Tivoli Application Dependency Discovery Manager
Version 7 Release 2.1

Administrator's Guide

IBM
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Discovered entities with descriptions</td>
</tr>
<tr>
<td>2.</td>
<td>The supported products versions</td>
</tr>
<tr>
<td>3.</td>
<td>Valid graph values and their relationship to the guid parameter</td>
</tr>
<tr>
<td>4.</td>
<td>Operator names of a TADDM MQL query</td>
</tr>
<tr>
<td>5.</td>
<td>State codes</td>
</tr>
<tr>
<td>6.</td>
<td>User tasks with corresponding integration capabilities to use</td>
</tr>
<tr>
<td>7.</td>
<td>Topics that contain more information about discovery using IBM Tivoli Monitoring</td>
</tr>
<tr>
<td>8.</td>
<td>Topics that contain more information about change events</td>
</tr>
<tr>
<td>9.</td>
<td>Topics that contain more information about the self-monitoring tool</td>
</tr>
<tr>
<td>10.</td>
<td>Topics that contain more information about launch in context</td>
</tr>
<tr>
<td>11.</td>
<td>Port configuration</td>
</tr>
<tr>
<td>12.</td>
<td>Communication ports used by the firewall</td>
</tr>
<tr>
<td>13.</td>
<td>Communication ports on the primary storage server used by the firewall</td>
</tr>
<tr>
<td>14.</td>
<td>Communication ports on the discovery server used by the firewall</td>
</tr>
<tr>
<td>15.</td>
<td>SSH keys</td>
</tr>
<tr>
<td>16.</td>
<td>Buffer pool size guidelines</td>
</tr>
<tr>
<td>17.</td>
<td>Buffer pool size guidelines (db_cache_size)</td>
</tr>
</tbody>
</table>
About this information

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

The *IBM Tivoli Application Dependency Discovery Manager Troubleshooting Guide* and the troubleshooting topics in the information center include information on the following items:

- How to identify the source of a software problem
- How to gather diagnostic information, and what information to gather
- Where to get fixes
- Which knowledge bases to search
- How to contact IBM® Support

Conventions used in this information

This information describes the conventions that are used in the IBM Tivoli Application Dependency Discovery Manager (TADDM) documentation for denoting operating system-dependent variables and paths and for denoting the `COLLATION_HOME` directory. It also indicates the location of the `collation.properties` file, which is referenced throughout the TADDM documentation, including in the messages.

**Operating system-dependent variables and paths**

This information uses the UNIX convention 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**

The `COLLATION_HOME` directory is the directory where TADDM is installed plus the `dist` subdirectory.

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

This information contains the terms and definitions for important concepts in the IBM Tivoli Application Dependency Discovery Manager (TADDM).

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
One or more computer programs or software components that provide functionality in direct support of a specific business process or processes.

business service
A group of diverse but interdependent applications and other system resources that interact to accomplish specific business functions.

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.

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>Solaris</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. For more information, see Converting from a synchronization server deployment to 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.
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 1. Discovered entities with descriptions

<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>• J2EE application servers and configurations</td>
</tr>
<tr>
<td></td>
<td>• J2EE and J2SE 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 the IBM Tivoli Application Dependency Discovery Manager (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.

