exceeds 192 characters, it is cut to required length.

**Limitations**

When you run a Level L1 discovery, Configuration Items which are already present in the database are not updated. As a result, location tags are assigned only to the newly discovered objects.

**Static location tagging**

Static location tagging assigns the `locationTag` attribute to all objects that are discovered or loaded using the IdML book import based on static configuration of the TADDM or anchor server.

**TADDM server**

To configure the location tag value for CIs that are created on a TADDM server, specify the following property in the `collation.properties` file:

```plaintext
com.ibm.cdb.locationTag=location
```

where `location` is the location tag value you want to use.

**Anchor**

To configure the location tag value for CIs that are created on an anchor, configure the `anchor_location_n` attribute in the `$COLLATION_HOME/etc/anchor.properties` file. The following example entries from the anchor.properties file indicate how the location information for anchors is set:

```plaintext
anchor_host_1=192.168.1.13
anchor_scope_1=FIRST_SCOPE
anchor_zone_1=FIRST_ZONE
```
If a location tag is not specified for an anchor, the location of each of the CIs that are created on the anchor is set to the location that is specified for either the TADDM server to which the CIs are connected.

If the location tag value is not specified for the anchor or the TADDM server, no location information is set for that CI.

**Dynamic location tagging**

Dynamic location tagging sets the `locationTag` attribute using a value specified for a specific discovery or IdML book import.

**Discovery**

To specify a location tag value during discovery, launch the discovery from the command line and specify the location tag using the optional `-l` or `-myLocation` option, as in the following example:

```sh
api.sh -u administrator -p collation discover start -n discovery1 -p myProfile -l myLocation myScope
```

where `locationTag` is the location tag value you want to use. The value you specify overrides any static location tag value for objects created during this specific discovery.

**Note:** If the location tagging is not enabled in the `collation.properties` file, specifying a location tag during discovery request causes a discovery exception.

**IdML book import**

To specify a location tag value while importing an IdML book, specify the location tag using the optional `-l` option, as in the following example:

```sh
loadidml.sh -f idml_book.xml -l locationTag
```

where `locationTag` is the location tag value you want to use. If you want to import multiple IdML books with different location tags, each book must be loaded separately.

**Access list**

You can create access list entries with a location tag assigned.

The location tag attribute is obligatory but can be changed later. The credentials are filtered by location, that is why only the access entries for specific locations are used. This limits the possibility of sniffing the password from other customers or locations. If you run discovery without a location tag, none of the tagged credentials are used.

When a new access entry is added with the location tag set to the asterisk character (*), it is used as the last access entry that is tried during a discovery while establishing a session to the endpoint.

The asterisk character (*) is the default value and can be changed by setting the following parameter:

```
com.ibm.cdb.locationTag.global=GLOBAL
```
In such case, the access entry with the GLOBAL tag is the last one that is tried when running a discovery. The preceding location tag is used only for the access list and has no influence on location tags that are assigned to CIs that are discovered during a discovery.

**BIRT reports**

Business Intelligence and Reporting Tools (BIRT) reports can be filtered to generate the data for specific customer location.

If location tagging is enabled, the text field is on the BIRT reports pane below the list of reports. You can run a BIRT report against any location tag so it sees the data that belongs only to that location.

None of the Out-of-the-Box reports can handle the location tags. If you need to use the BIRT reports, they must be updated manually to support filtering by location tag.

**Maintenance and tuning**

To maximize TADDM performance, you might want to perform additional configuration steps and ongoing maintenance tasks.

**Bulk load parameters tuning**

You can customize the behavior of the Bulk Loader by specifying particular parameters at run time or configuring the bulkload.properties file.

There are three distinct phases for loading data using the Bulk Loader:

1. Analyze the objects and relationships to determine the graphs in the data.
   - Typically, 1 - 5% of execution time
2. Construct model objects and build graphs.
   - Typically, 2 - 5% of execution time
3. Pass the data to the application programming interface (API) server.
   - Typically, 90 - 99% of execution time

There are two options for loading data:

- Data can be loaded one record at a time. This is the default mode. You must load records one at a time for the following files:
  - Files with errors.
  - Files with extended attributes.

- Data can be loaded in bulk. This is called graph writing because a entire graph is loaded, rather than just one record.

Bulk loading with the graph write option is faster than loading records one at a time. (Reference the Bulk Load measurements for details). The following example shows the graph write option, where -g=buffer and blocks of data are passed to the API server:

```
./loadidml.sh -g -f /home/confignia/testfiles/sample.xml
```

The following parameters in the bulkload.properties can be used to improve performance when loading data in bulk:

- `com.ibm.cdb.bulk.cachesize=2000`

The `cachesize` parameter controls the number of objects processed in a single write operation when bulk loading with the graph write option. Increasing the
cache size value improves performance at the risk of running out of memory either on the client or at the server. Alter the number only when specific information is available to indicate that processing a file with a larger cache provides some benefit in performance. The default cache size value is 2000, and the maximum cache size value is 40000.

`com.ibm.cdb.bulk.allocpoolsize=1024`

This value specifies the maximum amount of memory that can be allocated to the Bulk Loader process. It is an Xmx value that is passed to the main Java class of the Bulk Loader. Specify the value in megabytes.

Make sure that a Java virtual machine is not running out of memory. You can do that by collecting thread dumps of TADDM processes and reviewing them. If necessary, increase the memory size.

**Tip:** Tests that were run on the ITNMIP book indicate that the performance is optimal when you set the bulk load process properties and parameters to the following values:

```
com.ibm.cdb.bulk.cachesize=4000
com.ibm.cdb.bulk.allocpoolsize=4096
value-Xms768M|-Xmx1512M|-DTaddm.xmx64=6g|
```

It is also important that you run the **RUNSTATS** command frequently during the bulk loading process.

**Database maintenance**

To maintain optimal performance of your system, you must plan for and complete regularly scheduled maintenance and tuning of the TADDM database.

**Default database configurations**

The default database configurations that are provided with TADDM are sufficient for proof of concept, proof of technology, and small pilot implementations of TADDM.

**Tuning guidelines for both DB2 and Oracle databases**

The following tuning guidelines apply to both DB2 and Oracle databases:

1. Based on storage capacity alone, do not limit the number of physical disk drives that are available to your database.
2. Ideally, the following components should be placed on separate disk drives or arrays:
   - Application data (such as tables and indexes)
   - Database logs
   - Database temporary space: used for sort and join operations
3. Use the fastest disks that are available for your log files.
4. Enable Asynchronous I/O at the operating system level.

For more information on DB2 and Oracle database tuning, refer to *Database Performance Tuning on AIX* at [http://www.redbooks.ibm.com/redbooks/pdfs/sg245511.pdf](http://www.redbooks.ibm.com/redbooks/pdfs/sg245511.pdf)

For more information that is specific to DB2 database tuning, also refer to *Relational Database Design and Performance Tuning for DB2 Database Servers* at
Deleting old database records

The number of data records in the tables grows over time, and depending on the amount of storage space available, from time to time, you might want to remove data manually to maintain the tables at a smaller size. After you clear the CHANGE_HISTORY_TABLE, you can remove the corresponding entries in the CHANGE_CAUSE_TABLE. You can also improve performance and usability of the data integrity tool by deleting old records from the ALIASES_JN table.

Deleting records from CHANGE_HISTORY_TABLE and CHANGE_CAUSE_TABLE:

You can remove old records to improve performance and maintain the tables at a smaller size. After you remove the records from the CHANGE_HISTORY_TABLE, you can safely remove the corresponding entries from the CHANGE_CAUSE_TABLE.

To free storage space in TADDM databases, use SQL queries to remove old data manually from the CHANGE_HISTORY_TABLE. The following command is an example of such an SQL query, where the integer 1225515600000 represents the date, 1 November 2008, expressed in the same format as that returned by the System.currentTimeMillis() Java method, or a number equal to the difference, measured in milliseconds, between the current time and midnight, 1 January 1970 UTC:

```
DELETE FROM CHANGE_HISTORY_TABLE
WHERE PERSIST_TIME < 1225515600000
```

To convert a date to a Java time stamp, use the following code:

```
import java.util.*;
import java.text.*;
import java.sql.Timestamp;

public class DateToString {
    public static void main(String args[]) {
        try {
            String str = args[0];
            SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy");
            Date date = formatter.parse(str);
            long msec = date.getTime();
            System.out.println("Date is \"+date+\"\n            System.out.println("MillSeconds is \"+msec+\"; 
        } catch (ParseException e) {
            System.out.println("Exception :\"+e; 
        }
    }
}
```

Run the code as follows:
```
java DateToString 1/11/2008
Date is Sat Nov 01 00:00:00 EST 2008
MillSeconds is 1225515600000
```

Use the resulting Java time stamp in the SQL query.
If an exceptional number of records exist in the CHANGE_HISTORY_TABLE, you might want to perform incremental deletes (deleting a subset of records at a time) to avoid filling transaction logs in the database.

After you clear the CHANGE_HISTORY_TABLE, you can safely remove the corresponding entries in the CHANGE_CAUSE_TABLE. The CHANGE_CAUSE_TABLE is a link table that is used for change propagation. For example, if you add a new software component to the operating system, the table links this change to the computer system on which the operating system runs. You can remove records in the CHANGE_CAUSE_TABLE with the following command:

delete from change_cause_table where cause_id not in (select id from change_history_table)

**Timeframes for removing data**

To limit database growth over time, you can manage the size of the change history data stored by TADDM. When determining the optimal time frame for removing data from the change history table, consider what you use the change history data for, and whether the change history information is used by other applications.

If the change history information is being used by another application, ensure that you perform application synchronizations more frequently than the number of weeks of change history data maintained in the CHANGE_HISTORY_TABLE.

The following examples illustrate some typical scenarios:

- If you are using change history data for problem determination and you want to investigate problems that occurred five weeks ago, keep at least five weeks of data in CHANGE_HISTORY_TABLE.
- If you synchronize Tivoli Business Service Manager (TBSM) weekly, maintain more than one week of change history data in the TADDM change history table.

It is important to note that in synchronization server deployments, a large amount of change history data on the domain servers increases the time a full synchronization takes to complete.

**Data maintenance in a synchronization server deployment**

In a domain server deployment, you can base data maintenance decisions solely on the data needs for the domain. However, in a synchronization server deployment, you must coordinate the removal of change history data between each domain server database and the synchronization server database, and you must remove the data from all of these databases.

In a synchronization server deployment, use the following guidelines for data maintenance:

- Keep change history data at the domain level for a period of time that is greater than the period of time between each scheduled synchronization of the domain server databases with the synchronization server database. For example, if the synchronization occurs on a weekly schedule, maintain at least two weeks of change history data in each domain server database.
- Remove data from a domain server database first. Then remove data from the synchronization server database.
- The best practice is to maintain the same number of weeks of change history data in all TADDM databases. However, the period that change history data is
kept in the synchronization server database can vary from the period that such
data is kept in the domain server databases.

- After you determine a timeframe for data removal that meets the specific needs
  of your environment, the best practice is to remove the data just after the
  occurrence of a synchronization between the domain server databases and the
  synchronization server database.

**Deleting records from the ALIASES_JN table:**

When old records are deleted from the ALIASES_JN table, it can improve
performance and usability of the data integrity tool, and can free additional space
in the database.

**About this task**

The ALIASES_JN table contains the history of changes to the ALIASES table. The
data integrity tool requires the gathered data to find possible configuration item
over merges in the database. Over time, the number of data records in the
ALIASES_JN table grows to a significant size. The size of this table affects both the
performance and the usability of the data integrity tool, and increases the need for
storage space on the TADDM database.

The topology agent AliasesJnTableCleanup performs the cleanup of the
ALIASES_JN table.

By default it removes all rows older than 30 days. You can change the age at which
records are deleted by configuring the following property in the
collation.properties file:

```
com.ibm.cdb.topomgr.topobuilder.agents.AliasesJnTableCleanupAgent.removeOlderThanDays=30
```

If you set the property to a value of -1, the agent is disabled. If you set the age
value too low, the verify-data tool with the over merge option might not produce
complete results.

By default, the agent runs for no longer than 1800 seconds (30 minutes). If this
length of time is not enough to remove all of the aged rows, an attempt to delete
the remaining ones is made the next time the agent is run. You can set the agent
timeout value by configuring the following property in the collation.properties
file:

```
com.ibm.cdb.topomgr.topobuilder.agents.AliasesJnTableCleanupAgent.timeout=1800
```

**DB2 database maintenance**

You must maintain the TADDM DB2 database regularly to ensure acceptable
performance.

**About this task**

The following DB2 utilities are available:

**REORG**

After many changes to table data that are caused by the insertion, deletion,
and updating of variable length columns activity, logically sequential data
might be on non-sequential physical data pages. Because of that, the
database manager performs extra read operations to access data.
Reorganize DB2 tables to eliminate fragmentation and reclaim space by
using the **REORG** utility. Use the **REORG** utility as needed, if **RUNSTATS** takes
longer than typically to complete, or the DB2 REORGCHK command indicates a need for it. Shut down the TADDM server when you run the REORG utility, as during an offline table or index reorganization (data defragmentation), applications can access but not update the data in tables. Since TopologyBuilder runs frequently, even without discovery, such locks might cause unpredictable results within the application.

**RUNSTATS (manual statistics collection)**

The DB2 optimizer uses information and statistics in the DB2 catalog to determine the best access to the database, which is based on the query that is provided. Statistical information is collected for specific tables and indexes in the local database when you run the RUNSTATS utility. When significant numbers of table rows are added or removed, or if data in columns, for which you collect statistics, is updated, the RUNSTATS command must be used to update the statistics. For optimal performance, complete the RUNSTATS task weekly, or daily in situations, where there is high database activity. Lack of updated statistics might cause severe performance degradation within TADDM. You can run the RUNSTATS utility while the TADDM server is running. TADDM requires specific RUNSTATS format that is described later, and DB2 AUTO_RUNSTATS option must be turned off.

**AUTO_RUNSTATS (automatic statistics collection)**

You can enable the automatic statistics collection, also known as auto-runstats, to let DB2 decide whether the TADDM database statistics must be updated. The RUNSTATS utility is run in the background and the database statistics are always up to date.

To enable automatic statistics collection, you must set the parameters AUTO_MAINT, AUTO_TBL_MAINT, and AUTO_RUNSTATS to ON. Run the following command:

```
CONNECT TO <db_alias>
UPDATE DB CONFIG USING AUTO_MAINT ON AUTO_TBL_MAINT ON AUTO_RUNSTATS ON
```

where `<db_alias>` is the name of your database.

**Restriction:** You can use this utility only when you have DB2 APAR IT05733 installed and the `DB2_SELECTIVITY=DSCC` parameter set. The DB2 APAR IT05733 is included in the following and later releases of DB2:

- 9.7 Fix Pack 11
- 10.1 Fix Pack 6
- 10.5 Fix Pack 7

To set the `DB2_SELECTIVITY=DSCC` parameter on DB2 version 10.x, run the following command:

```
db2set -immediate DB2_SELECTIVITY=DSCC
```

**Note:** DB2 9.7 does not support the `-immediate` parameter. To set the `DB2_SELECTIVITY=DSCC` parameter in this version, run the `db2set DB2_SELECTIVITY=DSCC` command and restart DB2.

**Note:** If the TADDM user upgrades the DB2 version in a TADDM installation, then compatible version of the driver should also be updated. You can ask your DBA for db2jcc.jar from the TADDM DB2 server, or you can download the one appropriate for your version of DB2 here: http://www-01.ibm.com/support/docview.wss?uid=swg21363866.

Once you have it, stop TADDM, copy it to `dist/lib/jdbc/`, confirm...
permissions are correct so that the TADDM user can read the file and then start TADDM. Repeat this step on all TADDM servers in your environment.

**DB2 HEALTH MONITOR**

It is good practice to run the DB2 health monitor against the TADDM database to proactively monitor if conditions changed such that **RUNSTATS**, or **REORG**, or any other tuning is required. The health monitor can alert a database administrator of potential system health issues. The health monitor proactively detects issues that might lead to hardware failures, or to unacceptable system performance, or capability. Thanks to the proactive health monitoring, you can address an issue before it becomes a problem that affects system performance.

**DB2 PERFORMANCE ANALYSIS SUITE**

When a DB2 problem is suspected, the Performance Analyst tool can quickly analyze a DB2 snapshot that is taken during the time of the problem and suggest actions. You can download this tool at [https://www.ibm.com/developerworks/community/groups/community/perfanalyst](https://www.ibm.com/developerworks/community/groups/community/perfanalyst).

To take a DB2 snapshot for TADDM, complete the following steps:

1. Connect to your TADDM database from the DB2 server and run the following command:
   ```
db2 -tf updmont.sql
   ```
   where the `updmont.sql` file contains the following entries:
   ```
   UPDATE MONITOR SWITCHES USING BUFFERPOOL ON;
   UPDATE MONITOR SWITCHES USING LOCK ON;
   UPDATE MONITOR SWITCHES USING SORT ON;
   UPDATE MONITOR SWITCHES USING STATEMENT ON;
   UPDATE MONITOR SWITCHES USING TABLE ON;
   UPDATE MONITOR SWITCHES USING UOW ON;
   UPDATE MONITOR SWITCHES USING TIMESTAMP ON;
   RESET MONITOR ALL
   ```

2. After step 1 is completed, run the “DB2 get monitor switches” command to check whether they are all set. They all must have status ON.

3. Run the process, with which you have performance issues.

4. At appropriate intervals, while the slow process is running, run the following command from DB2:
   ```
db2 get snapshot for all on <dbname> > <dbname>-dbsnap.out
   ```
   Run this command from the same window that you ran the command in step 1. This command cannot be run with the use of a script.

5. Run the snapshots by using a different time stamped output file each time. Run them with such intervals that there are three, or four snapshots during the process, but the time between the runs does not exceed 1 hour.

Once the snapshot is collected, analyze them with the Performance Analyst tool, starting with the last snapshot. For example, high CPU and high average execution time on the statement tab for a query that is run many times generally indicates an optimization issue, which can be resolved with the **RUNSTATS** utility. A high overflow percent on the tables tab can indicate a need for the **REORG** utility. Check the buffer pool tab to ensure that there are no alerts, too small buffer pool might lead to poor performance.
Before you begin

After any major maintenance that generates a schema change, for example, after you apply a fix pack, you must generate the TADDM_table_statistics.sql file on the TADDM storage server. The file is needed for the RUNSTATS database maintenance tasks that you must perform regularly. TADDM requires a special format to update database statistics because of a DB2 limitation when handling columns with large common prefixes such as the class names, which are used extensively within TADDM. For this reason, do not use DB2 AUTO_RUNSTATS option, use the RUNSTATS syntax that you generate by completing the following steps. However, if you have DB2 APAR IT05733 installed and the DB2_SELECTIVITY=DSCC parameter set, you can use the AUTO_RUNSTATS option.

Note: The following instructions are given for the Linux and UNIX operating systems. To perform database maintenance on the Windows operating system, use the corresponding .bat script instead of the .sh script.

To generate the TADDM_table_stats.sql file, complete the following steps:

1. Run the following command:
   ```
   cd $COLLATION_HOME/bin
   ```

2. Run the following command, where tmpdir is a directory, where this file can be created:
   ```
   ./gen_db_stats.jy > tmpdir/TADDM_table_stats.sql
   ```
   In a streaming server deployment, run this command on the primary storage server.

3. Copy the file to the database server, or provide it to your database administrator (DBA) to run against the TADDM database as shown in step 2 in the Procedure. Update database statistics at least weekly, or more often, if there are large changes to any tables.

Procedure

To perform maintenance on a DB2 database, complete the following steps:

1. To use the REORG utility, complete the following steps:
   a. On the database server, place the following SQL query, which generates the REORG TABLE commands, in a file:
      ```
      select 'reorg table '||CAST(RTRIM(creator) AS VARCHAR(40))||'.
      "'||substr(name,1,60)||'" ;' from sysibm.systables where creator = 'dbuser' and type = 'T' and name not in ('CHANGE_SEQ_ID')
      order by 1;
      ```
      where dbuser is the value from com.collation.db.user=.
      
      Note: Make sure that the letter case of dbuser is the same as for the value that is specified in the database's sysibm.systables table, column creator.
   b. Stop the TADDM server.
   c. At a DB2 command line, connect to the database and run the following commands:
      ```
      db2 -x -tf temp.sql > cmdbreorg.sql
      db2 -tvf cmdbreorg.sql > cmdbreorg.out
      ```
   d. Make sure that the REORG utility was successful by checking the cmdbreorg.out file for errors.
2. To use the **RUNSTATS** utility, complete the following steps. Automate the process to run at least weekly.
   a. On the database server, run the TADDM-specific **RUNSTATS** command by using the output that you generated earlier:
      
      ```
      db2 -tvf tmpdir/TADDM_table_stats.sql > table_stats.out
      ```
   
   b. Make sure that the **RUNSTATS** utility was successful by checking the `table_stats.out` file for errors.

### DB2 for z/OS database maintenance

These maintenance and tuning guidelines apply to IBM DB2 for z/OS® databases.

#### Procedure

These guidelines assume that the `DB_USER` is the primary DB2 database user ID and `ARCHIVE_USER` is the secondary DB2 database user ID.

1. Use the Discovery Management Console to run a discovery. This method populates the domain database with data.

2. Stop the TADDM server.

3. Generate and run the REORG control statement for each table space used by TADDM.

   ```
   SELECT 'REORG TABLESPACE '||DBNAME||'.'||NAME FROM SYSIBM.SYSTABLESPACE WHERE CREATOR IN ('DB_USER', 'ARCHIVE_USER') ORDER BY 1;
   ```

4. Generate and run the REORG control statement for indexes used by TADDM.

   ```
   SELECT 'REORG INDEX '||CREATOR||'.'||NAME FROM SYSIBM.SYSINDEXES WHERE CREATOR IN ('DB_USER', 'ARCHIVE_USER');
   ```

5. Generate and run the RUNSTATS control statement for table spaces used by TADDM.

   ```
   SELECT 'RUNSTATS TABLESPACE '||DBNAME||'.'||NAME||' INDEX(ALL) SHRLEVEL REFERENCE' FROM SYSIBM.SYSTABLESPACE WHERE CREATOR IN ('DB_USER', 'ARCHIVE_USER') ORDER BY 1;
   ```

6. Regenerate and run the UPDATE index statistics statements for each TADDM DB user.

   ```
   SELECT 'UPDATE SYSIBM.SYSINDEXES SET FIRSTKEYCARDF=FULLKEYCARDF' 
   WHERE NAME = '||IXNAME||' 
   AND CREATOR = '||IXCREATOR||' 
   AND TBNAME = '||NAME||' 
   FROM SYSIBM.SYSINDEXES a 
   WHERE IXCREATOR in ('DB_USER', 'ARCHIVE_USER') 
   AND NAME IN 
   (SELECT IXNAME FROM SYSIBM.SYSKEYS B WHERE A.CREATOR = B.IXCREATOR AND A.NAME = B.IXNAME 
   AND COLNAME = 'PK__JDOIDX') 
   AND TBNAME in 
   (SELECT NAME FROM SYSIBM.SYSTABLES C WHERE A.TBCREATOR = C.CREATOR AND A.TBNAME = C.NAME 
   AND CARDF > 0);
   ```

   where `DB_USER` is the primary DB2 database user ID, and `ARCHIVE_USER` is the secondary DB2 database user ID.
7. Regenerate and run the UPDATE column statistics statements for each TADDM DB user.

```sql
SELECT 'UPDATE SYSIBM.SYSCOLUMNS SET COLCARDF=(SELECT FULLKEYCARDF FROM SYSIBM.SYSCOLUMNS WHERE NAME = ''|''||CAST(RTRIM(name) AS VARCHAR(40))||''|''||CAST(RTRIM(creator) AS VARCHAR(40))||''|''||CAST(RTRIM(tbname) AS VARCHAR(40))||''|''||''|'' FROM sysibm.sysindexes a WHERE tbcreator in (''DB_USER'', ''ARCHIVE_USER'') AND NAME IN (SELECT IXNAME FROM SYSIBM.SYSKEYS B WHERE A.CREATOR = B.IXCREATOR AND A.NAME = B.IXNAME AND COLNAME = ''PK__JDOIDX'') AND TBNAME IN (SELECT NAME FROM SYSIBM.SYSTABLES C WHERE A.TBCREATOR = C.CREATOR AND A.TBNAME = C.NAME AND CARDF > 0);

8. Regularly monitor your largest tables based on your use of TADDM, and adjust their storage attributes if necessary. In particular, monitor the size of the following database tables, which can become large:
   - ALIASES
   - CHANGE_CAUSE_TABLE
   - CHANGE_HISTORY_TABLE
   - MSSOBJLINK_REL
   - PERSOBJ
   - SUPERIORS

Use ALTER statements to modify the PRIQTY and SECPQTY attributes according to the needs of your environment. If appropriate, consider moving tables to separate table spaces.

9. Use the REBIND command on the following packages with the KEEPDYNAMIC(YES) option:
   - SYSLH200
   - SYSLH201
   - SYSLH202

Oracle database maintenance

These maintenance and tuning guidelines apply to Oracle databases.

1. Run the dbms_stats package on the database tables. Oracle uses a cost-based optimizer. The cost-based optimizer needs data to decide on the access plan, and this data is generated by the dbms_stats package. Oracle databases depend on data about the tables and indexes. Without this data, the optimizer must estimate.

Rebuilding the indexes and running the dbms_stats package is critically important for optimal performance with Oracle databases. After the database is populated, this should be done on a regularly scheduled basis, for example, weekly.

   - **REBUILD INDEX**: After many changes to table data, caused by insertion, deletion, and updating activity, logically sequential data might be on
non-sequential physical data pages, so that the database manager must perform additional read operations to access data. Rebuild the indexes to help improve SQL performance.

a. Generate the `REBUILD INDEX` commands by running the following SQL statement on the Oracle database, where `dbuser` is the value from `com.collation.db.user`:

```sql
select 'alter index dbuser.'||index_name||' rebuild tablespace ' ||tablespace_name||'';' from dba_indexes where owner = 'dbuser' and index_type not in ('LOB');
```

This generates all of the `ALTER INDEX` commands that you need to run.

b. Run the commands in SQLPLUS or some comparable facility. Rebuilding the indexes on a large database takes 15 - 20 minutes.

2. **DBMS_STATS**: Use the Oracle RDBMS to collect many different kinds of statistics as an aid to improving performance. The optimizer uses information and statistics in the dictionary to determine the best access to the database based on the query provided. Statistical information is collected for specific tables and indexes in the local database when you run the `DBMS_STATS` command. When significant numbers of table rows are added or removed, or if data in columns for which you collect statistics is updated, run the `DBMS_STATS` command again to update the statistics.

- The `gen_db_stats.jy` program in the `$COLLATION_HOME/bin` directory outputs the database commands for either an Oracle or DB2 database to update the statistics on the TADDM tables. The following example shows how the program is used:

  a. cd `$COLLATION_HOME/bin`
  
  b. Run this SQL statement, where `tmpdir` is a directory where this file is created:

  ```sql
  ./gen_db_stats.jy > tmpdir/TADDM_table_stats.sql
  ```

  In a streaming server deployment, run this statement on the primary storage server.

  c. After this is complete, copy the file to the database server and run the following command:

  ```shell
  SQL > @\{file\}
  ```

  d. Run the commands in SQLPLUS or some comparable facility.

3. Buffer pool: A buffer pool or buffer cache is a memory structure inside Oracle System Global Area (SGA) for each instance. This buffer cache is used for caching data blocks in the memory. Accessing data from the memory is significantly faster than accessing data from disk. The goal of block buffer tuning is to efficiently cache frequently used data blocks in the buffer cache (SGA) and provide faster access to data. Tuning block buffer is a key task in any Oracle tuning initiative and is a part of the ongoing tuning and monitoring of production databases. The Oracle product maintains its own buffer cache inside the SGA for each instance. A properly sized buffer cache can usually yield a cache hit ratio over 90%, which means that nine requests out of ten are satisfied without going to disk. If a buffer cache is too small, the cache hit ratio will be small and more physical disk I/O results. If a buffer cache is too big, parts of the buffer cache are underutilized and memory resources are wasted.

<table>
<thead>
<tr>
<th>Number of CIs</th>
<th>Guideline buffer pool size</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 500,000</td>
<td>38000</td>
</tr>
</tbody>
</table>

Table 37. Buffer pool size guidelines (`db_cache_size`)
Table 37. Buffer pool size guidelines (db_cache_size) (continued)

<table>
<thead>
<tr>
<th>Number of CIs</th>
<th>Guideline buffer pool size</th>
</tr>
</thead>
<tbody>
<tr>
<td>500,000 - 1,000,000</td>
<td>60000</td>
</tr>
<tr>
<td>&gt; 1,000,000</td>
<td>95000</td>
</tr>
</tbody>
</table>

4. You can double the size of maximum open cursors, if the discovery or bulkloading takes too long to complete and NRS contain the following error:

```
com.ibm.tivoli.namereconciliation.service.NrsService
getAliases(masterGuid)
SEVERE: NOTE *** SQL State = 60000. SQL Code = 604. SQL Message =
ORA-00604: error occurred at recursive SQL level 1
ORA-01000: maximum open cursors exceeded
ORA-01000: maximum open cursors exceeded
```

5. Verify that the versions of your Oracle JDBC driver and the Oracle server are the same. If necessary, replace the Oracle JDBC driver file in the following locations.

**Note:** This only applies when BIRT Report Viewer is enabled.

- TADDM 7.3.0 - $COLLATION_HOME/deploy-tomcat/birt-viewer/WEB-INF/platform/plugins/org.eclipse.birt.report.data.oda.jdbc_2.2.1.r22x_v20070919/drivers/
- TADDM 7.3.0.1, and later - $COLLATION_HOME/apps/birt-viewer/WEB-INF/platform/plugins/org.eclipse.birt.report.data.oda.jdbc_2.2.1.r22x_v20070919/drivers/
- $COLLECTION_HOME/lib/jdbc/

**Database communication**

When the database is unavailable, the storage server tries to establish the connection again.

When there is no connection between the database and the storage server, the storage server waits as long as it is specified in the com.ibm.cdb.db.timeout property and makes an attempt to connect to the database. The number of the retry attempts to establish the connection is specified in the com.ibm.cdb.db.max.retries property.

To learn more about the database properties, go to the [Database properties](#) section.

**Discovery performance tuning**

You can update the com.collation.discover.dwcount, com.collation.discover.observer.topopumpcount, and com.ibm.cdb.discover.observer.topopump.threshold properties in the collation.properties file to influence the discovery rate and the rate at which the discovery results are stored in the TADDM database, or to limit the number of threads that are responsible for storing data.

For details about these properties, see "Performance properties" on page 86.

If you increase the values of the com.collation.discover.dwcount or com.collation.discover.observer.topopumpcount properties, you might also have to increase the amount of installed memory by increasing the maximum heap size setting for the following Java Virtual Machines (JVMs):

**For the dwcount property:**

139
In a streaming server deployment:
  – Discover
  – DiscoverService

In a domain server deployment:
  – Discover

For the topopumpcount property:
  • In a streaming server deployment:
    – StorageService
  • In a domain server deployment:
    – Topology

For more information, see “Java Virtual Machine: IBM parameters tuning” on page 142.


**Discovery rate tuning**

The discovery rate attribute is the area with the most potential for tuning. The property with the most impact on performance is the number of discover worker threads. You can also use the in-progress sensors to monitor performance or improve it by specifying the session pool sizes.

A discover worker thread is a thread that runs sensors. The following property specifies the maximum number of discover worker threads:

```
com.collation.discover.dwcount=32
```

If the server has sufficient spare capacity, you can increase this number and allow more sensors to run in parallel.

**In-progress sensors**

To monitor performance, you can look at the in-progress sensors. An in-progress sensor can be in one of three stages of execution:

- **started**
  A sensor in this stage is discovering a CI or CIs.

- **discovered**
  A sensor in this stage finished discovering a CI or CIs, but is waiting for its results to be saved in the data store.

- **storing**
  A sensor in this stage saves its discovery results in the database.

To order the in-progress sensors by execution stage, click the Description column.

By observing a discovery run and comparing the number of in-progress sensors that are in the started stage versus the number of in-progress sensors in the discovered or storing stages, you can assess whether attribute discovery is faster or slower than attribute storage for a particular environment. As with all changes to the `collation.properties` file, you must restart the server for the change to take effect.
Examples:

In-progress sensors: STARTED, DISCOVERED, STORING.

If the number of (DISCOVERED + STORING) is smaller than STARTED, it might indicate that discovery is the performance bottleneck.

If the number of (DISCOVERED + STORING) exceeds STARTED, it might indicate that storage is the performance bottleneck.

Session and gateway pool sizes

To discover attributes of a particular CI, a sensor requires an SSH or WMI session with its host computer. To improve performance, these sessions are pooled and cached. The default pool sizes are sufficient in most cases but if they are not large enough, they can limit the discovery rate. You can change the following property to true to monitor for this condition:

\[\text{com.collation.platform.session.ExtraDebugging=false}\]

You must restart the discovery server for the change to take effect. After you run a discovery, you can search the DiscoverManager logs for waiting time issues that are related to the session pools. To do so, search the logs for pool lock. The following is an example of performance degradation that is caused by session pool contention:

```
2006-08-04 16:11:50,733 DiscoverManager [DiscoverWorker-34]
WindowsComputerSystemAgent(192.168.16.181)
INFO session.SessionClientPool -
Session client [3x ssh2:/admlxz@151.179.84.85]#9612508
waited 158.682 seconds for pool lock
```

You can increase the pool size if the waiting time for a session is too long. There are two ways to do it. You can globally change the pool size for sessions per host by editing the following property in the `collation.properties` file:

\[\text{com.collation.platform.session.PoolSize=3}\]

It is however unlikely that the contention concerns the sessions for all or even most hosts in the environment. The contention is likely restricted to a smaller number of larger hosts that are used by many sensors. The discovery server uses a scoped property which means that many of the properties in the `collation.properties` file use one value for general targets and a different one for specific targets. You can adjust this property by adding an IP address or a discovery server scope name, like in the following example:

\[\text{com.collation.platform.session.PoolSize.10.10.250.1=20}\]

In this case, the pool size for 10.10.250.1 is 20 but for all other hosts it is 3. You can look at the log messages like the one from the DiscoverManager logs and determine for which hosts the default session pool size is insufficient, and make the appropriate changes to the `collation.properties` file.

A related setting is the gateway pool size. It sets the number of sessions that are allowed between the discovery server and the Windows gateway. You can specify it by editing the following property:

\[\text{com.collation.platform.session.GatewayPoolSize=10}\]

If your environment consists mainly of Windows computer systems, adjust this property upwards so it is equal to the number of discover worker threads.
Storage tuning

Storage is the second major area for tuning. If the number of sensors in the storing stage is approximately the value of the property that specifies the number of parallel storage threads, storage of the discovery results is causing the performance bottleneck. To improve performance, you can also limit the number of threads that are responsible for storing data.

The following property specifies the number of parallel storage threads. It is one of the main settings for controlling the discovery storage performance:

com.collation.discover.observer.topopumpcount

To improve storage performance when the topology agents are running, you can limit the number of threads that are responsible for storing data during a discovery. As a result, a discovery takes less time to complete. To specify the limit of threads that are running, edit the following properties in the collation.properties file:

com.ibm.cdb.discover.observer.topopump.threshold
   This property specifies the number of storage threads to limit.

com.ibm.cdb.discover.observer.topopump.threshold.<agentGroupName>
   This property specifies the number of storage threads to limit when the specified agent group is running.

The following table shows how much the com.ibm.cdb.discover.observer.topopump.threshold property can improve the discovery performance. The calculations concern a database with 76 000 configuration items.

<table>
<thead>
<tr>
<th>Threshold property value</th>
<th>Percentage time improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>55</td>
</tr>
<tr>
<td>0.5</td>
<td>33</td>
</tr>
<tr>
<td>0.7</td>
<td>13</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Java Virtual Machine: IBM parameters tuning

You can set Java Virtual Machine (JVM) parameters that reduce fragmentation of the Java heap and can help improve performance.

Fragmentation of the Java heap can occur as the number of objects that are processed increases. There are a number of parameters that you can set to help reduce fragmentation in the heap.

- A kCluster is an area of storage that is used exclusively for class blocks. It is large enough to hold 1280 entries. Each class block is 256 bytes long. This default value is usually too small and can lead to fragmentation of the heap. Set the kCluster parameter, -Xk, as follows to help reduce fragmentation of the heap. These are starting values and might have to be tuned in your environment. An analysis of a heap dump would be best to determine the ideal size.
  - Topology: -Xk8300
  - EventsCore: -Xk3500
  - DiscoverAdmin: -Xk3200
  - Proxy: -Xk5700
Implement these changes in the `collation.properties` file by adding entries in the JVM Vendor Specific Settings section. For example, to implement these changes for the Topology server, add the following line:

```
com.collation.Topology.jvmargs.ibm=-Xk8300
```

Another option for fragmentation issues is to allocate some space specifically for large objects; > 64K. Use the `-Xloratio` parameter. For example:

```
-Xloratio0.2
```

This command reserves x% of the active Java heap (not x% of -Xmx but x% of the current size of the Java heap), to the allocation of large objects (≥64 KB) only. If changed, -Xmx should be changed to make sure that you do not reduce the size of the small object area. An analysis of a heap dump would be best to determine the ideal setting for this parameter.

There are a few additional parameters that can be set that affect Java performance. To change an existing JVM option to a different value, edit one of the following files:

- For a domain server in TADDM 7.3.0, the `$COLLATION_HOME/deploy-tomcat/ROOT/WEB-INF/cmdb-context.xml` file.
- For a domain server in TADDM 7.3.0.1, and later, the `$COLLATION_HOME/apps/ROOT/WEB-INF/cmdb-context.xml` file.
- For a synchronization server in TADDM 7.3.0, the `$COLLATION_HOME/deploy-tomcat/ROOT/WEB-INF/ecmdb-context.xml` file.
- For a synchronization server in TADDM 7.3.0.1, and later, the `$COLLATION_HOME/apps/ROOT/WEB-INF/ecmdb-context.xml` file.
- For a discovery server in TADDM 7.3.0, the `$COLLATION_HOME/deploy-tomcat/ROOT/WEB-INF/discovery-server-context.xml` file.
- For a discovery server in TADDM 7.3.0.1, and later, the `$COLLATION_HOME/apps/ROOT/WEB-INF/discovery-server-context.xml` file.
- For a storage server in TADDM 7.3.0, the `$COLLATION_HOME/deploy-tomcat/ROOT/WEB-INF/storage-server-context.xml` file.
- For a storage server in TADDM 7.3.0.1, and later, the `$COLLATION_HOME/apps/ROOT/WEB-INF/storage-server-context.xml` file.

To edit one of these files to change the settings for one of the TADDM services, first find the service in the file. The following example shows the beginning of a service definition in the XML file:

```xml
<bean id="Discover"
  class="com.collation.platform.service.ServiceLifecycle" init-method="start"
  destroy-method="stop">
  <property name="serviceName">
    <value>Discover</value>
  </property>
</bean>
```

Within the definition, there are some elements and attributes that control the JVM arguments. For example:

```xml
<property name="jvmArgs">
  <value>-Xms8M;-Xmx512M;
   -Djava.nio.channels.spi.SelectorProvider=sun.nio.ch.PollSelectorProvider
  </value>
</property>
```
The JVM arguments can be set as a semicolon separated list in the following element:

<property name="jvmArgs"><value>

You can also modify the JVM properties that are in the collation.properties file. Those properties can have one of the following forms:

**com.collation.JVM.jvmargs.VENDOR**
Such property is added to the values that are read from the *-config.xml file.

**com.collation.jvmargs.VENDOR**
Such property is added to all TADDM JVMs.

**com.collation.JVM.jvmargs**
Such property overwrites all the values that are specified in the *-config.xml file.

where

- JVM is Proxy, Topology, EventsCore, ExcmdbCore, DiscoverAdmin, StorageService, DiscoveryService
- VENDOR is ibm or sun

### Java Virtual Machine properties tuning

In the collation.properties file, the default values for the Java Virtual Machine (JVM) properties that apply to the TADDM Discovery Management Console are based on the number of server equivalents (SEs) in your environment.

#### Default values for JVM properties that apply to Discovery Management Console

- Small environment (fewer than 1000 SEs):
  - com.collation.gui.initial.heap.size=128m
  - com.collation.gui.max.heap.size=512m
- Medium environment (1000–2500 SEs):
  - com.collation.gui.initial.heap.size=256m
  - com.collation.gui.max.heap.size=768m
- Large environment (2500–5000 SEs):
  - com.collation.gui.initial.heap.size=512m
  - com.collation.gui.max.heap.size=1024m

### Network tuning

After a system is implemented, the network should be monitored to ensure that its bandwidth is not being consumed more than 50%.

The network can influence the overall performance of your application, and it is typically a factor in performance when a time delay exists in the following situations:

- A delay between when a client system sends a request to the server and when the server receives this request
- A delay between when the server system sends data back to the client system and when the client system receives this data
DNS tuning

TADDM is sensitive to the performance of the deployed DNS infrastructure. Even if DNS performance is adequate for other applications, some configuration might be required to optimize performance for TADDM.

TADDM performs a large number of DNS lookup queries in order to resolve meaningful display names for components and events. Unlike most other applications, TADDM uses primarily reverse lookups (mapping IP addresses to names) rather than forward lookups (mapping names to IP addresses).

Because of this usage pattern, problems with DNS performance can have a larger effect on TADDM performance than on other applications. For example, a DNS response time of 500 milliseconds would probably not significantly affect a typical application, but might cause noticeable performance issues for TADDM because of the large number of DNS queries it performs. Additionally, because other applications perform only forward lookups, a performance problem with reverse lookups would not affect most applications, but would affect TADDM.

In general, performance problems with the DNS infrastructure should be addressed in order to benefit all consumers of DNS services. If this is not possible, there are several ways in which you can mitigate the effect of DNS performance problems on TADDM:

- Make sure in-addr.arpa delegation for reverse lookups is configured correctly. Delegation problems can cause long pauses or hangs during reverse lookups as the TADDM server tries to reach servers that do not exist. This type of configuration issue affects only applications (like TADDM) that perform reverse lookups.
- Set up at least one caching/forwarding DNS server on a TADDM server system, and configure the TADDM servers to use that DNS server for lookups. This allows DNS lookups to be cached in the local TADDM environment, based on the TTL rules for the zones. This type of server is stateless and therefore requires minimal maintenance and adds little overhead.
- Set up at least one DNS slave server on a TADDM server system, and configure the TADDM servers to use that DNS server for lookups. This allows DNS lookups to be performed within the local TADDM environment without communicating with the broader DNS infrastructure. A DNS slave server maintains state automatically and therefore requires minimal maintenance and adds little overhead.
- Use an alternative method for lookups, such as a hosts file, instead of DNS. (This approach can have significant maintenance requirements.)

Note: Do not change the default DNS cache parameters in the java.security file. Although caching parameters can affect DNS performance, changes to this configuration file are not preserved when TADDM maintenance fixes are applied. Instead, use one of the approaches described in this topic to optimize DNS performance.

Synchronization server tuning

The performance of the synchronization server is highly dependent on database processing and therefore, on database maintenance and tuning. If you experience performance problems with synchronization processing, refer to the information on database tuning, and especially note the buffer pool settings for DB2 databases, the buffer cache settings for Oracle databases, and the information about database maintenance.
Specifically for the synchronization server, update the DB2 database configuration by entering the following command:

```
UPDATE DATABASE CONFIG FOR TADDM USING
  UTIL_HEAP_SZ 5000
  LOGBUFFS 1024
  LOCKLIST 20000
  SORT_HEAP 2048
  PCKCACHESZ AUTOMATIC
```

### Windows system tuning

To assign more memory for TADDM services, tune the Windows systems.

Complete the following tasks:

- The system paging file must not be located on the same drive as the operating system. If possible, put the system paging file on a separate disk drive.
- Configure the database and application server to maximize data for networking applications.

### Reporting

You can create and add custom reports to the Data Management Portal using external report viewers, JSP report viewers, or the BIRT reporting system.

### External report viewers

You use an external report viewer to run an external program that generates a report. The external program uses the TADDM API through a command line to access data. The report is then shown in the user interface.

#### Creating the external report viewer logic

An external report can be implemented inside any executable program. Examples are a Perl script, a shell script, or Java program. The external program must output a valid HTML file through standard output for the resulting report to appear in the Data Management Portal.

#### About this task

A typical implementation of an external report viewer uses a shell script to query the TADDM API and output the XML results of the query to a temporary file. The shell script then starts an XSLT processor to transform the query results into HTML output that is output to STDOUT.

**Important:** External report viewers that use the TADDM API must supply credentials to the command line program `api.sh` script on Linux and UNIX and the `api.bat` file on Windows. Because the credentials are command line arguments to the `api.sh` script and `api.bat`, they might be visible to other users of the system via process lists. To prevent disclosure of sensitive passwords, it might be useful to set up a dummy account that has read access to the objects that should appear in externally-generated reports.

The following example is a simple external report bourne shell script implementation. Copy the following contents into a new file, `$COLLATION_HOME/sdk/bin/appServers.sh`, and make the file readable and executable by the user that the TADDM server runs under:
#!/bin/sh
# Set environment variables for called scripts
export COLLATION_HOME=/opt/ibm/taddm/dist

# Invoke the query via API and output to $COLLATION_HOME/sdk/bin/appServers.xml
# NOTE: Change 'restrictedUser' and 'restrictedPassword' to your dummy account credentials.
sh $COLLATION_HOME/sdk/bin/api.sh -l log -H localhost -u restrictedUser -p restrictedPassword \ find AppServer > $COLLATION_HOME/sdk/bin/appServers.xml

# Invoke the XSLT processor
sh $COLLATION_HOME/sdk/bin/xslt.sh -XSL $COLLATION_HOME/sdk/bin/appServers.xsl

The following example is of the appServers.xsl style sheet used to transform the appServers.xml file generated by the shell script. The report displays application server names and their product versions. Copy the contents into a new file, $COLLATION_HOME/sdk/bin/appServers.xsl and make the file readable by the user the TADDM server runs under.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version='1.0' xmlns:xsl='http://www.w3.org/1999/XSL/Transform' xmlns:coll='urn:www-collation-com:1.0' xmlns:xhtml='http://www.w3.org/1999/xhtml'>
  <xsl:variable name="nl">
    </xsl:text>
  </xsl:variable>
  <xsl:variable name="pageheadertext">Simple Application Server report</xsl:variable>
  <xsl:variable name="pagefootertext">End Simple Application Server report</xsl:variable>

  <xsl:template match="/">
    <html>
      <head>
        <link rel="stylesheet" type="text/css" media="all" href="styles.css" />
      </head>
      <body>
        <h3><xsl:value-of select="$pageheadertext"/></h3>
        <table border="1" width="100%">
          <tr>
            <th>Product Version</th>
            <th>Name</th>
          </tr>
          <xsl:apply-templates select="document('appServers.xml')/coll:results"/>
        </table>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```
To test the report logic, run the appServer.sh script from a command line. Valid HTML output is displayed.

**Adding the external report viewer to the Data Management Portal**

Reports are added to the Data Management Portal by modifying the reports.xml file. The reports.xml file is located in the $COLLATION_HOME/etc/cdm/xml/ directory.

**Procedure**

To add the external report viewer to the Data Management Portal, complete the following steps:

1. Using a text editor, open the $COLLATION_HOME/etc/cdm/xml/reports.xml file.
2. In the reports.xml file, specify the report descriptor, report group, report name, and external script for the report definition. The following example shows how to create an external report named Application Servers that is located in the Inventory Reports group and specifies the sdk/bin/appServers.sh file:

   ```xml
   <bean class="com.collation.cdm.reports.viewer.ExternalReportViewer" id="AppServers1">
   <property name="reportGroup"><value>Inventory Reports</value></property>
   <property name="reportName"><value>Application Servers</value></property>
   <property name="script"><value>sdk/bin/appServers.sh</value></property>
   </bean>
   ```

3. Save the $COLLATION_HOME/etc/cdm/xml/reports.xml file.
4. The report is now displayed in the Data Management Portal.

**JSP report viewers**

A JSP report viewer provides additional flexibility and security for users who have knowledge of Java Server Pages (JSP) authoring. The reporting logic, including any API access, is placed in a JSP page that is then rendered by the Data Management Portal. When using JSP report viewers, security credentials are automatically inherited from the logged in user.

**Creating the JSP report viewer logic**

The logic for a JSP report viewer is contained in a JSP that is called by the Data Management Portal. A typical implementation of a JSP report uses a Java helper class called TMSDataHelper to query the TADDMM API. The results of the query are objects that can be manipulated using Java methods. For more information about the TADDMM API and model, consult the SDK documentation in $COLLATION_HOME/sdk/doc.

**About this task**

The following example is a simple JSP report viewer implementation. Copy the following contents into a new file, $COLLATION_HOME/deploy-tomcat/reports.war/WEB-INF/view/custom.jsp if you use TADDMM 7.3.0, or $COLLATION_HOME/apps/reports.war/WEB-INF/view/custom.jsp if you use TADDMM 7.3.0.1, and later, and make the file readable and executable by the user that runs the TADDMM server.

The following example shows the appServers.xsl style sheet used to transform the appServers.xml file generated by the shell script. The report displays application server names and their product versions. Copy the contents into a new file, $COLLATION_HOME/sdk/bin/appServers.xsl and make the file readable by the user that runs the TADDMM server.

```
<%@ page language="java" %>
<%@ page import="com.collation.cdm.common.util.TMSDataHelper" %>
<%@ page import="java.lang.StringBuffer" %>
<%@ page import="com.collation.cdm.reports.util.ReportsParser" %>
```
java.util.Locale locale = com.collation.cdm.common.util.CDMUtil.checkLocale(request.getLocale());
    if (null == session.getAttribute(org.apache.struts.Globals.LOCALE_KEY)) {
        session.setAttribute(org.apache.struts.Globals.LOCALE_KEY, locale);
    }

//TMSDataHelper is a utility class for running MQL queries against the DB
TMSDataHelper tms = new TMSDataHelper(locale);

//Perform a query for all ComputerSystems
ModelObject dataIn[] = tms.doModelObjectQuery("SELECT * FROM ComputerSystem",null);

//Build an HTML report based on the API output
StringBuffer output = new StringBuffer();
    output.append("<p>");
    output.append("<table border="1"">");
        int c = 0;
        int s = dataIn.length;
        while (cs) {
            ComputerSystem tmo = (ComputerSystem)dataIn[c];
            String csName = null;
            String csLabel = null;
            if (tmo.hasName()) {
                try {
                    csName = tmo.getName();
                } catch (AttributeNotSetException e) {
                    csName = "unknown";
                } 
            }
            if (tmo.hasSignature()) {
                try {
                    csLabel = tmo.getSignature();
                } catch (AttributeNotSetException e) {
                    csLabel = "";
                }  
            
            output.append("<tr><td colspan="2" bgcolor="#9999FF">");
            output.append("ComputerSystem<br>");
            output.append("Name: " + csName + "<br>");
            output.append("Signature: " + csLabel);
            output.append("</tr></table>" );
            c++;
        }
    output.append("</table>");
    String bpstring = output.toString();
</html>
<body>
Adding the JSP report viewer to the Data Management Portal

Reports are added to the Data Management Portal by modifying the reports.xml file. The reports.xml file is located in the $COLLATION_HOME/etc/cdm/xml/ directory.

Procedure

To add the JSP report viewer to the Data Management Portal, complete the following steps:

1. Using a text editor, open the $COLLATION_HOME/etc/cdm/xml/reports.xml file.
2. In the reports.xml file, specify the report descriptor, report group, report name, and external script for the report definition. The following example shows how to create an external report named Custom Report that is located in the Inventory Reports group and specifies the /WEB-INF/view/custom.jsp script:

   ```xml
   <bean class="com.collation.cdm.reports.viewer.JSPReportViewer" id="CustomReport">
     <property name="reportGroup">Inventory Reports</property>
     <property name="reportName">Custom Report</property>
     <property name="script">/WEB-INF/view/custom.jsp</property>
   </bean>
   ```

3. Save the $COLLATION_HOME/etc/cdm/xml/reports.xml file.

Reporting with Tivoli Common Reporting

As viewing BIRT reports in BIRT Report Viewer is not safe and is disabled by default, you can import the BIRT reports for TADDM into Tivoli Common Reporting. It enables cross-product reporting that includes TADDM data. You can also use Tivoli Common Reporting features such as report scheduling, or use Tivoli Common Reporting as a central repository for reports.

For some tasks, the steps you must complete differ depending on the version of Tivoli Common Reporting that you are using or the database that you are using.

Fix Pack 1  If you have TADDM 7.3 Fix Pack 1, or later, see also best practices guide The enhanced Cognos model in TADDM 7.3 FPx.

Tivoli Common Reporting overview

The Tivoli Common Reporting tool is a reporting feature provided with certain Tivoli products, and provides a centralized approach to viewing and administering reports with a consistent look and feel across multiple products.

Tivoli Common Reporting includes a data store for storing and organizing reports, and interfaces for managing, running, scheduling, and viewing reports. Tivoli Common Reporting uses both the Cognos® and BIRT runtime engines.

Important: Tivoli Common Reporting is provided on the IBM Jazz™ for Service Management installation disc. If you do not plan to install IBM Jazz for Service Management, you can use the integrated BIRT reporting capability.

If you already have Tivoli Common Reporting installed on your system, you can optionally import the predefined TADDM reports, which are compatible with Tivoli Common Reporting. You can then use Tivoli Common Reporting as a central
repository for Tivoli product reports. You can also use enhanced reporting options, including cross-product reporting, role-based security, and report scheduling.

To see supported versions of the products, go to the “Supported versions” on page 183 section.

Note: If you are using TADDM with IBM Tivoli Change and Configuration Management Database (CCMDB) or IBM SmartCloud Control Desk, see the CCMDB or IBM SmartCloud Control Desk documentations for information about which Tivoli Common Reporting versions are supported.

For more information about Tivoli Common Reporting, go to https://www.ibm.com/developerworks/community/groups/service/html/communityview?communityUuid=9caf63c9-15a1-4a03-96b3-8fc700f3a364.

Installing Tivoli Common Reporting and IBM Cognos Framework Manager
You must install Tivoli Common Reporting and IBM Cognos Framework Manager.

Procedure
To install Tivoli Common Reporting and IBM Cognos Framework Manager, complete the following steps:

1. Install Tivoli Common Reporting using the default options presented to you. If you are using an Oracle database, you must use Tivoli Common Reporting 2.1, or 3.1.
2. Install the IBM Cognos Framework Manager package available in the CognosModeling folder. Use the default options presented to you.
3. If available, install also the security patch available in the CognosModelingFix folder. Use the default options presented to you.

Installing and configuring the database client
If you have installed Tivoli Common Reporting on a computer other than the TADDM database server, you must install a database client to connect to the database. You can use either a DB2 or Oracle database client, depending on the TADDM database type. If you have installed Tivoli Common Reporting on the same server as the TADDM database, you do not need to install a database client.

Procedure
To install and configure the database client, complete the following steps:

Complete one of the following tasks:
• If you want to use the DB2 database client, complete the following steps:
  1. Install the DB2 client on the machine on which TCR is installed, using the default options presented.
  2. Ensure that the TADDM database has been cataloged. This step is necessary for Tivoli Common Reporting to successfully connect to the DB2 server, using the DB2 client.
• If you want to use the Oracle database client, complete the following steps to install and configure it using the Oracle Universal Installer wizard and the Oracle Net Configuration Assistant wizard:
  1. In the Select Installation Type page of the Oracle Universal Installer wizard, select Administrator as the installation mode.
2. In the Specify Home Details page, specify the name of the installation and the path to the location to where you want to install the product.

3. In the Product-Specific Prerequisite Checks page, ensure that each of the requirements for installation and configuration are met. Do not continue with the installation until each check has a status of Succeeded.

4. In the Welcome page of the Oracle Net Configuration Assistant wizard, ensure that the Perform typical configuration check box is not selected.

5. In the Naming Methods Configuration, Select Naming Method page, set Local Naming as the naming method.

6. In the Net Service Name Configuration, Service Name page, type the service name of your remote Oracle database server, for example, ORCL.

7. In the Net Service Name Configuration, Select Protocols page, select TCP as the protocol to use to connect to the database.

8. In the Net Service Name Configuration, TCP/IP Protocol page, type the host name of the computer on which the database is running. Select Use the standard port number of 1521.

9. In the Net Service Name Configuration, Test page, select Yes, perform a test.
   If your database user name and password are correct, the following text is displayed:
   Connecting... Test successful.
   If a successful connection to the database is not made, you might need to change your login credentials. To change the database login credentials click Change Login and specify a valid database user name and password.

10. In the Net Service Name Configuration, Net Service Name page accept the default service name, which should be the service name you specified earlier.

11. Create a Windows system variable called TNS_ADMIN and set the value to the full path of the folder containing the tnsnames.ora file. During installation the tnsnames.ora file is created in the %ORACLE_HOME%/client_1/NETWORK/ADMIN folder, for example C:/oracle/product/10.2.0/client_1/NETWORK/ADMIN.

12. Set the TNS_ADMIN variable in the startTCRserver.sh/bat script to point to the location of the tnsnames.ora file, for example %ORACLE_HOME%/client_1/NETWORK/ADMIN.

13. Reboot the computer to ensure that the new system variable is available.

**Configuring IBM Cognos Framework Manager**

You must update IBM Cognos 10 Framework Manager properties with appropriate values.

**About this task**

**Note:** The following procedure applies to configuring IBM Cognos 10 Framework Manager for Tivoli Common Reporting 3.1. However, it is the same for IBM Cognos 8 Framework Manager for Tivoli Common Reporting 2.1.

When you install Tivoli Common Reporting the IBM Cognos Configuration program is installed and some property values are updated. When you install IBM Cognos 10 Framework Manager, a different version of the IBM Cognos Configuration program is installed, but not all of the properties are updated. You must manually copy some property values from the Tivoli Common Reporting
version of IBM Cognos Configuration to the IBM Cognos 10 Framework Manager version of IBM Cognos Configuration.

Procedure

To configure IBM Cognos 10 Framework Manager, complete the following steps:

1. Open the version of IBM Cognos Configuration installed by Tivoli Common Reporting. To open this program click Start > Programs > Tivoli Common Reporting 3.1 > IBM Cognos Configuration.
2. Open the version of IBM Cognos Configuration installed by IBM Cognos 10. To open this program click Start > Programs > IBM Cognos 10 > IBM Cognos Configuration.
3. For each version of IBM Cognos Configuration, click Local Configuration > Environment.
5. Copy the value of the External dispatcher URI property from the Tivoli Common Reporting version of IBM Cognos Configuration to the Dispatcher URI for external applications property of the IBM Cognos 10 version of IBM Cognos Configuration. The URI syntax is: http://tcrhost:16310/tarf/servlet/dispatch.
6. Save changes made to the IBM Cognos 10 version of IBM Cognos Configuration.

Generating the TADDM model

Fix Pack 1

You can generate the TADDM model to have the up-to-date snapshot of the TADDM database content, including definitions of all extended attributes. If you do not use extended attributes, you can skip this procedure and use the prebuilt TADDM Cognos model file, the $COLLATION_HOME/etc/reporting/tcr/model.xml file.

Before you begin

The generated TADDM model includes all Common Data Model classes that are supported by TADDM and extended attributes definitions that are stored in the TADDM database. You can publish the TADDM model in Tivoli Common Reporting server and use it in Cognos reports. You can generate the TADDM model many times. Each time the model is regenerated, it contains the updated TADDM database content.

Notes:

• If extended attribute definitions are removed from the TADDM database after the TADDM model is published in Tivoli Common Reporting server, the Cognos reports that use them might stop working.
• If you use the Windows operating system, change the extension of the scripts that are used in the following procedure from .sh to .bat.
Procedure

1. On the TADDM server, open the $COLLATION_HOME/bin directory.
2. Refresh the extended attributes views by completing the following steps:
   a. If you created any extended attribute views, remove them by running the following command:
      ./extattr_views.sh remove
   b. Generate SQL scripts with extended attribute views definitions by running the following command:
      ./extattr_views.sh scripts
   c. Create the extended attribute views with the use of the generated SQL scripts by running the following command:
      ./extattr_views.sh create
3. To generate the Cognos model file, run the following command:
   ./genCognosModel.sh

   The generated TADDM model is stored in the model.xml file and it is placed in the $COLLATION_HOME/etc/reporting/tcr directory. The log messages of the command are in the $COLLATION_HOME/log/genCognosModel.log file.

What to do next

You can publish the generated TADDM model in Tivoli Common Reporting server by using IBM Cognos Framework Manager. For more information, see "Publishing the model using IBM Cognos Framework Manager" on page 158. For more information about extended attributes views, see the Extended attributes views topic in the TADDM SDK Developer’s Guide.

Importing the model and sample reports into Tivoli Common Reporting

You can import sample TADDM reports into Tivoli Common Reporting version 2.1 and 3.1.

About this task

This procedure applies to Tivoli Common Reporting version 2.1.

Procedure

To import the model and sample reports into Tivoli Common Reporting 2.1, complete the following steps:

1. Copy the $COLLATION_HOME/etc/reporting/TADDMPackage.zip package from the TADDM server to the TCRComponent/cognos/deployment folder of the Tivoli Common Reporting server.
2. Open Tivoli Common Reporting home page.
3. Click Reporting > Common Reporting.
4. From the Launch menu, select Administration. The Administration pane is displayed.
5. Click the Configuration tab.
6. Click the New Import icon. The New Import wizard is displayed.
7. From the list of available packages, select TADDMPackage. Click Next.
8. Optional: In the **Description** field, type a description of the package. Click **Next**.

9. Select the check box next to the package name.

10. In the **Options** section, click **The owner from the source** and **New and existing entries**. From the **Recording level** menu, select **Basic**. Click **Next**.

11. Click **Save and run once**. Click **Next**.

**About this task**

This procedure applies to **Tivoli Common Reporting version 3.1**.

**Procedure**

To import the model and sample reports into Tivoli Common Reporting 3.1, complete the following steps:

1. Copy the `$COLLATION_HOME/etc/reporting/TADDMPackage.zip` package from the TADDM server to the `reporting/cognos/deployment` folder of the JazzSM installation.

2. Open Tivoli Common Reporting home page.

3. Click **Reporting > Common Reporting**.

4. From the **Launch** menu, select **IBM Cognos Administration**. The Administration pane is displayed.

5. Click the **Configuration** tab.

6. Go to Content Administration. Click the **New Import** icon. The New Import wizard is displayed.

7. From the list of available packages, select **TADDMPackage**. Click **Next**.

8. Optional: In the **Description** field, type a description of the package. Click **Next**.

9. Select the check box next to the package name. Click **Next**.

10. In the **Entry ownership** section, click **The owner from the source** and **New and existing entries**. From the **Recording level** menu in the **Deployment record** section, select **Basic**. Click **Next**.

11. Verify that the provided values are correct. Click **Next**.

12. Click **Save and run once**. Click **Finish**.

13. Click **Run**.

**Data views in the TADDM model**

You can generate reports from the TADDM data model file, `model.xml` file.

The data model is organized into several namespaces. The namespace is a logical container in which all the names are unique. Each namespace contains query subjects, query items, and objects. The following namespaces are present after you import the TADDM `model.xml` file:

### CDM namespaces

These views contain the query subjects for almost all Common Data Model classes, including discovery-related classes, split into several namespaces by their package names. Package names are sorted alphabetically. The Simplified Model classes are distinguished by the `simple` prefix in the namespace name. You can use this data to generate reports that contain different types of CDM objects.
The query subjects in CDM namespaces contain predefined relationships that are related to the Parent attributes. For example, the app.j2ee.J2EEDomain class has the attribute Servers of the type app.j2ee.J2EEServer[]. And the app.j2ee.J2EEServer class has the attribute Parent of the app.j2ee.J2EEDomain type. Therefore, between all compatible pairs of CDM classes, there are predefined relationships, for example:

- app.j2ee.J2EEDomain [0..1] - [0..n] app.j2ee.J2EEServer
- app.j2ee.J2EEDomain [0..1] - [0..n] app.j2ee.jboss.JBossServer
- app.j2ee.J2EEDomain [0..1] - [0..n] app.j2ee.weblogic.WebLogicServer
- app.j2ee.jboss.JBossDomain [0..1] - [0..n] app.j2ee.jboss.JBossServer
- app.j2ee.websphere.WebSphereCell [0..1] - [0..n] app.j2ee.websphere.WebSphereServer

In TADDM 7.3.0.1, some query subjects in CDM namespaces are defined for non-persistent attributes of the array type. In TADDM 7.3.0.2, query subjects are defined for all attributes of the array type. Their names have the following format: "[name of the class declaring the array attribute]-->[array attribute name]". For example, the simple.SGroup class has the GroupMembers attribute of the ModelObject[] type, so the query subject is "SGroup-->GroupMembers". These query subjects contain predefined relationships between the described array attributes and all CDM classes that contain these attributes. For example, for the mentioned GroupMembers attribute, the following relationships are defined among others:

- simple.SGroup [1..1] - [0..n] simple."SGroup-->GroupMembers"
- simple.SBaseCollection [1..1] - [0..n] simple."SGroup-->GroupMembers"
- app.biztalk.BizTalkGroup [1..1] - [0..n] simple."SGroup-->GroupMembers"
- app.hacmp.HACMPResourceGroup [1..1] - [0..n] simple."SGroup-->GroupMembers"

To use attributes of the array type, you must define a relationship between an attribute of the array type and the required CDM class by using its PK_C attribute or, in case of non-persistent attribute of the array type (ModelObject[]), its Guid attribute. For example:

- **Fix Pack 2** To create a Cognos report that shows sys.zOS.ZReportFile objects as ZReportfiles of the sys.ComputerSystem objects, you must define a join between the following columns in IBM Cognos Report Studio:

  sys."ComputerSystem-->ZReportfiles".PK__ZReportfiles_C
  [0..n]-[0..1] sys.zOS.ZReportFile.PK_C

- **Fix Pack 2** To create a Cognos report that shows app.AppServer objects as GroupMembers of the simple.SBaseCollection objects, you must define a join between the following columns in IBM Cognos Report Studio:

  simple."SGroup-->GroupMembers".GroupMembersGuids [0..n]-
  [0..1] app.AppServer.Guid

In many cases, there is no need to manually create joins for attributes of the array type because there are corresponding Parent attributes of the dependent objects. The Cognos model contains
relationships for them. For example, you do not need to manually create joins to create a report that shows sys.FileSystem objects as FileSystems of the sys.ComputerSystem objects, because the sys.FileSystem objects have the Parent attribute that points to the sys.ComputerSystem objects.

In TADDM 7.3.0.3, and later, the Cognos model contains query items of Data Time type for all attributes of the time stamp type. For example, sys.aix.AixUnitaryComputerSystem query subject contains the following query items:

- LastStoredTime of the type Int64, which points to the LASTSTOREDTIME_C column in the building block view. Example value in the column: 1445417251307.
- LastStoredTimeT of the type Data Time, which points to the LASTSTOREDTIME_T column in the building block view. Example value in the column: Oct 21, 2015 10:47:31 AM.

The LastStoredTimeT query item is the equivalent of the LastStoredTime query item, only it is expressed in the Coordinated Universal Time format, instead of UNIX epoch (long integer). The query items that contain the T suffix are the time stamp equivalents of the original long integer attribute.

**WebSphere namespace**

**Note:** WebSphere namespace is deprecated in TADDM 7.3.0.1, and later.

This view contains the primary query subjects for a WebSphere environment. You can use this data to generate WebSphere-specific reports, such as listing the properties or JVM settings of WebSphere servers. This WebSphere Server query subject is linked to the AppServer query subject contained in the shared namespace. The WebSphere cluster and WebSphere cell query subjects are linked to the AppServer cluster and J2EE Domain query subjects contained in the shared namespace.

**Shared namespace**

**Note:** Shared namespace is deprecated in TADDM 7.3.0.1, and later.

This view contains query subjects that are considered to be key classes and can be used as a bridge to join data between different namespaces. The shared namespace contains information about computer systems and collection classes. You can use this data to create inventory reports.

**Business applications namespace**

**Note:** Business applications namespace is deprecated in TADDM 7.3.0.1, and later.

This view contains query subjects for a business application, namely, Application and FunctionalGroup query subjects. The functional group query subject is linked to the shared namespace through the Collection query subject. You can use this data to create reports showing business applications and their members.

**Database namespace**

**Note:** Database namespace is deprecated in TADDM 7.3.0.1, and later.

This view contains query subjects related to database and database servers.
You can use the All Databases query subject to generate general database reports rather than vendor-specific database reports. The database content is linked to the shared namespace through the AppServers query subject.

Dependencies and relationships namespace

**Note:** Dependencies and relationships namespace is deprecated in TADDM 7.3.0.1, and later.

This view contains query subjects that represent generated relationships and dependencies, such as IP dependencies or switch to device relationships. You can use the general purpose relationship(unlinked) query subject to create manual links when creating a report or query. The SwitchToDevice query subject joins switches to computer system objects in the shared namespace. There are three query subjects related to server affinity. The Server query subject shows the union of all computer systems, application servers, and service objects in the database. The Affinity (target-linked) query subject, joins each affinity relationship to its target in the Server query subject. The Affinity (source-linked) query subject joins each affinity relationship to its source in the Server query subject. The server content is linked to the shared namespace through the computer system, application server, and service query subjects. You can use this data to generate a general report showing relationship between configuration items in the network.

**Publishing the model using IBM Cognos Framework Manager**

If you want to add objects to the TADDM data model (model.xml file), you must edit the file, and then import it using IBM Cognos 10 Framework Manager.

**About this task**

The following procedure applies to IBM Cognos 10 Framework Manager. However, it is the same for IBM Cognos 8 Framework Manager.

**Procedure**

To import the data model using IBM Cognos 10 Framework Manager, complete the following steps:

1. Start IBM Cognos 10 Framework Manager.
2. Create a new project.
3. When prompted to do so, enter credentials for the Tivoli Common Reporting server. You might be prompted to enter these credentials more than once.
4. Close the IBM Cognos 10 Framework Manager.
5. Copy the following file from the TADDM server to the Cognos Framework project folder:
   `$COLLATION_HOME/etc/reporting/tcr/model.xml`
   Overwrite the existing model.xml file in the Cognos Framework project folder.
6. Start IBM Cognos 10 Framework Manager and open the project you created earlier.
7. In the Project Viewer pane, click **Data Sources** > `content_manager_data_source_name`.
8. If you are using a DB2 database with a different name than the one that is defined in the Cognos database source, replace the contents of the **Schema** field with the DB2 instance name used for the TADDM database.
9. Save the project.
10. In the Project Viewer pane, click Packages.
11. Right-click the package name and select Publish Packages. Verifying and publishing the TADDM Cognos model might take several minutes.

**Configuring the data source in Tivoli Common Reporting**

You can use Tivoli Common Reporting to configure the data source.

**Before you begin**

Ensure that one of the following situations is true:

- The TADDM database is cataloged locally.
- Tivoli Common Reporting is running on the server hosting the TADDM database.

If you are using a DB2 database, check that the schema name matches the DB2 instance name. The schema name specifies the DB2 database name used to authorize access to the specified database. You specified the DB2 instance name when installing TADDM. The default instance name specified in the TADDM model.xml file is DB2INST1. If required, change the name of the schema.

If you are using an Oracle database, ensure that the schema name is blank.

**Procedure**

To configure the data source using Tivoli Common Reporting, complete the following steps:

1. Open Tivoli Common Reporting home page.
2. Click Reporting > Common Reporting.
3. From the Launch menu, depending on which version of Tivoli Common Reporting you use, select one of the following menu items:
   - Version 2.1 - Administration.
   - Version 3.1 - IBM Cognos Administration.

   The Administration pane is displayed.
4. Click the Configuration tab.
5. Click the New Data Source icon. The New Data Source wizard is displayed.
6. In the Name field, type CMDBTCR. The name CMDBTCR is referenced in the data model so you must give the new data source the same name.
7. From the Type menu, select the type of database you are using.
8. Complete one of the following steps:
   - If your database type is DB2, in the DB2 database name field, type the TADDM database name or the alias of the cataloged TADDM database.
   - If your database type is Oracle, in the SQL*Net connect string field, type the service name of the Oracle database, for example, ORCL. You specified the Oracle database service name when configuring the Oracle Database Client. You can check the Oracle database service name in the %TNS_ADMIN%/tnsnames.ora file. Search for the following string:

```
SERVICE_NAME =
```

9. In the Signon section, specify the database user name and password.
10. To test the database connection, click Test. In the View the Results page of the New Data Source wizard, the status of the test is displayed.
Importing the TADDM report package to Tivoli Common Reporting

To import the predefined TADDM reports into Tivoli Common Reporting, you can import the TADDM report package.

Before you begin

You must first have the Tivoli Common Reporting feature installed on your system. Tivoli Common Reporting is provided with some Tivoli products but is not currently included with TADDM.

About this task

A Tivoli Common Reporting report package is a .zip file containing one or more reports or report designs along with their required resources, in a format that can be used by Tivoli Common Reporting. The predefined BIRT reports for TADDM are provided in a report package that you can import to Tivoli Common Reporting.

For some BIRT reports, there are different versions of the same report available depending on which server the report is being run, for example, the TADDM_SNAPSHOT_CHANGE report on the domain server or storage server, and the TADDM_SNAPSHOT_SYNC_CHANGE report on the synchronization server. Generally, only the appropriate version of a report is available, but after importing BIRT reports into Tivoli Common Reporting, both versions of a report might be available. Ensure that you use only the version of the report that is appropriate to the server on which you want to run the report.

After importing BIRT reports into Tivoli Common Reporting, some reports with the text “Drill-through only” in the report name might be available. These reports are intended to be run by drilling down through selected data in another report and you must not run them separately.

The Server Affinity by Scope report cannot be imported into Tivoli Common Reporting.

For more information about importing report packages, refer to the Tivoli Common Reporting documentation.

Procedure

To import TADDM reports, complete the following steps:

1. If you are using Tivoli Common Reporting 1.3, complete the following steps:
   a. In the Tivoli Common Reporting report navigation window, go to the Navigation tab.
   b. Right-click the root node of the navigation tree (Report Sets).
   c. Click Import Report Package.
   d. In the Import Report Package window, specify the location of the TADDMReports.zip report package file. This file is located in the $COLLATION_HOME/etc/reporting directory.
   e. Expand Advanced Options and do the following:
      1) Select the Overwrite check box. This ensures that any previously installed copies of the reports are overwritten.
      2) In the Security Set field, type the name of the security set into which you want to import the contents of the report package.
f. Click Import. The TADDM report package is imported into the Tivoli Common Reporting data store.

2. If you are using Tivoli Common Reporting 2.1, complete the following steps:
   a. Open a command line and navigate to TIP_install_dir/tipv2Components/TCRComponent/bin.
   b. Run the import command:
      trcmd -user userID -password password -import -bulk pkgFile

      where pkgFile is the path to the TADDMReports.zip report package file copied to the Tivoli Common Reporting server from $COLLATION_HOME/etc/reporting on the TADDM server.
   c. TADDM report package is imported into the Tivoli Common Reporting data store.

3. If you are using Tivoli Common Reporting 3.1, complete the following steps:
   a. Open a command line and navigate to JazzSM_install_dir/reporting/bin.
   b. Run the import command:
      trcmd -user userID -password password -import -bulk pkgFile

      where pkgFile is the path to the TADDMReports.zip report package file copied to the Tivoli Common Reporting server from $COLLATION_HOME/etc/reporting on the TADDM server.
   c. TADDM report package is imported into the Tivoli Common Reporting data store.

What to do next

After you import the TADDM reports, you must reconfigure the JDBC data source for each report.

Configuring TADDM BIRT reports in Tivoli Common Reporting

After importing the TADDM reports to Tivoli Common Reporting, you must configure the JDBC data source used by each report.

Before you begin

Before you configure JDBC access, make sure the appropriate JDBC driver files are installed in the Tivoli Common Reporting drivers directory. For Tivoli Common Reporting 1.3 they are in the following directory:

```
tcr_install_dir/products/tcr/lib/birt-runtime-2_2_1/ReportEngine/plugins/
    org.eclipse.birt.report.data.oda.jdbc_2.2.1.r22x_v20070919/drivers
```

For Tivoli Common Reporting 2.1 they are in the following directory:

```
tip_install_dir/tip21Components/TCRComponent/lib/birt-runtime-2_2_2/ReportEngine/plugins/
    org.eclipse.birt.report.data.oda.jdbc_2.2.2.r22x_v20071206/drivers
```

For Tivoli Common Reporting 3.1 they are in the following directory:

```
JazzSM_install_dir/reporting/lib/birt-runtime-2_2_2/ReportEngine/plugins/
    org.eclipse.birt.report.data.oda.jdbc_2.2.2.r22x_v20071206/drivers
```

If you are using an Oracle database, ensure that either ojdbc14.jar or ojdbc5.jar is included in the directory.
About this task

Imported reports are initially configured to use a default data source. You must modify the data source properties of each TADDM report to use the database where discovery data is stored. The TADDM reports do not use a shared data source. Therefore, complete the following steps to configure data source properties of all the TADDM reports.

Procedure

To configure JDBC data sources for Tivoli Common Reporting, complete the following steps:

1. If you are using Tivoli Common Reporting 1.3, complete the following steps:
   a. In the Tivoli Common Reporting Reports table, right-click the TADDM report that you want to configure.
   b. Click Data Sources from the pop-up menu.
   c. In the Report Data Sources window, type the JDBC driver, URL, user ID and password information. You can find the correct values for these settings in the collation.properties file in the $COLLATION_HOME/etc directory.
   d. Repeat the previous steps for each of the TADDM reports you want to configure.

2. If you are using Tivoli Common Reporting 2.1, or 3.1 complete the following steps:
   a. Open a command line and navigate to tip_install_dir/tip21Components/TCRComponent/bin for Tivoli Common Reporting 2.1, or to JazzSM_install_dir/reporting/bin for Tivoli Common Reporting 3.1.
   b. To configure all JDBC sources for all reports, run the modify command on one line:

   Important: The following commands include the name of the directory that contains BIRT reports, 'IBM Tivoli Products'. This name applies to TADDM 7.3.0.1, and later. If you use TADDM 7.3.0, replace this name with 'Tivoli Products'.

   trcmd -user user1D -password password -modify -datasources -reports -reportname "/content/package[@name='IBM Tivoli Products']/folder[@name='TADDM Reports']//report" -setdatasource odaDriverClass=driverClass odaURL=odaURL odaUser=odaUser odaPassword=odaPassword

   For example, if you use a DB2 database, enter the following command on one line:

   trcmd -user tipadmin -password tipadmin -modify -datasources -reports -reportname "/content/package[@name='IBM Tivoli Products']/folder[@name='TADDM Reports']//report" -setdatasource odaDriverClass=com.ibm.db2.jcc.DB2Driver odaURL=jdbc:db2://100.101.102.103:50000/SAMPLEDB odaUser=db2inst1 odaPassword=db2inst1

   For example, if you use an Oracle database, enter the following command on one line:

Verifying TADDM reports
You can check to ensure that TADDM reports are being displayed correctly in Tivoli Common Reporting.

Procedure
To verify that TADDM reports are being displayed correctly in Tivoli Common Reporting, complete the following steps:
1. Open Tivoli Common Reporting home page.
2. Click Reporting > Common Reporting.
3. Ensure that the TADDM and Tivoli Products folders are displayed.
4. Click TADDM.
5. Click the Run icon to run one of the reports. The report is displayed.
6. Ensure that the report is displayed correctly and in full.
7. Using the breadcrumb trail, return to Public Folders.
8. Click Tivoli Products > TADDM Reports. Click the Run icon to run one of the reports. The report is displayed.
9. Ensure that the report is displayed correctly and in full.

Reporting with BIRT
You can use the Business Intelligence and Reporting Tools (BIRT) reporting feature to run predefined and custom reports based on data from the TADDM database.

Overview of BIRT reporting
In addition to the built-in reports available from the Data Management Portal, you can also design, develop, and install reports based on the open source Business Intelligence and Reporting Tools (BIRT) system.

Important: Viewing the BIRT reports in Data Management Portal in BIRT Report Viewer (BIRT runtime engine) is not safe, therefore it is disabled. It is preferred to view the BIRT reports by using the Tivoli Common Reporting (TCR) after you import the TADDM reports to TCR.

If you are aware of the risks, you can restore BIRT Report Viewer and use it as specified in the following paragraphs.

TADDM includes the open source BIRT runtime engine as an integrated component. In addition, TADDM also includes hundreds of predefined database views and predefined reports. In addition to the predefined reports, you can also use the BIRT designer tool to create new reports to use with the TADDM BIRT runtime engine. These reports can use JDBC data sources that extract data by using predefined database views.

The Data Management Portal interface provides ways to manage these BIRT reports. You can add new reports, download selected reports, delete reports you have uploaded, or run reports. The predefined reports are also packaged so they can be used with the Tivoli Common Reporting tool.
Business Intelligence and Reporting Tools

Business Intelligence and Reporting Tools (BIRT) is an open source, based on Eclipse system for designing, developing, and running reports. You can develop BIRT reports for TADDM, designing them to use JDBC data sources and SQL queries of predefined database views.

Important: BIRT reports must not use data taken directly from the TADDM database tables. Instead, always design your reports to use a JDBC data source and the TADDM database views that are documented in TADDM SDK Developer’s Guide.

The BIRT system includes two major components:

- The BIRT designer, a graphical tool for designing and developing new reports
- The BIRT runtime engine, which provides support for running reports and rendering published report output

TADDM includes the BIRT runtime engine, which you can use to run the predefined reports. If you want to create your own BIRT reports, you must download the BIRT designer tool that matches the version of the BIRT runtime engine included with TADDM (currently version 2.2.1).

For more information about the BIRT project, including how to download the BIRT designer tool, see [http://www.eclipse.org/birt](http://www.eclipse.org/birt).

Related tasks:

“Restoring BIRT Report Viewer” on page 179

If you aware of the risks related to security and still want to use BIRT Report Viewer, you can restore it.

Predefined BIRT reports

The predefined BIRT reports included with TADDM provide information about discovered systems, operating systems, and server processes.

AppServer Inventory Report:

The AppServer Inventory Report includes all of the application servers discovered by TADDM. When you run the report, you can specify a parameter value limiting the report to application servers of a specific type. The report groups the discovered application servers by system, listed by fully qualified host name.

The data for this report is taken from the CM_APP_SERVERS_PER_HOST_V database view.

Computer System Inventory Report:

The Computer System Inventory Report includes all computer systems in the TADDM database that have been assigned IP addresses, listed by fully qualified host name. This report has no parameters.

This report is designed to be exported to a comma-separated file that can be imported into a spreadsheet application. If a system does not have an IP address, it is not included in the report. The same computer system name might be listed multiple times in the report, once for each unique IP address (including the 127.0.0.1 loopback address).
The data for this report is taken from the CM_COMPUTER_SYSTEMS_V database view.

**Computer System Inventory Report by operating system type:**

The Computer System Inventory Report by operating system type includes all discovered computer systems whose operating systems were also discovered. This report has no parameters.

This report is designed to be exported to a comma-separated file that can be imported into a spreadsheet application. The same computer system name might be listed multiple times in the report, once for each unique IP address (including the 127.0.0.1 loopback address). To be included in this report, an operating system must be associated with a system in the TADDM database. Similarly, any system that does not have an operating system defined in the TADDM database is not included.

Click the name of a system in the report to open a drill-through Inventory Details Report for that system.

The data for this report is taken from the following database views:
- DP_UNITARY_COMP_GENERAL_V
- DP_UNITARY_COMP_OS_V
- DP_UNITARY_COMP_IP_INTERFACE_V
- BB_OPERATINGSYSTEM62_V

**ITNM IP Report:**

Provides information about the installed instances of the Network Manager product and lists all Network Manager resources that have a relationship to a computer system.

The Network Manager Inventory Report is available in the TADDM Domain Manager console. The report has the following sections:

**Server Summary**
Provides information about the installed instances of the Network Manager product, including the Network Manager versions installed, the host addresses of the servers where Network Manager is installed, and the URLs to access the Network Manager GUI.

**Resource Summary**
Lists all Network Manager resources that have a relationship to a computer system, including information on their IP address, manufacturer, type of resource (for example, router), and unique identifier in the Network Manager database.

**Concise Computer System Inventory Report:**

The Concise Computer System Inventory Report allows you to view the IP addresses discovered using a Level 1 discovery profile. For each IP address, the report also shows the associated computer system name, as well as the operating system or control software name (if this information has been discovered).

Although the Concise Computer System Inventory Report is intended for use after a Level 1 discovery, you can also use it after a Level 3 discovery. However, other
reports such as the Computer System Inventory Report provide more detailed information after a discovery with credentials.

**Fibre Channel Network Report:**

The Fibre Channel Network Report displays fibre channel connections between a selected Fibre Channel switch and other computer systems.

To run the report, you specify the worldwide name (WWN) of the Fibre Channel switch to view the fibre channel connections between this switch and other computer systems. In the Parameter window, type the name (WWN) or select it from the drop-down list of discovered Fibre Channel switches.

The following information is displayed in the report for each connected computer system:

- Computer System (display name; WWN in the case of Fibre Channel switches)
- Manufacturer
- Model
- Serial Number

You can click a computer system display name in the report to open another Fibre Channel Network Report. This report shows the fibre channel connections between the selected computer system and other computer systems.

**Host Bus Adaptor Inventory Report:**

The Host Bus Adaptor Inventory Report displays a list of all discovered host bus adapters and the computer systems on which they are installed.

For each host bus adapter discovered, the following information is displayed in the report:

**Host Bus Adaptor Name**

The name of the host bus adapter.

**Fully Qualified Domain Name**

The fully-qualified domain name of the computer systems on which the host bus adapter is installed.

**Host uses storage arrays**

A Boolean value indicating whether the hosting computer system uses any storage volumes located in a storage array.

**Inventory Summary:**

The Inventory Summary includes a pie chart of the operating systems installed on discovered computer systems, based on the scopes discovered by TADDM. Each segment of the chart represents an operating system type and indicates the overall count of discovered servers running that operating system. This report has no parameters.

Click any segment of the chart to open a drill-through Computer System Inventory Report for the selected operating system type.

The data for this report is taken from the BB_OPERATINGSYSTEM62_V database view.
Monitoring Coverage reports:

The Monitoring Coverage reports show details about various components in your environment. You can generate a report for operating systems, databases, Microsoft applications, VMware servers, and System p components in your environment. These components are monitored by IBM Tivoli Monitoring 6.1, or later agents. You can run this report from the BIRT Reports pane of the Data Management Portal.

Table 38 lists the coverage reports available. The report monitoring coverage for operating systems can be populated by the IBM® Tivoli® Monitoring Scope sensor. However, the remaining reports require the IBM® Tivoli® Monitoring discovery library adapter (DLA) to populate the reports.

The reports contain three sections:

Coverage by type

This section displays the number of monitored, unmonitored, and total instances grouped by report type. The Coverage Details window shows a graphical representation of the following statistics:

- Overall Coverage
- Coverage by platform

Coverage details

This section displays the fully qualified domain name, managed system name, and monitoring status grouped by report type. The monitoring status is listed, along with agent version information, if monitored. Clicking the MSN of a monitored system opens the Agent Details window.

Agent details

This section displays detailed information about the agent and the operating system running on it. The information displayed depends on whether the agent is monitored or unmonitored. Affinity and source token information is included along with a launch in context link to the Tivoli Enterprise Portal view of IBM Tivoli Monitoring.

The Management Software System section provides an inventory of the installed IBM Tivoli Monitoring agents and a launch in context link to workspaces in IBM Tivoli Monitoring. The Monitoring Coverage Summary provides a list of monitored and unmonitored systems, which you can use to monitor and maintain monitoring agents.

A level 1 discovery can use the IBM Tivoli Monitoring Scope sensor to populate the monitoring coverage for operating systems report. The other reports must be populated by the IBM Tivoli Monitoring discovery library adapter (DLA). See the TADDM Administrator’s Guide for information about IBM Tivoli Monitoring DLA.

Table 38 lists the coverage reports available.

<table>
<thead>
<tr>
<th>Report name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring coverage for operating systems</td>
<td>This report shows the details for the operating system in your environment.</td>
</tr>
<tr>
<td>Monitoring coverage for databases</td>
<td>This report shows the details for the DB2 instance and SQL sever in your environment.</td>
</tr>
</tbody>
</table>
Table 38. Monitoring Coverage reports (continued)

<table>
<thead>
<tr>
<th>Report name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring coverage for Microsoft applications</td>
<td>This report shows the details for the Active Directory, Cluster Server, Exchange Server, Host Integration Server, Hyper-V server role enabled, and Internet Information Services server.</td>
</tr>
<tr>
<td>Monitoring coverage for VMware</td>
<td>This report shows the details for the VMware ESX servers and VMware Virtual Center servers.</td>
</tr>
<tr>
<td>Monitoring coverage for System p</td>
<td>This report shows the details for the System p, Hardware Management Console, Virtual I/O Server, and AIX logical partitions.</td>
</tr>
</tbody>
</table>

Sensor reports:

The predefined sensor reports collate the information gathered on sensor metrics. Table 39 lists the predefined sensor reports available.

Table 39. Predefined sensor reports

<table>
<thead>
<tr>
<th>Report name</th>
<th>Description</th>
</tr>
</thead>
</table>
| TADDM_SENSORS_WEEKLY_METRICS_ALL     | This report shows the weekly percentage success rate for sensors that are enabled in a Level 1 discovery profile, Level 2 discovery profile, or a Level 3 discovery profile. The following information is displayed:  
  • Date  
  • Level 1 (L1) % Success  
  • Level 2 (L2) % Success  
  • Level 3 (L3) % Success  
  • L1, L2 % Success  
  • All % Success  
  The second “TADDM_SENSORS_WEEKLY_METRICS” report contains the same information but presents the information by using a bar chart. |
<p>| TADDM_SENSORS_WEEKLY_METRICS         |                                                                                                                                               |</p>
<table>
<thead>
<tr>
<th>Report name</th>
<th>Description</th>
</tr>
</thead>
</table>
| TADDM_SENSORS_SUMMARY_Total                    | This report shows the total number of sensors that ran and completed successfully. The following information is displayed:  
• Level  
• Runs with stored CIs  
• Successes  
• Failures  
In addition, a summary is displayed showing the discovery profile levels and the overall percentage success and failure rates for each level.  
The “TADDM_SENSORS_SUMMARY” report shows the percentage success or failure rate for individual sensors during a discovery. The following information is displayed:  
• Level  
• Sensor  
• Runs  
• Successes  
• Failures  
• % Success  
• % Failure                                                                                                                                 |
| TADDM_SENSORS_SUMMARY                          |                                                                                                                                                                                                            |
| TADDM_SENSORS_SERVER_SCANS_IP                  | This report shows the status after scanning a server by specifying the IP address. The following information is displayed:  
• Week  
• Status  
The start of the report shows summary information about the IP address, host name, fully qualified domain name, first scan date and status, last scan data and status.                                                                                                                                 |
| TADDM_SENSORS_SERVER_SCANS_HOSTNAME            | This report shows the status after scanning a server by specifying the host name. The following information is displayed:  
• Week  
• Status  
The start of the report shows summary information about the host name, IP address, fully qualified domain name, first scan date and status, last scan data and status.                                                                                                                                 |
| TADDM_SENSORS_MONTHLY_COVERAGE                 | This report shows a bar chart showing the monthly coverage of the Session sensor. Includes information about the number of scans run and the number of scans that were successful or not successful. The Session sensor creates a session between the TADDM server and the target computer system. |
| TADDM_SENSORS_METRICS_LEVEL_1_AND_2            | This report shows a bar chart showing for a specified week the percentage success rate for individual sensors when carrying out a Level 1 and Level 2 discovery.  
The bar chart for the report “TADDM_SENSORS METRICS_LEVEL3” shows the metrics for individual sensors when carrying out a Level 3 discovery. |
| TADDM_SENSORS_METRICS_LEVEL3                   |                                                                                                                                                                                                            |
Table 39. Predefined sensor reports (continued)

<table>
<thead>
<tr>
<th>Report name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TADDMSENSORS_FAILED_LEVELS_1_2_3</td>
<td>This report shows a pie chart for a given week based on failures when carrying out a Level 1, Level 2, or Level 3 discovery. Each segment of the chart represents session issues, sensor issues, connect issues, and other issues. The pie chart for the report “TADDM_SENSORS_FAILED_LEVEL” report shows the metrics for a specified discovery level.</td>
</tr>
<tr>
<td>TADDMSENSORS_FAILED_LEVEL</td>
<td></td>
</tr>
<tr>
<td>TADDMSENSORS_EVENTS_SENSOR_IP</td>
<td>This report shows the event data for a specified sensor and IP address. The following information is displayed: Date, Sensor Details, Severity, Description. The “TADDM_SENSORS_EVENTS_SENSOR” report contains the same information but shows the event data for a specified sensor. The “TADDM_SENSORS_EVENTS_IP” report contains the same information but shows the event data for a specified IP address. The “TADDM_SENSORS_DONE_EVENTS_RUN” report contains the same information but shows the event data for a specified discovery run.</td>
</tr>
<tr>
<td>TADDMSENSORS_EVENTS_SENSOR</td>
<td></td>
</tr>
<tr>
<td>TADDMSENSORS_EVENTS_IP</td>
<td></td>
</tr>
<tr>
<td>TADDMSENSORS_DONE_EVENTS_RUN</td>
<td></td>
</tr>
</tbody>
</table>

**Server Affinity by Scope:**

The Server Affinity by Scope report displays relationships between servers, arranged according to the source and target of each relationship. The first table displays all servers within the specified scope that are sources of relationships, and the connections from those servers to other servers. The second table displays all servers within the specified scope that are targets of relationships, and the connections to those servers from other servers.

The Server Affinity by Scope report is available only in domain server deployments.

To view a graph that visualizes the server-to-server communications, click **Launch the Affinity Graph**. The graph shows transactional and service dependencies between computer systems, with dependencies indicated by links drawn between systems. The graph includes all dependency links that include at least one system within the discovery scope, with systems that are members of the scope highlighted in yellow.

The links that are shown in the affinity graph can represent either transactional or service relationships. The direction of a link indicates which system is the source and which is the target of the dependency relationship. The source and target objects can be of several types, depending on the relationship:

- Computer system
- Application server
- Service

Links on the graph are always drawn between computer systems. For a relationship involving an application server or service, the link connects to the host...
computer system. To see more information about a dependency relationship (including the source, target, command name, and port number involved), move your mouse pointer over the link in the diagram.

The Server Affinity by Scope report cannot be imported into Tivoli Common Reporting.

**Snapshot reports:**

The predefined snapshot reports collate the information captured by one or more snapshots.

A snapshot is a copy of the discovered computer information taken at a specific point in time. For more information about creating snapshots, see "Using the snapshot tool" on page 180.

The specific report name depends on the server on which you are running and viewing BIRT reports. If you are using the Data Management Portal on the domain server or the storage server, run the standard report, for example, TADDM_SNAPSHOT_CHANGE. If you are using the Data Management Portal on the synchronization server, run the report with "SYNC" in the name, for example, TADDM_SNAPSHOT_SYNC_CHANGE. The following reports are exceptions and have the same name on all servers:

- TADDM_SNAPSHOT_FRAME
- TADDM_SNAPSHOT_HOST

When you import a BIRT report into Tivoli Common Reporting, a changed report name is displayed, for example, the TADDM_SNAPSHOT_SYNC_SESSION_FAILED report is displayed as “TADDM: Details about failed sessions (Enterprise)”.

Table 40 lists the predefined snapshot reports available.

**Table 40. Predefined snapshot reports**

<table>
<thead>
<tr>
<th>Report name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TADDM_SNAPSHOT_FRAME</td>
<td>Displays the following detailed information about discovered servers:</td>
</tr>
<tr>
<td></td>
<td>• frame name</td>
</tr>
<tr>
<td></td>
<td>• serial number</td>
</tr>
<tr>
<td></td>
<td>• manufacturer</td>
</tr>
<tr>
<td></td>
<td>• model</td>
</tr>
<tr>
<td></td>
<td>• CPU type</td>
</tr>
<tr>
<td></td>
<td>• CPU speed</td>
</tr>
<tr>
<td></td>
<td>• number of CPUs</td>
</tr>
<tr>
<td></td>
<td>• memory</td>
</tr>
<tr>
<td></td>
<td>• location</td>
</tr>
<tr>
<td></td>
<td>• support area</td>
</tr>
<tr>
<td></td>
<td>• last discovered</td>
</tr>
<tr>
<td>Report name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| TADDM_SNAPSHOT_HOST | Displays the following detailed information about physical and virtual servers:  
  • frame name  
  • system name  
  • IP address  
  • OS type  
  • host type  
  • managed system name  
  • last discovered |
| TADDM_SNAPSHOT_SESSION_FAILED | Displays name and IP address information about discovered servers for which TADDM was unable to get L2 information because of failed sessions. |
| TADDM_SNAPSHOT_SYNC_SESSION_FAILED |  |
| TADDM_SNAPSHOT_CHANGE | Compares two snapshots taken at different times. For each server that was added or removed in the time between the two snapshots, it displays the following information:  
  • name  
  • IP address  
  • virtual  
  It also displays information about the change in the ratio of physical to virtual servers in the time between the two snapshots. |
| TADDM_SNAPSHOT_SYNC_CHANGE |  |
| TADDM_SNAPSHOT_DISCOVERY_ERROR | Displays information about errors generated during any discoveries. |
| TADDM_SNAPSHOT_SYNC_DISCOVERY_ERROR |  |
| TADDM_SNAPSHOT_FQDN_OS_CHANGES | Displays information about any servers with changed fully-qualified domain name (FQDN), or operating system information, in the time between two snapshots. |
| TADDM_SNAPSHOT_SYNC_FQDN_OS_CHANGES |  |
| TADDM_SNAPSHOT_REFERENCE | Compares a snapshot with a reference list. It displays information about the servers in the reference list but not in the snapshot, and the servers in the snapshot but not in the reference list. |
| TADDM_SNAPSHOT_SYNC_REFERENCE |  |
| TADDM_SNAPSHOT_RECONCILIATION_SUMMARY | Prompts you for a snapshot, and displays the following summary information about discovered servers:  
  • baseline host name  
  • baseline IP address  
  • TADDM host name  
  • TADDM IP address  
  • status  
  • error reason |
Table 40. Predefined snapshot reports (continued)

<table>
<thead>
<tr>
<th>Report name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TADDM_SNAPSHOT_RECONCILIATION_DETAIL</td>
<td>Prompts you for a snapshot, and displays the following detailed information about discovered servers:</td>
</tr>
<tr>
<td>TADDM_SNAPSHOT_SYNC_RECONCILIATION_DETAIL</td>
<td>- baseline host name</td>
</tr>
<tr>
<td></td>
<td>- baseline IP address</td>
</tr>
<tr>
<td></td>
<td>- TADDM host name</td>
</tr>
<tr>
<td></td>
<td>- TADDM IP address</td>
</tr>
<tr>
<td></td>
<td>- status</td>
</tr>
<tr>
<td></td>
<td>- error reason</td>
</tr>
<tr>
<td></td>
<td>- error description</td>
</tr>
<tr>
<td></td>
<td>- scope name</td>
</tr>
<tr>
<td></td>
<td>- filtered exclusion</td>
</tr>
<tr>
<td></td>
<td>- TADDM frame</td>
</tr>
<tr>
<td></td>
<td>- TADDM host name</td>
</tr>
<tr>
<td></td>
<td>- TADDM FQDN</td>
</tr>
<tr>
<td></td>
<td>- TADDM name</td>
</tr>
<tr>
<td></td>
<td>- TADDM DisplayName</td>
</tr>
<tr>
<td></td>
<td>- TADDM JdoClass</td>
</tr>
<tr>
<td></td>
<td>- TADDM OsDerived</td>
</tr>
<tr>
<td></td>
<td>- TADDM OsName</td>
</tr>
<tr>
<td></td>
<td>- TADDM IP address</td>
</tr>
<tr>
<td></td>
<td>- TADDM serial number</td>
</tr>
<tr>
<td></td>
<td>- TADDM manufacturer</td>
</tr>
<tr>
<td></td>
<td>- TADDM model</td>
</tr>
<tr>
<td></td>
<td>- TADDM host type</td>
</tr>
<tr>
<td></td>
<td>- TADDM virtual</td>
</tr>
<tr>
<td></td>
<td>- TADDM type</td>
</tr>
<tr>
<td></td>
<td>- TADDM discovery date</td>
</tr>
</tbody>
</table>

**Storage Arrays by Host Report:**

The Storage Arrays by Host Report displays a list of the storage volumes and storage arrays that are used by a specified computer system.

When running the report, you are prompted to enter the host name of the computer system for which you want to view storage information. In the Parameter window, type the host name or select it from the drop-down list.

The following information is displayed in the report:
- Storage Volume
- Storage Array
- Manufacturer
- Model
- Serial Number
- Available Capacity
- Allocated Capacity
Storage Array Consumers Report:

The Storage Array Consumers Report displays a list of the computer systems and application servers that use a specified storage array.

When running the report, you are prompted to enter the name of a storage array. In the Parameter window, type the name of the storage array or select it from the drop-down list.

The report is displayed in the form of the following three tables:

Computer systems using the storage array storage_array_name
This table lists all discovered computer systems that use the specified storage array.

Application servers using the storage array storage_array_name
This table lists all discovered application servers that use the specified storage array.

Business applications using the storage array storage_array_name
This table lists all discovered business applications that use the specified storage array.

System Connection Topology Report:

The System Connection Topology Report displays a textual report of computer systems with network connections to or from other computer systems. When you run the report, you must enter the configuration item for which you want to run the report, and you must specify whether it is a computer system or business application.

If the report is run for a computer system, all computer systems with network connections to or from the selected computer system are displayed in a table, along with metrics for each network connection. If the report is run for a business application, all computer systems with network connections to or from the selected business application are displayed in a table.

You can view the system connection topology for each computer system by clicking the name of the system in the report.

System Utilization Hourly Peak Report:

The System Utilization Hourly Peak Report displays system hourly utilization peak values for systems in the specified scope, on the specified date.

The utilization metrics include the following information:
• hourly 95 percent CPU utilization
• hourly peak percent memory utilization
• hourly peak network bandwidth utilization
• hourly peak disk I/O utilization

System Utilization Report:

The System Utilization Report displays generic server operating system configuration, and associated utilization information.

Server operating system configuration data includes the following information:
• CPU
• memory
• file system

It is the latest server configuration information available to TADDM. Server utilization data includes the following:
• CPU
• memory
• network
• disk

Unknown Servers Report:

The Unknown Servers Report includes all of the discovered server processes that are not recognized by TADDM. The report groups the discovered server processes by system, listed by fully qualified host name. This report has no parameters.

Unknown servers are identified after a discovery by a topology build agent. The topology build agent runs in the background on a periodic basis, depending on the value of the configured frequency, so unknown servers might not be recognized immediately after a discovery completes. Every four hours is the default frequency at which the topology build agent runs.

For this reason, if you run the Unknown Servers Report before the topology build agent has completed, the report might not list all unknown servers.

The following information is displayed in the report:

**Name**  
The name of the computer on which the unknown server process is running.

**Context IP**  
The IP address of the computer on which the unknown server process is running.

**PID**  
The process ID of the unknown server process.

**PPID**  
The process ID of the parent process of the unknown server process.

**Command Line**  
The command being used to run the unknown server process.

The data for this report is taken from the BB_RUNTIMEPROCESS15_V database view.

**Running a BIRT report**

You can use the Analytics section of the Data Management Portal to run a BIRT report.

**About this task**

**Important:** Running a BIRT report in Data Management Portal is only possible if you have BIRT Report Viewer enabled. BIRT Report Viewer is disabled because of security issues. The alternative way to view the BIRT reports is by using the Tivoli Common Reporting (TCR) after you import the TADDM reports to TCR. If you are aware of the risks, you can restore BIRT Report Viewer.

Administering  175
Procedure

To run a BIRT report, complete the following steps:

1. In the Functions pane, click **Analytics**.

2. In the Analytics section, click **BIRT Reports**. The **TADDM BIRT Reports** list opens, displaying all of the available BIRT reports.

3. In the **TADDM BIRT Reports** list, click to highlight the report you want to run.

4. Optional: Specify the location tag value. The `com.ibm.cdb.locationTaggingEnabled` value in the `COLLATION_HOME/etc/collation.properties` file must be set to true. Only the report data for this specific location tag is displayed.

   **Note:** The BIRT reports included with TADDM do not currently support location filtering without additional customization.

5. Click **Run Report**. If the report has any parameters, you are then prompted to specify the parameter values. When you have finished specifying parameter values, click **OK**.

Results

The formatted report is displayed in the BIRT Report Viewer window. Click the icons at the top of the report to page forward and backward in the report, print the report, or export it to a file. To open a drill-through report showing additional details about a subset of the report data, click a link in the report.

**Note:** Exported reports in .doc format are compatible with Microsoft Word 2003 or later.

Running a BIRT report from the command line interface

You can run a BIRT report from the command line interface of the TADDM server.

Procedure

To run a BIRT report from the command line interface, complete the following steps:

1. Open a command prompt and depending on the version of TADDM you use, navigate to one of the following directories:
   - 7.3.0: `COLLATION_HOME/deploy-tomcat/birt-viewer/WEB-INF/resources`
   - 7.3.0.1, and later: `COLLATION_HOME/apps/birt-viewer/WEB-INF/resources`

2. Set the **BIRT_HOME** variable. Do one of the following:
   - On Linux, depending on the version of TADDM you use, run one of the following commands:
     - 7.3.0:
       ```bash
       export BIRT_HOME=$COLLATION_HOME/deploy-tomcat/birt-viewer
       ```
     - 7.3.0.1, and later:
       ```bash
       export BIRT_HOME=$COLLATION_HOME/apps/birt-viewer
       ```
   - On Windows, depending on the version of TADDM you use, run one of the following commands:
     - 7.3.0:
       ```bash
       set BIRT_HOME=%COLLATION_HOME%/deploy-tomcat/birt-viewer
       ```
     - 7.3.0.1, and later:
       ```bash
       set BIRT_HOME=%COLLATION_HOME%/apps/birt-viewer
       ```
3. Run the BIRT report. Do one of the following:
   • On Linux, run the following command:
     
     ```
     ./genReport.sh -f format -o output -F parameters report
     ```
   • On Windows, run the following command:
     
     ```
     genReport.bat -f format -o output -F parameters report
     ```

   The following command-line options are used with the `genReport` program:

   **format**
   The output format of the report file. Valid values are PDF and HTML.

   **output**
   The path to the report file you want to produce. For example, `/home/cognos/utilization.pdf` on Linux, or `C:\data\utilization.pdf` on Windows.

   **parameters**
   (Optional) The path to a properties file, where each property represents a parameter required by the report. For example `/home/cognos/utilization.properties` on Linux, or `C:\data\utilization.properties` on Windows.

   The following text is an example of the contents of a properties file:

   ```
   scope=All
   Windows
   Machines
   metric=ALL
   operator=N/A
   value1=N/A
   value2=N/A
   appdeps=N/A
   ```

   You must ensure that spaces in a parameter name are escaped with the backslash character. For example, if the parameter name is `Snapshot ID Parameter`, the entry in the properties file should be `Snapshot\ ID\ Parameter=my_id`

   **report**
   The path of the report you want to run, with the string “compiled” appended to the name. For example:
   • On Linux and TADDM 7.3.0: `$COLLATION_HOME/deploy-tomcat/birt-viewer/WEB-INF/report/taddm_server_utilization.rptdesigncompiled`
   • On Linux and TADDM 7.3.0.1, and later: `$COLLATION_HOME/apps/birt-viewer/WEB-INF/report/taddm_server_utilization.rptdesigncompiled`
   • On Windows and TADDM 7.3.0: `%COLLATION_HOME%\deploy-tomcat\birt-viewer\WEB-INF\report\taddm_server_utilization.rptdesigncompiled`
   • On Windows and TADDM 7.3.0.1, and later: `%COLLATION_HOME%\apps\birt-viewer\WEB-INF\report\taddm_server_utilization.rptdesigncompiled`

   **Results**

   **Note:** The `genReport` command does not generate drill-through reports. Therefore, links in the generated report do not function.
**Importing a BIRT report**
You can use the Data Management Portal to add custom reports by importing BIRT report designs.

**Before you begin**
To add a custom report, you must first design and develop the report using the BIRT designer tool. The report design must be saved to a .rptdesign file that you can access from the client system.

**Note:** In TADDM 7.3.0.3, and later, the columns in extended attributes database views have specific data types, for example, VARCHAR. In the previous TADDM releases, the columns had only CLOB type. Therefore, after you upgrade to Fix Pack 3, the BIRT reports that use extended attributes might stop working. For example, if extended attributes columns are not cast to a specific data type, for example, VARCHAR, errors might be generated.

**Procedure**
To import a BIRT report, complete the following steps:
1. In the Functions pane of the Data Management Portal, click **Analytics**.
2. In the Analytics section, click **BIRT Reports**. The **TADDM BIRT Reports** list opens, displaying all of the available BIRT reports.
3. Click **New**.
4. When prompted, specify the details of the new report, including the name, description, and the location of the report design file. The name and description are used to identify the report in the **TADDM BIRT Reports** list.
5. Click **OK**.

**Results**
The report design is uploaded to the server, and the new report becomes available from the Data Management Portal.

**Note:** If the report already exists on the server, the import fails. This can happen even if the existing report is not visible in the Data Management Portal. (For example, the Server Affinity report is not supported on the synchronization server, and is therefore not shown in the Data Management Portal even if it exists on the server.)

**Deleting a BIRT report**
You can use the Data Management Portal to delete BIRT reports from the server.

**Before you begin**
Deleting a report from the server removes the .rptdesign file used by the report from the report directory of the server. If you want to save the report design for future use, make sure you have a backup copy of the .rptdesign file before you delete the report.

**Procedure**
To delete a BIRT report, complete the following steps:
1. In the Functions pane, click **Analytics**.
2. In the Analytics section, click **BIRT Reports**. The **TADDM BIRT Reports** list opens, showing all of the available BIRT reports.

3. Select the report that you want to delete.

4. Click **Delete**.

5. To refresh the **TADDM BIRT Reports** list, click **Refresh**.

**Results**

The selected report is deleted from the server and is no longer displayed in the **TADDM BIRT Reports** list in the Data Management Portal. In addition, the .rptdesign file for the report is deleted from the report directory of the TADDM server.

**Exporting a BIRT report design**

You can use the Data Management Portal to export a BIRT report design from the server.

**About this task**

You might want to export a report design if you want to use an existing report as the basis for a new custom report, or if you want to import the report design into a different server.

**Procedure**

To export a BIRT report design, complete the following steps:

1. In the Functions pane, click **Analytics**.

2. In the Analytics section, click **BIRT Reports**. The **TADDM BIRT Reports** list opens, showing all of the available BIRT reports.

3. Select the report that you want to export.

4. Click **Download**.

5. When prompted by your browser, specify that you want to save the file, and specify a location.

**Results**

The design used by the selected report is saved in the location you specified as a .rptdesign file. You can open and modify this file using the BIRT designer tool.

**Restoring BIRT Report Viewer**

If you are aware of the risks related to security and still want to use BIRT Report Viewer, you can restore it.

**Procedure**

1. In the collation.properties file, set the com.ibm.taddm.birtviewer.enabled property to true:

   ```
   com.ibm.taddm.birtviewer.enabled=true
   ```

2. Restart the TADDM server.

   **Note:** In case of the TADDM server upgrade, this flag is set to false by default.
Using the snapshot tool
You can use the snapshot tool to take a copy of the computer system information, discovery events, and the server applications running at the time of the snapshot.

You can also use the snapshot tool to load information used in reconciliation processing, for example:
- load a list of expected servers, also known as a reference list
- load a list of excluded servers

You can use reports to query the information captured by the snapshot tool, for example:
- which servers have been added or removed
- the ratio of physical to virtual servers
- which servers could not be fully discovered because an ssh session was not successfully established
- the delta between the list of discovered servers and the list of expected servers

Restriction: Take snapshots after the discovery and topology agents finished their run. If you take a snapshot before the topology agents finish processing the discovered information, some snapshot reports, such as Snapshot Session Failed Report, might be incomplete.

The snapshot.sh command syntax:

You can use the snapshot.sh command to take a snapshot of the system, and the associated events and servers. The snapshot.sh command is in the $COLLATION_HOME/bin directory.

You can run the snapshot.sh command on the TADDM server. In a streaming server deployment, you must run the snapshot.sh command on the primary storage server.

Command syntax

snapshot.sh action [action_parameter]

Parameters

addexclude filename [exclude_list]
    Adds the exclude list to the file, or replaces an existing instance of it in the file.

addref filename [reference_list]
    Adds the reference list to the file, or replaces an existing instance of it in the file.

clear
    Clears all snapshot data and drops tables.

compare [snapshot_A snapshot_B]
    Shows the delta between the last two snapshots, or snapshot_A and snapshot_B, based on the host name.

compareref [snapshot_A reference_list]
    Shows the delta between the snapshot and the reference list.
**comparesig** [snapshot_A snapshot_B]
 Shows the delta between the last two snapshots, or snapshot_A and snapshot_B, based on the signature for changes in the host name or operating system.

**compsys**
 Shows the computer systems.

**detail** [snapshot_A]
 Shows all details of the computer systems in the latest snapshot, or snapshot_A.

**detailos** [snapshot_A]
 Shows operating system information of the computer systems in the latest snapshot, or snapshot_A.

**help**
 Shows detailed help on the usage of the **snapshot.api** command.

**list** [snapshot_A]
 Shows the latest snapshot, or snapshot_A.

**listall** [default]
 Shows all snapshots.

**listexclude** [exclude_list]
 Shows the latest exclude list, or the one specified by name.

**listref** [reference_list]
 Shows the latest reference list, or the one specified by name.

**listallexclude**
 Shows all exclude lists.

**listallref**
 Shows all reference lists.

**nosession** [snapshot_A]
 Shows computer systems that failed to host a session in the latest snapshot, or snapshot_A.

**remove** snapshot_A [type]
 Removes snapshot A, or remove all snapshots of the specified type.

**removeexclude** exclude_list
 Removes the exclude list specified by name.

**removeref** reference_list
 Removes the reference list specified by name.

**session** [snapshot_A]
 Shows the computer systems that hosted a session in the latest snapshot or snapshot_A.

**sensorerror** [snapshot_A]
 Shows all sensor errors from the latest snapshot or snapshot_A.

**take** [type] [description]
 Takes a snapshot, including type and description information, if specified.

**Using the snapshot tool to help reduce the number of physical servers:**

You can use the snapshot tool when replacing many physical servers with fewer physical servers, running virtual servers.
Procedure

To get information that is useful when attempting to reduce the number of physical servers used, complete the following steps:

1. Perform a discovery of all known systems.
2. Using the snapshot tool, take a snapshot:
   
   ```
   snapshot.sh take
   ```

   Optionally, you can add type and description information to the snapshot:
   
   ```
   snapshot.sh take type description
   ```

3. In the Data Management Portal, run the TADDM_SNAPSHOT_SESSION_FAILED report. The report returns information about which systems were not discovered because an ssh session could not be established.
4. Ensure that ssh sessions can be established with all systems. It might be necessary to update the authentication details in TADDM.
5. Perform a discovery of only the systems that were not accessed as part of the first discovery to ensure that all connection problems have been resolved.
6. After an appropriate length of time, for example, one month, perform a discovery of all known systems.
7. In the Data Management Portal, run the TADDM_SNAPSHOT_CHANGE report. The report returns information about the new systems visible since the snapshot was taken, the systems no longer present, and the physical to virtual ratio as percentages.

Using the snapshot tool to reconcile expected and actual system lists:

You can use the snapshot tool and the predefined reports to verify that the list of servers available on the network matches the list of expected servers.

Procedure

To reconcile the expected and actual systems, complete the following steps:

1. Prepare a reference list containing the list of expected servers. The reference list is a text file in comma-separated values (CSV) format, with the following fields:
   - host name
   - IP address
   - frame
   - operating system
   - host type
   - comments
   - support area
   - location

   For more information about the syntax of the reference file, run the `snapshot.sh` command with the help parameter:
   
   ```
   snapshot.sh help
   ```

2. If required, prepare an exclusion list containing the list of servers to be ignored in the reconciliation process. The exclusion list is a text file in CSV format, with the following fields:
   - host name
   - exclude type
For more information about the syntax of the exclusion file, run the `snapshot.sh` command with the help parameter:

`snapshot.sh help`

3. Using the snapshot tool, take a snapshot:

`snapshot.sh take`

Optionally, you can add type and description information to the snapshot:

`snapshot take type description`

4. In the Data Management Portal, run one of the following BIRT reports:

- `TADDM_SNAPSHOT_RECONCILIATION_SUMMARY`
- `TADDM_SNAPSHOT_RECONCILIATION_DETAIL`

**Using snapshot reports in a synchronization server deployment:**

You can gather information in a synchronization server deployment by running the enterprise version of the predefined snapshot reports.

**Procedure**

To run predefined snapshot reports in a synchronization server deployment, complete the following steps:

1. If it has not already been created, set up the snapshot table. To do this, complete the following steps:
   a. On each TADDM server, run the `snapshot.sh` command with no parameters.
   b. Restart TADDM at each domain and synchronization server.

   This procedure creates the snapshot tables if they do not exist already. The snapshot tables need to be set up only once per TADDM environment.

2. Run a discovery at each TADDM domain, taking snapshot at each domain, when required.

3. Perform a sync at the synchronization server. Ensure that you include all domains.

4. Create an enterprise snapshot. At the synchronization server, run the following command:

   `snapshot.sh take`

5. Run the reports at each domain. Use the regular version of each snapshot report, for example `TADDM_SNAPSHOT_CHANGE`.

6. Run the reports at the synchronization server. Use the enterprise version of each snapshot report, for example `TADDM_SNAPSHOT_SYNC_CHANGE`.

**Integrating TADDM with other Tivoli products**

For extended capabilities in managing your IT environment, you can integrate IBM Tivoli Application Dependency Discovery Manager (TADDM) with other Tivoli products, including IBM Tivoli Business Service Manager, IBM Tivoli Monitoring, and event management systems such as IBM Tivoli Netcool/OMNIbus.

**Supported versions**

You can use the following table to see which versions of the products that TADDM can be integrated with are supported.
The following table shows the supported versions of the products that TADDM can be integrated with.

Table 41. The supported products versions.

<table>
<thead>
<tr>
<th>Product name</th>
<th>Supported version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context Menu Service and Data Integration Service (CMS/DIS)</td>
<td></td>
</tr>
<tr>
<td>IBM Control Desk (ICD)</td>
<td>• 7.6</td>
</tr>
<tr>
<td>IBM SmartCloud® Control Desk (SCCD)</td>
<td>• 7.5.1 - use the latest Fix Pack level available</td>
</tr>
<tr>
<td>IBM Tivoli Business Service Manager (TBSM)</td>
<td>• 4.2.1 - use the latest Fix Pack level available</td>
</tr>
<tr>
<td></td>
<td>• 6.1.0 - use the latest Fix Pack level available</td>
</tr>
<tr>
<td></td>
<td>• 6.1.1 - use the latest Fix Pack level available</td>
</tr>
<tr>
<td>IBM Tivoli Change And Configuration Management Database (CCMDB)</td>
<td>• 7.2.1</td>
</tr>
<tr>
<td>IBM Tivoli Integration Composer (ITIC)</td>
<td>• 7.5.1 - use the latest Fix Pack level available</td>
</tr>
<tr>
<td>IBM Tivoli Monitoring (ITM)</td>
<td>• 6.2.1</td>
</tr>
<tr>
<td></td>
<td>• 6.2.2 - FP3</td>
</tr>
<tr>
<td></td>
<td>• 6.2.3</td>
</tr>
<tr>
<td></td>
<td>• 6.3</td>
</tr>
<tr>
<td>IBM Tivoli Netcool®/OMNIbus</td>
<td>• 7.3</td>
</tr>
<tr>
<td></td>
<td>• 7.4</td>
</tr>
<tr>
<td></td>
<td>• <em>Fix Pack 1 8.x - supported with TADDM 7.3.0.1, and later</em></td>
</tr>
<tr>
<td>IBM Tivoli Network Manager IP (ITNMIP)</td>
<td>• 3.9</td>
</tr>
<tr>
<td></td>
<td>• 4.1</td>
</tr>
<tr>
<td>Jazz for Service Management (JazzSM)</td>
<td>• 1.1</td>
</tr>
<tr>
<td>Tivoli Common Reporting (TCR)</td>
<td>• 1.3</td>
</tr>
<tr>
<td></td>
<td>• 2.1.1</td>
</tr>
<tr>
<td></td>
<td>• 3.1</td>
</tr>
<tr>
<td>Tivoli Directory Integrator (TDI)</td>
<td>• 7.0</td>
</tr>
<tr>
<td></td>
<td>• 7.1</td>
</tr>
<tr>
<td></td>
<td>• 7.1.1</td>
</tr>
<tr>
<td>Tivoli Netcool/IMPACT</td>
<td>• 7.1</td>
</tr>
<tr>
<td>Tivoli Workload Scheduler (TWS)</td>
<td>• 8.5.1</td>
</tr>
<tr>
<td></td>
<td>• 8.6</td>
</tr>
</tbody>
</table>

For more information about products that you integrate with TADDM, refer to their documentation:
For information about Context Menu Service and Data Integration Service (CMS/DIS), see the Configuring for the Context Menu Service and Data Integration Service topic in the TADDM Installation Guide.

- IBM Control Desk (ICD)
- IBM SmartCloud Control Desk (SCCD)
- IBM Tivoli Business Service Manager (TBSM)
- IBM Tivoli Change and Configuration Management Database (CCMDB)
- IBM Tivoli Integration Composer (ITIC)
- IBM Tivoli Monitoring (ITM)
- IBM Tivoli Netcool/OMNIbus
- IBM Tivoli Network Manager IP (ITNMIP)
- Jazz for Service Management (JazzSM)
- Tivoli Common Reporting (TCR)
- Tivoli Directory Integrator (TDI)
- Tivoli Netcool/Impact
- Tivoli Workload Scheduler (TWS)

**Integrating TADDM with IBM Tivoli Monitoring via OSLC Automation**

TADDM can be integrated with IBM Tivoli Monitoring by using OSLC Automation. If you want to integrate TADDM with IBM Tivoli Monitoring 6.3, it is advised to use OSLC Automation. The old method of integration with the use of IBM Tivoli Monitoring Scope sensor is deprecated and will be removed from future releases.

TADDM uses the IBM Tivoli Monitoring infrastructure in the following two ways:

- TADDM obtains the list of IBM Tivoli Monitoring endpoints from the Tivoli Enterprise Portal Server via OSLC Automation Session.
- TADDM both runs CLI commands on the target systems for the sensors in Level 2 and Level 3 discovery and captures the output of those commands.

If you encounter any problems, see the ITM OSLC Execute Automation Service Provider problems topic in the TADDM Troubleshooting Guide.

**Fix Pack 5**

**Prerequisites:**

If you are using Windows 7 and above you will need:

1. PowerShell version 2+
2. TEMS SOAP URL
3. Check that you can connect to both TEMS and TEPS

The following table provides steps, which you must complete to successfully enable integration of TADDM with IBM Tivoli Monitoring via OSLC Automation.

**Table 42. Integrating TADDM with IBM Tivoli Monitoring via OSLC Automation**

<table>
<thead>
<tr>
<th>Step</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure ITM Tivoli Enterprise Monitoring Server (TEMS) and ITM TEPS hosts</td>
<td>“Configuring ITM Tivoli Enterprise Monitoring Server (TEMS) and ITM TEPS hosts” on page 189</td>
</tr>
</tbody>
</table>
Table 42. Integrating TADDM with IBM Tivoli Monitoring via OSLC Automation (continued)

<table>
<thead>
<tr>
<th>Step</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install OSLC Execute Automation Service Provider on IBM Tivoli Monitoring</td>
<td>&quot;Installing OSLC Execute Automation Service Provider on IBM Tivoli Monitoring&quot; on page 192</td>
</tr>
<tr>
<td><strong>Note:</strong> Make sure that you meet all prerequisites specified in &quot;Prerequisites for installing OSLC Execute Automation Service Provider on IBM Tivoli Monitoring&quot; on page 189.</td>
<td></td>
</tr>
<tr>
<td>Configure TADDM to use OSLC Execute Automation Service Provider</td>
<td>&quot;Configuring TADDM to use OSLC Execute Automation Service Provider&quot; on page 198</td>
</tr>
<tr>
<td>Configure TADDM for discovery:</td>
<td>&quot;Configuring for discovery over OSLC Automation Session&quot; on page 107</td>
</tr>
<tr>
<td>• Configure automation properties in the collation.properties file.</td>
<td></td>
</tr>
<tr>
<td>• Create a new access list entry of type &lt;&quot;Integration&quot;&gt;&quot;OSLC Automation&quot; in the access list.</td>
<td></td>
</tr>
</tbody>
</table>

After you complete these steps, you can run a discovery by using ITM OSLC Execute Automation Service Provider.

**Related concepts:**
"Integrating TADDM with other products via OSLC Automation” on page 197
TADDM can be integrated with other products using Open Services for Lifecycle Collaboration (OSCL) Automation. TADDM connects to OSLC Execute Automation Service Provider that provides data about other products infrastructure, which can be discovered by TADDM using OSLC Automation Session.

**ITM OSLC Execute Automation Service Provider**
ITM OSLC Execute Automation Service Provider is used to import data about IP addresses of endpoints managed by IBM Tivoli Monitoring to TADDM and to discover IBM Tivoli Monitoring endpoints via OSLC Automation Session.

**Figure 1:** illustrates TADDM connected to ITM OSLC Execute Automation Service Provider that collects the data about ITM-managed infrastructure by using KT1 commands.
TADDM gets the destination of OSLC Execute Automation Service Provider from JAZZ SM Registry Services or from the collation.properties file. Figure 2 illustrates TADDM using JAZZ SM Registry Services to obtain the address of OSLC Execute Automation Service Provider.

Figure 4. TADDM connected to ITM OSLC Execute Automation Service Provider that collects the data about ITM-managed infrastructure by using KT1 commands.

TADDM gets the destination of OSLC Execute Automation Service Provider from JAZZ SM Registry Services or from the collation.properties file. Figure 2 illustrates TADDM using JAZZ SM Registry Services to obtain the address of OSLC Execute Automation Service Provider.

Figure 5. TADDM using JAZZ SM Registry Services to obtain the address of OSLC Execute Automation Service Provider.
TADDM can be connected directly to many OSLC Execute Automation Service Providers and to a single JAZZ SM Registry Services, where many providers can be registered. Figure 3 illustrates TADDM downloading addresses of OSLC Execute Automation Service Providers deployed on several ITM TEPS (Portal Servers) from JAZZ SM Registry Services.

Figure 6. TADDM downloading addresses of OSLC Execute Automation Service Providers deployed on several ITM TEPS (Portal Servers) from JAZZ SM Registry Services.

Related concepts:

"OSLC Execute Automation Service Provider” on page 198
OSLC Execute Automation Service Provider is used to populate data about IP addresses of endpoints managed by other products to TADDM. The data is used for discovery of endpoints using OSLC Automation Session.
ITM OSLC Execute Automation Service Provider installation

To import data about IP addresses of endpoints managed by IBM Tivoli Monitoring (ITM) to TADDM, or to run a discovery, you must install OSLC Execute Automation Service Provider on IBM Tivoli Monitoring.

If you encounter any problems, see the ITM OSLC Execute Automation Service Provider problems topic in the TADDM Troubleshooting Guide.

Prerequisites for installing OSLC Execute Automation Service Provider on IBM Tivoli Monitoring:

Before you install OSLC Execute Automation Service Provider on IBM Tivoli Monitoring (ITM), you must configure your environment to meet all prerequisites.

ITM OSLC Execute Automation Service Provider must be installed on ITM Tivoli Enterprise Portal Server (TEPS) host. The supported version of IBM Tivoli Monitoring is IBM Tivoli Monitoring 6.3.

Configuring ITM Tivoli Enterprise Monitoring Server (TEMS) and ITM TEPS hosts

Step 1 - Reconfiguring TEMS and TEPS

The TEMS and TEPS configuration is best done using the Manage Tivoli Enterprise Monitoring Services (MTEMS) GUI.

For the Windows OS, start the ITM kinconfig.exe process to bring up the Manage Tivoli Enterprise Monitoring Services (MTEMS) GUI.

For Unix/Linux you can start the MTEMS GUI using the CLI command “./itmcmd manage”.

For each of the two ITM components (TEMS and TEPS):

- Highlight the component on the MTEMS UI
- Right click and select Reconfigure

This will take you into the TEMS or the TEPS configuration window.

Provide all the TEMS parameters correctly (TEMS Type, TEMS Name and protocol in the main configuration settings and Hostname/IP Address and Port in the Advanced settings) and select “OK”.

Select “Enable the dashboard data provider” in the TEPS configuration window.

Step 2 - Executing TADDM Scripts and Configuration

This step includes the execution of two main TADDM scripts to help setup the integration of TADDM to ITM.

1. Configure the provider.properties file

ITM TEMS host

Enable KT1 commands (tacmd get/put/execute) by running one of the following scripts on ITM TEMS host:

- For Linux operating system:
  
  TADDM_CD_ISO/itm-discovery-support/configure_tems.sh -i <ITM_HOME>
  
  [-t <TEMP-DIR>]

- For Windows operating system:
  
  TADDM_CD_ISO/itm-discovery-support/configure_tems.ps1 -i <ITM_HOME>
  
  [-t <TEMP-DIR>]
where \(<\text{ITM\_HOME}>\) is the directory of ITM TEPS installation, for example \(/\text{opt/IBM/ITM}\), and \(<\text{TEMP\_-DIR}>\) is the temporary files destination directory. The default value of the \(<\text{TEMP\_-DIR}>\) parameter is \(/\text{var/log/automation\_provider}\).

When completed you should see the following lines at the end of the script execution.

INFO: Stopping ITM TEPS...
INFO: ITM TEPS has stopped.
INFO: Starting ITM TEPS...
INFO: ITM TEPS has started.
INFO: Checking if TEPS is running...
INFO: Checking if OSLC Automation Provider is installed...
INFO: Installation of OSLC Automation Provider successfully finished.

Note: For an initial user id validation (\(\text{tacmd get/put/execute}\)), use \(\text{http:1920}\) or \(\text{https:3661}\) then use \(\text{ip.pipe:1920}\) or \(\text{ip.spipe:3660}\) for KT1 work from an automation provider to the discovered target. These protocols shall be enabled in ITM to complete the discovery.

**ITM TEPS host**

Verify whether ITM dashboard data provider is installed or enabled. If not, optionally install or enable it. See the [Verifying the dashboard data provider is enabled](#) topic in the IBM Tivoli Monitoring documentation.

**Important:** If you use 64-bit Windows operating system, make sure that your system PATH (or path entry in the \(\text{kfwenv}\) file) points to 64-bit TMAITM6 directory. If it is not there, add it manually. For example, if you have ITM installed in the \(\text{C:\IBM\ITM}\) directory, you must have \(\text{C:\IBM\ITM\TMAITM6\_x64}\) specified in either system path environment variable or in the PATH directive in the \(\text{C:\IBM\ITM\CNPS\kfwenv}\) file.

**2. Run the automation\_provider script from ITM TEPS host**

Execute the \(\text{automation\_provider}\) script on your server:

- For Linux operating system:
  
  \(\text{automation\_provider.sh install -t /tmp/log -i /opt/IBM/ITM -c /tmp/provider.properties}\)

- For Windows operating system:
  
  \(\text{automation\_provider.ps1 install -t /tmp/log -i /opt/IBM/ITM -c /tmp/provider.properties}\)

An important point to note here is that even if a provider.properties file is created prior to running the automation\_provider script, this is ignored and a 'default' provider.properties is created in the ITM TEPS config path \(\text{($\text{ITM\_HOME/iw/profiles/ITMProfile/installedApps/ITMCell/ itmautomationprovider.ear/itmautomationprovider.war/WEB-INF/ provider.properties)}\}

You will have to locate this file, manually make the changes to the parameters and then restart the TEPS for the file to be effective.

**Required files**

The following files must be present in the directory, from which you run the installation script:

- The installation script:
- For Windows operating system: TADDM_CD_ISO/itm-discovery-support/automation_provider.ps1, and its submodules located in the TADDM_CD_ISO/itm-discovery-support/mod/ps/.

- ITM automationprovider.ear - the package with ITM OSLC Execute Automation Service Provider. The exact location of the file is TADDM_CD_ISO/itm-discovery-support/ear/itmautomationprovider.ear.
- provider.properties - the example configuration file for ITM OSLC Execute Automation Service Provider. The file can be configured manually and passed to install script as a parameter. If not passed, you must provide the required parameters during the installation. The exact location of the file is TADDM_CD_ISO/itm-discovery-support/template_provider.properties.
- KT1 support libraries for the appropriate operating system and its architecture, 32 or 64 bit.
  - For Linux operating system:
    TADDM_CD_ISO/itm-discovery-support/linux32
    TADDM_CD_ISO/itm-discovery-support/linux64
  - For AIX operating system:
    TADDM_CD_ISO/itm-discovery-support/aix32
    TADDM_CD_ISO/itm-discovery-support/aix64
  - For Linux on IBM System Z (zLinux):
    TADDM_CD_ISO/itm-discovery-support/linuxz32
    TADDM_CD_ISO/itm-discovery-support/linuxz64
  - For Windows operating system:
    TADDM_CD_ISO/itm-discovery-support/win32
    TADDM_CD_ISO/itm-discovery-support/win64

Configuring the provider.properties file

You shall optionally configure the provider.properties file by setting the following parameters:

  
  **Note**: Modify the localhost to represent the hostname or IP address of TEPS server.

  
  **Note**: Modify localhost in this case to represent the hostname or IP address of your TEPS server.

  
  **Note**: Modify localhost in this case to represent the hostname or IP address of your hub TEMS server, which may or may not be on the same host as your TEPS (most production environments should have TEMS and TEPS on separate servers).
**Note:** If you have nondefault configuration of ITM CURI or ITM SOAP, or if you configured SSL security on ITM TEPS, or both, make sure that you specify correct URL addresses for the `com.ibm.automationprovider.itm.curi.url` and `com.ibm.automationprovider.itm.soap.url` properties.

The values of the parameters specified in the `provider.properties` file have precedence over the values of the parameters defined from command line.

**Fix Pack 5** If you intend to run a discovery on the RTEMS then ensure that you have the parameter "KT1_TEMS_SECURE=YES" enabled in the environment file.

**Registering OSLC Execute Automation Service Providers in JAZZ SM Registry Services (FRS)**

You can optionally register OSLC Execute Automation Service Providers in JAZZ SM Registry Services. Choose one of the following methods:

- Add the following parameters to the `provider.properties` file:
  - `com.ibm.automationprovider.frs.url` - specifies FRS URL address for registration of OSLC Execute Automation Service Provider. The full URL address of the collection is required, for example `http://9.122.100.100:9083/oslc/pr/collection`.
  - `com.ibm.automationprovider.frs.user` - specifies the user name that is used for connection to FRS.
  - `com.ibm.automationprovider.frs.password` - specifies the password that is used for connection to FRS.
  - `com.ibm.automationprovider.registration.initialdelay=5000` - specifies the time between the start of OSLC Execute Automation Service Provider and the first attempt to register in FRS. The default value is 5000 and is expressed in milliseconds. To disable the registration, set the value to -1.

- Add the `-f` option in the command line, for example `./automation_provider.sh -f`, and during the provider installation, when prompted, provide the required parameters.

**Installing OSLC Execute Automation Service Provider on IBM Tivoli Monitoring:**

To install OSLC Execute Automation Service Provider on IBM Tivoli Monitoring (ITM), you must run the `automation_provider` script. OSLC Execute Automation Service Provider can be installed in a noninteractive or interactive mode.

**Procedure**

To install OSLC Execute Automation Service Provider, run the following `automation_provider` script from ITM TEPS host:

- For Linux operating system:
  ```
  ```

- For Windows operating system:
  ```
  ```

where:
-i <ITM-HOME>
is the directory of ITM TEPS installation, for example /opt/IBM/ITM.

-t <TEMP-DIR>
is the temporary files destination directory. The default value is /var/log/automation_provider.

-h <TEPS-IP>
is the IP address of ITM TEPS host.

-p <TEPS-PORT>
is the HTTP port of ITM TEPS.

-c <CONFIG-FILE>
is the destination of the provider.properties file, which contains OSLC Execute Automation Service Provider configuration.

-f is a flag that you can use to be prompted during the installation to provide the required parameters to register OSLC Execute Automation Service Providers in JAZZ SM Registry Services

Important: All parameters of the installation script are optional. You can specify them in any order.

Examples:
 automation_provider.sh install -t /tmp/log -i /opt/IBM/ITM -h 9.100.100.200 -p 15210
 automation_provider.ps1 install -i /opt/IBM/ITM

• You can install OSLC Execute Automation Service Provider in a noninteractive mode. Complete the following steps:
  1. Configure the provider.properties file. See the "Configuring the provider.properties file" on page 191 section.
  2. Run the following automation_provider script from ITM TEPS host:
    - For Linux operating system:
      automation_provider.sh install -t /tmp/log -i /opt/IBM/ITM -c /tmp/provider.properties
    - For Windows operating system:
      automation_provider.ps1 install -t /tmp/log -i /opt/IBM/ITM -c /tmp/provider.properties

Note: If you have nondefault configuration of ITM CURI or ITM SOAP, or if you configured SSL security on ITM TEPS, or both, install OSLC Execute Automation Service Provider in a noninteractive mode. Make sure that you specify correct URL addresses for the com.ibm.automationprovider.itm.curi.url and com.ibm.automationprovider.itm.soap.url properties.

• You can install OSLC Execute Automation Service Provider in an interactive mode. During the installation, provide values for required parameters as specified in the "Configuring the provider.properties file" on page 191 section.

Verifying OSLC Execute Automation Service Provider installation:

You can verify whether the OSLC Execute Automation Service Provider was successfully installed on IBM Tivoli Monitoring.

Procedure
1. Ensure that ITM TEMS has any running Windows, Linux, or UX agents by running the following commands:
2. Ensure that each ITM TEMS has an Automation Plan. The plans must contain IP addresses of ITM endpoints. Open the following web addresses in your web browser:

http://<ITM_TEPS>:<ITM_PORT>/itmautomationprovider
http://<ITM_TEPS>:<ITM_PORT>/itmautomationprovider/services/plans

Example

http://9.100.200.100:15210/itmautomationprovider/services/plans

Checking the status of ITM OSLC Execute Automation Service Provider:

You can check the status of ITM OSLC Execute Automation Service Provider installation.

Procedure

Run the following automation_provider script:

• For Linux operating system:
  automation_provider.sh status [i- <ITM-HOME>] [t- <TEMP-DIR>]

• For Windows operating system:
  automation_provider.ps1 status [i- <ITM-HOME>] [t- <TEMP-DIR>]

where:

i- <ITM-HOME>
  is the directory of ITM TEMS installation, for example /opt/IBM/ITM.

t- <TEMP-DIR>
  is the temporary files destination directory. The default value is /var/log/automation_provider.

Important: All parameters of the script are optional. You can specify them in any order.

Examples

automation_provider.sh status
automation_provider.ps1 status -i /opt/IBM/ITM
automation_provider.sh status -t /tmp/log -i /opt/IBM/ITM

Uninstalling OSLC Execute Automation Service Provider:

You can uninstall ITM OSLC Execute Automation Service Provider by running the automation_provider script.

Procedure

Run the following automation_provider script:

• For Linux operating system:
  automation_provider.sh uninstall [i- <ITM-HOME>] [t- <TEMP-DIR>]

• For Windows operating system:
  automation_provider.ps1 uninstall [i- <ITM-HOME>] [t- <TEMP-DIR>]

where:

i- <ITM-HOME>
  is the directory of ITM TEMS installation, for example /opt/IBM/ITM.
t- `<TEMP-DIR>`

is the temporary files destination directory. The default value is
/var/log/automation_provider.

**Important:** All parameters of the script are optional. You can specify them in any
order.

**Examples**