**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**
- `lsof -nP -i` to get the port information
- `ps auxw` 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:
- `GenericServerSensor.log`
- `DiscoverManager.log`
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 db2tcp 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="C1A992327AFF33409C41D5C71046DBB9"
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="B599AED918F436C99FDA0E8EDA578F02"
lastModified="117755771475"
parent="C1A992327AFF33409C41D5C71046DBB9"
xsi:type="coll:com.collation.platform.model.discovery.template.FilterSet">
    <filterList array="1"
guid="BBE4D351653B37E38BFFD2DEBD532EE8"
lastModified="117755771476"
parent="B599AED918F436C99FDA0E8EDA578F02"
xsi:type="coll:com.collation.platform.model.discovery.template.Filter">
      <displayName>unknown</displayName>
      <operand1>db2tcp</operand1>
      <operator>contains</operator>
      <part>Program Name</part>
    </filterList>
    <filterList array="2"
guid="63816C902B0A317F8C3B24C7A1EEBC17"
lastModified="117755771471"
parent="B599AED918F436C99FDA0E8EDA578F02"
xsi:type="coll:com.collation.platform.model.discovery.template.Filter">
      <displayName>unknown</displayName>
      <operand1>db2agent</operand1>
      <operator>contains</operator>
      <part>Program Name</part>
    </filterList>
  </filterSet>
  <boolExp>1</boolExp>
</Template>
```

Administering 5
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
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 http://www.ibm.com/developerworks/wikis/display/tivoliaddm/A+Flexible+Approach+to+Discovery

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_7.1.0/Db2Sensor.xml
• $COLLATION_HOME/osgi/plugins/
  com.ibm.cdb.discover.sensor.app.db.db2windows_7.1.0/Db2WindowsSensor.xml

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 discovery and in script-based discovery, a discovery script that is
provided by the sensor is run against the target system rather than having the
sensor run individual commands.

Not all sensors support asynchronous and script-based discovery. Only sensors that
provide a discovery script can support these types of discovery.

The TADDM Sensor Reference contains information about the asynchronous
discovery sensor and about which sensors support asynchronous and script-based
discovery.
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.

- In asynchronous or script-based discovery, if a problem occurs when processing an application instance, a warning message might be used to inform the user of the problem.
  
  In a nonscript-based discovery, an error message is used.

Asynchronous discovery:

Asynchronous discovery is the running of a discovery script on a target system 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.

The discovery script contains a main script and several sensor scripts. Each sensor script provides a discovery capability that is similar to the function of the sensor during a typical credentialed discovery.

The output of the discovery script is an archive file that contains the discovery results and is stored in a directory on 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 credentialed discovery, it is called “asynchronous”.

The asynchronous discovery sensor is required for asynchronous discovery.

Script-based discovery:

Sensors that provide a sensor script in support of asynchronous discovery can be configured to use that discovery script during a credentialed discovery.

Script-based discovery is the use, in a credentialed discovery, of the same sensor scripts that sensors provide in support of 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 credentialed discovery, you must create an access list entry 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.

You can also use scripts to extend custom server templates.

**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**

Beginning with TADDM 7.2 Fix Pack 4, 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, the 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
 | 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:

```bash
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.

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. 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:**

all normally in seconds 30.367

**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) 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
- 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.

Related reference:
- “Managing credentials cache - cachemgr utility” on page 118
- “Access credentials caching properties” on page 91

You can use the `cachemgr.sh` or `cachemgr.bat` command to list and delete the content of credentials cache.

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.

Fix Pack 4
Exporting data
The process queries the TADDM database to export configuration item information to external systems. For example, integration with Foundation Registry Services is implemented as a topology agent.

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

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>
<thead>
<tr>
<th>Product name</th>
<th>Supported version</th>
</tr>
</thead>
</table>
| IBM Tivoli Monitoring (ITM) | • 6.2.1  
• 6.2.2 FP3  
• 6.2.3  
• 6.3 |
| IBM Tivoli Business Service Manager (TBSM) | • 4.2.1  
• 6.1.0  
• 6.1.1 |
| Change And Configuration Management Database / IBM SmartCloud Control Desk (CCMDB/SCCD) | • 7.2 CCMDB  
• 7.2.1 CCMDB  
• 7.5 SCCD  
• 7.5.1 SCCD |
| IBM Tivoli Network Manager IP / Context Menu Service and Data Integration Service (ITNMIP (MSDIS)) | • 3.8  
• 3.9 |
| Tivoli Workload Scheduler (TWS) | • 8.5.1  
• 8.6 |
| Netcool/OMNIbus | • 7.3  
• 7.3.1.1  
• 7.4 |
| IBM Tivoli Integration Composer (ITIC) | • 7.2.2  
• 7.5 |
Table 2. The supported products versions. (continued)

<table>
<thead>
<tr>
<th>Product name</th>
<th>Supported version</th>
</tr>
</thead>
<tbody>
<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>Jazz for Service Management (JazzSM)</td>
<td>• 1.1</td>
</tr>
<tr>
<td>Netcool/IMPACT</td>
<td>• 6.1.0</td>
</tr>
<tr>
<td></td>
<td>• 6.1.1</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:
./run_cms_dis_registration.sh [ register [guid] | clean [guid [classtype]] | re-register-all | help ]

• Windows systems:
  run_cms_dis_registration.bat [ register [guid] | clean [guid [classtype]] | re-register-all | 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 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.

**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:
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:

```sh
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:

```sh
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.

  - **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 3 on page 22 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:

BA2842345F693855A3165A4B5F0D8BDE

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 3 lists the valid values for the graph parameter and indicates whether the guid parameter is optional or required based on the respective graph value.

### Table 3. 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><strong>business applications</strong></td>
<td>Optional</td>
<td></td>
</tr>
<tr>
<td>applicationinfrastructure</td>
<td>Optional</td>
<td></td>
</tr>
<tr>
<td>physicalinfrastructure</td>
<td>Optional</td>
<td></td>
</tr>
<tr>
<td>For business application objects:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• app_software for Business Application Software Topology</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• app_physical for Business Application Physical Topology</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>For business service objects:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• bus_svc_software for Business Service Software Topology</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>• bus_svc_physical for Business Service Physical Topology</td>
<td></td>
<td></td>
</tr>
<tr>
<td>For collection objects:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• collection_relationship for Collection Relationship Topology</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>• collection_physical for Collection Physical Topology</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3. Valid graph values and their relationship to the guid parameter (continued)

<table>
<thead>
<tr>
<th>Synchronization 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>physicalinfrastructure</td>
<td>Optional</td>
</tr>
<tr>
<td>For business application objects:</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>• app_software for Business Application Software Topology</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• app_physical for Business Application Physical Topology</td>
<td></td>
<td></td>
</tr>
<tr>
<td>For business service objects:</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>• bus_svc_software for Business Service Software Topology</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• bus_svc_physical for Business Service Physical Topology</td>
<td></td>
<td></td>
</tr>
<tr>
<td>For collection objects:</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>• collection_relationship for Collection Relationship Topology</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• collection_physical for Collection Physical Topology</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**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:

• Change History pane
• Details pane
• Discovery Management Console, 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**

http://home.taddm.com:9430/cdm/servlet/LICServlet?username=administrator
&password=adminpwd&guid=BA2B4234563855A3165A0B5F0D8BDE
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

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 Tivoli Netcool/OMNIbus, including the Event Integration Facility (EIF) probe

To see supported versions of the products, go to the “Supported versions” on page 15 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.
   b. Customize the omnibus.eif.properties file. For more information about customizing an EIF property file, see the Configuring support for TADDM events in your integrated environment at http://www-01.ibm.com/support/knowledgecenter/SSSHTQ_7.4.0/com.ibm.netcool_OMNIbus.doc_7.4.0/omnibus/wip/install/task/omn_con_ext_configuringtaddmevents.html?lang=en in the IBM Tivoli Netcool/OMNIbus documentation.

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.

```
<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 28.

**[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>TADDM 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 4. Operator names of a TADDM MQL query.
<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.
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.
(IBM Tivoli Monitoring only) is the port, which the Universal Agent POST Data Provider specified in KUMENV as KUMP_POST_DP_PORT.

(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 at http://www-01.ibm.com/support/knowledgcenter/SSSHQ_7.4.0/com.ibm.netcool_OMNIbus.doc_7.4.0/omnibus/wip/install/concept/omn_con_ext_enablingtaddmevents.html?lang=en in the IBM Tivoli Netcool/OMNIbus documentation. 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 IBM 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:

```plaintext
# 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 TADDEvent_Slot properties property values:

   TADDEvent_Slot_object_name=$TADDM_OBJECT_NAME
   TADDEvent_Slot_change_type=$TADDM_CHANGE_TYPE
   TADDEvent_Slot_change_time=$TADDM_CHANGE_TIME
   TADDEvent_Slot_class_name=$TADDM_CLASS_NAME
   TADDEvent_Slot_attribute_name=$TADDM_ATTRIBUTE_NAME
   TADDEvent_Slot_old_value=$TADDM_OLD_VALUE
   TADDEvent_Slot_new_value=$TADDM_NEW_VALUE
   TADDEvent_Slot_host=$TADDM_HOST
   TADDEvent_Slot_port=$TADDM_PORT
   TADDEvent_Slot_guid=$TADDM_GUID
   TADDEvent_Slot_origin=$TADDM_ORIGIN

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 IBM 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:

If you are running the Universal Agent on a Windows system, 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{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}
```

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 -u 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
```
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.
   b. At the command line, type the following text:

```
validate KUMPOST
```