```
automation_provider.sh uninstall
automation_provider.ps1 uninstall -i /opt/IBM/ITM
automation_provider.sh uninstall -t /tmp/log -i /opt/IBM/ITM
```

**Configuring the discovery for ITM OSLC Execute Automation Service Provider**

When you use ITM OSLC Execute Automation Service Provider, you can configure
the discovery process by setting the following properties.

```
com.collation.discover.dwcount=32
```

The default value is 32.

This property is a TADDM server property which defines the number of
discovery worker threads.

For the best results, set the `com.collation.discover.dwcount` and
`KT1_RPC_THREADS` properties to the same value.

```
com.ibm.automationprovider.kt1.concurrenttasks.limit=100
```

The default value is 100.

This property is an ITM OSLC Execute Automation Service Provider
property that can be edited in the `provider.properties` file. It defines the
number of concurrent requests that the provider issues to TEMS. Excessive
requests are queued at the provider level.

**Note:** Change the value of this property only if additional throttling
between TADDM and TEMS is required, or more than 100 KT1 worker
threads are set.

```
KT1_RPC_THREADS=10
```

The default value is 10.

This is an ITM TEMS property that can be edited in the
`ITM_HOME/config/kbbenv.ini` file. It defines the number of worker threads
that respond to KT1 requests.

For the best results, set the `KT1_RPC_THREADS` and
`com.collation.discover.dwcount` properties to the same value.

**Troubleshooting OSLC automation discovery**

This section lists the best practices for TADDM ITM OSLC discovery with regard
to performance. To discover multiple targets via TADDM ITM OSLC based
discovery, consider the following:

1. Open a case with ITM Support for the following code fixes, if they are not
available in your ITM release.

   **APAR IJ02368 ITM OSLC TADDM discovery performance**
   TADDM discoveries through ITM takes too long to complete and TACMD
may hang. It also allows to override the timeout through the environment variable: KT1_SOAP_EXECUTE_TIMEOUT=20 defined in the TEPS cq.ini file.

A request timeout of 20 seconds is enough for most requests. Using the discovery rate of 300 seconds (default), limits in an environment where even a small number of requests do timeout.

**APAR IJ02662 TADDM discoveries via ITM monopolizes the hub's soap thread and take too long**

This APAR supports Node Status caching and eliminates the bulk of these requests. Caching is enabled through the following environment variable, where 120 is the number of seconds that a cache entry remains in the cache.

KT1_NODE_STATUS_CACHE_TTL=120 defined in the TEPS cq.ini file.

APAR IJ01062, another fix optimizes requests made to the Node List table that is included in IJ02662.

To install, do the following:
- Replace a couple of libraries used by TEPS
- Define two new environment variables
- Restart TEPS

There are no changes on the Hub side.

2. Open a case with TADDM support to obtain performance fixes for the automation provider as per APAR IJ12778. Currently the files in this fix are Execute.class. June14 KT1Wrapper.class.may23

Take the backup and replace the files on TEPS. Remove the date suffix, confirm the proper permissions and restart TEPS. On the lab systems, the files are in the below directories:

```
/opt/IBM/ITM/1x8266/iw/profiles/ITMProfile/installedApps/ITMCell
/itmautomationprovider.ear/itmautomationprovider.war/WEB-INF/classes/com/ibm/cdb/integration/actions/Execute.class

and KT1Wrapper.class is here;
```

```
/opt/IBM/ITM/1x8266/iw/profiles/ITMProfile/installedApps/ITMCell
/itmautomationprovider.ear/itmautomationprovider.war/WEB-INF/classes/com/collation/platform/session/
```

and restart TEPS

It is important to apply all the ITM and TADDM patches in one go. All APARs are TEPS fixes, more specifically, they are fixes to the KT1 component on which TEPS relies to issue discovery requests to the Hub. Once TEPS is restarted, check after 10 minutes if the plans are available. If they are not, check TEPS SystemOut.log for errors.

If they are available, run the OSLCAutomationAgent on TADDM to refresh the database with the new content:

```
dist/support/bin/runtopobuild.sh -w -a OSLCAutomationAgent--forceScopeSetRefresh true
```

Here are additional best practices from prior issues you may want to consider.

**collation.properties suggestions:**

- If you are not doing any direct discovery of servers (discovery without ITM), set this to true as auth caching can cause issues during OSLC discovery if too many threads are going after the same auth at once:

```
com.ibm.cdb.security.auth.cache.disabled=false
```
Note: In APAR IJ01289, the default value is true with a new **ITM only** property `com.ibm.cdb.security.auth.cache.itm.disabled` so, if you have FP5, skip this step.

- If you ever use the out of the box 'Level 3 Discovery' profile and expect it to use OSLC, set this to true as the default is false:
  
  `com.ibm.cdb.session.prefer.OSLCAutomation.Level_3_Discovery=false`

- If agents can move around between TEMS, set this to true on the PSS to ensure the cache is fully refreshed each time:
  
  `com.ibm.cdb.topobuilder.integration.oslc.automation.scope.always refresh=true`

You can manually run runtopobuild before discovery to ensure the agents are correctly assigned or if they do not move that frequently, leave it as it is. By default, the agent should run several times a day (see `com.ibm.cdb.topobuilder.groupinterval.integration` property). If you start seeing errors like 'CTJTP1404E Request failed with the following error returned from the Automation Provider: Could not execute remote command automation plan: Specified ip address 1.2.3.4 is not allowed for given AutomationPlan” it means the TADDM cache is out dated and the agent needs to be run.

Ensure the TEMS soap thread count is higher than the dwcount (2x dwcount is considered the best).

The default dwcount is 32 in dist/etc/collation.properties

`dwcount` represents how many sensors can run at one time. The value of this property is based on your capacity on both TADDM and ITM. Usually the default value is sufficient. If you increase the dwcount from the default value, you should increase the maximum memory for the TADDM DiscoveryService and Discover jvms via the `-Xmx jvmarg.`

To set the soap thread count to 2x dwcount see: https://www.ibm.com/developerworks/community/blogs/0587adbc-8477-4318c689226aedeal1ed/entry/tacmd_commands_running_slow_and_the_number_of_SOAP_processes?lang=en

**Additional ITM changes:**

- **KT1_RPC_THREADS=10**
  
  The default value is 10.

This is an ITM TEMS property that can be edited in the `ITM_HOME/config/kbbenv.ini` file. It defines the number of worker threads that respond to KT1 requests.

For best results, set the `KT1_RPC_THREADS` and `com.collation.discover.dwcount` properties to the same value.

Under normal operation, the following properties should be false in `provider.properties` otherwise the output files will remain until manually cleared:

- `com.ibm.automationprovider.temp.remote.keepoutputs=false`
- `com.ibm.automationprovider.temp.local.keepfiles=false`
- `com.ibm.automationprovider.temp.remote.keepscripts=false`

**Integrating TADDM with other products via OSLC Automation**

TADDM can be integrated with other products using Open Services for Lifecycle Collaboration (OSCL) Automation. TADDM connects to OSLC Execute Automation
Service Provider that provides data about other products infrastructure, which can be discovered by TADDM using OSLC Automation Session.

The discovery with the usage of OSLC Execute Automation Service Provider is a generic process, which can be enhanced to include the discovery of other products that implement their own OSLC Execute Automation Service Providers. During the discovery, one port per OSLC Execute Automation Service Provider host or Jazz SM Registry Services, or both, is opened. It ensures better security control.

The following table lists topics that contain more information about discovery over OSLC.

<table>
<thead>
<tr>
<th>Information</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuring for discovery</td>
<td>“Configuring for discovery over OSLC Automation Session” on page 107</td>
</tr>
<tr>
<td>TADDM server properties</td>
<td>“Properties for discovery using OSLC Automation Session” on page 79</td>
</tr>
<tr>
<td>Sensors that support discovery using OSLC Automation Session</td>
<td>See the Sensors that support discovery using OSLC Automation Session topic in the TADDM Sensor Reference</td>
</tr>
</tbody>
</table>

**OSLC Execute Automation Service Provider**

OSLC Execute Automation Service Provider is used to populate data about IP addresses of endpoints managed by other products to TADDM. The data is used for discovery of endpoints using OSLC Automation Session.

TADDM can get the destination of OSLC Execute Automation Service Provider from Jazz SM Registry Services or from the collation.properties file.

TADDM can be connected directly to many OSLC Execute Automation Service Providers and to a single instance of Jazz SM Registry Services, where many OSLC Execute Automation Service Provider can be registered. Every OSLC Execute Automation Service Provider stores information about an instance of a particular product with which TADDM integrates, for example about ITM HUB.

**Related reference:**

“ITM OSLC Execute Automation Service Provider” on page 186

ITM OSLC Execute Automation Service Provider is used to import data about IP addresses of endpoints managed by IBM Tivoli Monitoring to TADDM and to discover IBM Tivoli Monitoring endpoints via OSLC Automation Session.

**Configuring TADDM to use OSLC Execute Automation Service Provider:**

To be able to run discovery using OSLC Automation Session, you must configure TADDM to use OSLC Execute Automation Service Provider.

**Procedure**

To configure TADDM to use OSLC Execute Automation Service Provider, complete the following steps:

1. Ensure that OSLC Execute Automation Service Provider is installed and that it is running.
2. Connect TADDM to OSLC Execute Automation Service Provider. You can do it in two ways, directly or by using Jazz for Service Management Registry Services. These methods can be combined in case of more than one OSLC Execute Automation Service Provider.

   • To directly connect TADDM to OSLC Execute Automation Service Provider, add the addresses of OSLC Execute Automation Service Providers to the com.ibm.cdb.topobuilder.integration.oslc.automationprovider property in the collation.properties file.
   
   • To connect TADDM to OSLC Execute Automation Service Provider by using Jazz for Service Management Registry Services, enable TADDM Jazz SM Registry Services lookup of OSLC Execute Automation Service Providers. Complete the following steps:
     
     a. Ensure that JAZZ SM Registry Services is running.
     
     b. Ensure that OSLC Execute Automation Service Providers are connected to Jazz SM Registry Services.
     
     c. Configure one of following properties in the collation.properties file to provide address to Jazz SM Registry Services:
        
        com.ibm.cdb.topobuilder.integration.oslc.frsurl
        com.ibm.cdb.topobuilder.integration.oslc.automation.frsurl

3. Restart the TADDM server.

Results

After you configure TADDM, you can run discovery using OSLC Automation Session.

Related reference:

"Properties for discovery using OSLC Automation Session” on page 79

These properties apply to discovery using OSLC Automation Session.

Command line interface for OSLCAutomationAgent

OSLCAutomationAgent is used to collect data from OSLC Execute Automation Service Providers. You can use commands to run the agent manually, and to refresh or update the scope sets that it creates.

The addresses of OSLC Execute Automation Service Providers are configured in the collation.properties file or downloaded from Jazz SM Registry Services, or both. The agent connects to every OSLC Execute Automation Service Provider to get the list of Automation Plans that are compatible with TADDM. The Automation Plans consist of the IP addresses that the agent uses to cache and create discovery scope sets. For example, when TADDM integrates with IBM Tivoli Monitoring, the Automation Plans consist of the IP addresses of ITM TEMS servers and endpoints (agents) that are managed by IBM Tivoli Monitoring.

OSLCAutomationAgent caches and creates scope sets with IP addresses of IBM Tivoli Monitoring agents. Each ITM TEMS has a separate scope set.

OSLCAutomationAgent runs periodically in the Integration Agents' group.

You can use the following commands on OSLCAutomationAgent.

• To run the agent manually, use the following command:
  
  /taddm/dist/support/bin/runtopobuild.sh -a OSLCAutomationAgent

• To refresh scope sets, use the following command:
  
  /taddm/dist/support/bin/runtopobuild.sh -a OSLCAutomationAgent -s true
Note: Scope sets are refreshed only in case of changes in ITM Automation Provider's Automation Plan. To force the scope sets refresh, use the following command:

```
/taddm/dist/support/bin/runtopobuild.sh -a OSLCAutomationAgent --forceScopeSetRefresh true
```

The scope sets are available in Discovery Management Console in the **Scopes** pane.

- To display the cached scope sets, use the following commands:
  
  ```
  /taddm/dist/support/bin/runtopobuild.sh -a OSLCAutomationAgent -d true
  /taddm/dist/support/bin/runtopobuild.sh -a OSLCAutomationAgent --displayCache true
  ```

The scope sets are displayed in the following TADDM log files:

- `<COLLATION_HOME>/dist/log/services/TopologyBuilder.log`
- `<COLLATION_HOME>/dist/log/agents/OSLCAutomationAgent.log`

The following example shows the output, which can be found in the `<COLLATION_HOME>/dist/log/agents/OSLCAutomationAgent.log` file:

```
```

Related concepts:

- [“Topology building process overview” on page 14](#)

TADDM runs the topology building process on a periodic basis. Until the topology building process has completed after a discovery or after a bulk load operation, unreconciled objects might exist in the TADDM database, and the topology relationships might be incomplete.

**Integrating TADDM with IBM Tivoli Monitoring (old method)**

Depending on the specific tasks that you must do in your IT environment, you can use the integration capabilities that are available between IBM Tivoli Application Dependency Discovery Manager (TADDM) and IBM Tivoli Monitoring. You can integrate TADDM with IBM Tivoli Monitoring by using IBM Tivoli Monitoring Scope sensor.

**New integration method**

**Important**: Starting with the TADDM version 7.3.0, it is advised to integrate with IBM Tivoli Monitoring 6.3 using OSLC Automation. The old method of integration with the use of IBM Tivoli Monitoring Scope sensor is deprecated and will be removed from future releases.

Learn more about TADDM integration with IBM Tivoli Monitoring by using OSLC Automation at ["Integrating TADDM with IBM Tivoli Monitoring via OSLC Automation” on page 185](#) and about sensors that support discovery using OSLC Automation in the [Sensors that support discovery using OSLC Automation](#) topic of the TADDM Sensor Reference.
The old integration method

All of the following sections apply to the old method of integration. You can still use it, but remember that this method is deprecated and will be removed from future releases.

Table 1. correlates some tasks that you might need to do with the integration capabilities that you should use, and the remaining sections provide an overview of these integration capabilities.

Table 44. User tasks with corresponding integration capabilities to use

<table>
<thead>
<tr>
<th>Task</th>
<th>Integration capability to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gain insight into availability by viewing the operating system settings, application settings, and change history of systems that are monitored by IBM Tivoli Monitoring.</td>
<td>• “Discovery using IBM Tivoli Monitoring”&lt;br&gt; • “Launch in context” on page 203</td>
</tr>
<tr>
<td>Ensure that operating systems that are discovered by TADDM are monitored for availability.</td>
<td>• “Discovery using IBM Tivoli Monitoring”&lt;br&gt; • “Monitoring Coverage reports” on page 203</td>
</tr>
<tr>
<td>View the availability and performance of systems that are discovered by TADDM.</td>
<td>• “IBM Tivoli Monitoring DLA” on page 202&lt;br&gt; • “Monitoring Coverage reports” on page 203</td>
</tr>
<tr>
<td>Monitor a business application for configuration changes.</td>
<td>• “Discovery using IBM Tivoli Monitoring”&lt;br&gt; • “Change events” on page 203&lt;br&gt; • “Launch in context” on page 203</td>
</tr>
</tbody>
</table>

Discovery using IBM Tivoli Monitoring

TADDM can perform Level 1, Level 2, and some Level 3 discoveries using an IBM Tivoli Monitoring 6.2.1 or later infrastructure. TADDM discovers configuration items in the IBM Tivoli Monitoring environment by using only the credentials for your Tivoli Enterprise Portal Server rather than the credentials for each computer that the portal server monitors.

TADDM leverages the Tivoli Monitoring infrastructure in the following two ways:
- TADDM obtains the list of Tivoli Monitoring endpoints from the Tivoli Enterprise Portal Server both to create basic Level 1 discovery information and to create scopes for deeper Level 2 and 3 discovery.
- TADDM uses the Tivoli Monitoring infrastructure both to run CLI commands on target systems for the sensors in Level 2 and 3 discovery and to capture the output of those commands.

This capability provides the following benefits:
- Rapid deployment of TADDM in existing Tivoli Monitoring environments
- No need for TADDM anchor and gateway servers
- No need to define scope sets that contain computers to scan. Only a scope with a single entry for the Tivoli Enterprise Portal Server is required.
- No need to define an access list (operating system credentials) for discovery targets
• Only a single access list entry for the Tivoli Enterprise Portal Server GUI logon is required.

Table 45. Topics that contain more information about discovery using IBM Tivoli Monitoring

<table>
<thead>
<tr>
<th>Information</th>
<th>Location of information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuring for discovery using IBM Tivoli Monitoring</td>
<td>“Configuring for discovery using IBM Tivoli Monitoring (old method)” on page 106</td>
</tr>
<tr>
<td>TADDM server properties that apply to discovery using IBM Tivoli Monitoring</td>
<td>“Properties for discovery using IBM Tivoli Monitoring (old method)” on page 77</td>
</tr>
</tbody>
</table>

• Sensors that support discovery using IBM Tivoli Monitoring
• IBM Tivoli Monitoring Scope sensor, including information about configuring the sensor and about troubleshooting any known problems that might occur when deploying or using the sensor
  TADDM Sensor Reference

IBM Tivoli Monitoring DLA

The IBM Tivoli Monitoring discovery library adapter (DLA) extracts configuration data from Tivoli Monitoring about the computer systems and databases that Tivoli Monitoring monitors. The output of the DLA is a formatted XML file that contains these components and their relationships. The output of the DLA also includes data that represents Tivoli Monitoring agents and data that is used for launching availability views from TADDM. For detailed information about loading the DLA-exported data into TADDM, see the The bulk load program topic in the TADDM User’s Guide.

To run DLA, complete the following steps:


2. Copy the DLA output file to the TADDM host.

3. Use the bulk load program to load DLA from ITM to TADDM. Use the following command:

   $COLLATION_HOME/bin/loadidml.sh -u user -p password -f path_to_DLA

When you install new Tivoli Monitoring agents, they can provide additional support to the Tivoli Monitoring DLA. The agents provide information to populate the monitoring coverage reports, only the monitoring coverage for operating systems report does not require a DLA.

When you install an agent, you must enable application support for those agents to ensure that the agent participates in the output generated by the DLA. Not all agents support the Tivoli Monitoring DLA.

For information about configuring application support for non-standard agents, see the appropriate documentation. To verify that an agent supports the Tivoli Monitoring DLA, see the documentation for the IBM Tivoli Composite Application Manager agent.
Monitoring Coverage reports

The Monitoring Coverage reports show details about various components in your environment. You can generate a report for operating systems, databases, Microsoft applications, VMware servers, and System p components in your environment. These components are monitored by IBM Tivoli Monitoring 6.1 or later agents.

For more information about Monitoring Coverage reports, see TADDM User’s Guide.

Change events

You can configure TADDM to notify IBM Tivoli Monitoring when a change to a discovered resource is detected.

Table 46. Topics that contain more information about change events

<table>
<thead>
<tr>
<th>Information</th>
<th>Location of information</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Configuring TADDM to send change events</td>
<td>“Sending change events to external systems” on page 211</td>
</tr>
<tr>
<td>• Configuring an IBM Tivoli Monitoring data provider</td>
<td></td>
</tr>
<tr>
<td>• Configuring change events for a business system</td>
<td></td>
</tr>
</tbody>
</table>

Launch in context

With launch in context, you can view TADDM data within the Tivoli Enterprise Portal views of IBM Tivoli Monitoring.

By configuring topology views to show in the Tivoli Enterprise Portal, you can view physical infrastructure, application infrastructure, and business system topologies within Tivoli Enterprise Portal availability views.

Table 47. Topics that contain more information about launch in context

<table>
<thead>
<tr>
<th>Information</th>
<th>Location of information</th>
</tr>
</thead>
<tbody>
<tr>
<td>URLs that are required for display of topology views</td>
<td>“Configuring for launch in context” on page 208</td>
</tr>
<tr>
<td>Instructions for configuring launch in context for viewing the operating system settings, application settings, and change history for incoming change events</td>
<td>“Creating detail links in configuration change event reports in IBM Tivoli Monitoring” on page 221</td>
</tr>
</tbody>
</table>

Registering configuration items for the Context Menu Service and Data Integration Service

If you are using the Context Menu Service (CMS) and Data Integration Service (DIS) to enable flexible cross-product launch points, you must register TADDM configuration items (CIs) in the CMS/DIS database.

Before you begin

Before you can use the Context Menu Service and Data Integration Service, you must first set up the CMS/DIS database.
About this task

TADDM CIs are registered in the CMS/DIS database in two ways:
- Initial registration using the CMS/DIS registration script
- Periodic automatic updates by the CMSDISAgent topology builder agent

Running initial registration

To complete initial registration of TADDM configuration items (CIs) in the Context Menu Service and Data Integration Service database, you must manually run the run_cms_dis_registration script. The CMSDISAgent topology builder agent does not automatically update CI registration until after this initial registration is complete.

About this task

If you are using a streaming server deployment, run the registration script on the primary storage server. If you are using a synchronization server deployment, run the registration script on the synchronization server.

Procedure

To complete initial registration of TADDM CIs:
1. At a command prompt, navigate to the $COLLATION_HOME/bin directory.
2. Run the run_cms_dis_registration script for your operating system:
   - Linux and UNIX systems:
     - register [guid] |
     - clean [guid [classtype]] |
     - re-register-all | register-menu | help ]
   - Windows systems:
     - run_cms_dis_registration.bat [ register [guid] |
       - clean [guid [classtype]] |
       - re-register-all | register-menu | help ]

Where:

register [guid]

Registers TADDM data in the Context Menu Service and Data Integration Service database. You can optionally specify the globally unique identifier (GUID) of a model object you want to register.

The first time you run the script with the register option and no GUID specified, all TADDM data is registered in the database, and all launch points are registered with the Context Menu Service. Subsequent runs with this option register only changes to TADDM data that have occurred since the previous run. This option is the default option.

If you specify a GUID, only the model object with the specified GUID is registered.

Note: Initial registration of all TADDM data can take a long time.

clean [guid [classtype]]

Unregisters TADDM data currently in the database.
If you do not specify a GUID, all TADDM data is unregistered. If you specify a GUID, only the model object with the specified GUID is unregistered. If the model object with the specified GUID is no longer available in TADDM, you must also specify the model object type.

re-register-all
Unregisters all TADDM data and launch points and then repeats the initial registration. This option is equivalent to running the script with the clean option and then with the register option.

register-menu
Updates the menu definitions only that are registered in the Context Menu Service database. Use this option when the TADDM data is registered but you want to update the menu definitions only.

help
Shows help information for the script.

Example
- This example registers all TADDM data with the Context Menu Service and Data Integration Service when run for the first time; on subsequent runs, it registers all changes since the last run:
  ```
  ./run_cms_dis_registration.sh
  ```
- This example registers only the model object with the specified GUID.
  ```
  ./run_cms_dis_registration.sh register 3950DF835FA0337A829D864415CC1384
  ```
- This example removes all registered TADDM data:
  ```
  ./run_cms_dis_registration.sh clean
  ```
- This example removes the object with the specified GUID and model object type:
  ```
  ./run_cms_dis_registration.sh clean 3950DF835FA0337A829D864415CC1384
  LinuxUnitaryComputerSystem
  ```
- This example removes all registered TADDM data and then repeats the registration:
  ```
  ./run_cms_dis_registration.sh re-register-all
  ```

What to do next
If you want to run the registration script again at a later time, first disable the CMSDISAgent topology builder agent to stop incremental updates. To disable the agent, edit the $COLLATION_HOME/etc/collation.properties file and set the following property:

```
com.ibm.cdb.DisCmsIntegration.enabled=false
```

After the script finishes, you must then re-enable the agent by setting the property to true.

Configuring the CMSDISAgent
The CMSDISAgent runs periodically as a topology builder agent and updates the registration of TADDM configuration items (CIs) in the Context Menu Service and Data Integration Service database, registering any new or modified CIs and deregistering any deleted CIs.

About this task
If enabled, the CMDDISAgent begins running after you complete the initial registration of TADDM CIs using the `run_cms_dis_registration` script. You can
modify the agent configuration to change how the agent runs.

Procedure

- To enable or disable the CMSDISAgent, edit the $COLLATION_HOME/etc/collation.properties file and set the following property:

  
  com.ibm.cdb.DisCmsIntegration.enabled=value

  
  where value is either true or false. If the value is set to true, the agent runs periodically after the initial registration is completed. (This property does not affect the operation of the run_cms_dis_registration script, which can be run at any time.)

- To customize which CIs are registered in the database, modify the following files in the $COLLATION_HOME/etc/cmsdis directory:

  classtype-changehistory.list
  
  Lists the model object types of CIs for which TADDM has launch-in-context support for the change history report.

  classtype-detailPanel.list
  
  Lists the model object types of CIs for which TADDM has launch-in-context support for the details panel.

You can remove any model object types that are not needed for other products to launch TADDM in context. Do not add any types to these files; TADDM might not support launching in context for additional types. After modifying the class type list files, disable the agent and then run the run_cms_dis_registration script again, specifying the re-register-all option.

Creating a Discovery Library store

A Discovery Library store is a directory or folder on a computer in the data center, and it represents the common location for all Discovery Library Adapters (DLAs) to write the XML files that contain resource information. XML data files to be bulk loaded into a TADDM system are placed in the Discovery Library store. To use the bulk loader program, you must create a Discovery Library store.

Before you begin

A DLA is a software program that extracts data from a source application such as IBM Tivoli Monitoring or IBM Tivoli Business Service Manager.

Each DLA writes XML files that contain resource information in a particular XML format called the Identity Markup Language (IdML). Any XML file that is written in the IdML format is commonly referred to as a book. To see the Tivoli collection of books that can load the TADDM database with data from other Tivoli products, see [http://www.ibm.com/software/brandcatalog/ismlibrary/](http://www.ibm.com/software/brandcatalog/ismlibrary/).

DLAs are specific to a particular product because each product has a distinct method of accessing the resources from the environment. The configuration and installation of a DLA is different for every application. A typical DLA is installed on a system that has access to the data of a particular application. For example, the DLA for IBM Tivoli Monitoring is installed on a computer that has access to the IBM Tivoli Monitoring enterprise management system database. All DLAs are run using the command-line interface and can be scheduled to run using any type of scheduling program in your environment (for example, cron).
You can create a DLA to extract information from existing products or databases in your environment.

For more information about how to create a DLA and about the IdML specification, or for additional details about the Discovery Library store, see the TADDM Discovery Library Adapter Developer’s Guide.

About this task

Typically, the Discovery Library store is located on the TADDM server. If you do not set up the Discovery Library store on the TADDM server, you must ensure that the TADDM bulk load program that runs on the TADDM server can access the Discovery Library store. Other applications can run on the same computer that hosts the Discovery Library store.

Procedure

To create the Discovery Library store, complete the following steps:

1. Create a directory to store the XML files on a computer, with a distinct directory name (for example, c:\IBM\DLFS). Optionally, you can create subdirectories in the main Discovery Library store for each DLA that is used.
2. Set up a File Transfer Protocol Server (FTP) with at least one user ID. The user ID must have the following permissions: write, rename, and read access to the directory that stores the Discovery Library XML files. If you are not using FTP to transfer the XML files to the Discovery Library store, ensure that the tool you use and the user ID used to run the tool have write permissions to the Discovery Library store directory.
3. Ensure that the various Discovery Library Adapters have access to the name of the system (host name) that hosts the Discovery Library store. Most Discovery Library Adapters copy XML files to the Discovery Library store.
4. Ensure that the various Discovery Library Adapters have the user ID and password to connect to the FTP server.
5. If the DLA does not use FTP, copy your XML files (books) that you want the bulk loader program to access into that shared directory. The shared directory must be accessible by the bulk loader program.

The book writers and the administrator are not responsible for getting the books into the Discovery Library store. For example, set up a cron job to send the produced IdML books to the Discovery Library store using FTP.

What to do next

If you are creating a Discovery Library store and want to set up a TADDM database to contain DLA books, a local drive on the domain server can be the networked Discovery Library store. This directory must be defined in the $COLLATION_HOME/etc/bulkload.properties file on the domain server where the data is loaded. If you have multiple domain servers, configure the correct bulk loader program to access the corresponding shared directory. The bulk loader does not delete XML files from the Discovery Library store. You must maintain the files in the Discovery Library store. Ensure that there is enough disk space on the server for the files in the directory. If new XML files are added to the directory frequently, you must regularly clean up the directory.

If you have a synchronization server deployment, you must choose from the following options:
• If the resources that are referenced in a book are contained in the scope definitions that are defined on one domain server, load that book into the respective domain server.
• If the resources that are referenced in a book are not contained in the scope definitions that are defined on one domain server, load all of the books into the synchronization server.

Configuring for launch in context
To see more detailed information about components in your environment, you can launch TADDM views from other Tivoli applications. To configure your application to launch TADDM views in context, you must specify a URL.

Views that you can launch from other Tivoli applications
From other Tivoli applications, you can launch Data Management Portal views. You can also launch the details and change history report for a specified configuration item (CI).

In the Data Management Portal views, you can see more information for the following component groupings:
• Business applications
• Business services
• Collections

If both the TADDM server and the application from which TADDM is being launched are not configured for a single sign-on, a sign-on window is shown. Before you can view additional information in the Data Management Portal, you must provide a user name and password.

Specifying the URL to launch TADDM views
To launch TADDM views in context from other Tivoli applications, you must specify a URL.

The URL format for launching in context is:
Protocol://TADDMHostname:TADDMPort/ContextRoot/?queryString

The following list describes the valid values for each variable in the URL format:

Protocol
The Web protocol to use. Valid values are http or https.

TADDMHostname
The host name for the TADDM server to which you are launching.

TADDMPort
The port number for the TADDM server to which you are launching. The default value is 9430.

ContextRoot
The following values are valid:

cdm/servlet/LICServlet
The relative path to the Java servlet that is deployed in the Apache Tomcat server for TADDM 7.3.0, and in the WAS Liberty Profile server for TADDM 7.3.0.1, and later, and later.
cdm/queryHomePage.do

The relative path to the Query Home Page, when launched from IBM Tivoli Monitoring, using single sign-on, and specifying search text.

queryString

Contains name-value pair parameters that are delimited by separators. The format for a name-value pair is name=value. Use = to separate names and values, and use & to separate name-value pairs.

The following list describes the valid name-value pairs that can be used in the queryString variable:

view

Specifies that you want to display change history.

The only valid value is changehistory.

days_previous

Specifies the time period (the number of past days) for which to show the change history of a particular configuration item.

The valid value is a positive integer.

hoursback

Specifies the time period (the number of past hours) for which to show the change history of a particular configuration item.

The valid value is a positive integer.

guid

Specifies the Globally Unique Identifier (GUID) for a configuration item.

For the domain server and the synchronization server [Table 48 on page 210] lists the valid values for the graph parameter and indicates whether the guid parameter is optional or required based on the respective graph value.

If the graph parameter is specified with any of the following values, the guid parameter is optional:

- businessapplications
- applicationinfrastructure
- physicalinfrastructure

If the graph parameter is specified with any other type of topology graph, the guid parameter is required.

The valid value is a valid string representation of a GUID, as shown in the following example:

BA2842345F693855A3165A4B5F0088DE

You should specify only one GUID for each URL request for launch in context.

graph

Specifies the type of topology graph to be launched.

If you also specify a configuration item by providing its GUID on the guid parameter, the requested configuration item is then selected, if it is found in the topology graph that is specified on this graph parameter.

For the domain server and the synchronization server [Table 48 on page 210] lists the valid values for the graph parameter and indicates whether the guid parameter is optional or required based on the respective graph value.

Administering 209
Table 48. Valid graph values and their relationship to the guid parameter

<table>
<thead>
<tr>
<th>Domain server</th>
<th>Valid value</th>
<th>Is the guid parameter optional or required with this graph value?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>businessapplications</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>applicationinfrastructure</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>physicalinfrastructure</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>For custom collection objects:</td>
<td>Required</td>
</tr>
<tr>
<td></td>
<td>• ba_infrastructure</td>
<td></td>
</tr>
<tr>
<td>Synchronization server</td>
<td>businessapplications</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>physicalinfrastructure</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>For custom collection objects:</td>
<td>Required</td>
</tr>
<tr>
<td></td>
<td>• ba_infrastructure</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The other graph types that were used in the previous TADDM releases to render particular grouping entities by GUID are deprecated. However, to ensure compatibility with older versions, if you specify an old graph type with a GUID, the request is redirected to the new topology type.

**Username**
- Specifies the user name used to log in to TADDM.

**Password**
- Specifies the password used to log in to TADDM.

**Launchsource**
- The only valid value is ITM. It is always used with the searchtext=search_term name-value pair.

The search is confined to the configuration items of type ComputerSystem and TMSAgent, listed in the $COLLATION_HOME/etc/cdm/xml/itm_query_components.xml configuration file.

From the Query Home Page results, for each configuration item listed, you can launch the following views:
- Change History pane
- Details pane
- Data Management Portal, displaying the Details pane

**Searchtext**
- Specifies the search term. It is always used with the launchsource=ITM name-value pair.

**Examples of how to specify the URL**

The following examples show how to specify the URL to launch TADDM views:

**URL for launching the Data Management Portal, without entering authorization information separately**


If you are using a trusted connection, you must use only credentials as part of the URL for launching in context because the user name and password are not encrypted.
URL for launching the Query Home Page window for IBM Tivoli Monitoring when using single sign-on and searching for a configuration item that matches the search text

http://home.taddm.com:9430/cdm/queryHomePage.do?launchsource=itm&searchtext=127.0.0.1

URL for showing a topology of a custom collection indicated by the guid parameter


Sending change events to external systems

You can configure TADDM to notify an external event-handling system when a change to a discovered resource is detected.

To send change events from TADDM, you must have one or more of the following event-handling systems installed:

- IBM Tivoli Monitoring 6.2.1 Fixpack 2, or later
- IBM TivoliNetcool/OMNIbus, including the Event Integration Facility (EIF) probe

To see supported versions of the products, go to the “Supported versions” on page 183 section.

When a discovery completes, TADDM checks for changes to items being tracked by external event-handling systems. If any are detected, they are sent, using EIF, directly to IBM Tivoli Netcool/OMNIbus and to IBM Tivoli Monitoring using the Universal Agent.

The Universal Agent converts the received notifications to asynchronous events, and forwards the data to the IBM Tivoli Enterprise Monitoring Server component of IBM Tivoli Monitoring. The IBM Tivoli Monitoring Server stores the events and uses them to evaluate situations. The events are then passed to the IBM Tivoli Enterprise Portal for display.

IBM Tivoli Netcool/OMNIbus servers process received events according to their internal rules and display them.

To set up the sending of change events from TADDM to external event-handling systems, you must enable change events in TADDM, and configure each external recipient to handle incoming events, as appropriate.

Configuring TADDM to send change events

To send change events, you must configure TADDM with information about the event-handling systems to which you want to send change events.

About this task

Depending on your type of TADDM deployment, make the following changes on the following TADDM servers:

- In a domain server deployment, make the changes on the domain server.
- In a synchronization server deployment, make the changes on the synchronization server.
- In a streaming server deployment, make the changes on the primary storage server.
Procedure

To enable the sending of change event information, complete the following steps:

1. To enable change events, in the `$COLLATION_HOME/etc/collation.properties` file, set the following property: `com.ibm.cdb.omp.changeevent.enabled=true`

2. To configure which resources are tracked for changes and to which event-handling systems the events are sent, edit the `$COLLATION_HOME/etc/EventConfig.xml` file.

For information about the format that you must use to specify information in the `EventConfig.xml` file, see "TADDM OMP Change event module configuration."

When you upgrade TADDM, the `EventConfig.xml` file from the previous TADDM release is kept to ensure that you do not lose the customized settings that you had configured. Information about new features and how to use them is available in the `$COLLATION_HOME/etc/EventConfigDefault.xml` file. The `EventConfigDefault.xml` file is for reference only. If you want to use any of the new features, you must update `EventConfig.xml` based on the appropriate examples in `EventConfigDefault.xml`.

3. If you specified an IBM Tivoli Netcool/OMNIbus event-handling system in the `EventConfig.xml` file, create a corresponding EIF property file for the system type. To do this, complete the following steps:
   a. Create a `$COLLATION_HOME/etc/omnibus.eif.properties` property file.

TADDM OMP Change event module configuration:

To enable sending change events, you must edit the `EventConfig.xml` file to define event listeners and recipients.

Event listeners

You can define a listener by providing the necessary criteria for a TADDM query. The resulting objects that are selected by the query are checked for changes after each discovery. There can be many listeners. Both a listener and a corresponding recipient block must exist for an event routing to occur.

Use the following format to specify a listener.

```xml
<listener object="[OBJECT_TYPE]"
    enabled="true|false">
    sendCauses="true|false"
    sendOriginGuid="true|false">
    <alert recipient="[RECIPIENT_SYSTEM_NAME]"/>
    <attribute name="[ATTRIBUTE_NAME]" operator="[OPERATOR]">
        <value>
            [ATTRIBUTE_VALUE]
        </value>
    </attribute>
</listener>
```
where:

**[OBJECT_TYPE]**

is a type of model object that is represented in TADDM, for example, ComputerSystem or ITSystem. For more examples, see the TADDM Data Dictionary at http://taddmserverhost:9430/cdm/datadictionary/model-object/index.html.

**enabled**

is an attribute that allows for sending the events. The value must be set to true for the listener to be active.

**sendCauses**

is an optional attribute that defines whether the listener sends events about changes that were propagated to the model object. For example, if a change to a Windows Operating System causes a change to a ComputerSystem object, and the sendCauses attribute is set to true for a ComputerSystem listener, the listener sends an event for the change both to the ComputerSystem and to the Windows Operating System. The default value of the sendCauses attribute is false.

**sendOriginGuid**

is an optional attribute that is used with the sendCauses attribute. When the sendOriginGuid attribute is set to true, an object that matches the listener is considered the logical origin of changes that are propagated to the object. Events that are sent about propagated changes contain the unique identifier of the origin object. For example, if a change to a ConfigFile object causes a change to a ComputerSystem object, and both sendCauses and sendOriginGuid attributes are set to true for a ComputerSystem listener, the event about the ConfigFile change contains the unique identifier of the ComputerSystem object in addition to the unique identifier of the ConfigFile object. This function is available only for the Netcool/OMNIbus event recipients. The default value of the sendOriginGuid attribute is false.

**[RECIPIENT_SYSTEM_NAME]**

is an alert recipient. See “Event recipients” on page 215.

**[ATTRIBUTE_NAME]**

is the name of an attribute on [OBJECT_TYPE], which is queried.

**[OPERATOR]**

is the operator name of a TADDM MQL query. The following values are allowed.

<table>
<thead>
<tr>
<th>Operator</th>
<th>TADD MQL equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>contains-with</td>
<td>contains</td>
</tr>
<tr>
<td>ends-with</td>
<td>ends-with</td>
</tr>
<tr>
<td>equals</td>
<td>equals</td>
</tr>
<tr>
<td>greater-or-equal</td>
<td>&gt;=</td>
</tr>
<tr>
<td>greater-than</td>
<td>&gt;</td>
</tr>
<tr>
<td>less-or-equal</td>
<td>&lt;=</td>
</tr>
</tbody>
</table>

---

Table 49. Operator names of a TADD MQL query.
Table 49. Operator names of a TADDM MQL query. (continued).

<table>
<thead>
<tr>
<th>Operator</th>
<th>TADDM MQL equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>less-than</td>
<td>&lt;</td>
</tr>
<tr>
<td>not-equals</td>
<td>not-equals</td>
</tr>
<tr>
<td>starts-with</td>
<td>starts-with</td>
</tr>
</tbody>
</table>

**[ATTRIBUTE_VALUE]**

is the value against which the attribute is evaluated.

**<causeFilter>**

is an attribute that provides a means to filter the object types of cause events that are passed when the sendCauses attribute is enabled. If you specify this attribute, only cause events of the specified object type are sent. However, propagated events are still sent, for example, the ones that are a part of the object type that is specified in the listener. If the causeFilter attribute is not specified, all cause events that are found by the listener are sent to the receiver.

For example, a change to a WindowsService causes a change to the Windows Operating System and consequently to the ComputerSystem. If you set the causeFilter attribute to WindowsService, only the ComputerSystem and the WindowsService changes are shown, the Windows Operating System change is not shown.

When you set the causeFilter attribute, you can optionally set a value for the sendOriginGuid attribute. By default, the causeFilter attribute inherits the sendOriginGuid setting of the listener that is the parent of the causeFilter attribute. When you use the sendOriginGuid attribute in a causeFilter attribute, only the listener setting for that causeFilter attribute is overridden.

If you want to update objects like WindowsService or ConfigFile, whose changes are propagated to a higher-level object like ComputerSystem, capture such objects by using a combination of the sendCauses and causeFilter attributes, rather than a separate listener.

**[CAUSEFILTER_OBJECT_TYPE]**

is the class name of the object as defined in the CDM. You can use the full name, for example, `com.collation.platform.model.topology.sys.windows.WindowsService`, or the short name, for example, `WindowsService`.

**Examples of event listeners**

In the following example, a change that is detected in any ComputerSystem whose FQDN contains the "mycompany" string is sent to the recipient "enterprise-eventhost-itm".

```xml
<listener object="ComputerSystem" enabled="true">
  <alert recipient="enterprise-eventhost-itm"/>
  <attribute name="fqdn" operator="contains-with">
    <value>
      mycompany
    </value>
  </attribute>
</listener>
```

In the following example, changes to all objects of a specified type are detected.

```xml
<causeFilter objectType="WindowsService"/>
```
In the following example, a change that is detected in any object of the ComputerSystem type is sent to the recipient "enterprise-eventhost-omnibus".

```xml
<listener object="ComputerSystem" enabled="true">
  <alert recipient="enterprise-eventhost-omnibus"/>
  <attribute name="guid" operator="not-equals">
    <value>0</value>
  </attribute>
</listener>
```

In the following example, only changes that are caused by a change in a ConfigFile on a Linux Computer System are sent.

```xml
<listener object="ITSystem" enabled="true" sendCauses="true">
  <alert recipient="enterprise-eventhost-itm"/>
  <attribute name="name" operator="ends-with">
    <value>ShoppingCart</value>
  </attribute>
  <causeFilter object="ConfigFile"/>
  <causeFilter object="LinuxUnitaryComputerSystem"/>
</listener>
```

Event recipients

An event recipient is an instance of IBM Tivoli Monitoring or OMNIbus, which can receive events from the change event module. When changes are located by the change listeners, notification is sent to the corresponding recipients. You can define multiple recipients of a different or the same type simultaneously. Both a listener and a corresponding recipient block must exist for an event routing to occur.

Use the following format to specify a recipient.

```xml
<recipient name="[RECIPIENT_NAME]" type="[RECIPIENT_TYPE]">
  <address>[RECIPIENT_FQDN]</address>
  <port>[EVENT_ROUTING_PORT]</port>
  <config>[PATH_TO_EIF_CONFIGURATION]</config>
</recipient>
```

where:

[RECIPIENT_NAME]

is the name of the system referred to in the listener.

[RECIPIENT_TYPE]

is the type of software that is used for receiving events. The following types are supported:

- itm - IBM Tivoli Monitoring 6 with Universal Agent POST Data provider.
- omnibus - Netcool/OMNIbus with EIF adapter.

[RECIPIENT_FQDN]

(IBM Tivoli Monitoring only) is the fully qualified domain name of the host where the Universal Agent is located.
[EVENT_ROUTING_PORT]  
(IBM Tivoli Monitoring only) is the port, which the Universal Agent POST Data Provider specified in KUMENV as KUMP_POST_DP_PORT.

[PATH_TO{EIF_CONFIGURATION}]  
(OMNIBUS only) is the path to the EIF configuration, which is read from the properties file. Use the full path to the file.

Examples of event recipients

The following example defines a Netcool/OMNibus event recipient.

```xml
<recipient name="enterprise-eventhost-omnibus" type="omnibus">
  <config>/opt/IBM/taddm/dist/etc/omnibus.eif.properties</config>
</recipient>
```

The following example defines an IBM Tivoli Monitoring event recipient.

```xml
<recipient name="enterprise-eventhost-itm" type="itm">
  <address>itm-ua.mycompany.com</address>
  <port>7575</port>
</recipient>
```

Configuring IBM Tivoli Netcool/OMNibus

You can configure IBM Tivoli Netcool/OMNibus Version 7.3 or later to receive change events sent by TADDM. You can aggregate and customize the event data that is displayed in previous versions of Tivoli Netcool/OMNibus and you can define event-handling logic.

Before you begin

To configure IBM Tivoli Netcool/OMNibus Version 7.3 or later to receive change events that TADDM sends, see the Enabling support for TADDM events topic in the IBM Tivoli Netcool/OMNibus documentation at [http://www-01.ibm.com/support/knowledgecenter/SSSH2TQ/landingpage/NetcoolOMNibus.html?lang=en](http://www-01.ibm.com/support/knowledgecenter/SSSH2TQ/landingpage/NetcoolOMNibus.html?lang=en). The Tivoli Netcool/OMNibus documentation also includes information about the tivoli_eif_taddm.rules file. This file contains the logic to process details of configuration changes that were detected during a TADDM discovery.

In an environment where high availability, or failover, computing is used, TADDM can be configured to support automatic failover. This support occurs when TADDM events are sent to Tivoli Netcool/OMNibus. You can specify primary and secondary EIF probe addresses and their associated ports in the EIF properties file. The following example shows where to add these properties:

```
# Hostname where the NetCool/OMNibus EIF probe resides. Specify up to 8 locations.
# Each location should be separated by a comma.
# The event is sent to the first available probe in the list.
# Example:
#   ServerLocation=netcool.mycompany.com,netcool2.mycompany.com
#   ServerLocation=netcool.mycompany.com,netcool2.mycompany.com

# Port the NetCool/OMNibus EIF probe is listening on.
# There must be a port entry for each probe specified under ServerLocation.
# Example:
#   ServerPort=9998,9998
#   ServerPort=9998,9998

```

Each probe address entered must have the associated port specified in the ServerPort property. Failure to specify the port for each probe address results in an error when the event is sent. When an event cannot be sent to the primary port it
is sent to first available port on the list. Up to eight probe addresses can be specified in the *ServerLocation* property.

### About this task

In IBM Tivoli Netcool/OMNibus versions before Version 7.3, the default behavior is for all events from an event module to be combined into a single event, with the Count attribute set to display the number of events that are contained in the combined event. The following steps describe how to change that default behavior.

### Procedure

1. On the TADDM server, open the following file for editing:
   
   $COLLATION_HOME/etc/omnibus.eif.properties

2. Set the following TADDMEvent_Slot properties property values:

   ```
   TADDMEvent_Slot_object_name=$TADDM_OBJECT_NAME
   TADDMEvent_Slot_change_type=$TADDM_CHANGE_TYPE
   TADDMEvent_Slot_change_time=$TADDM_CHANGE_TIME
   TADDMEvent_Slot_class_name=$TADDM_CLASS_NAME
   TADDMEvent_Slot_attribute_name=$TADDM_ATTRIBUTE_NAME
   TADDMEvent_Slot_old_value=$TADDM_OLD_VALUE
   TADDMEvent_Slot_new_value=$TADDM_NEW_VALUE
   TADDMEvent_Slot_host=$TADDM_HOST
   TADDMEvent_Slot_port=$TADDM_PORT
   TADDMEvent_Slot_guid=$TADDM_GUID
   TADDMEvent_Slot_origin=$TADDM_ORIGIN
   ```

### What to do next

If you have any problems configuring IBM Tivoli Netcool/OMNibus, see the *Integrating TADDM with other products problems* topic in the TADDM Troubleshooting Guide.

### Configuring an IBM Tivoli Monitoring data provider

You can configure the Universal Agent initialization file to define a new data provider.

### Before you begin

If you are using Tivoli Monitoring version 6.2.2 or earlier, make sure there are no tab or space characters in the KUMPOST configuration file.

### Procedure

To configure an IBM Tivoli Monitoring data provider, complete the following steps:

1. On the Windows system where the Universal Agent is installed, click *Start* > *IBM Tivoli Monitoring* > *Manage Tivoli Monitoring Services*.
2. Right-click the Universal Agent and click *Reconfigure*.
3. In each of the two Agent Advanced Configuration windows, click *OK*.
4. To update the Universal Agent initialization file, click *Yes*. The KUMENV file is opened in the system text editor.
5. Set the KUMA_STARTUP_DP value to *POST*:

   ```
   KUMA_STARTUP_DP=POST
   ```
**Note:** If the Universal Agent is already configured to use another data provider, specify both values separated with a comma, as in the following example:

```
KUMA_STARTUP_DP=ASFS,POST
```

6. Add the required POST parameter information to the KUMENV file:

```
*----------------------------------------*
* TADDM POST DP Parameters             *
*----------------------------------------*
KUMP_POST_DP_PORT=7575                   
KUMP_POST_GROUP_NAME=TADDM              
KUMP_POST_APPL_TTL=14400                 
```

7. Save the KUMENV file, and close it.

8. To configure the agent, click **Yes**.

9. In the Manage Tivoli Enterprise Monitoring Services window, click **Universal Agent > Start**.

10. In the system text editor, create a text file. Enter the following information in the file:

```
//APP1 CONFIGCHANGE
//NAME dpPost E 3600
//ATTRIBUTES '
Post_Time T 16 Caption{Time}
Post_Origin D 32 Caption{Origin}
Post_Ack_Stamp D 28 Caption{Event time stamp}
Comp_Type D 512 Caption{Component type}
Comp_Name D 512 Caption{Component name}
Comp_Guid D 512 Caption{Component GUID}
Change_Type D 512 Caption{Change type}
Chg_Det_Time D 512 Caption{Change detection time}
Chg_Attr D 512 Caption{Changed attribute}
Srv_Addr D 512 Caption{TADDM server}
Srv_Port D 16 Caption{TADDM port}
```

11. Save the file as `%ITM_HOME%\TMAITM6\metafiles\KUMPOST`.

**Note:** Ensure that you spell the file name, KUMPOST, with uppercase letters, as shown here.

12. Open a Windows command prompt and navigate to the `%ITM_HOME%\TMAITM6` folder.

13. Run the KUMPCON.exe program to validate and import the KUMPOST metafile.

14. In the Manage Tivoli Monitoring Services window, right-click the Universal Agent, and select **Recycle**.

If you are running the Universal Agent on a Linux or UNIX system, complete the following steps:

1. Reconfigure the universal agent using the following command:

```
itmcmd config -A um
```

   When you are prompted for the data provider, enter **POST**.

**Note:** If the Universal Agent is already configured to use another data provider, specify both values separated with a comma (for example, ASFS,POST).

2. In the `%ITM_HOME%\config` directory, make a backup copy of the `um.ini` file, and then add the following entries to the original copy of the file:

```
# TADDM POST DP Parameters
KUMP_POST_DP_PORT=7575
KUMP_POST_GROUP_NAME=TADDM
KUMP_POST_APPL_TTL=14400
```

218  Application Dependency Discovery Manager: Administering
3. In the $ITM_HOME/interp/um/metafiles directory, create a text file. Enter the following information in the file:

```
//APPL CONFIGCHANGE
//NAME dpPost E 3600
//ATTRIBUTES ';
Post_Time T 16 Caption(Time)
Post_Origin D 32 Caption(Origination)
Post_Ack_Stamp D 28 Caption(Event time stamp)
Comp_Type D 512 Caption(Component type)
Comp_Name D 512 Caption(Component name)
Comp_Guid D 512 Caption(Component GUID)
Change_Type D 512 Caption(Change type)
Chg_Det_Time D 512 Caption(Change detection time)
Chg_Attr D 512 Caption(Changed attribute)
Srv_Addr D 512 Caption(TADDM server)
Srv_Port D 16 Caption(TADDM port)
```

4. Save the file as KUMPOST.

**Note:** Ensure that you spell the file name, KUMPOST, with uppercase letters, as shown here.

5. Restart the Universal Agent using the following commands:
   ```
   itmcmd agent stop um
   itmcmd agent start um
   ```

6. To validate and refresh the KUMPOST metafile, complete the following steps:
   a. Run the $ITM_HOME/bin/um_console command with the following parameters:
      ```
      um_console -h <ITM directory>
      ```
   b. At the command line, type the following text:
      ```
      validate KUMPOST
      ```
   c. When prompted for the action you want to perform on the metafile, type the following text:
      ```
      Refresh
      ```
   d. Type Yes to confirm.

**What to do next**

To verify that the Universal Agent configuration has been successful, check the change event report in the Tivoli Enterprise Portal.
To open the change event report using IBM Tivoli Monitoring 6.2.1, or later, complete the following steps:

1. Navigate to the Universal Agent that was configured to send and receive event notifications from TADDM.
2. Expand the CONFIGCHANGE node.
3. Click the DPPOST node.

**Creating configuration change situations in IBM Tivoli Monitoring**

You can use the Situation function in the Tivoli Enterprise Portal to monitor change events and to trigger situations that are based on the information in a change event.

**Procedure**

To create a configuration change situation in IBM Tivoli Monitoring, complete the following steps:

To create a configuration change situation if you are using IBM Tivoli Monitoring 6.2.1, complete the following steps:

1. In the Navigator pane of IBM Tivoli Enterprise Portal navigate to the Universal Agent that was configured to send and receive event notifications from TADDM.
2. Expand the CONFIGCHANGE node.
3. Right-click the DPPOST node. Click *Situations*.
4. In the “Situations for node_name” window, right-click *Universal Data Provider*. Click *Create New*. The Create Situation or Rule window is displayed.
5. In the *Name* field, type the name of the situation. For example, *ConfigurationChanged*.
6. In the *Description* field, type the description of the situation. For example, *A change to a tracked object was detected by TADDM*.
7. From the *Monitored Application* list, select *Universal Data Provider*.
8. Ensure that the *Correlate Situations across Managed Systems* check box is clear.
9. Click *OK*. The “Select condition” window is displayed.
10. From the *Attribute Group* list, select DPPOST.
11. From the *Attribute Item* list, select *Component name*.
12. Click *OK*. The *Formula* tab for the situation is displayed.
13. Configure the situation so that it is triggered when the component name matches the name of the resource in your environment that you want to monitor.
14. Click *OK*.

To create a configuration change situation if you are using IBM Tivoli Monitoring 6.2.2, or later, complete the following steps:

1. In the Navigator pane of IBM Tivoli Enterprise Portal navigate to the Universal Agent that was configured to send and receive event notifications from TADDM.
2. Expand the CONFIGCHANGE node.
3. Right-click the DPPOST node. Click *Situations*. 
4. In the “Situations for node_name” window, click **Create new Situation**. The Create Situation window is displayed.
5. In the **Name** field, type the name of the situation. For example, ConfigurationChanged.
6. In the **Description** field, type the description of the situation. For example, A change to a tracked object was detected by TADDM.
7. From the **Monitored Application** list, select **Universal Data Provider**.
8. Click **OK**. The “Select condition” window is displayed.
9. From the **Attribute Group** list, select **DPPOST**.
10. From the **Attribute Item** list, select **Component name**.
11. Click **OK**. The **Formula** tab for the situation is displayed.
12. Configure the situation so that it is triggered when the component name matches the name of the resource in your environment that you want to monitor.
13. Click **OK**.
14. In the Navigator pane of IBM Tivoli Enterprise Portal, right-click the node that contains the change event report. Click **Situations**.
15. In the “Situations for node_name” window, right-click the **ConfigurationChanged** situation you created and click **Start Situation**.

**Results**

When configuration change events are received, their component name is checked. If the component name matches that of the component you have specified in the situation formula, the configured situation is triggered.

**Creating detail links in configuration change event reports in IBM Tivoli Monitoring**

You can create links in a report table to a workspace displaying change history and details directly from the TADDM server. These links give more detailed information than what is displayed in a report.

**Procedure**

To create a link, in a configuration change event report, to more detailed change event information, complete the following steps:

1. To create a workspace to display the information, complete the following steps:
   a. In the Navigator pane, right-click the node within which you want to contain the workspace. Click **File > Save workspace as**. The Save Workspace As window is displayed.
   b. In the **Name** field, type the name of the workspace. For example, ConfigChangeDetails.
   c. In the **Description** field, type a description of the workspace. For example, Generic workspace for the change event table.
   d. Select the **Only selectable as the target of a Workspace Link** check box.
   e. Click **OK**.
2. To configure the workspace using IBM Tivoli Monitoring 6.2.1, or later, complete the following steps:
   a. Configure the workspace to have one navigator pane and two browser panes.
b. Click Edit > Properties.
c. In the Browser pane, select the first instance of Getting Started.
d. In the Style pane, select Use Provided Location.
e. Click OK.
f. In the Location field of one of the browser panes, type the URL of the Change History view in TADDM. When you have typed the URL on one line, do not press Enter.

```
```

The hoursback parameter specifies the number of hours for which change events are displayed. For example, setting hoursback to 6 displays all change events in the previous six hours.
g. In the Browser pane, select the second instance of Getting Started.
h. In the Style pane, select Use Provided Location.
i. Click OK.
j. In the Location field of the second of the browser panes, type the URL of the Object Details view in TADDM. When you have typed the URL on one line, do not press Enter.