   Messages similar to the following are displayed:

```
KUMP001I Console input accepted.
KUMP025I Processing input metafile /opt/IBM/ITM//lx8266/um/metafiles/KUMPOST
KUMP026I Processing record 0001 -> //APPl CONFIGCHANGE
  Note: APPL names starting with letters A-M are designated for
  Best Practices and Business Partner UA solutions.
KUMP026I Processing record 0002 -> //NAME dpPost E 3600
KUMP026I Processing record 0003 -> //ATTRIBUTES ';'
KUMP026I Processing record 0004 -> Post_Time T 16 Caption{Time}
KUMP026I Processing record 0005 -> Post_Origin D 32 Caption{Origination}
KUMP026I Processing record 0006 -> Post_Ack_stamp D 28 Caption{Event time stamp}
KUMP026I Processing record 0007 -> Comp_Type D 512 Caption{Component type}
KUMP026I Processing record 0008 -> Comp_Name D 512 Caption{Component name}
KUMP026I Processing record 0009 -> Comp_Guid D 512 Caption{Component GUID}
KUMP026I Processing record 0010 -> Change_Type D 512 Caption{Change type}
KUMP026I Processing record 0011 -> Chg_Det_Time D 512 Caption{Change detection time}
KUMP026I Processing record 0012 -> Chg_Attr D 512 Caption{Changed attribute}
KUMP026I Processing record 0013 -> Srv_Addr D 512 Caption{TADDM server}
KUMP026I Processing record 0014 -> Srv_Port D 16 Caption{TADDM port}
KUMP000I Validation completed successfully
KUMP090I Application metafile validation report saved in file
  /opt/IBM/ITM//lx8266/um/metafiles/KUMPOST.rpt.
```

   c. When prompted for the action you want to perform on the metafile, type the following text:

```
Refresh
```

**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. 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.

- **Optional:** `-P port`
  This value specifies the TADDM server port. The default the value is 9530.

- **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.

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:

   ```plaintext
   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 IBM 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:

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:

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:

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:

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 (TBSM). To use these capabilities, you must have TBSM 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 (TBSM) from TADDM. You can use the BusinessServiceLifecycle program to list information about a business service or to set the lifecycle state of a business service.

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 service 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>
<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 IBM Tivoli Monitoring**

Depending on the specific tasks that you must do in your IT environment, you can use the integration capabilities that are available between the IBM Tivoli Application Dependency Discovery Manager (TADDM) and IBM Tivoli Monitoring.

Table 6 on page 40 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 6. User tasks with corresponding integration capabilities to use

<table>
<thead>
<tr>
<th>Task</th>
<th>Integration capability to use</th>
</tr>
</thead>
</table>
| Gain insight into availability by viewing the operating system settings, application settings, and change history of systems that are monitored by IBM Tivoli Monitoring. | • “Discovery using IBM Tivoli Monitoring”  
• “Launch in context” on page 42                                                                          |
| Ensure that operating systems that are discovered by TADDM are monitored for availability.          | • “Discovery using IBM Tivoli Monitoring”  
• “Monitoring Coverage reports” on page 42                                                                |
| View the availability and performance of systems that are discovered by TADDM.                          | • “IBM Tivoli Monitoring DLA” on page 41  
• “Monitoring Coverage reports” on page 42                                                                 |
| Monitor a business application for configuration changes.                                                 | • “Discovery using IBM Tivoli Monitoring”  
• “Change events” on page 42                                                                                 |
| Monitor the availability of TADDM.                                                                         | • “Self-monitoring tool” on page 42                                                               |

**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 7. 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” on page 123</td>
</tr>
</tbody>
</table>
Table 7. Topics that contain more information about discovery using IBM Tivoli Monitoring (continued)

<table>
<thead>
<tr>
<th>Information</th>
<th>Location of information</th>
</tr>
</thead>
<tbody>
<tr>
<td>TADDM server properties that apply to discovery using IBM Tivoli Monitoring</td>
<td>“Properties for discovery using IBM Tivoli Monitoring” on page 102</td>
</tr>
<tr>
<td>• Sensors that support discovery using IBM Tivoli Monitoring</td>
<td>TADDM Sensor Reference</td>
</tr>
<tr>
<td>• 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</td>
<td></td>
</tr>
</tbody>
</table>

**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 topic about the bulk load program 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:
   ```bash
   $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 the 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 8. Topics that contain more information about change events

<table>
<thead>
<tr>
<th>Information</th>
<th>Location of information</th>
</tr>
</thead>
</table>
| • Configuring TADDM to send change events  
• Configuring an IBM Tivoli Monitoring data provider  
• Configuring change events for a business system | “Sending change events to external systems” on page 24 |

Self-monitoring tool

The TADDM self-monitoring tool provides detailed tracking of performance and availability of the TADDM server and its component processes. The self-monitoring tool is instrumented with IBM Tivoli Monitoring 6.x.

Table 9. Topics that contain more information about the self-monitoring tool

<table>
<thead>
<tr>
<th>Information</th>
<th>Location of information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>“Tracking TADDM server performance and availability” on page 77</td>
</tr>
<tr>
<td>Installation instructions, including troubleshooting the installation of the tool</td>
<td>Installation Guide, “Installing the self-monitoring tool”</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.
### Integrating TADDM with Jazz for Service Management

Fix Pack 4

TADDM supports integration with Open Services for Lifecycle Collaboration (OSLC) platforms. OSLC, when used with TADDM, enables you to get discovery data presented in the form of 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:**

Fix Pack 4

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:**

Fix Pack 4

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.

---

**Table 10. 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>&quot;Configuring for launch in context&quot; on page 20</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>&quot;Creating detail links in configuration change event reports in IBM Tivoli Monitoring&quot; on page 34</td>
</tr>
</tbody>
</table>
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 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: Fix Pack 4

Jazz for Service Management Foundation 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://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: Fix Pack 4

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 61.

By default, security is turned off, so incoming requests might not contain credentials, yet the requested feeds are presented to the user. This is because 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 Foundation Registry Service using OSLCAgent:**

You can use the OSLCAgent topology agent to export configuration item (CI) information to Foundation Registry Service (FRS).

OSLCAgent is an automated solution for exporting data from TADDM to FRS. The agent periodically performs the following tasks:
- Queries for objects that can be registered in FRS
- 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 FRS, you must specify the FRS address and access entry details.

Configure the FRS address in the following property:

```
com.ibm.cdb.topobuilder.integration.oslc.frsurl
```

Specify the FRS address in the following format:

```
protocol://ip_or_hostname:port
```

For example, http://192.0.2.24:9081

Create an access list entry of **Integration/Foundation Registry Service** type.
Specify the user name and password for FRS.
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.frsretrycnt**
If an attempt made by a job to register a CI is rejected by FRS, the attempt is tried again, while time allows. This property specifies the maximum number of times the attempt is tried again.

**com.ibm.cdb.topobuilder.integration.oslc.enablecrtvtype.CRTVType**
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
- ServiceInstance
- SoftwareModule
- SoftwareServer

## Registering configuration items with Foundation Registry Service: Fix Pack 4

This topic lists the configuration items (CIs) discovered by TADDM that are queried for registration in Foundation Registry Service (FRS), 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>crtv:serverAccessPoint</td>
<td>The serviceAccessPoint resource is registered, along with the IpAddress resource to which it points, using crtv:ipAddress.</td>
</tr>
<tr>
<td>version</td>
<td>crtv:version</td>
<td></td>
</tr>
<tr>
<td>vendorName</td>
<td>crtv:manufacturer</td>
<td></td>
</tr>
<tr>
<td>host</td>
<td>crtv:runsOn</td>
<td>crtv:runsOn points to ComputerSystem</td>
</tr>
<tr>
<td>home</td>
<td>crtv:instancePath</td>
<td>For DatabaseServer and Db2Instance only.</td>
</tr>
<tr>
<td>dataPath</td>
<td>crtv:instancePath</td>
<td>For MQQueueManager only.</td>
</tr>
</tbody>
</table>

rdf:type is set to one of the following values:
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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>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>
</tbody>
</table>
| VMID            | crtv:vmid       | If CPUType and Model are set:  
  • For intel, VMID is set to null and an attempt is made to set crtv:systemBoardUUID with systemBoardUUID or convertedUUID.  
  • For power, CS is ignored if it has VMID set. |

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>
</tbody>
</table>
| VMID            | crtv:VMID       | If OSRunning is set to WindowsOperatingSystem, VMID is set to null.  
  If OSRunning is set to Hpiux, VMID, model, and serialNumber are set to null. |
| systemBoardUUID or convertedUUID | crtv:systemBoardUUID | For FCSwitch, TapeLibrary, and TapeMediaChanger only. |
| worldWideName   | crtv:hostid     |                   |

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**

The CRTV ServiceInstance type contains the following TADDM classes and attributes:

• BusinessSystem
  - name
• Application
  - 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>SAPSystemSID:systemHome name</td>
<td>If neither name nor displayName are set.</td>
<td></td>
</tr>
<tr>
<td>parentGUID or NULL crtv:parentServiceInstance</td>
<td>For SAPSystem only.</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: Fix Pack 4
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 FRS, 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 FRS, 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 password
   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 FRS 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 FRS 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, run the following commands:

```
update taddmdb2user.table set oslcregistrationtime_x = null, oslchangeregistrationtime_x = null
update archdb2user.table set oslcregistrationtime_x = null, oslchangeregistrationtime_x = null
```