```
http://$taddm_server$:$taddm_port$/cdm/servlet/LICServlet?console=web&guid=$taddm_guid$
```
k. To save the new workspace, click File > Save. Immediately after you have typed the URL into the Location field, do not press Enter, but save the workspace.

4. Click Link To > Link Wizard. The Welcome page of the Workspace Link Wizard is displayed.
5. Click Create a new link. Click Next. The Link Name page of the Workspace Link Wizard is displayed.
6. In the Name field, type the name of the link. For example, Show Details.
7. In the Description field, type a description of the link. For example, Link to details.
8. Click Next. The Link Type page of the Workspace Link Wizard is displayed.
9. Click Absolute. Click Next. The Target Workspace page of the Workspace Link Wizard is displayed.
10. In the Navigator panel, select the node containing the workspace you created. In the Workspace panel, select the workspace you created.
11. Click Next. The Parameters page of the Workspace Link Wizard is displayed.
12. You must add three symbols: "taddm_server", "taddm_port", and "taddm_guid". To add a symbol, complete the following steps:
   a. Click Add Symbol. The Add Symbol window is displayed.
   b. In the Symbol field, type the name of the symbol.
   c. Click OK.
13. For each symbol you create, you must link it to an attribute representing the correct column in the report.
   - Link the "taddm_server" symbol to the TADDM server attribute.
   - Link the "taddm_port" symbol to the TADDM web console port number.
   - Link the "taddm_guid" symbol to the Component GUID attribute.
To link a symbol to an attribute, complete the following steps:

a. In the Parameters page of the Workspace Link Wizard, select the symbol you want to link to a report column.

b. Click **Modify Expression**. The Expression Editor window is displayed.

c. Click **Symbol**. The Symbols window is displayed.

d. Navigate to **Attributes**, and select the attribute you want to link to the symbol. Click **OK**.

e. In the Expression Editor window, click **OK**. The Parameters page of the Workspace Link Wizard is displayed.

14. Click **Next**. The Summary page of the Workspace Link Wizard is displayed.
15. Click **Finish**.

**Results**

If you have active events in your change event report, a link icon is displayed next to each table row. To move to the target workspace, click the link icon and select **Show Details**. In the table row, values are substituted for symbols. In the workspace, the Change History and Object Details panels are launched in context.

**Configuring change events for a business system**

You can use the change event functionality to send a change event whenever a business system is changed.

**About this task**

By default, TADDM does not indicate a business system as changed if one of the computers it depends on has changed.

**Procedure**

To enable the sending of change events for business systems, complete the following steps:

1. Open `$COLLATION_HOME/etc/propagationserver.xml` in an appropriate editor.
2. In the Computer System section, for the application and business system relationship elements, set the value of the `enabled` attribute to `true`. For example:

   ```xml
   <relationship enabled="true" source="sys.ComputerSystem" attribute="groups"
   target="app.Application" targetAttribute="true"
   collectionType="app.FunctionalGroup" radius="1"/>

   <relationship enabled="true" source="sys.ComputerSystem" attribute="components"
   target="sys.BusinessSystem" targetAttribute="true"/>
   ```

3. Restart TADDM.
4. Create a listener for the business system in the change event configuration `$COLLATION_HOME/etc/EventConfig.xml`. In the following example, the event recipient is `mycompany-itm`, and the business system name is `MyBiz`.

   ```xml
   <listener object="ITSystem" enabled="true">
   <alert recipient="mycompany-itm"/>
   <attribute name="name" operator="equals">
   <value>MyBiz</value>
   </attribute>
   </listener>
   ```
Scheduling jobs with IBM Tivoli Workload Scheduler

You can use the IBM Tivoli Workload Scheduler to schedule jobs in TADDM. The IBM Tivoli Workload Scheduler is a software automation tool that provides the backbone for automated workload management and monitoring.

Use IBM Tivoli Workload Scheduler 8.5.1 or later. You must install the Master Domain Manager and Fault Tolerant Agent on the TADDM server. For information about how to install and configure Tivoli Workload Scheduler, see [http://www-01.ibm.com/support/knowledgecenter/SSGSPN_8.5.1.1/com.ibm.tivoli.itws.doc_8.5.1.1/ic-homepage.html?lang=en](http://www-01.ibm.com/support/knowledgecenter/SSGSPN_8.5.1.1/com.ibm.tivoli.itws.doc_8.5.1.1/ic-homepage.html?lang=en) Scheduling objects are managed with the composer command-line program and are stored in the Tivoli Workload Scheduler.

The Tivoli Workload Scheduler jobs use the `invokejob.sh` script to run the required operation. The `invokejob.sh` script is provided by TADDM installation.

The following parameters are common to all uses of the script:

**Required:** `-u user`
This value specifies the user running the API command.

**Required:** `-p password`
This value specifies the password that authenticates the user.

**Required:** `--profile profile`
This value defines the discovery profile.

**Optional:** `-H host`
This value specifies the TADDM server host name. The default name is `localhost`. If you use the `-T` parameter, you must also specify the `-H` parameter.

**Optional:** `-P port`
This value specifies the TADDM server port. The default value is 9433.

**Optional:** `-v version`
This value specifies the version name or number. The default value is 0.

**Optional:** `-t timeout`
This value specifies the amount of time before the job is automatically interrupted.

**Optional:** `-T | --truststorefile truststore`
This value specifies the location of the truststore file, `jssecacerts.cert`, with a certificate for connection to the TADDM server. This parameter is required for secure connection to TADDM. If you use this parameter, you must also specify the `-H` parameter.

To schedule a job, complete the following steps:

1. From the Tivoli Workload Scheduler, enter the TADDM job definition file in an edit file. The following example shows a template job definition:

```bash
WORKSTATION_ID#TADDM_JOB
SCRIPTNAME "/opt/IBM/taddm/dist/bin/invokejob.sh -u ^TADDM_USERNAME^ -p ^TADDM_PASSWORD^ command [parameters]"
STREAMLOGON taddmuser
TASKTYPE UNIX
RECOVERY STOP

^TADDM_USERNAME^ and ^TADDM_PASSWORD^ are variables that must be defined in Tivoli Workload Scheduler. These variables are mapped to values..."
that are stored in the database. For security reasons, use variables, especially when coding passwords, to ensure that the values are not visible as open text.

2. Use the composer to add the edit file to the database.

3. Add the job to a job stream, and schedule the job stream to run. The IBM Tivoli Workload Scheduler agent starts and monitors the action of the invokejob.sh script.

**Scheduling a discovery job**

The following example runs a discovery on scope 127.0.0.1:

```bash
dist/bin/invokejob.sh -u USER -p PASSWORD --timeout 60000 discover start --profile "Level 3 Discovery" 127.0.0.1
```

The following example runs a discovery on the MyScopeSet scope set, which must already exist in the scope list:

```bash
dist/bin/invokejob.sh -u USER -p PASSWORD --timeout 60000 discover start --profile "Level 3 Discovery" MyScopeSet
```

In the preceding examples, the last parameter specifies the scope element or scope set to be included in the discovery run. The `profile` parameter is required. The `name` parameter, which is the name of the discovery run, is optional.

The following command is an example of how to stop a currently running discovery:

```bash
dist/bin/invokejob.sh -u USER -p PASSWORD --timeout 60000 discover stop
```

The `discover stop` command does not take any additional arguments.

**Scheduling a domain synchronization job**

The following example shows the command-line syntax and options for the TADDM script invokejob.sh to run a domain synchronization in a synchronization server deployment:

```bash
dist/bin/invokejob.sh -u USER -p PASSWORD --timeout 60000 sync start TestDomain
```

Both the `sync start` and `sync stop` commands require one argument, the name of the domain for which to start or stop the synchronization job.

**Integrating TADDM with IBM Tivoli Business Service Manager**

Depending on the specific tasks that you must do in your IT environment, you can use the integration capabilities that are available between TADDM and IBM Tivoli Business Service Manager. To use these capabilities, you must have IBM Tivoli Business Service Manager 4.2.1 Interim Fix 3, but no additional configuration of TADDM is required.

**Updating the lifecycle state for business applications**

You can use the lifecycle state to filter objects for synchronization into IBM Tivoli Business Service Manager from TADDM. You can use the `BusinessServiceLifecycle` program to list information about a business application or to set the lifecycle state of a business application.

The IBM Tivoli Business Service Manager IT systems application includes only business applications. For this reason, the `BusinessServiceLifecycle` program supports only business applications.
The **BusinessServiceLifecycle** program is in the following location:

- For Linux and UNIX operating systems, the `BusinessServiceLifecycle` script is in the `$COLLATION_HOME/bin` directory.
- For Windows operating systems, the `BusinessServiceLifecycle.bat` batch file is in the `%COLLATION_HOME%\bin` folder.

Use the **BusinessServiceLifecycle** program with the following command-line options:

```
BusinessServiceLifecycle -u TADDM_username -p TADDM_password -l | -s guid state
```

Use the `-l` option to list business application lifecycle information, or use the `-s` option, along with a guid parameter and a state code parameter, to set a lifecycle state. You cannot use the `-l` option and the `-s` option at the same time.

The following table lists the valid state codes:

*Table 50. State codes*

<table>
<thead>
<tr>
<th>Code</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Unknown</td>
</tr>
<tr>
<td>1</td>
<td>Other</td>
</tr>
<tr>
<td>2</td>
<td>Ordered</td>
</tr>
<tr>
<td>3</td>
<td>Received</td>
</tr>
<tr>
<td>4</td>
<td>In Test</td>
</tr>
<tr>
<td>5</td>
<td>Tested</td>
</tr>
<tr>
<td>6</td>
<td>Installed</td>
</tr>
<tr>
<td>7</td>
<td>Enabled</td>
</tr>
<tr>
<td>8</td>
<td>Disabled</td>
</tr>
<tr>
<td>9</td>
<td>Maintenance</td>
</tr>
<tr>
<td>10</td>
<td>Retired</td>
</tr>
<tr>
<td>11</td>
<td>Archived</td>
</tr>
<tr>
<td>12</td>
<td>Accepted</td>
</tr>
<tr>
<td>13</td>
<td>Build</td>
</tr>
<tr>
<td>14</td>
<td>Development</td>
</tr>
<tr>
<td>15</td>
<td>Draft</td>
</tr>
<tr>
<td>16</td>
<td>Inventory</td>
</tr>
<tr>
<td>17</td>
<td>Offline</td>
</tr>
<tr>
<td>18</td>
<td>Postproduction</td>
</tr>
<tr>
<td>19</td>
<td>Production</td>
</tr>
<tr>
<td>20</td>
<td>Production Ready</td>
</tr>
<tr>
<td>21</td>
<td>Sunset</td>
</tr>
<tr>
<td>22</td>
<td>Validate</td>
</tr>
</tbody>
</table>

**Integrating TADDM with Jazz for Service Management**

TADDM supports integration with Open Services for Lifecycle Collaboration (OSLC) platforms. OSLC, when used with TADDM, enables you to get discovery
data presented in a the form standard resource definitions. Jazz for Service Management platform is an IBM integration tool that is based on OSLC open community specifications.

Jazz for Service Management provides a single point of configuration and administration of, but not limited to, all Tivoli products. Jazz for Service Management shows end-to-end view of IT resource, application, and business relationships.

**TADDM OSLC REST Communication**

TADDM Representational State Transfer (REST) Service provides OSLC integration across a number of OSLC REST feeds. The service specifies the media types that are returned when it is run and describes the security aspects that are connected with the service.

Common Resource Type Vocabulary (CRTV) is an IBM and OSLC community-defined data model that TADDM supports, along with the Tivoli Common Data Model (CDM). TADDM support for OSLC makes CDM discovery data available in the form of CRTV-defined resources.

**OSLC REST interface:**

A REST interface is available in TADDM for Open Services Lifecycle Collaboration (OSLC). You can use the OSLC REST interface to get information about registered configuration items (CIs), their attributes, and change history.

You can get information about CI attributes only if the attributes are supported by the Common Resource Type Vocabulary (CRTV) or TADDM vocabulary.

Each valid request must have a GUID that identifies the concrete CI.

There are two service types:

**Configuration service**

This service provides an interface to retrieve extended attributes for a CRTV resource.

**Change history service**

This service provides an interface to retrieve change history for a specified period of time, for a CRTV resource.

For each service, you can view the following three types of content:

- RDF representation
- OSLC compact view
- HTML preview

The following URL is the base address:

http[s]://taddm_host:port/cdm/oslc/provider_name/ci_guid

where

- *port* is the port, on which the Tomcat server (TADDM 7.3.0) or WAS Liberty Profile server (TADDM 7.3.0.1, and later) listens. The default value is 9430.
- *provider_name* is one of the following two values, depending on which service you want to use:
  - configuration
• changehistory

• ci_guid is the ID of the CI in TADDM

To view the HTML preview for a CI, use the following URL:
• http[s]://taddm_host:port/cdm/oslc/provider_name/ci_guid/preview

The OSLC REST interface accepts only HTTP-GET requests. You can use the HTTP accept header to specify the returned content type.

To view the OSLC compact view for the given CI, specify the following accept header:
application/x-oslc-compact+xml

You view the RDF representation of the given CI, specify the following accept header:
application/rdf+xml

This is the default behavior if no value is provided for the accept header.

**OSLC compact view:**

The OSLC compact view is an XML representation of a target resource.

The OSLC compact view is a preview provided by the OSLC REST interface. To view a preview of a target resource, a representation of the resources, as defined in the OSLC specification, must be supplied by the provider.

You can get this representation of the resource by using a HTTP GET request to the URI of the target resource along with the application/x-oslc-compact+xml access header.

If the provider supports the preview mechanism, it responds with a compact representation that includes information that the consumer can use to display links and a preview of the target resource.

**Jazz for Service Management HTML preview:**

Jazz for Service Management Registry Services provides an HTML user interface to deliver information about registered elements from connected external systems.

All elements that have data that are provided by TADDM have a HTML preview that provides a fast overview of selected element data directly from TADDM server.

TADDM provides Jazz for Service Management with a feeding service at the following address:
http[s]://host_name:port/cdm/oslc/configuration/guid/preview

where host_name and port are the host name and the port number of the TADDM server and guid is the unique element identifier.

The URL shows a page with overview information about selected element. The page is displayed automatically in the Jazz for Service Management user interface.
The page content is similar to the General tab in the Inventory Summary Details view available in the TADDM Data Management Portal.

**Security:**

You can configure TADDM so that access to the feeds provided by the OSLC REST interface requires authentication.

To access the REST interface, you must authenticate using one of the following methods:

**Basic HTTP authentication**

Credentials must be placed in the authorization request header. The value of that header must adhere to Basic HTTP authentication rules.

**Single sign-on**

When using single sign-on, all requests submitted to the REST interface must carry a Lightweight Third-Party Authentication (LTPA) token. To verify the token, TADDM must be configured to use WebSphere Virtual Member Manager (VMM) as a user repository.

For more information about configuring VMM, see “Configuring the TADDM server to use WebSphere federated repositories” on page 25.

To provide requested feeds to be presented without authentication, the following property in the collation.properties file must be configured with a valid Registry Services URL:

\[com.ibm.cdb.topobuilder.integration.oslc.frsurl\]

Then, a pre-configured user name and password are used if valid credentials are not included with the request.

The user name and password are taken from the web.xml deployment descriptor file of the Common Data Model web application. You can configure this customization using the following OSLCFilter init parameters:

**OSLC_LOGIN_OFF**

If this parameter is set to `true`, then the user name and password specified by the OSLC_USER and OSLC_PASSWORD parameters are used if incoming requests do not contain their own, valid credentials.

If this parameter is set to `false`, the incoming request must contain valid credentials.

The default value is `true`.

**OSLC_USER**

This parameter is set to the user name that is used if valid credentials are not included with the request. If required, you can change the user name used.

The default value is `administrator`.

**OSLC_PASSWORD**

This parameter is set to the password that is used if valid credentials are not included with the request. If you change the administrator’s password using the TADDM UI, you must update the password value set by this parameter.

The default value is `collation`.
Exporting data to Registry Services using OSLCAgent
You can use the OSLCAgent topology agent to export configuration item (CI) information to Registry Services.

OSLCAgent is an automated solution for exporting data from TADDM to Registry Services. The agent periodically performs the following tasks:

- Queries for objects that can be registered in Registry Services
- Translates them into RDF-formatted messages
- Posts them using HTTP

The OSLCAgent belongs to the Integration group. The time interval between runs is specified in the following entry in the collation.properties file:
com.ibm.cdb.topobuilder.groupinterval.integration

The OSLCAgent can act as a configuration provider and a change history provider. These two roles can be enabled separately. To enable the configuration provider role, set the following property to true:
com.ibm.cdb.topobuilder.integration.oslc.enable.configurationsp

To enable the change history provider role, set the following property to true:
com.ibm.cdb.topobuilder.integration.oslc.enable.changehistorysp

To configure the OSLCAgent to connect to Registry Services, you must specify the Registry Services address and access entry details.

Configure the Registry Services address in the following property:
com.ibm.cdb.topobuilder.integration.oslc.frsurl

Specify the Registry Services address in the following format:
protocol://fqdn_or_ip_or_hostname:port

For example, http://192.0.2.24:9081

**Note:** Fully Qualified Domain Name (FQDN) or Fully Qualified Hostname are preferred to IP address to provide consistency with other products and avoid integration issues. However, if all other products that are used with TADDM use IP address, you must specify IP address. If no products are used with TADDM, it is still preferred to use FQDN in case other products are added later.

Create an access list entry of Integration/Registry Service type. Specify the user name and password for Registry Services.

You can fine-tune how the OSLCAgent works using the following properties:

**com.ibm.cdb.topobuilder.integration.oslc.maxtimeperrun**
This property specifies the maximum time (in minutes) for which the OSLCAgent is allowed to run. This time can be exceeded for each provider by the length of time that is consumed by jobs that are submitted into the pool before the timeout event. If the property is not configured or is set to -1, the time that is allowed for a single run of the OSLCAgent is unlimited.

**com.ibm.cdb.topobuilder.integration.oslc.jobspoolsize**
This property specifies the maximum number of concurrent jobs that are allowed to run. Each job registers a single CI. If the property is not configured, the default value is 10.
com.ibm.cdb.topobuilder.integration.oslc.frshttp_timeout
This property specifies, in milliseconds, the timeout for HTTP connections. The default value is 5000.

com.ibm.cdb.topobuilder.integration.oslc.frssafarfastafter
This property specifies the number of consecutive HTTP connection timeouts after which an agent considers Registry Services unavailable and stops attempting to connect. The default value is 5.

com.ibm.cdb.topobuilder.integration.oslc.unregisterableIPs
This property specifies IP addresses that should not be registered as they are not public and might cause the overmerge of resource records. This property can contain regular expressions. The default value is 127.0.0.1,01111111000000000000000000000001,0:0:0:0:0:0:0:1,
0000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000.

com.ibm.cdb.topobuilder.integration.oslc.unregistrableFQDNs
This property specifies fully qualified domain names that should not be registered as they are not public and might cause the overmerge of resource records. This property can contain regular expressions. The default value is localhost.localdomain,localhost,loopback6(.?\u005c\w*)*.

com.ibm.cdb.topobuilder.integration.oslc.enablecrtvtype.CRTVTy
This property specifies that the OSLCAgent processes only CIs of a particular Common Resource Type Vocabulary (CRTV) type. If it is enabled, only CIs of that CRTV type are registered, updated, and unregistered.

For example, setting
com.ibm.cdb.topobuilder.integration.oslc.enablecrtvtype.ComputerSystem=true

specifies that only computer systems are processed.

The following CRTV types are allowed:
• ComputerSystem
• Database
• Fix Pack 2 SAPSystem
• ServiceInstance
• SoftwareModule
• SoftwareServer

com.ibm.cdb.topobuilder.integration.oslc.history.days_previous
This property specifies the number of days of change history information available using the launch in context (LIC) URL. The default value is 5.

You can use the days_previous property to control the amount of storage space and processing time required for managing change history information.

The LIC URL is listed in the OSLC Change History HTML preview.

If the value is greater than 0, then the days_previous parameter is applied to the LIC URL to limit the amount of change history displayed.

If the value is 0, or less than 0, then the LIC URL does not contain the days_previous parameter, and you can see the entire change history for that CI.
Command line interface for OSLCAgent

You can use the OSLCAgent command line interface (CLI) to manually export configuration item (CI) information to Registry Services.

For the OSLCAgent, you can pass a combination of commands and switches to the runtopobuild script or batch file. Each command and switch has a short, single-letter format, and a longer, more descriptive format. You can use any combination of command and switch formats.

The following commands are available:

- **-R | –refreshAll true|false**
  This command registers all eligible CIs, even if they have been registered already.

- **-r | -refreshGuid guid**
  This command registers the CI that has the specified GUID, even if it has been registered already.

- **-l | -refreshIgnored true|false**
  If a CI is discovered in a position that is not deep enough, it might not have correctly formed naming rules. By default, the OSLCAgent ignores such CIs. This command forces the OSLCAgent to process these CIs again.

To specify particular actions, you can pass a switch with any command. There are two types of switches available.

You can use the following switches to enable or disable the processing of particular CRTC types:

- **-c | --enableComputerSystem true|false**
- **-d | --enableDatabase true|false**
- **-i | --enableServiceInstance true|false**
- **-m | --enableSoftwareModule true|false**
- **-s | --enableSoftwareServer true|false**

For example, if you don’t want to re-register computer systems, use the -c false switches.

You can use the following switches to enable or disable the configuration and change history roles:

- **-h | --enableChangeHistoryProvider true|false**
- **-p | --enableConfigurationProvider true|false**

For example, if you don’t want to perform re-registration as a change history provider, use the -h false switches.

If you want default values to be used if you do not pass a command or a switch when running the runtopobuild script or batch file, configure the following properties in the collation.properties file:

- **com.ibm.cdb.topobuilder.integration.oslc.refreshAll=true|false**
- **com.ibm.cdb.topobuilder.integration.oslc.refreshGuid=guid**
- **com.ibm.cdb.topobuilder.integration.oslc.enablecrtvtype.crtv_type**
For a full list of the available parameters and switches, go to $COLLATION_HOME/support/bin and run the runtopobuild script or batch file with the -H switch. For example,

`./runtopobuild.sh -H`

**Registering configuration items with Registry Services**

This topic lists the configuration items (CIs) discovered by TADDM that are queried for registration with Registry Services, and lists the attributes that are set, along with detailed mapping information.

If a particular CI is not registered, every registering thread produces log information about why the CI is not registered. The list of unset naming rule attributes are listed in the log. To configure the correct logging level, set the following property value in the collation.properties file:

```
com.collation.log.level.vm.Topology=DEBUG
```

The following attributes are common for every CRTV type:

- **guid** — Set with the GUID value of the CI.
- **name** — Set with the value of the name, label, or displayName attribute.
- **description** — Set with the value of the description attribute.
- **lastDiscoveredTime** — Set with the value of the lastModifiedTime attribute.

**SoftwareServer**

The CRTV SoftwareServer type contains the following TADDM classes and attributes:

- **WebSphereServer**
  - host
  - node
  - node.cell
- **Db2Instance**
  - home
  - host
- **MQQueueManager**
  - displayName | label | name
- **AppServer**
  - displayName | label | name
  - host
- **CommunityServer**
  - displayName | label
- **SametimeServer**
  - displayName | label
- **MeetingServer**
  - displayName | label
- **SpecialityServer**
  - displayName | label | name
• AgentManager
  − displayName | label
• SharePointRole
  − displayName | label | name

TADDM attributes are mapped to CRTV attributes in the following way:

<table>
<thead>
<tr>
<th>TADDM attribute</th>
<th>CRTV attribute</th>
<th>Other information</th>
</tr>
</thead>
<tbody>
<tr>
<td>PrimarySAP</td>
<td>cvt:serverAccessPoint</td>
<td>The serviceAccessPoint resource is registered, along with the IpAddress resource to which it points, using cvt:ipAddress.</td>
</tr>
<tr>
<td>version</td>
<td>cvt:version</td>
<td></td>
</tr>
<tr>
<td>vendorName</td>
<td>cvt:manufacturer</td>
<td></td>
</tr>
<tr>
<td>host</td>
<td>cvt:runsOn</td>
<td>cvt:runsOn points to ComputerSystem</td>
</tr>
<tr>
<td>home</td>
<td>cvt:instancePath</td>
<td>For DatabaseServer and Db2Instance only.</td>
</tr>
<tr>
<td>dataPath</td>
<td>cvt:instancePath</td>
<td>For MQQueueManager only.</td>
</tr>
</tbody>
</table>

rdf:type is set to one of the following values:
• J2EEServer
• WebSphereServer
• IBMHTTPServer
• WebServer
• Db2Instance
• OracleInstance
• MQQueueManager
• WebServer
• DatabaseInstance
• CICSRegion

**ComputerSystem**

The CRTV ComputerSystem type contains the following TADDM classes and attributes:
• ComputerSystem
  One of the following combinations of attributes is set:
  − systemId &VMID
  − systemId
  − serialNumber & model & manufacturer &VMID
  − serialNumber & model & manufacturer
  − systemBoardUUID
  − ipInterfaces

TADDM attributes are mapped to CRTV attributes in the following way:
<table>
<thead>
<tr>
<th>TADDM attribute</th>
<th>CRTV attribute</th>
<th>Other information</th>
</tr>
</thead>
<tbody>
<tr>
<td>label or displayName</td>
<td>crtv:name</td>
<td></td>
</tr>
<tr>
<td>OSVersion or OSRunning</td>
<td>crtv:version</td>
<td></td>
</tr>
<tr>
<td>hostSystem</td>
<td>crtv:dependsOn</td>
<td></td>
</tr>
<tr>
<td>fqdn</td>
<td>crtv:fqdn</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>crtv:shortHostname</td>
<td>If name is set and it is a valid hostname. For SunSPARCComputerSystem only.</td>
</tr>
<tr>
<td>ipInterface</td>
<td>crtv:ipAddress</td>
<td>All FQDNs for those IP addresses are merged into crtv:fqdn.</td>
</tr>
</tbody>
</table>

crtv:type is set with one of the following values
- Generic
- SunFire
- SunSPARC
- SystemP
- Unitary
- Virtual
- WPAR

For a LinuxUnitaryComputerSystem, additional attributes are mapped in the following way:

<table>
<thead>
<tr>
<th>TADDM attribute</th>
<th>CRTV attribute</th>
<th>Other information</th>
</tr>
</thead>
<tbody>
<tr>
<td>manufacturer</td>
<td>crtv:manufacturer</td>
<td></td>
</tr>
<tr>
<td>model</td>
<td>crtv:model</td>
<td></td>
</tr>
<tr>
<td>serialNumber</td>
<td>crtv:serialNumber</td>
<td></td>
</tr>
<tr>
<td>VMID</td>
<td>crtv:vmid</td>
<td>If CPUPType and Model are set:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For intel, VMID is set to null and an attempt is made to set</td>
</tr>
<tr>
<td></td>
<td></td>
<td>crtv:systemBoardUUID with systemBoardUUID or convertedUUID.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For power, CS is ignored if it has VMID set.</td>
</tr>
</tbody>
</table>

For a SunSPARCUnitaryComputerSystem, additional attributes are mapped in the following way:

<table>
<thead>
<tr>
<th>TADDM attribute</th>
<th>CRTV attribute</th>
<th>Other information</th>
</tr>
</thead>
<tbody>
<tr>
<td>systemId</td>
<td>crtv:hostid</td>
<td></td>
</tr>
<tr>
<td>VMID</td>
<td>crtv:vmid</td>
<td></td>
</tr>
</tbody>
</table>
For any other computer system, additional attributes are mapped in the following way:

<table>
<thead>
<tr>
<th>TADDM attribute</th>
<th>CRTV attribute</th>
<th>Other information</th>
</tr>
</thead>
<tbody>
<tr>
<td>manufacturer</td>
<td>crtv:manufacturer</td>
<td></td>
</tr>
<tr>
<td>model</td>
<td>crtv:model</td>
<td></td>
</tr>
<tr>
<td>serialNumber</td>
<td>crtv:serialNumber</td>
<td></td>
</tr>
<tr>
<td>VMID</td>
<td>crtv:VMID</td>
<td>If OSRunning is set to WindowsOperatingSystem, VMID is set to null. If OSRunning is set to HpUx, VMID, model, and serialNumber are set to null.</td>
</tr>
<tr>
<td>systemBoardUUID or convertedUUID</td>
<td>crtv:systemBoardUUID</td>
<td></td>
</tr>
<tr>
<td>worldwideName</td>
<td>crtv:hostid</td>
<td>For FCSwitch, TapeLibrary, and TapeMediaChanger only.</td>
</tr>
</tbody>
</table>

**Database**

The CRTV Database type contains the following TADDM classes and attributes:
- Db2Database
  - name | displayName
- IDSDatabase
  - name | displayName
- IMSDatabase
  - name | displayName
- OracleDatabase
  - name | displayName
- SqlServerDatabase
  - name | displayName
- SybaseDatabase
  - name | displayName
- DominoDatabase
  - name | displayName

TADDM attributes are mapped to CRTV attributes in the following way:

<table>
<thead>
<tr>
<th>TADDM attribute</th>
<th>CRTV attribute</th>
<th>Other information</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>crtv:name</td>
<td></td>
</tr>
<tr>
<td>fileName</td>
<td>crtv:name</td>
<td>For DominoDatabase only.</td>
</tr>
<tr>
<td>parent</td>
<td>crtv:dbInstance</td>
<td></td>
</tr>
</tbody>
</table>

**ServiceInstance**

Depending on whether the compatibility with earlier versions is enabled, the CRTV ServiceInstance type contains the following TADDM classes and attributes:
- When compatibility with earlier versions is enabled:
- BusinessSystem
  - name
- Application
  - name
- ServiceInstance
  - name
- ServiceInfrastructure
  - name
- SAPSystem
  - SAPSystemSID | systemHome

• When compatibility with earlier versions is disabled:
  - CustomCollection (with „BusinessApplication” type only)
    - collectionId

TADDM attributes are mapped to CRTV attributes in the following way:

<table>
<thead>
<tr>
<th>TADDM attribute</th>
<th>CRTV attribute</th>
<th>Other information</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>crtv:name</td>
<td></td>
</tr>
<tr>
<td>SAPSystemSID:systemHome</td>
<td>crtv:name</td>
<td>If neither name nor displayName are set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For SAPSystem only.</td>
</tr>
<tr>
<td>parentGUID or NULL</td>
<td>crtv:parentServiceInstance</td>
<td></td>
</tr>
<tr>
<td>collectionId</td>
<td>crtv:name</td>
<td></td>
</tr>
</tbody>
</table>

**SoftwareModule**

The CRTV SoftwareModule type contains the following TADDM classes and attributes:

• SoftwareModule
  - fileName
  - name
  - parent.name
• MQQueue
  - name
  - queueManager

TADDM attributes are mapped to CRTV attributes in the following way:

<table>
<thead>
<tr>
<th>TADDM attribute</th>
<th>CRTV attribute</th>
<th>Other information</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent</td>
<td>deployedTo</td>
<td></td>
</tr>
<tr>
<td>fileName</td>
<td>crtv:fileName</td>
<td></td>
</tr>
</tbody>
</table>

rdf:type is set to one of the following values

• J2EEApplication
• MQQueue
 Troubleshooting OSLC

This topic describes common problems that occur with OSLC and presents solutions for those problems.

The TADDM URL configured does not include a port number

Problem

The TADDM URL property that is configured in the collation.properties file, taddmURL, must include a port number.

If the property is not configured with a port number, you must update the TADDM URL to include a port number, clear the information about Registry Services, or specific providers, and clear TADDM timestamps.

Solution

To update the TADDM URL to include a port number, complete the following steps:

1. In the collation.properties file, set the taddmURL property in the following way:
   
   taddmURL=http://server.domain:port

2. On the computer with Registry Services, complete the following steps:
   a. Go to /opt/IBM/JazzSM/registry/etc.
   b. In the CLI.properties file, configure credentials for the following properties:
      - ds.jdbc.user
      - ds.jdbc.password
      - appserver.user
      - appserver.password
   c. Go to /opt/IBM/WebSphere/AppServer/bin.
   d. Run the stopServer.sh script to stop the WebSphere Application Server.
      
      .stopServer.sh server_name -user user_name -p password

      for example,
      
      .stopServer.sh server1 -user wasadmin -p passw0rd
   e. Go to /opt/IBM/JazzSM/registry/bin.
   f. Run the frs.sh script with the appropriate parameters:
      
      ./frs.sh uninstall -type db -properties ../etc/CLI.properties
   g. Check that the database has been dropped. If not, run the following commands:
      
      db2 drop db db_name
      
      db2 create db db_name

      where db_name is the name of the Registry Services database.
   h. Go to /opt/IBM/JazzSM/registry/bin.
   i. Run the frs.sh script with the appropriate parameters:
      
      ./frs.sh install -type db -properties ../etc/CLI.properties
   j. Go to /opt/IBM/WebSphere/AppServer/bin.
   k. Run the startServer.sh script to start the WebSphere Application Server.
      
      ./startServer.sh server_name -user user_name -p password
for example,

```
./startServer.sh server1 -user wasadmin -p passw0rd
```

l. Run the frs.sh script with the appropriate parameters:

```
./frs.sh uninstall -type container -properties ../etc/CLI.properties
```

m. Run the frs.sh script with the appropriate parameters:

```
./frs.sh install -type container -properties ../etc/CLI.properties
```

You might be able to remove an item from Registry Services for a specific provider by using the following command:

```
./frs.sh deleteProvider -providerUrl url -properties cli.properties
```

3. On the computer with the TADDM database, complete the following steps:

a. Go to $COLLATION_HOME/support/bin.

b. Run the runtopobuild script or batch file with the appropriate parameters, for example:

```
./runtopobuild.sh -a OSLCAgent -R
```

### Tivoli Directory Integrator

When you purchase the IBM Tivoli Application Dependency Discovery Manager (TADDM), you also receive the Tivoli Directory Integrator, which enables you to integrate TADDM with other data sources.

**Tivoli Directory Integrator documentation at Knowledge Center**


**TADDM integration scenarios at Tivoli Application Dependency Discovery Manager Wiki**


### Business entities compatibility with earlier versions

A new function was introduced to enable integration between TADDM and products that read data from TADDM by using DataApi or directly from TADDM database by using SQL. The examples of such products are IBM Tivoli Business Service Manager (TBSM), IBM SmartCloud Control Desk (SCCD), Tivoli Directory Integrator (TDI). The current Business Application data model relies on CustomCollection interface, which has nothing in common with former Application and ITSystem interfaces. The new function enables integration with other products without introducing any modifications to those systems.

In the future versions of TBSM and SCCD, the Business Application model will be introduced with new features. The aim is to generate previous business entities, which are copies of custom collection instances.

The new function, which provides compatibility with earlier versions, consists of the following features.

**Additional step while running BizAppsAgent**

The additional step generates business entities (services, application, collection) compatible with earlier versions for each custom collection that is generated by the agent.
To activate this step, a new property was added to the collation.properties file, com.ibm.cdb.serviceinfrastructure.earlier.ver.compatibility. The default value of this property is TRUE for the upgrade scenario and FALSE for the fresh installation scenario.

**OSLC support**

OSLC agent was modified and is able to register either old business entities or new custom collections. When compatibility flag is set to TRUE, old business entities are registered. Otherwise, custom collections are used to produce a content for Jazz for Service Management (JazzSM).

In the future, a full reload of business entities will be required when integrating product starts to load data using new model objects (custom collections and nodes). Old business applications (applications) and new business applications (custom collections) cannot have the same GUID. To avoid duplicates, before loading new custom collections, users will have to remove old business applications.

**Creating functional groups**

New business applications, unlike old business applications, do not have functional groups. However, a new tiers functionality was introduced to serve similar purposes. For each unique tier, to ensure compatibility with earlier versions, a functional group with a name corresponding to the tier name is created.

For more information, see the Business application tiers topic in the TADDM User’s Guide.

**Integrating BigFix**

**Purpose**

TADDM uses Anchors and Gateways to discover machines/applications/network that are behind firewall. The use of Anchors/Gateways can be avoided by using IBM Netcool monitoring tools (ITM) currently. Alternatively, TADDM integration with BigFix architecture can be utilized to avoid using anchors and gateways. BigFix architecture consists of BigFix server (BES Server) and multiple BigFix endpoints (BES clients) where the BES clients are the secure machines which are accessible through BES server. BigFix infrastructure can be reused/utilized for running the TADDM script packages on the BES clients via BES server automatically.

**Key Advantages of this Integration for TADDM Administrators are:**

1. Ability to discover firewalled zones without anchors.
2. Ability to reuse the BigFix Architecture (e.g. open secure ports) to access endpoints and saving set up time to discover the same targets using the standard TADDM method.
3. Align with strategic direction for TADDM script based sensors.
4. Minimal intervention required from TADDM administrator.
5. Provides alternative method of discovery for firewalled zone machines without using TADDM-ITM integration.

**Reference**

Documentation for TADDM
The following table shows the supported versions of the products that TADDM can be integrated with.

For more information about products that you integrate with TADDM, refer to their documentation:

- **TADDM 7.3 and Sensors Knowledge Center (Official Documentation)**
  

- **TADDM 7.3 sensors and supported target systems**
  
  https://www.ibm.com/developerworks/community/groups/service/html/communityview?communityUuid=7d5ebce8-2dd8-449c-a58e-4676134e3eb8#fullpageWidgetId=Wea1cb2531f10_4c9d9d7_6ab0334cb21f&file=e70bf323-31f1-45ba-8992-4cb491feab4a

- **TADDM configuring Asynchronous Script Discovery (ASD)**
  

- **IBM BigFix Configuration Guide**
  

- **TADDM support website**
  

- **TADDM Wiki**
  
  https://www.ibm.com/developerworks/mydeveloperworks/wikis/home?lang=en#/wiki/Tivoli%20Application%20Dependency%20Discovery%20Manager/page/Home This is a good source of up-to-date information and best practices for TADDM. Bookmark this page and get comfortable with it.

- **TADDM Forum**
  

- **Request for Enhancement Community**
  
  http://www.ibm.com/developerworks/rfe/?BRAND_ID=90 This community should be used for requesting enhancements to the product directly to IBM developers.

**Solution Architecture**

TADDM-BigFix integration is based on enhancing and automating the current behaviour of Asynchronous Script Discovery (ASD) which requires manual intervention from TADDM administrator. This integration utilizes the connectivity that BigFix infrastructure provides to the firewall zone machines for running the discovery through the TADDM script packages.

In ASD, TADDM administrator must perform the following steps manually:

1. Executes a script at TADDM server to create a discovery package that includes all the sensors to be run on target.
2. Transfer this package to the target system.
3. Execute the discovery package at target system.
4. Transfer the result file generated at target system back to TADDM server.

With the current solution, the manual steps have been automated and thus the solution is also termed as Automated Asynchronous Script Discovery (AASD). TADDM administrator just has to execute a script to start the discovery at TADDM server and rest steps are performed automatically.
Steps in BigFix Discovery:

Step details of Big Fix discovery

Step 1: Big Fix Integration Script

A script “runBigFixDiscovery.sh” has been developed that will kick start the AutoASD discovery (AASD) from TADDM discovery server. The script can be executed on demand. This script takes Discovery Scope and Discovery Profile name as inputs (besides BigFix access credentials) and supports below modes:

• DISCOVER mode – for initiating the BigFix discovery
• POLL mode – for polling the results of the BigFix discovery
• CLEANUP mode – for on-demand purging the discovery result packages from BES Root Server
• REDISCOVER mode – to rerun the previous discovery again

a) Create AutoASD Sensor Package

• Specified Discovery Profile is used to fetch list of sensors and only valid scripted sensors subset is considered for AASD scripts package creation. This feature supports only a subset of the script sensors, which are supported in standard ASD mode.
• Other non-scripted sensors in the discovery profile are ignored
• AASD package is OS agnostic - as a result, some of the sensors may fail on BigFix endpoints, if not present
• Generated AASD script package is uploaded to BigFix root server using /api/upload REST API

b) Create a BigFix Task

• Specified Discovery Scope is used to create the “Relevance” XML, which BigFix understands
• BigFix Task XML is generated with “Relevance” and dummy “ActionScript”
• Generate Task title based on current date time
• Use /api/tasks/custom/TADDM REST API to create a BigFix task under custom site namely “TADDM” at BigFix server

c) Start the BigFix Task

• Use <SourcedFixletAction> to start the action execution for above created BigFix task
• /api/actions BigFix REST API is used to start the “ActionScript” execution on target endpoint

Step 2: Run Script

As part of “ActionScript” execution, TADDM AASD package will be uncompressed, and contained sensor scripts (based on discovery profile) will be executed at the BigFix endpoints

Step 3: Collect Zip

• At end of “ActionScript” execution, result package generated at BES client from execution of TADDM AASD package will be copied over to BES Root Server
Step 4: Import Results back to TADDM

- TADDM will continuously poll BigFix Server DB to check for the results file uploaded to BES server
- If the DB shows that new result files are present, TADDM will make HTTP request to fetch the encrypted result files, decrypt them and save them
- TADDM will then process these result files based on the scope and profile configured and stores the discovered objects in the database

TADDM Bigfix Integration

BigFix enhanced TADDM discovery solution focuses on end-to-end functional execution of discovery for Windows, Linux, AIX and Solaris OS and associated sensors (for multiple endpoints) and with SSL communication supported between TADDM and BigFix via BigFix REST APIs. Discovery can be initiated at TADDM Discovery server and the discovery results should be retrieved automatically and visible on the TADDM GUI.

Assumptions:

Following assumptions have been considered into account for running the Discovery:

1. BigFix server and clients are having version mentioned in section 2.1.
2. BigFix clients have appropriate rights to execute the discovery task or action script uploaded by the BigFix server.
3. BigFix SQL Database User configured in collation.properties must have READ access to BFEnterprise Database.
4. Sensor Script Packages executed via BigFix Agents will need write access to configured TEMP directory (e.g., “C:\Windows\Temp”). Temp directory can be configured in collation.properties and assumed to have directory path without spaces.
5. Cleanup of script request package is not handled on TADDM discovery server, and its assumed that Administrator will manage them.
6. Since TADDM-BigFix Integration is based on currently existing ASD framework in TADDM, hence performance characteristics of this integration will be based on ASD framework benchmarks.
7. Only "taddmusr" can be used for running the BigFix discovery script on TADDM discovery server, and root user will not be permitted.
8. Bigfix Root Server cleanup will be invoked on each TADDM Startup and also periodically as per configured duration com.collation.bigfix.root.cleanup.interval = Default 1 day. This will delete result files older than configured time (com.collation.bigfix.root.cleanup.days = Default 5 days).
9. TADDM server cleanup of all result files created/copied on TADDM server during discovery and having name containing "taddmasd" and ending with "DONE" will be handled. (At least one threshold specified in Configurable Properties > Cleanup, should be configured to enable cleanup on TADDM server).
10. Endpoint Cleanup Discovery Endpoint cleanup is enabled by default and can be controlled by configuring below property setting:

- "com.collation.bigfix.endpoint.cleanup" set to value “N” will disable cleanup on discovery endpoint

11. High Availability or Disaster Recovery scenario is not supported. A Discovery Server only connects with the single BigFix Server configured in its collation.properties to launch this enhanced discovery process.

**Prerequisite:**

Before starting the discovery from the TADDM server, following pre-requisite must be met:

1. Select ASDSensor, ASDPingSensor, Generic Server Sensor and uncheck PingSensor, PortSensor and SessionSensor mandatorily during Discovery Profile creation.

2. All the configuration steps specified in section "Configurable Properties used in the integration" have been completed.

3. Bigfix Action script have used native powershell command to unzip the request package in Windows Endpoint and native tar command in Linux.

**Note:** Based on specific requirement, ActionScript customization can be done. This customization is supported by updating customer modifiable file ActionScript_Pre_Post.txt located in `$COLLATION_HOME/etc/` folder. E.g., to enable download and use of custom unzip software (executable distribution placed at BigFix root server). Sample Example Snippet is provided below:

```bash
%WIN_PRE_START%
if (not exists file "C:\Windows\System32\unzip.exe")
prefetch unzip.exe sha1:e1652b058195db3f4575947ab430652ae04a50b8

// Make sure that environment is set appropriately and "unzip" utility is available in the windows PATH
copy "__Download\unzip.exe" "C:\Windows\System32\unzip.exe"
endif
%WIN_PRE_END%

%WIN_POST_START%
%WIN_POST_END%

%LIN_PRE_START%
%LIN_PRE_END%
...```

4. User running the discovery shall have Read or Write permissions for the result folder.

5. Sufficient disk space, processing capacity and memory should be available to cater to the request and result packages that gets processed on TADDM server, BigFix Root Server and Discovery targets.

6. Bigfix Agent (Endpoint computer) should be configured with sufficient value for "_BESClient_ArchiveManager_MaxArchiveSize" setting to enable successful result uploads to BigFix root server.
7. Correct Scope and Profile must be set (please refer to section 4.1 for setting Scope and Profile).

8. Site name "TADDM" must be configured and existing on BigFix server.

9. All the pre-requisites required for standard ASD sensor scripts are applicable in case of BigFix discovery as well.
   - Powershell executable should be installed and configured properly in case of discovery involving Windows2003 Endpoint

10. For Rediscovery or repeated discovery with interval, the scope and profile used to initiate Big Fix discovery should not be deleted in between. In case of such deletion user may have to delete their unprocessed files which arrived on TADDM server from Big Fix Server.

**Limitations:**

Following Limitation are associated with current release:

1. Any discovery targets specified during discovery and not reachable from BigFix root server will not be visible in Discovery History.

2. PingSensor, PortSensor, and SessionSensor gets enabled automatically when other sensors are chosen and enabled, and these must be manually disabled during the discovery profile creation.

3. Request Package cleanup on TADDM Discovery server or BigFix Root server is not supported by design. Consideration is that it might be used during Rediscovery (triggered by Rediscover mode).

4. Rediscovery is supported only from same TADDM discovery server where original discovery was initiated.

**Configuration:**

Follow the steps mentioned in this section to set the desired configuration.

*Basic Configurations for Bigfix Discovery:*

1. Set the following mandatory properties in the $COLLATION_HOME/etc/collation.properties:

   **BigFix Integration Feature Settings**
   - com.collation.bigfix.enabled=true

   **BigFix Server Settings**
   - com.collation.bigfix.host=<ip or fqdn of the BigFix server>
   - com.collation.bigfix.port=<port_no>
   - com.collation.bigfix.uid=<userid for BigFix server console access>
   - com.collation.bigfix.pwd=<password to access BigFix server console>

   **BigFix DB Settings**
   - com.collation.bigfix.db.type=<MSSQL/DB2>
   - com.collation.bigfix.db.host=<IP or FQDN of the BigFix server DB>
• com.collation.bigfix.db.port=<Port at which TADDM will connect with BigFix DB>
• com.collation.bigfix.db.dbname=<BigFix DB name>
• com.collation.bigfix.db.domain=<User Domain> Optional Parameter - only required when Windows Based authentication is configured for BigFix DB>
• com.collation.bigfix.db.uid=<User id to access the BigFix DB>
• com.collation.bigfix.db.pwd=<Password to access the BigFix DB>

Result Processing Thread Settings
• com.ibm.cdb.discover.asd.ProcessUnreachableIPs=true
• com.ibm.cdb.discover.asd.autodiscovery.enabled=true

2. Set the following properties in the $COLLATION_HOME/etc/collation.properties, only in case SSL configuration is enabled on BigFix Server:

Bigfix Certificate
• com.collation.bigfix.certificate.type=<PKCS12/JKS>
• com.collation.bigfix.certificate.file=<Full path to the certificate file>
• com.collation.bigfix.certificate.pwd=<Password to use Certificate>

Note: There are more properties apart from these mandatory properties that can be configured. Please refer 'Configurable Properties' for complete list of properties and their details.

Note: Certificate generated with Blank password is supported.

3. Execute script “encryptprops.sh” to encrypt the properties (refer 'Example of scripts execution' to check format for running this script). Otherwise, Discovery scripts (runBigFixDiscovery.sh/.bat) will fail with missing or Invalid arguments error (refer 'Error Codes and Description' for error code details), since non-encrypted passwords are not accepted.

4. Create $COLLATION_HOME/var/asdd folder to store the result files at TADDM. If var or asdd folder is not to be used then property “com.ibm.cdb.discover.asd.AsyncDiscoveryResultsDirectory” must be set with specific folder where administrator wants the result files to be downloaded.

5. Restart TADDM.

6. Create a Discovery Profile with the required sensors listed in section 2.1. The profile should have Mandatory sensors, in addition to other sensors that are being discovered.

7. Create a discovery scope with BigFix target endpoint(s) where discovery needs to be run.

Other Configuration:

On BigFix server a site name “TADDM” is to be created.

1. Open the “BigFix console” -> goto “Tools” tab -> select “Create Custom Site..” -> provide site name as “TADDM”.

246 Application Dependency Discovery Manager: Administering
2. Click on “TADDM” -> select “Computer Subscription” tab -> subscribe computers based on the requirement (it should include all computers that BigFix server wants to be connected to through TADDM).

Customization supported:

Customization supported in TADDM BigFix integration.

ActionScript customization

- ActionScripts customization is supported for discovery
- User can modify the Pre/Post section as per OS in “ActionScript_Pre_Post.txt” file located in $COLLATION_HOME/etc/ folder, if user wants some specific actions be performed before/after the discovery
  - E.g. below snippet enables deployment of custom unzip software (executable distro placed at BigFix root server), instead of requiring them to be pre-installed on each endpoint
  ```
  %WIN_PRE_START%
  if (not exists file "C:\Windows\System32\unzip.exe")
  prefetchunzip.exe sha1:e1652b058195db3f57754b7ab430652ae04a50b8 size:167936
  //Make sue the environment is set appropriatly and "unzip" utility is available in the windows PATH
  copy"_Download\unzip.exe""C:\Windows\System32\unzip.exe"
  endif
  %WIN_PRE_END%
  %WIN_POST_START%
  %WIN_POST_END%
  %LIN_PRE_START%
  %LIN_PRE_END%
  ...
  
  OS customization
  
  Default ActionScripts is created for all 4 supported platforms (Linux, Windows, AIX and SunOS)
  
  If user wants to limit the ActionScript logic to particular set of platforms, below collation property can be utilized:
  - com.collation.bigfix.action.enable.os
  
  Example: com.collation.bigfix.action.enable.os=WIN,LIN,SUN,AIX

Relevance customization

- TADDM discovery scopes are based on IP addresses/ranges/network masks. With this customization, user can expand discovery scopes which are not necessarily IP based, but can be dynamic based on specified relevance/criterion.
- E.g., If user want to run discovery on all Windows computers, and want to skip the specified TADDM “Discovery scope”, below collation property shall be used:
  - com.collation.bigfix.relevance.appendscope=false
  - com.collation.bigfix.relevance=Windows operating system
  - com.collation.bigfix.relevance.appendscope
    - If true, then Custom Relevance query will be used, in addition to the specified discovery scope

...
– If false, then only Custom Relevance query will be used, instead of specified discovery scope

*com.collation.bigfix.relevance*

– Valid Relevance query for identifying a set of endpoints for given discovery

**Note:**

*Custom relevance is supported at TADDM discovery server level*

*Validation of Custom Relevance is not supported, and will be transparently used*

*Log file and troubleshooting tips:*

In case of any failure encountered during discovery, following things can be checked. Confirm that all prerequisites are satisfied and followed:

**TADDM Discovery Server**

*To check logs for Bigfix Discovery script execution*

– `$COLLATION_HOME/log/BigFixDiscovery.log`

*To check the logs for Result File is reaching TADDM Server or not:*

– `$COLLATION_HOME/log/services/ApiServer.log` (search for keywords 'BigfixDiscoveryServerController' and 'AASDiscoveryServerController')

**BigFix Root Server**

To check the status of the discovery and action execution using IBM BigFix Console on BigFix root server

1. Open IBM Bigfix Console.
2. Select Site (Custom->TADDM) -> Fixlets and Tasks.
3. Select Task (given during script execution).
4. Check Details and Action History.
6. Check the status and for line by line execution double-click it.
7. Click OK to return to the Discovery Profiles window.

**BigFix Agent/Discovery Target**

*Check that result file is present at %wintemp%/taddm7.3.0.4/asd folder (only when com.collation.bigfix.endpoint.cleanup property is set to "N")*

*allErrors.txt file (present in %wintemp%/taddm7.3.0.4/asd) can be referenced for any errors during sensors script execution*

**Running Bigfix Discovery:**

Running Bigfix Discovery

*Creating Scope:*

Open TADDM server GUI to create a discovery scope. Scope should include all the BigFix target end points. The target endpoints can be set as individuals host or by specifying domain/network range.
Creating Profile:

A Discovery profile should be created through TADDM server GUI. Profile should include the sensors for those applications that administrator wants TADDM to discover. Refer to section 2.1 for details about sensors that needs to be mandatorily included/excluded from the created discovery profile.

Running Script:

To run discovery, script “runBigFixDiscovery.sh” is to be executed from $COLLATION_HOME/bin. The script can be executed in 4 modes ‘discovery’, ‘poll’, ‘cleanup’ and ‘rediscovery’. In discovery mode, the discovery is started. In poll mode, current discovery status will be fetched. In cleanup mode, the result files cleanup will be triggered on BigFix root server and In rediscovery mode, the previously run Discovery can be executed again.

1. Discovery mode-

TADDM provides Jazz for Service Management with a feeding service at the following address:

```
./runBigFixDiscovery.sh -d -o <output dir> -s <scope> -p <profile>
```

Where,

- `-d` – for discovery mode
- `-o` - output directory where discovery package would be created
- `-s` – input scope, having BigFix endpoints target that are to be discovered.
- `-p` – input profile, having sensors to be run

Once, the command is executed in discovery mode, the discovery is started. This will show you the status of the steps performed and will give an id of “Action”. This Action is can be used in poll mode to check the status of the Action at each BigFix endpoint

Note:

- “ActionID” created for the discovery (shown in the console output, e.g. 2090 in below example) can be reused for polling purpose.
- Please keep the newly created bigfix task’s name (“TaskName” shown in the console output, e.g. 20180130125432) associated with given scope and profile, which can be reused for rediscovery.

2. Poll mode-

```
./runBigFixDiscovery.sh -p -r <repetition> -i <Action id>
```

Where,

- `-p` – for poll mode
- `-r` – no of times polling will be done to BigFix server
- `-i` – Action id obtained from discovery mode command.

3. Cleanup mode-

```
./runBigFixDiscovery.sh -c -d <No of Days>
```

Where,

- `-c` – for cleanup mode
- `-d` – Files older than specified number of days to be removed

4. Rediscovery mode-

```
./runBigFixDiscovery.sh -r --rediscover -i <TaskName>
```

Where,

- `-r` – for rediscover mode
- `-i` – TASK NAME corresponding to the previous discovery, that needs to be run again
**Note:** More detail about this command with all possible options is captured in Appendix B and an example for running the command is captured in Appendix C.

**Script Parameter options in different modes:**

`./runBigFixDiscovery.sh` (or.bat) is the TADDM tool to run a BigFix enhanced discovery or to query an existing discovery Action.

**Mode: DISCOVER**


where,

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ac, --actionConstraint &lt;arg&gt;</td>
<td>File specified with different action constraint related parameter. File should contain below parameter, with sample values given as example: com.collation.bigfix.action.constraint.starttime=10M com.collation.bigfix.action.constraint.endtime=3DT5H20M com.collation.bigfix.action.constraint.timerange.starttime=01:15:00 com.collation.bigfix.action.constraint.timerange.endtime=05:30:00 com.collation.bigfix.action.constraint.days=sat,sun This means a script can run on target only when these constraints are met which is starting from 10 minutes from now (10M) till 3 days 5 hours and 20 minutes (3DT5H20M), discovery can run only on Saturday and Sunday and between time 01:15:00 to 05:30:00 (24 hours format). This option will be helpful in both single discovery and in discovery started with interval option.</td>
</tr>
<tr>
<td>-c, --compressMethod &lt;arg&gt;</td>
<td>[Default : ZIP] Possible Values : [ZIP, TAR].</td>
</tr>
<tr>
<td>-freq, --frequency &lt;arg&gt;</td>
<td>[Default : 1] Number of times discovery needs to be run.</td>
</tr>
<tr>
<td>-h, --help</td>
<td>show help.</td>
</tr>
<tr>
<td>-intr, --interval &lt;arg&gt;</td>
<td>[Default : P1D] Time interval between rerun of discovery. Supported values : [PT15M, PT30M, PT1H, PT2H, PT4H, PT6H, PT8H, PT12H, P1D, P2D, P3D, P5D, P7D, P15D, P30D].</td>
</tr>
<tr>
<td>-o, --output &lt;arg&gt;</td>
<td>REQUIRED: Output Directory where the Bigfix Discovery Package will be generated.</td>
</tr>
<tr>
<td>-p, --profile &lt;arg&gt;</td>
<td>REQUIRED: Profile name will be used for Discovery Package creation to include sensors.</td>
</tr>
<tr>
<td>-s, --scope</td>
<td>REQUIRED: Scope/ScopeGroup name/s (Comma seperated. Wrap names in quotes that contain spaces).</td>
</tr>
</tbody>
</table>

**Mode: POLL**

usage: `bin/runBigFixDiscovery.sh -p/--poll [-h] -i <arg> [-r <arg>] [-t <arg>]`

where,
Table 52.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d, --detail &lt;arg&gt;</td>
<td>[Default : true] Polling result for each Endpoint</td>
</tr>
<tr>
<td>-h, --help</td>
<td>show help.</td>
</tr>
<tr>
<td>-r, --repeat &lt;arg&gt;</td>
<td>[Default : 1] Number of times to poll for Action Status</td>
</tr>
<tr>
<td>-i, --id &lt;arg&gt;</td>
<td>REQUIRED: Action ID to POLL</td>
</tr>
<tr>
<td>-t, --timeout &lt;arg&gt;</td>
<td>[Default : 1] Interval between consecutive POLL in seconds.</td>
</tr>
</tbody>
</table>

Mode: CLEANUP

usage: bin/runBigFixDiscovery.sh -c/--cleanup [-d <arg>] [-h]

where,

Table 53.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-h, --help</td>
<td>show help.</td>
</tr>
<tr>
<td>-d, --days &lt;arg&gt;</td>
<td>[Default : 5] Clean Result Files older than specified number of Days</td>
</tr>
</tbody>
</table>

Mode: REDISCOVER

usage: bin/runBigFixDiscovery.sh -r/--rediscover [-freq <arg>] [-h] [-intr <arg>]

where,

Table 54.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-freq, --frequency &lt;arg&gt;</td>
<td>[Default : 1] Number of times discovery needs to be run</td>
</tr>
<tr>
<td>-h, --help</td>
<td>show help.</td>
</tr>
<tr>
<td>-intr, --interval &lt;arg&gt;</td>
<td>[Default : P1D] Time interval between rerun of discovery. Supported values : [PT15M, PT30M, PT1H, PT2H, PT4H, PT6H, PT8H, PT12H, P1D, P2D, P3D, P5D, P7D, P15D, P30D].</td>
</tr>
</tbody>
</table>

Discovery Result Processing:

The poll mode of the “runBigFixDiscovery.sh” command gives the status of the Action executed at each BigFix endpoint. Based on the status, the result file will be created and processed.

1. Once, the action is completed successfully for the endpoint, a result file for that endpoint will be downloaded at the result folder configured (by default it is var/asdd. Check section 'Configurable Properties' for configuring the result folder).

2. Once the result files are processed successfully, the result can be available on the TADDM GUI in history tab.

3. The processed result data will be stored in the TADDM database and will be available on the Data Management Portal or PSS of TADDM.
Possible Failure Scenario:

In case of any failure encountered during discovery, poll, cleanup or rediscover mode, following things can be checked:

1. Confirm that all prerequisite mentioned in section 2.2 are followed:

2. Check the TADDM logs at path:
   - $COLLATION_HOME/log/BigFixDiscovery.log – for discovery and script execution logs
   - $COLLATION_HOME/log/services/ApiServer.log – for result fetching and parsing logs

3. Check the BigFix server logs and BigFix console for any failure status.

4. Bigfix ActionScript execution logs could be verified in case 'Action Polling' gets failure from 'Root Server' and 'Endpoint'.

Configurable Properties
Following are the configurable properties which can be configured in collation.properties

1. BigFix feature enabled

Table 55.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation. bigfix.enabled</td>
<td>true/false</td>
<td>If true, BigFix feature will be enabled. After setting this property to true, TADDM restart is needed.</td>
<td>Y</td>
</tr>
</tbody>
</table>

2. BigFix server

Table 56.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation. bigfix.host</td>
<td>IP or FQDN</td>
<td>IP or FQDN of the BigFix Server.</td>
<td>Y</td>
</tr>
<tr>
<td>com.collation. bigfix.port</td>
<td>&lt;port no&gt;</td>
<td>Port at which TADDM will send request to BigFix server.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Default=52311</td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.collation. bigfix.uid</td>
<td>&lt;userid&gt;</td>
<td>User id to access BigFix server console.</td>
<td>Y</td>
</tr>
<tr>
<td>com.collation. bigfix.pwd</td>
<td>&lt;password&gt;</td>
<td>Password to access BigFix server console. Will be stored in encrypted form.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.collation. bigfix.connectTo</td>
<td>&lt;time_period&gt;</td>
<td>TADDM will wait for this period in sec before HTTP/RestAPI connect timeout.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Default=20 sec</td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.collation. bigfix.responseTo</td>
<td>&lt;time_period&gt;</td>
<td>TADDM will wait for this period in sec before HTTP response timeout.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Default=20 sec</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 56. (continued)

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation.</td>
<td>&lt;site_type&gt;</td>
<td>Type of the Site at BigFix server with which TADDM will connect.</td>
<td>N</td>
</tr>
<tr>
<td>bigfix.site.type</td>
<td>Default=custom</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Visibility.Control.</td>
<td>&lt;site_name&gt;</td>
<td>Name of the Site at BigFix server with which TADDM will connect.</td>
<td>N</td>
</tr>
<tr>
<td>Automation</td>
<td>Default=TADDM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.collation.</td>
<td>&lt;Request Package</td>
<td>Maximum allowed Request Package generated by BigFix discovery script.</td>
<td>N</td>
</tr>
<tr>
<td>bigfix.</td>
<td>size&gt; Default=1024</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aasdpkgmaxsize</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3. Big Server Certificate

Table 57.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation.</td>
<td>&lt;PKCS12/JKS&gt;</td>
<td>Supported Client certificate Type.</td>
<td>N</td>
</tr>
<tr>
<td>bigfix.certificate.type</td>
<td>Default=JKS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.collation.</td>
<td>&lt;Path&gt;</td>
<td>Client Certificate File Location.</td>
<td>N</td>
</tr>
<tr>
<td>bigfix.certificate.file</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.collation.</td>
<td>&lt;Password&gt;</td>
<td>Password of Client Certificate</td>
<td>N</td>
</tr>
<tr>
<td>bigfix.certificate.pwd</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4. BigFix Server DB

Table 58.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation.</td>
<td>MSSQL or DB2</td>
<td>Type of the DB used by BigFix server.; MSSQL for Windows based BigFix server or DB2 for Linux based.</td>
<td>Y</td>
</tr>
<tr>
<td>bigfix.db.type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.collation.</td>
<td>IP or FQDN</td>
<td>IP or FQDN of the BigFix DB.</td>
<td>Y</td>
</tr>
<tr>
<td>bigfix.db.host</td>
<td>&lt;port no&gt;</td>
<td>Port at which TADDM will connect with BigFix DB.</td>
<td>Y</td>
</tr>
<tr>
<td>com.collation.</td>
<td>&lt;db name&gt;</td>
<td>Name of the BigFix DB.</td>
<td>Y</td>
</tr>
<tr>
<td>bigfix.db.dbname</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.collation.</td>
<td>&lt;user domain&gt;</td>
<td>Domain of user; Mandatory in case of Windows based Authentication.</td>
<td>N</td>
</tr>
<tr>
<td>bigfix.db.domain</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.collation.</td>
<td>&lt;userid&gt;</td>
<td>User id to access the BigFix DB.</td>
<td>Y</td>
</tr>
<tr>
<td>bigfix.db.domain</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.collation.</td>
<td>&lt;password&gt;</td>
<td>Password to access the BigFix DB; will be stored in encrypted form.</td>
<td>Y</td>
</tr>
<tr>
<td>bigfix.db.pwd</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note:
• In case of TADDM-DB connection break, TADDM will try to reconnect as per the “com.collation.bigfix.result.wait” setting.
• In case of any change in the above settings, TADDM restart is required.

5. TADDM - Result Fetching or Processing thread

Table 59.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation.bigfix.result.wait</td>
<td>&lt;value in sec&gt; Default=60 sec</td>
<td>When “com.collation.bigfix.enabled” is enabled, Results Fetching Thread will be spawned to periodically fetch the discovery result files from BigFix server. ”Results Fetching thread” will fetch the result packages from BigFix server as per the periodicity (defined in seconds) configured.</td>
<td>N</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.autodiscovery.enabled</td>
<td>True/false</td>
<td>If true, will enable the thread to process the stored ASD results files</td>
<td>Y</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.ProcessUnreachableIPs</td>
<td>True/false</td>
<td>The thread will process the ASD result for target that are unreachable.</td>
<td>Y</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.AsyncDiscoveryResultsDirectory</td>
<td>Path Default = var/asdd</td>
<td>path where results files are to be kept. Path is configurable, but is set to var/asdd by default.</td>
<td>N</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.autodiscovery.asdScope</td>
<td>&lt;scope name&gt; Default = ASD</td>
<td>The thread will pick the target mentioned in this scope to process result file. If this property is not mentioned, default ASD scope is processed.</td>
<td>N</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.autodiscovery.asdProfile</td>
<td>&lt;profile name&gt; Default = ASD</td>
<td>The thread will pick the sensors mentioned in this profile to process result file. If this property is not mentioned, default ASD profile is processed.</td>
<td>N</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.autodiscovery.filesThreshold</td>
<td>&lt;file threshold&gt; Default=20</td>
<td>Minimum Number of files required by thread, to start processing them. The thread will process the result if either the File threshold or Time threshold is met.</td>
<td>N</td>
</tr>
<tr>
<td>com.ibm.cdb.discover.asd.autodiscovery.timeThreshold</td>
<td>&lt;time threshold&gt; Default=60sec</td>
<td>Time threshold after which thread will process the result files, even if File Threshold is not met.</td>
<td>N</td>
</tr>
</tbody>
</table>

Note: 1. BigFix Discovery results will arrive asynchronously on TADDM server, and whenever one of the property (com.ibm.cdb.discover.asd.autodiscovery.filesThreshold, com.ibm.cdb.discover.asd.autodiscovery.timeThreshold) is met, then bunch of result files available will be processed and this will create a fresh “Discovery History”
entry. These properties shall be finetuned as per the specific requirements to control the number of “Discovery History” entries.

6. Cleanup

Table 60.

<table>
<thead>
<tr>
<th>S.No</th>
<th>Resources</th>
<th>TADDM Server</th>
<th>BES Root Server</th>
<th>BES Enspoint</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Created</td>
<td>Cleanup</td>
<td>Created</td>
</tr>
<tr>
<td>1.</td>
<td>Request Package</td>
<td>Y</td>
<td>N¹</td>
<td>Y</td>
</tr>
<tr>
<td>2.</td>
<td>Task</td>
<td>-</td>
<td>-</td>
<td>Y</td>
</tr>
<tr>
<td>3.</td>
<td>Action</td>
<td>-</td>
<td>-</td>
<td>Y</td>
</tr>
<tr>
<td>4.</td>
<td>Result Package</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>5.</td>
<td>Fileset</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Note:
- Request Package Cleanup on TADDM server is not supported
- Request Package cleanup on BES Root server is not supported. (Request package may get reused during rediscovery)
- Only Actions that were created by TADDM and which are in Expired state will be considered for cleanup (Except for Action created with name TADDMCLEANUP).
- Tasks created by TADDM will be remove, only when all actions associated with that task are already removed.
- Tasks created by TADDM will be remove, only when all actions associated with that task are already removed.
  - To exclude a specific task and its associated actions from cleanup (for Rediscovery support), retainBigFixTask.sh/.bat can be used as per details below:
    Usage: ./retainBigFixTask.sh <TaskName> <enable/disable>

Cleanup on TADDM Server

Table 61.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation.bigfix.taddm.cleanup.volume</td>
<td>&lt;limit size with suffix&gt;</td>
<td>&lt;Limited size after which the old processed files will be removed e.g. 50MB, 2GB, etc&gt;</td>
<td>N</td>
</tr>
<tr>
<td>com.collation.bigfix.taddm.cleanup.time</td>
<td>&lt;limit time with suffix&gt;</td>
<td>&lt;To check the processed files older than configured units e.g. 1D, 5H, 30M, etc&gt;</td>
<td>N</td>
</tr>
<tr>
<td>com.collation.bigfix.taddm.cleanup.runtime</td>
<td>No of minutes</td>
<td>TADDM cleanup thread will wait for configured no. of minutes after run.</td>
<td>N</td>
</tr>
</tbody>
</table>

Note:
• Result files cleanup on TADDM server will be done, only when atleast one of the properties (com.collation.bigfix.taddm.cleanup.volume, com.collation.bigfix.taddm.cleanup.time) is configured.

Cleanup on Endpoint

Table 62.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation.bigfix.endpoint.cleanup</td>
<td>&lt;Y or N&gt;</td>
<td>&lt;Default=Y&gt;</td>
<td>When set to Y, it will remove the request package zip, Extracted Request Package directory and newly created Result package zip from endpoint</td>
</tr>
</tbody>
</table>

Cleanup on BigFix Root Server

Table 63.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation.bigfix.root.cleanup.interval</td>
<td>&lt;no of days&gt;</td>
<td>&lt;Default=1&gt;</td>
<td>Periodicity of running the cleanup task to remove Result packages, Tasks and Expired Actions from BES Root Server.</td>
</tr>
<tr>
<td>com.collation.bigfix.root.cleanup.days</td>
<td>&lt;no of days&gt;</td>
<td>&lt;Default=5&gt;</td>
<td>Result files older than given number of days will be considered for removal.</td>
</tr>
</tbody>
</table>

7. Custom Relevance

Table 64.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation.bigfix.relevance.appendscope</td>
<td>true/false</td>
<td>If true, then custom Relevance query will be used, in addition to the specified scope</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>&lt;Default=true&gt;</td>
<td>If false, then only Custom Relevance query will be used, instead of specified scope</td>
<td></td>
</tr>
</tbody>
</table>
Table 64. (continued)

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation. bigfix. relevance</td>
<td>True/false</td>
<td>Relevance query for identifying a set of endpoints for given discovery. e.g. to discover endpoints with computer names starting with “nc04”, add this property in collation.properties as com.collation.bigfix.relevance=(it starts with &quot;nc04&quot;) of (computer name as lowercase). This will enable inclusion of endpoints whose computer names start with “nc04” along with the scope (depending on the value of the property com.collation.bigfix.relevance.appendscope).</td>
<td>N</td>
</tr>
</tbody>
</table>

8. Bigfix Package Temp Path Settings

Table 65.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation. bigfix.action.enable.os</td>
<td>&lt;action script for the configured os&gt;&lt;Default= Windows, AIX, Linux, SunOS&gt;</td>
<td>Action script for the configured OS will be included in the Bigfix ActionScript</td>
<td>N</td>
</tr>
<tr>
<td>com.collation. bigfix.temp.Windows Request Package Path</td>
<td>&lt;Default= C:\Windows\Temp&gt;</td>
<td>Path which will be used for the ASD Request Package. *Note: “\” would be required to give as “\” in windows path.</td>
<td>N</td>
</tr>
<tr>
<td>com.collation. asd.temp.Windows Result Package Path</td>
<td>&lt;Default= C:\Windows\Temp&gt;</td>
<td>Path which will be used for the ASD Result Packages.</td>
<td>N</td>
</tr>
<tr>
<td>com.collation. asd.temp.Unix Result Package Path</td>
<td>&lt;Default= /tmp&gt;</td>
<td>Path which will be used for the ASD Result Package.</td>
<td>N</td>
</tr>
<tr>
<td>com.collation. bigfix.temp.Linux Result Package Path</td>
<td>&lt;Default= /tmp&gt;</td>
<td>Path which will be used for the ASD Request Package.</td>
<td>N</td>
</tr>
<tr>
<td>com.collation. bigfix.temp.SunOS Result Package Path</td>
<td>&lt;Default= /tmp&gt;</td>
<td>Path which will be used for the ASD Request Package.</td>
<td>N</td>
</tr>
<tr>
<td>com.collation. bigfix.temp.AIX Result Package Path</td>
<td>&lt;Default= /tmp&gt;</td>
<td>Path which will be used for the ASD Request Package.</td>
<td>N</td>
</tr>
</tbody>
</table>
**Note:** Based on the above configured Temp path, the folder will be created on target endpoints, if it does not exists. E.g., In case of Windows 2003, default Temp path of “C:\Windows\Temp\” will be used and this folder will get created during the discovery.

**Examples of scripts execution**
The following shows examples of the scripts execution during this integration:

1. Executing script - encryptprops.sh

```
/opt/IBM/taddm/dist/bin/encryptprops.sh $COLLATION_HOME
```

2. Executing script – runBigFixDiscovery.sh

   TADDM Server – 9.167.42.227 (Linux)
   BigFix server – 10.160.161.195 (windows)
   BigFix endpoints – 10.160.161.196 (windows)
   10.160.161.212 (windows)
   Scope – ASD (having both BigFix endpoints)
   Profile – ASD (having sensors mentioned in section 2.2)
   Configuration – done as per section 3.

   a. Start discovery:

   ```
   [taddmusr@nc042227 bin]$ ./runBigFixDiscovery.sh -d -o /tmp -p ASD -s ASD
   BigFix Action will be applied total [1] times with [P1D] interval
   Task created on BES server with Name [20170828083852] and Action created with ID [633]
   DISCOVER: LAUNCH OK
   The Bigfix Discovery script exited successfully.
   ```

   b. Task Name – 20170828083852, Action id – 633

   c. Start Poll:

   ```
   [taddmusr@nc042227 bin]$ ./runBigFixDiscovery.sh -p -i 633
   Repeatedly poll the BigFix Action [1] number of times for every [1] seconds
   Total [1] computers with status : The action executed successfully.
   [Hostname] [Apply Count] [Line Number] [Start Time] [End Time]
   [PNC161196] [1] [98] [Mon, 28 Aug 2017 14:43:56 +0000] [Mon, 28 Aug 2017 14:44:11 +0000]
   [Hostname] [Apply Count] [Line Number] [Start Time] [End Time]
   [PRODUCTIONWAS8] [1] [37] [Mon, 28 Aug 2017 07:40:26 +0000] [Mon, 28 Aug 2017 07:40:26 +0000]
   POLL FINISHED
   The Bigfix Discovery script exited successfully
   ```

   Start Cleanup:

   ```
   [taddmusr@nc042227 bin]$ ./runBigFixDiscovery.sh -c
   CLEANUP TASK FOUND: TADDMCLEANUP with ID: 2067
   Cleanup Action created with ID: [2068]
   CLEANUP: LAUNCH OK
   The Bigfix Discovery script exited successfully
   ```

   Start Rediscovery:
The Bigfix Discovery script exited successfully.

Error Codes and Description

The following table lists the Error or Message IDs and their description that can come when user invokes the BigFix script - runBigFixDiscovery.sh (or .bat):

<table>
<thead>
<tr>
<th>Message ID</th>
<th>M:Message, C:Cause, E:Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTJTD1260E</td>
<td>M: Bigfix Discovery is not enabled, Please configure com.collation.bigfix.enabled in collation.properties</td>
</tr>
<tr>
<td></td>
<td>E: Bigfix Discovery script will not execute and Thread for fetching result will not be invoked</td>
</tr>
<tr>
<td>CTJTD1261E</td>
<td>M: Missing or incorrect arguments</td>
</tr>
<tr>
<td></td>
<td>C: Trying to execute Script with mode other than Discover, Poll, Cleanup or Rediscover</td>
</tr>
<tr>
<td></td>
<td>E: Script will not execute</td>
</tr>
<tr>
<td>CTJTD1262E</td>
<td>M: Incorrect number format provided</td>
</tr>
<tr>
<td></td>
<td>C: Properties or arguments specified in string format whereas required in number format</td>
</tr>
<tr>
<td></td>
<td>E: Bigfix Discovery script will not execute and Thread for fetching result will not function properly</td>
</tr>
<tr>
<td>CTJTD1263E</td>
<td>M: Failed to parse command line properties: &lt;property name&gt;</td>
</tr>
<tr>
<td></td>
<td>C: Arguments passed while running scripts are not supported</td>
</tr>
<tr>
<td></td>
<td>E: Script mode will not invoked</td>
</tr>
<tr>
<td>CTJTD1264E</td>
<td>M: &lt;Property name&gt; is missing in collation.properties</td>
</tr>
<tr>
<td></td>
<td>C: While executing script the required property(ies) are missing or invalid (check 'Configurable Properties')</td>
</tr>
<tr>
<td></td>
<td>E: Bigfix Discovery script will not execute and Thread for fetching result will not function properly</td>
</tr>
<tr>
<td>CTJTD1265I</td>
<td>M: Only Custom Relevance will be used, instead of specified scope</td>
</tr>
<tr>
<td>CTJTD1266I</td>
<td>M: Custom Relevance will be used, in addition to the specified scope</td>
</tr>
<tr>
<td>CTJTD1267E</td>
<td>M: Empty Scope specified, No elements found</td>
</tr>
<tr>
<td></td>
<td>C: Given scope/scopegroup doesn’t contains any element to define Endpoint</td>
</tr>
<tr>
<td></td>
<td>E: Discovery will not be invoked</td>
</tr>
<tr>
<td>CTJTD1268E</td>
<td>M: No sensors present or enabled in the specified profile</td>
</tr>
<tr>
<td></td>
<td>C: Given profile doesn’t contains any sensors</td>
</tr>
<tr>
<td></td>
<td>E: Discovery will not be invoked</td>
</tr>
</tbody>
</table>
### Integrating TADDM with ServiceNow

#### Purpose

TADDM offers a highly comprehensive discovery solution covering length and breadth of customer deployed infrastructure; whereas ServiceNow provides a flexible software as a service solution. The purpose of integrating TADDM with ServiceNow CMDB helps in enabling TADDM discovery available through ServiceNow UI along with other data processing or analysis facilities.

#### Key advantages of ServiceNow Integration

- With this integration in place, TADDM discovery data can be easily viewed on ServiceNow UI for any subsequent data processing or presentation formats.
- Reduction in development efforts for integrating new CMDB tables, as this can be done by updating the data transformation files.
- Integration Plugin provides the facility to present TADDM data into predefined customized reports and create role-based dashboards in a flash on ServiceNow UI.

---

**Table 66. (continued)**

<table>
<thead>
<tr>
<th>Message ID</th>
<th>M:Message, C:Cause, E:Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTJTD1269E</td>
<td>M: AASD Request Package does not exist</td>
</tr>
<tr>
<td></td>
<td>C: Problem in creating request package or doesn’t have permission</td>
</tr>
<tr>
<td></td>
<td>or not exists to upload</td>
</tr>
<tr>
<td></td>
<td>E: Discovery will not be invoked</td>
</tr>
<tr>
<td>CTJTD1270E</td>
<td>M: AASD Package size is more than configured</td>
</tr>
<tr>
<td></td>
<td>com.collation.bigfix.aasdpkgmaxsize threshold</td>
</tr>
<tr>
<td></td>
<td>C: Created Request package size is larger than the configured</td>
</tr>
<tr>
<td></td>
<td>E: Discovery will not be invoked</td>
</tr>
<tr>
<td>CTJTD1271E</td>
<td>M: Unable to connect to BigFix with reason : &lt;reason&gt;</td>
</tr>
<tr>
<td></td>
<td>C: Bigfix Web service connection problem due to invalid parameters</td>
</tr>
<tr>
<td></td>
<td>or invalid certificate. E: Package will not be uploaded and</td>
</tr>
<tr>
<td></td>
<td>Discovery will not be invoked</td>
</tr>
<tr>
<td>CTJTD1272E</td>
<td>M: Error caught while setup : &lt;reason&gt;</td>
</tr>
<tr>
<td></td>
<td>C: Unexpected scenario for code which is not handled</td>
</tr>
<tr>
<td></td>
<td>E: Script execution will not function properly</td>
</tr>
<tr>
<td>CTJTD1273I</td>
<td>M: Main script to (re)run a BigFix Discovery, OR to POLL a</td>
</tr>
<tr>
<td></td>
<td>specified discovery Action or to do manual cleanup</td>
</tr>
</tbody>
</table>
Reference

Reference links for TADDM and ServiceNow.

TADDM reference links

For more information about TADDM, refer to the following documentations:

- TADDM Wiki [https://www.ibm.com/developerworks/community/wikis/home/?%20lang=en#/wiki/Tivoli%20Application%20Dependency%20Discovery%20Manager/page/Home](https://www.ibm.com/developerworks/community/wikis/home/?%20lang=en#/wiki/Tivoli%20Application%20Dependency%20Discovery%20Manager/page/Home) This is a good source of up-to-date information and best practices for TADDM. Bookmark this page and get comfortable with it.
- Request for Enhancement Community [http://www.ibm.com/developerworks/rfe/?BRAND_ID=90](http://www.ibm.com/developerworks/rfe/?BRAND_ID=90) This community should be used for requesting enhancements to the product directly to IBM developers.

ServiceNow reference links

For more information about ServiceNow, refer to the following documentations:

- OAuth Configuration [https://docs.servicenow.com/bundle/london-platform-administration/page/administer/security/concept/c_OAuthApplications.html](https://docs.servicenow.com/bundle/london-platform-administration/page/administer/security/concept/c_OAuthApplications.html)
- ServiceNow Community [https://community.servicenow.com/community](https://community.servicenow.com/community)
- Identification Rule [https://docs.servicenow.com/bundle/london-servicenow-platform/page/product/configuration-management/task/t_CreateCIMIdentificationRule.html](https://docs.servicenow.com/bundle/london-servicenow-platform/page/product/configuration-management/task/t_CreateCIMIdentificationRule.html)

Solution Architecture

ServiceNow Integration is achieved by creating a generic tool or plugin framework capable of having software connectors for both source and target data flows. The plugin should be able to receive data from source data connector, perform the configured data transformations and pass on the transformed data to the target data connector. It is the sole responsibility of the connectors to interface with the end-points for data extraction and integration. This integration of TADDM discovery data with ServiceNow CMDB supports initial bulk migration and any subsequent Configuration Item (CI) dynamic updates using push data mechanism.

Migration

Initial data migration from TADDM database to ServiceNow CMDB.

1. Configure the different <name>.property files for proper initialization and providing default values.
2. Configure the configuration files: mapping.xml and <CI>.xml for the supported CIs and their respective transformation mappings.
3. Tool or Integration Plugin retrieves data from TADDM storage server using the standard APIs in the form of model objects.
4. Model object information is transformed per the defined mapping rules.
5. Transformed data is pushed towards ServiceNow instance via REST APIs and inserted in ServiceNow CMDB.

Dynamic Updates
Dynamic data update integration resulting from new TADDM discoveries into the ServiceNow CMDB.
1. Configure the different `<name>`.property files for proper initialization and providing default values.
2. Configure the configuration files: mapping.xml and `<CI>`.xml for the supported CIs and their respective transformation mappings.
3. Configure the `<CI>`.xml for listing the CIs for which change events will be generated by the TADDM change event framework.
4. Tool or Integration Plugin receives events from TADDM change event framework, which contains the updated CI.
5. Relevant information is retrieved from TADDM storage server using standard APIs in form of model objects.
6. Model object information is transformed per the defined mapping rules.
7. Transformed data is pushed towards ServiceNow instance via REST APIs and inserted in ServiceNow CMDB.

Running the Integration Plugin
To run the Integration Plugin, execute the following steps:

Procedure
1. Collect zip. TADDM iso and FP package contains IntegrationPlugin zip. Once installed or mounted, IntegrationPlugin package is present at /opt/IBM/taddm/dist/tools.
2. Extract zip.
   Command: `unzip IntegrationPlugin.zip`
   Integration package will be uncompressed and shall contain the following structure:
   /lib  lib directory contains all the required jars and IntegrationPlugin.jar
   /plugin.sh IntegrationPlugin script
   /resources resources directory contains properties files and configuration files
   /security security directory contains Java security policy file
   /external  external directory contains IBM Java JDK
3. User must perform the necessary configurations.
   • Configure supported Configuration Items (CIs) and their types in mapping.xml for transformation purpose from TADDM DB to ServiceNow CMDB
   • Update corresponding `<CI>`.xml for attribute mapping
   • Specify required target, source and plugin properties in properties files
   • Configure Access list with required parameters at TADDM side
   • Copy jdk-Linux-x86_64.zip from TADDM hosted machine (path:/opt/IBM/taddm/dist/external/jdk) to (path:<IntegrationPlugin path>/external) on the machine which hosts the tools/Integration Plugin.
   Set the path IBM_JAVA path in plugin.sh.
   – Unzip jdk-Linux-x86_64.zip in external folder
   – Command: `unzip jdk-Linux-x86_64.zip`  
   ```java
   $HOME = /opt/IBM/taddm/dist/tools/IntegrationPlugin/
   IntegrationPlugin/external/jdk-Linux-x86_64
   ```
- Copy TADDMSec.properties file from TADDM hosted machine (path:/opt/IBM/TADDM/dist/etc) to (path:<IntegrationPlugin path>/security/etc) on the machine which hosts the tool or Integration Plugin.

4. Enable Migration and Change Event process in plugin.properties.

5. Run script.

Integration Plugin package contains plugin.sh script in /script.

Starting the Integration Plugin:

Option 1:
[Command] ./plugin.sh start
Taddm User Id and Password – will read from internal properties file
[Command] ./plugin.sh start &
Taddm User Id and Password – will read from internal properties file
& - for running the plugin in background

Note: User is recommended to exit gracefully from the terminal from which Integration plugin is invoked and put in background. This can be done using commands "exit" or "disown".

Option 2:
[Command] ./plugin.sh -u <username> start
-u, --user <Taddm User Id>
Password – to be entered on subsequent command line prompt (for safety reasons)

Option 3:
[Command] ./plugin.sh -u <username> -p <password> start
-u, --user <Taddm User Id>
-p, --password <Taddm User password>

Stopping the Integration Plugin:

Option 1:
[Command] ./plugin.sh stop

Option 2:
[Command] ./plugin.sh stop force

Status of Integration Plugin:

Option 1:
[Command] ./plugin.sh status

Integrating TADDM with ServiceNow CMDB

In TADDM 7.3 Fixpack 6, this Integration focuses on migrating or pushing discovery data from TADDM to ServiceNow instance via REST APIs.

Prerequisites

1. ServiceNow instance should be running and accessible from Tool or Integration Plugin for sending the transformed data.

2. TADDM storage server must be running and accessible from Tool or Integration Plugin for retrieving discovery data.

3. There must be network connectivity from TADDM server to the machine hosting the Tool or Integration Plugin for receiving the dynamic change events.

4. Copy jdk-Linux-x86_64.zip from TADDM hosted machine (path: /opt/IBM/taddm/dist/external/jdk) to (path:<IntegrationPlugin path>/external) on the machine which hosts the tools/Integration Plugin. Set the path IBM JAVA path in plugin.sh.
   - Unzip jdk-Linux-x86_64.zip in external folder
   - Command : unzip jdk-Linux-x86_64.zip
javaHome=<IntegrationPlugin path>/external/jdk-Linux-x86_64

5. If Integration Plugin is running on TADDM machine, specify taddmHome path in IntegrationPlugin/resources/config/taddm.properties.
    taddmHome=/opt/IBM/taddm/dist.

6. If Integration Plugin is not running on TADDM discovery server then copy TADDMSec.properties from TADDM machine
    (/opt/IBM/taddm/dist/etc) to <IntegrationPlugin path>/security/etc on plugin installed machine. taddmHome=<IntegrationPlugin path>/security/etc.

    Note: If machine does not have network connection, enable network connectivity first and enter below configuration in
    /etc/hosts.XX.XX.XX servicenow service-now.dev345.com

7. Ensure that all the necessary configuration is done.

Limitations
Following limitations are associated with current release:

1. Support for TADDM and ServiceNow CMDB Integration starts with TADDM 7.3 Fixpack 6 release.
2. ServiceNow CMDB Integration Plugin or tool is currently supported for RHEL 7.3, 7.4 and 7.5 platforms only.
3. Not all TADDM CIs are presently supported using this integration in this FP.

Configuration
Follow the steps mentioned in this section to set the desired configuration.

Basic configuration:

Follow the steps mentioned in this section to set the desired configuration:

<IntegrationPlugin path>/resources/config/plugin.properties

Basic Integration Plugin configuration

• enableChangeEventManagement= <enables and disables processing for change event updates>
• enableMigration= <enables and disables the migration of CI data>
• runMigrationAgain= <run migration process again for all CIs>
• migrationThreadPoolSize= <Number of threads for processing migration data>
• changeEventThreadPoolSize= <Number of threads for processing change event updates>
• sourceConnector= <Source endpoint>
• targetConnector= <Target endpoint>
• javaHome= <IBM Java home path>
• useMagicMethod= <Set true to run magic methods>
• sessionRetryTime=<Set time for session retry for source and target connector>

Log settings
• fileSize= <log file size in MB>
• backupIndex= <Count of backup log files before they are lost/rotated with latest logs>
• logLevel= <log level>

For graceful shutdown
• gracefulEventProcessingCount= <Count of number of events to be processed during shutdown time>
• gracefulEventProcessingTime = <Time for graceful shutdown processing (in milliseconds)>

<IntegrationPlugin path>/resources/taddm_snow/taddm.properties
TADDM configurations (Source)
• taddmHost= <IP Address of the TADDM machine>
• taddmPort= <TADDM server port>
• taddmUserName= <User name>
• taddmPassword= <TADDM user password and Password stored in encrypted form>
• taddmUseSSL= <If true, secure connection will be established>
• taddmTrustStorePath= <Specify SSL certificate path>
• taddmHome= <Path of collation home>
• tlsVersion= <Same TLS version as set in TADDM collation.properties file>

Change event
• listenIp= <IP address of machine on which Plugin is running and would listen for change event processing>
• listenPort= <Port on which Plugin would listen for change event processing>

<IntegrationPlugin path>/resources/taddm_snow/serviceNow.properties
ServiceNow configuration (Target)
• serviceNowInstanceUrl= <Service now instance URL>
• serviceNowOAuthUrl= <OAuth URL for generating access token>
• discoverySource= <discovery source type>
• relationTable= <table contains relations type>
• refreshTokenLifespan= <Refresh Token life time span in seconds>
• serviceNow.attribute.softdelete.status= <Status to be used for indicating soft delete of record>
• serviceNow.attribute.softdelete.name = <ServiceNow attribute to maintain the soft delete status>
• serviceNow.attribute.guid.name= justification <ServiceNow attribute to store TADDM GUID, a per-record unique identifier>

Proxy settings
• enableProxy= <Enable proxy while connecting to ServiceNow>
• proxyHost= <Proxy IP>
• proxtPort= <Proxy Host>
• proxyUserName= <User Name for Proxy>
• proxyPassword= <Proxy Password>

$COLLATION_HOME/etc/collation.properties
Enable ChangeEvent from TADDM
• com.ibm.cdb.omp.changeevent.enabled= <Enable ChangeEvent module to send out events from TADDM>

Newly added Properties:
• com.ibm.cdb.omp.changeevent.optimized.update = <Property to allow consolidation of update events for multiple attributes of the same Model Object>
• com.ibm.cdb.omp.changeevent.classnames.nonfriendly= <This property returns the CI class name instead of non-user friendly class name>
• com.ibm.cdb.omp.changeevent.deletefromportal= <Property to enable sending of delete events when the CI is deleted from portal>

Password encryption
• com.collation.integrationplugin.taddm.password= <encrypted TADDM password>
• com.collation.integrationplugin.target.proxy.password= <encrypted target proxy password>

OAuth configuration at ServiceNow instance
OAuth 2.0 lets users access instance resources through external clients by obtaining a token rather than by entering login credentials with each resource request.

Role required: security_admin

ServiceNow instance provides an endpoint for third-party clients to pull data from that instance. Using the below procedure, configure OAuth application endpoint for Integration Plugin to access the instance and set OAuth parameters.

1. Login into ServiceNow instance with credentials.
2. Activate the OAuth 2.0 plugin, set the system property
   com.snc.platform.security.oauth.is.active to true in sys_properties table.
3. Navigate to System OAuth > Application Registry.
4. Click New and then click Create an OAuth API endpoint for external clients.
5. Set below OAuth parameters. Enter the value of Name and Client Secret:
   • Name: A unique name
   • Client ID: Client ID automatically generated by ServiceNow OAuth server.
   • Client Secret: Client secret for the OAuth application
   • Refresh Token Lifespan: Time in seconds the Refresh Token will be valid
   • Access Token Lifespan: Time in seconds the Access Token will be valid
6. Record the value of Client ID and Client Secret from the previous step.
7. Click on Submit.

A new OAuth Entity Profile record is created for Integration Plugin.

Note: OAuth configuration is not mandatory to establish connectivity with ServiceNow. If OAuth configuration is provided, then it will be used for token-based communication with ServiceNow instance, else, the instance credentials (username and password) will be used in REST queries.
Certificate
Integration Plugin ensures secure communication with TADDM and provides support for TLS over this interface. To enable TLS, do the following configuration.

TADDM:
1. Refer following properties in $COLLATION_HOME/etc/collation.properties file and infer the TLS version that needs to be set on Integration Plugin side.
   - com.ibm.cdb.secure.server=true or false
   - com.ibm.cdb.rmi.ssl.protocol=<TLS version>
   - com.ibm.cdb.ssl.protocol=<TLS version>
2. Restart TADDM.
3. Login to TADDM Discovery Management Console.
4. Under the Discovery Management Console heading, select Show SSL Options.
5. Click Download Trust Store to download the truststore and select a directory in which to save the truststore file.
6. In the input box to the right of the Download Trust Store link, enter the name of the directory that contains the truststore file.

Integration plugin:
1. Set this certification path in taddmTrustStorePath property in <Integrationplugin path>/resources/taddm_snow/taddm.properties. If Integration Plugin is installed on different machine, copy certificate from TADDM installed machine, then set the complete path.
2. Set true for taddmUseSsl property in <Integrationplugin path>/resources/taddm_snow/taddm.properties.
3. Set tlsVersion property in <Integrationplugin path>/resources/taddm_snow/taddm.properties with the same protocol version as set in TADDM.

Password Encryption
To encrypt passwords in the collation.properties file, complete the following steps:
1. Edit the collation.properties for com.collation.integrationplugin.taddm.password property, or archive the user password using clear text, or both.
2. Run either the encryptprops.sh file (located in the $COLLATION_HOME/bin directory). This script encrypts the passwords.
3. Copy encrypted password from com.collation.integrationplugin.taddm.password property to taddmPassword property in <IntegrationPlugin path>/resources/config/taddm_snow/taddm.properties.

Processing Thread settings
Multithreading architecture has been used in Integration Plugin for bulk data processing. Processing threads can be configured to speed up the process and achieve higher performance.

User can configure number of threads for migration and change event processing using the below properties in <IntegrationPlugin path>/resources/config/plugin.properties file.
- migrationThreadPoolSize= <Number of threads for processing migration data >
- changeEventThreadPoolSize= <Number of threads for processing change event updates >

Other configuration:

Configure Mapping and Configuration item files for supported CIs. This requires prior understanding of TADDM and ServiceNow CMDB table and attributes.

<IntegrationPlugin path>/resources/config/mappingFiles/Mapping.xml

Mapping.xml is configuration file to configure supported CIs name and their type. <configurationItemMapping> is root element.

Procedure:
Describe <configureItem> attributes in <filename> tag to configure supported CIs name and their Id.
1. Configuration Item name.
2. Transformation mapping (XML) file for this CI.
3. Id for supported CI.

Note: ModelObject is the top most supported CI. First entry is always for modelObject and you should not provide id attribute for modelObject CI.

<IntegrationPlugin path>/resources/config/mappingFiles/CI.xml

Configuration item XML files are used to configure the attributes for transforming TADDM discovery data to ServiceNow CMDB and establish relationships between TADDM and ServiceNow CIs. These files have the name as the <CI name>.xml.

Describe <configurationItem> tag for any new CI and their type. It contains 1-to-1 mapping for the data attributes having details on:
- Relationships
- Data Source attribute
- Data Target attribute
- Data Target table
- Data customization hook

Creation of custom identifier rule

Identification rules are used to uniquely identify CIs in the ServiceNow CMDB, as part of the identification and reconciliation process. Each ServiceNow CMDB class can be associated with a single identification rule.

Role required : security_admin

In a CI identification rule, specify a CI identifier, identifier entries and related entries that uniquely identifies the CI.

Procedure.
1. **Login** to your ServiceNow instance.
2. **Navigate** to Configuration > CI Class Manager.
3. Click **Hierarchy** to display the CI Classes list. Select the **class** for which to create an identification rule.
4. In the class navigation bar, expand **Class Info** then click **Identification Rule**.
5. Click **Edit** to edit an existing rule or click **Add** in the Identification Rule section to create one. Fill out the form, then click **Save**.

Refer to ServiceNow document for detailed description:
https://docs.servicenow.com/bundle/london-servicenow-platform/page/product/configuration-management/task/t_CreateCIIdentificationRule.html

**Addition of Discovery Source**

Every record migrated or updated from TADDM towards ServiceNow will be having the "Discovery Source" set to "BM TADDM". This is done to distinguish TADDM integrated records from other coming in through another discovery source. If you want to customize this value further, below steps must be performed:

**ServiceNow instance**

1. **Login** to ServiceNow instance.
2. **Search** 'System Defination-Tables' in Filter Navigator.
3. **Search** 'cmdb_ci' table name in search section.
4. **Click** on 'cmdb_ci' table from search results.
5. In column section **search** 'Discovery source'.
6. **Click** on 'Discovery source', go to choices sub tab in third section.
7. **Click** on 'New button', provide mandatory field values the 'Integration Plugin' to reflect the changes.
   - Label: Specify label for choice
   - Value: Specify value for choice
8. **Click** on 'Submit button'. New choice will be added into choice list.

**Integration Plugin**

1. Mention the above mentioned choice value to discoverySource property of serviceNow.properties file.
   discoverySource=IBM TADDM

**Magic method manipulation**

Magic method provides extremely useful functionality to manipulate the value of mapping attributes. Magic method can be customized as per the user’s requirement. It converts the value of desired format before transforming it to target side.

**Procedure:**

1. **Configure** method name in `<magicMethod>` tag for a specific attribute in the CI transformation XML file.
2. **Create** the method with same name in `/magicMethod/MagicMethod.txt`.
3. Method should be written in Java with specified format.
4. **Save** the file.
5. **Set** Java Home path in javaHome property at `<IntegrationPlugin path>/resources/config/plugin.properties`.
6. **Set true** against use MagicMethod property value at `<IntegrationPlugin path>/resources/config/plugin.properties`.
7. **Restart** the Integration Plugin to reflect the changes.
Log configuration and troubleshooting tips:

Integration logs are created at /log directory. Each time Integration Plugin starts, a new log directory is created with its name represented by timestamp.

Log configuration: `<IntegrationPlugin path>/resources/config/plugin.properties`

- `fileSize`: size of log file
  Default size is 20MB. It can be increased or decreased as per the requirement.
- `backupIndex`: Count of backup file
  Default count is 5.
- `logLevel`: Define log level for Integration plugin
  Default is "info". Supported log level types are "info" and "debug".

Troubleshooting

In case of any failure encountered during Integration processing, following points can be validated.

1. Ensure that all the prerequisites are satisfied.
2. Integration processing logs can be referred.

- **Plugin core log**: It contains logs related to core plugin modules like process, connection, configuration, initialization and shutdown.
  `<IntegrationPlugin path>/log/<timestamp>/plugin.log`

- **Migration log**: It contains logs for migration record processing.
  `<IntegrationPlugin path>/log/<timestamp>/migration.log`

- **Change event management log**: It contains logs for change event processing corresponding to dynamic events received from TADDM.
  `<IntegrationPlugin path>/log/<timestamp>/changeEvent.log`

- **Transformation log**: It contains logs for transformation mapping and processing in JSON form.
  `<IntegrationPlugin path>/log/<timestamp>/transform.log`

- **Error log**: It contains any errors reported across the Integration plugin.
  `<IntegrationPlugin path>/log/<timestamp>/error.log`

Running integration

Integration Plugin works on transferring initial data via migration process along with sending subsequent CI updates coming from TADDM towards ServiceNow CMDB.

Migration:

Migration process migrates TADDM discovery data to ServiceNow CMDB for supported CIs. It's a one-time process.

Migration management

To enable migration process, set "enableMigration" property to true in /resources/config/plugin.properties. This requires the tool/Integration plugin to be restarted to take effect.

To re-run the migration process, set "runMigrationAgain" property to true in /resources/config/plugin.properties.

Migration status can be viewed in ./plugin.sh status. It displays CI names and their status respectively.
Change event management:

TADDM events are generated when a configuration change is detected in IT environment after discovery runs on that environment. To generate such dynamic events, some pre-configuration is required.

Configuration on TADDM side

To send change events, you must configure TADDM with information about the event-handling systems to which you want to send change events.

Procedure:
1. To enable change events, in the $COLLATION_HOME/etc/collation.properties file, set the following property:
   com.ibm.cdb.omp.changeevent.enabled=true.
2. To configure which resources are tracked for changes and to which event-handling systems the events are sent, edit the $COLLATION_HOME/etc/EventConfig.xml file.

   For information about the format that you must use to specify information in the EventConfig.xml file, see EVENT LISTNERS.

   In EventConfig.xml specify supported CI name and configure a listener and a corresponding recipient block with required information.

   Create CI entry correspond to which Event are required as below:

   ```xml
   <!--- ComputerSystem --->
   <listener object="ComputerSystem" enabled="true">
   <alert recipient="taddm-snow-plugin-host" />
   <attribute name="guid" operator="not-equals">
   <value>0</value>
   </attribute>
   </listener>
   Template for TADDM-SNOW Event recipient with IP and Port
   <recipient name="taddm-snow-plugin-host" type="itm">
   <address>Plugin host IP</address>
   <port>7575</port>
   </recipient>
   ```


3. To get the CI update events with attribute name, in the $COLLATION_HOME/etc/collation.properties file, set the following property:
   com.ibm.cdb.omp.changeevent.optimized.update=true.
4. To store and subsequently retrieve the class names of CI from events, in the $COLLATION_HOME/etc/collation.properties file, set the following property:
   com.ibm.cdb.omp.changeevent.classnames.nonfriendly=true.
5. To get delete events for the CIs manually deleted from TADDM Data Management Portal, configure the following:

   - In the $COLLATION_HOME/etc/collation.properties file, set the following property:
     com.ibm.cdb.omp.changeevent.deletefromportal=true
   - In the /opt/IBM/taddm/dist/etc/CiNamesForDeleteEvents.txt, input the CI names for getting delete events from portal
6. Restart TADDM after the setting these changes in files.

Event handling on Integration Plugin side
Integration Plugin handles all four types of events that are triggered by TADDM.

1. Create Event: corresponding to the new CIs post TADDM discovery.
2. Update Event: for a change in any attribute of same CI (not in stored references).
3. Delete Event: corresponding to any CI being deleted during discovery or manually from TADDM data management portal.
   Soft-delete will be performed on ServiceNow record entries. This means that the entry will not be explicitly removed, but existing unused attribute (lease contract) from base ServiceNow CI table will be used to mark that status.
4. Propagated Event: It is implicit Update event for scenarios when the CI references are updated.

**Running script:**

To run discovery, script “plugin.sh” is to be executed from IntegrationPlugin directory. The script can be executed with 3 different arguments: 'start', 'status' and 'stop'. Refer the details below:

**Starting the Integration Plugin**

Option 1:

[Command] ./plugin.sh start
start – parameter to start the Integration plugin
TADDM User Id and Password – will be read from internal properties file

[Command] ./plugin.sh start &
start – parameter to start the Integration plugin
TADDM User Id and Password – will be read from internal properties file
& – for running the plugin in background

Option 2:

[Command] ./plugin.sh -u <username> start
-u, --user <TADDM User Id>
start – parameter to start the Integration plugin
Password – to be entered on subsequent command line prompt (for safety reasons)

Option 3:

[Command] ./plugin.sh -u <username> -p <password> start
-u, --user <TADDM User Id>
-p, --password <TADDM User password>
start – parameter to start the Integration plugin

**Stopping the Integration Plugin**

Option 1:

[Command] ./plugin.sh stop
stop – parameter to gracefully stop the Integration plugin

Option 2:

[Command] ./plugin.sh -stop force
stop – parameter to stop the Integration plugin

**Status of Integration Plugin**

Option 1:

[Command] ./plugin.sh status
status – parameter for the status of Integration plugin

**Possible failure scenarios**

In case of failure encountered during running of Integration Plugin, following things can be checked:
1. Confirm that all prerequisite are followed.
2. Validate Integration Plugin error logs at <IntegrationPlugin path>/log/<timestamp>/error.log.

Failure scenarios and their resolutions:

These are the common errors, that can be occurred during Integration Plugin initialization:

Encryption exception

Exception: Did not get the api session
com.collation.platform.util.FIPSEncryptionException: CTJOP0165E"An IO error occurred while loading the TADDM cryptographic key".
security/TADDMSec.properties (No such file or directory).
Resolution:
1. Integration Plugin and TADDM both are installed on same machine, set taddmHome path at <Integrationplugin path>/resources/taddm_snow/taddm.properties.
2. If Integration Plugin is installed on a different machine, validate TADDMSec.properties present at <Integrationplugin path>/security/etc location.
3. If not, copy TADDMSec.properties from TADDM machine and locate at <Integrationplugin path>/security/etc location on Integration Plugin installed machine.

Credential not found Error

Error: [TSI] Credential not found.
Resolution: Configure Access list for ServiceNow credentials.

Connection failed with target retry

Error: [TSI] Connection failed with target retry count 5.
Resolution: Login to TADDM Discovery Management Console and validate whether the correct ServiceNow credentials are configured in Discovery > Access List and valid Instance URL is configured in /resources/taddm_snow/serviceNow.properties.

Appendix A: Properties used in the integration

Properties used in the integration.

Plugin.properties

Table 67.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableChangeEventManagement</td>
<td>true/false</td>
<td>If true, Change Event processing will be enabled. Plugin restart is not required after changing the property value.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default=true</td>
<td></td>
<td></td>
</tr>
<tr>
<td>enableMigration</td>
<td>true/false</td>
<td>If true, CI Migration processing will be enabled.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default=true</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 67. (continued)

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>runMigrationAgain</td>
<td>true/false</td>
<td>If true, Migration will run again for all the configured CIs.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Default=false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>migrationThread</td>
<td>Recommended</td>
<td>Specify ThreadPool count for CI migration processing. Increase the count if require to process bulk data.</td>
<td>Y</td>
</tr>
<tr>
<td>PoolSize</td>
<td>value&gt;1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Default=5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>changeEventThread</td>
<td>Recommended</td>
<td>Specify ThreadPool count for CI Change Event processing.</td>
<td>Y</td>
</tr>
<tr>
<td>PoolSize</td>
<td>value&gt;1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Default=5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sourceConnector</td>
<td>&lt;source&gt;</td>
<td>Currently supported source type is TADDM.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default=TADDM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>targetConnector</td>
<td>&lt;target&gt;</td>
<td>Currently supported target type is SNOW.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default=SNOW</td>
<td></td>
<td></td>
</tr>
<tr>
<td>javaHome</td>
<td>&lt;javahomepath&gt;</td>
<td>Set IBM Java home path to compile code written in magic methods. Provide the same IBM_JAVA path of plugin.sh script.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Default=/opt/IBM/taddm/dist/external/jdk-Linux-x86_64/jre</td>
<td></td>
<td></td>
</tr>
<tr>
<td>useMagicMethod</td>
<td>true/false</td>
<td>Set true to run magic methods used to dynamically update the transformed values.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Default=false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sessionRetryTime</td>
<td>&lt;time in</td>
<td>Set time for session retry for source and target connector both. Time should be in milliseconds.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>milliseconds&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Default: 2000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Log configuration

**Table 68.**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileSize</td>
<td>&lt;filesize&gt;</td>
<td>Specify file size for log files.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default=20MB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>backupIndex</td>
<td>&lt;count&gt;</td>
<td>Specify Count of backup file.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default=5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>logLevel</td>
<td>&lt;loglevel&gt;</td>
<td>Specify log level. Supported log level is info or debug.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default=info</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Graceful Shutdown settings
Table 69.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>gracefulEvent ProcessingCount</td>
<td>&lt;queuesize&gt;</td>
<td>Number of pending events to be processed before gracefully shutting down the threads.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default=5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gracefulEvent ProcessingTime</td>
<td>&lt;time in milliseconds&gt; 240000</td>
<td>Time interval during which Integration Plugin must process any pending queue events before it is completely shut down.</td>
<td>Y</td>
</tr>
</tbody>
</table>

**taddm.properties**

Table 70.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>taddmHost</td>
<td>IP</td>
<td>IP address of TADDM discovery server.</td>
<td>Y</td>
</tr>
<tr>
<td>taddmPort</td>
<td>Port Default=-1</td>
<td>Port on TADDM server to which queries need to be sent. Port 9433 is used for queries. Setting the port -1 in code, it conditionally set it to 9433.</td>
<td>Y</td>
</tr>
<tr>
<td>taddmUserName</td>
<td>&lt;userid&gt;</td>
<td>Username to access TADDM.</td>
<td>Y</td>
</tr>
<tr>
<td>taddmPassword</td>
<td>&lt;password&gt;</td>
<td>Password to access TADDM. Will be stored in encrypted form.</td>
<td>Y</td>
</tr>
<tr>
<td>taddmUseSSL</td>
<td>true/false</td>
<td>If true, secure connection will be established.</td>
<td>N</td>
</tr>
<tr>
<td>taddmUseSSL</td>
<td>Default=false</td>
<td></td>
<td></td>
</tr>
<tr>
<td>taddmTrustStorePath</td>
<td>&lt;path&gt;</td>
<td>Specify SSL certificate path.</td>
<td>N</td>
</tr>
<tr>
<td>taddmHome</td>
<td>&lt;path&gt;</td>
<td>Provide the $COLLATION_HOME path on taddm hosted machine.</td>
<td>Y</td>
</tr>
<tr>
<td>taddmHome</td>
<td></td>
<td>Provide the &lt;IntegrationPlugin path&gt;/security/etc, if plugin is running independently..</td>
<td></td>
</tr>
<tr>
<td>tlsVersion</td>
<td>TLS version</td>
<td>Specify the same value for TLS version as set in TADDM (e.g. TLSv1.0, TLSv1.1, TLSv1.2)</td>
<td>N</td>
</tr>
<tr>
<td>tlsVersion</td>
<td>Default: TLSv1.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Change event settings

Table 71.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>listenIp</td>
<td>IP</td>
<td>IP of change event listening machine.</td>
<td>Y</td>
</tr>
<tr>
<td>listenPort</td>
<td>&lt;port no&gt;</td>
<td>Port of change event listening machine.</td>
<td>Y</td>
</tr>
</tbody>
</table>
## ServiceNow.properties

ServiceNow configurations

**Table 72.**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceNowInstanceUrl</td>
<td>URL</td>
<td>URL of ServiceNow instance.</td>
<td>Y</td>
</tr>
<tr>
<td>serviceNowOAuthUrl</td>
<td>URL</td>
<td>URL of OAuth for generating access token.</td>
<td>N</td>
</tr>
<tr>
<td>discoverySource</td>
<td>&lt;discoverySource&gt;</td>
<td>Set discovery source type for payload.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default: IBM TADDM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>relationTable</td>
<td>&lt;table&gt;</td>
<td>ServiceNow table name which contains relations type for all ServiceNow CMDB relations.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default: cmdb_rel_type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>refreshTokenLifespan</td>
<td>&lt;time&gt;</td>
<td>Refresh Token life time span in seconds.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Default: 8640000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>serviceNow.attribute.softdelete.status</td>
<td>&lt;status&gt;</td>
<td>To mark record as inactive in case of soft delete.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default: InActive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>serviceNow.attribute.softdelete.name</td>
<td>&lt;attribute name&gt;</td>
<td>Configure ServiceNow attribute for storing the soft delete status.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default: lease_id</td>
<td></td>
<td></td>
</tr>
<tr>
<td>serviceNow.attribute.guid.name</td>
<td>&lt;attribute name&gt;</td>
<td>Configure ServiceNow attribute for storing TADDM CI identifier (GUID).</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Default: justification</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Proxy settings**

**Table 73.**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>enableProxy</td>
<td>True/false</td>
<td>If true, proxy will be enabled while connecting to ServiceNow.</td>
<td>N</td>
</tr>
<tr>
<td>proxyHost</td>
<td>&lt;IP or Host&gt;</td>
<td>Proxy IP.</td>
<td>N</td>
</tr>
<tr>
<td>proxyPort</td>
<td>&lt;Port&gt;</td>
<td>Proxy port.</td>
<td>N</td>
</tr>
<tr>
<td>proxyUserName</td>
<td>&lt;username&gt;</td>
<td>Username to access via proxy connection.</td>
<td>N</td>
</tr>
<tr>
<td>proxyPassword</td>
<td>&lt;password&gt;</td>
<td>Set Proxy Password. It will be stored in encrypted form. Encrypted password will be generated at com.collation.integrationplugin.target.proxy.password property in collation.properties.</td>
<td>N</td>
</tr>
</tbody>
</table>
**Collation.properties**

Collation.properties on TADDM.

Change event configuration

*Table 74.*

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.ibm.cdb.omp.changeevent.enabled</td>
<td>True/false</td>
<td>If true, change event module will be enabled to send change events. Plugin restart is required after changing the property value.</td>
<td>Y</td>
</tr>
<tr>
<td>com.ibm.cdb.omp.changeevent.optimized.update</td>
<td>True/false</td>
<td>Property to allow consolidation of update events for multiple attributes of the same Model Object. Plugin restart is required after changing the property value.</td>
<td>Y</td>
</tr>
<tr>
<td>com.ibm.cdb.omp.changeevent.classnames.nonfriendly</td>
<td>True/false</td>
<td>This property returns the CI class name instead of non-user-friendly class name. Plugin restart is required after changing the property value.</td>
<td>Y</td>
</tr>
<tr>
<td>com.ibm.cdb.omp.changeevent.deletefromportal</td>
<td>True/false</td>
<td>Property to enable sending of delete events when the CI is deleted from portal. Plugin restart is required after changing the property value.</td>
<td>Y</td>
</tr>
</tbody>
</table>

Password property

*Table 75.*

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Possible Values</th>
<th>Description</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.collation.integrationplugin.taddm.password</td>
<td>&lt;password&gt;</td>
<td>TADDM user password. Will be stored in encrypted form.</td>
<td>N</td>
</tr>
<tr>
<td>com.collation.integrationplugin.target.proxy.password</td>
<td>&lt;password&gt;</td>
<td>Proxy password for target connection. It shall be stored in encrypted form. Encrypted password is generated by running encryptprops.sh and set this password to proxyPassword property in serviceNow.properties.</td>
<td>N</td>
</tr>
</tbody>
</table>

**Appendix B: Integration Plugin parameter help for different modes**

`. /plugin.sh` – The script can be executed with 3 different arguments ‘start’, ‘status’ and ‘stop’.

Argument: START
usage: ./plugin.sh -u <username> -p <password> start [-h]
where,
-u <username> Provide TADDM user name
-p <password> Provide TADDM user password. Not recommended for security purpose.
-h,--help show help.

Example to check Integration plugin execution
Start:
./plugin.sh start

Argument: STATUS
Integration Plugin status displays CI name, their running status and processed number of records and count of failure and success of records.

./plugin.sh status
Migration Enabled=true
Change Event Management=true
Migration CI Status
ComputerSystem=COMPLETED|TOTAL=7|SUCCESS=4|FAILED=3
AppServer=COMPLETED|TOTAL=4|SUCCESS=4|FAILED=0
StorageVolume=COMPLETED|TOTAL=14|SUCCESS=14|FAILED=0
StoragePool=COMPLETED|TOTAL=0|SUCCESS=0|FAILED=0
DiskDrive=COMPLETED|TOTAL=6|SUCCESS=6|FAILED=0
FileSystem=COMPLETED|TOTAL=9|SUCCESS=9|FAILED=0

Plugin Status: Running

Argument: STOP
.
Plugin is stopping gracefully

.
Plugin stopped forcefully

Appendix C: Error codes and description

REST Messages sent to ServiceNow instance return the below mentioned HTTP response codes.

Table 76.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Message</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Success</td>
<td>Success with response body.</td>
</tr>
<tr>
<td>201</td>
<td>Created</td>
<td>Success with response body.</td>
</tr>
<tr>
<td>204</td>
<td>Success</td>
<td>Success with no response body.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>The request URI does not match the APIs in the system, or the operation failed for unknown reasons. Invalid headers can also cause this error.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized</td>
<td>The user is not authorized to use the API.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
<td>The requested operation is not permitted for the user. This error can also be caused by ACL failures, or business rule or data policy constraints.</td>
</tr>
<tr>
<td>404</td>
<td>Not found</td>
<td>The requested resource was not found. This can be caused by an ACL constraint or if the resource does not exist.</td>
</tr>
<tr>
<td>405</td>
<td>Method not allowed</td>
<td>The HTTP action is not allowed for the requested REST API, or it is not supported by any API.</td>
</tr>
</tbody>
</table>
Table 76. (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Message</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>406</td>
<td>Not acceptable</td>
<td>The endpoint does not support the response format specified in the request Accept header.</td>
</tr>
<tr>
<td>415</td>
<td>Unsupported media type</td>
<td>The endpoint does not support the format of the request body.</td>
</tr>
</tbody>
</table>

Appendix D: Supported CI list

CI name and their supported type:

1. ComputerSystem
   Type:
   - WindowsComputerSystem
   - LinuxUnitaryComputerSystem
   - NetwareUnitaryComputerSystem
   - SunSPARCUnitaryComputerSystem
   - HpUxUnitaryComputerSystem
   - AixUnitaryComputerSystem
   - StorageSubSystem
   - UnitaryComputerSystem
   -VmwareUnitaryComputerSystem

2. AppServer
   Type:
   - SqlServer
   - OracleInstance
   - Db2Instance
   - SybaseServer
   - WebLogicAdminServer
   - WebsphereServer
   - OracleAppHTTPServer
   - VirtualCenter
   - MySql
   - JavaServer
   - KVM
   - IBMTivoliMonitoringAgent
   - WebLogicServer

3. AppServerCluster
   Type:
   - WebLogicCluster

4. ComputerSystemCluster
   Type:
   - VMwareCluster

5. StorageVolume

6. StoragePool
7. DiskDrive
8. FileSystem
   Type:
   - NFSFileSystem
   - SMBFileSystem
   - UnixFileSystem
   - WindowsFileSystem

9. IpInterface
10. L2Interface
11. Function
    Type:
    - IpV4Router
    - IpV6Router
    - IpDevice
    - Bridge

---

Data Access Portal

Fix Pack 6

To access the TADDM Data Access Portal, user needs the viewer role.

Fix Pack 6

User with viewer role can only access the portal.

Creating viewer role

To provide access to users, you can create new viewer role with 'Read only' permission from the TADDM Data Management Portal.

Procedure

Note: Viewer role is default for new TADDM installation; for fix pack migration the new roles must be created from the TADDM data management portal.

To create viewer role, complete the following task:
1. Open the Data Management Portal.
2. Click Administration > Roles. A list of roles is displayed.
3. Click Create Role. The Create Role window is displayed.
4. Type 'viewer' name for the new role, and then select the Read Only permissions.
5. Click Create Role. The list of roles is displayed again with the new role included in the list.

Assigning viewer role

After the new viewer role is created, assign the role to TADDM users that they can access the Data Access Portal.
Procedure

To assign the viewer role to an existing TADDM user, complete the following steps:

1. Open the Data Management Portal.
2. Click Administration > User. A list of users is displayed.
3. Click the user name that you want to edit and then click Edit. The information for the user is displayed.
4. Change the role to 'viewer'.
5. To apply Change Role Assignment properties, click 'Change Role'.

Note: The same steps need to be followed for new users.


Configuring database

By default, Data access portal is configured with TADDM database. Based on the requirements you can change the configured database.

Procedure

To change the database, complete the following steps:

1. Specify the following values in the utility.properties file located at:/opt/IBM/taddm/dist/apps/dap/etc.
   - Database User Name: com.utility.db.user
   - Database User Password: com.utility.db.password
   - JDBC URL of Database to use: com.utility.db.url
   - JDBC Driver Name: The jar file that contains the JDBC driver should be present in libs path of the application com.utility.db.driver
   - Database Type: com.utility.db.type
2. Save the changes.
3. Restart the TADDM.
Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user’s responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan, Ltd.  
1623-14, Shimotsuruma, Yamato-shi  
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.
IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX  78758  U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM’s future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information in softcopy form, the photographs and color illustrations might not be displayed.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at [http://www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).
Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.