You must run the update command against the following tables:

- APPSRVR
- COMMSVR
- SAMETIME
- MTGSVR
- SPECSVR
- AGENTMGR
- SROLE
- COMPSYS
- DBTDBASE
- IDSDB
- IMSDB
- ORCLDBASE
- SQLDATABASE
- SYBSDBASE
- DOMINODB
- ITSYSTEM
- COLLECTION
- SOFTMODL
- MQQUEUE
**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**


---

**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.

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.

**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 view and 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 the Edit menu of the Discovery Management Console.

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

When you enable data-level security, you are allowed to view and change information about the configuration items that belong to the access collections to which you have the Read and Update permissions.

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

To enable data-level security, 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. 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 Permission to read objects in assigned access collections.
This is suitable for an operator role.

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

Read + Update + Discover Permission to read and update objects in assigned access collections and to start discovery operations.
This is suitable for a supervisor role.

Read + Update + Admin Permission to read and update objects in assigned access collections and to create users, roles, and permissions.

Read + Update + Discover + Admin All permissions.
This is suitable for an administrator role.
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.

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.
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.

TADDM uses the AES 128 algorithm from the FIPS-compliant IBMJCEFIPS security provider to encrypt the following items:

- Passwords, including entries in collation.properties and userdata.xml.
- Access list entries stored in the database.

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

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

To change the TADDM encryption key, use the bin/changekey.sh script, or equivalent batch script file. This script migrates encrypted entries in collation.properties and userdata.xml, as well as access list entries 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.

Use the changekey script as follows:

```
./changekey.sh $COLLATION_HOME admin_user admin_password
```

For 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, in collation.properties to true.

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 the following types of TADDM discoveries:

- Level 1 discoveries where the TADDM server and discovered systems are any TADDM-supported platforms.
- Level 2 discoveries where the TADDM server is Windows based and Windows Management Instrumentation (WMI) is used to discover Windows platforms.
- Level 1 and Level 2 discoveries through IBM Tivoli Monitoring, using the IBM Tivoli Monitoring Scope sensor, discovering systems managed by IBM Tivoli Monitoring. The TADDM server can run on any supported TADDM platform.

Other TADDM discovery types are not supported.

**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.

### 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
- 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 15 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 supports single sign-on for launch in context between IBM Tivoli CCMDB 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:
   ```properties```
   com.collation.security.usermanagementmodule=vmm
   ```properties```

3. Specify the WebSphere host name and port in the `collation.properties` file. For example:
   ```properties```
   com.collation.security.auth.websphereHost=localhost
   com.collation.security.auth.webspherePort=2809
   ```properties```

   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 uses, the default port is 9809.

4. Specify the WebSphere administrator user name and password in the `collation.properties` file. For example:
   ```properties```
   com.collation.security.auth.VMMAdminUsername=administrator
   com.collation.security.auth.VMMAdminPassword=password
   ```properties```

5. Make the following change to the authentication services configuration file:
   - For the Linux, Solaris, 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.
   ```properties```
   # 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, Solaris, 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 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, Solaris, 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 that allow groups and group memberships to be created and maintained in the Maximo® user and group applications.
When Tivoli CCMDB is configured for this, TADDM uses its own, separate repository from Tivoli CCMDB. Users must be created in both Tivoli CCMDB/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:
   ```
   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 or IBM Tivoli Business Service Manager.

To see supported versions of the products, go to the “Supported versions” on page 15 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), 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:
   ```
   $COLLATION_HOME/external/jdk-1.5.0-Linux-i686/jre/bin/keytool -genkey -alias truststore -keystore truststore.jks
   $COLLATION_HOME/external/jdk-1.5.0-Linux-i686/jre/bin/keytool -import -trustcacerts -alias default -file signer1.cert -keystore truststore.jks
   $COLLATION_HOME/external/jdk-1.5.0-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:
   ```
   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.
   ```
   # This is the URL for the ESS Authentication Service
   #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 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 (authnsvc_ctges) in the Name column. The Enterprise Applications > authnsvc_ctges pane is displayed.
4. In the Enterprise Applications > authnsvc_ctges pane, in the Detailed Properties list, click Security role to user/group mapping. The Enterprise Applications > authnsvc_ctges > Security role to user/group mapping pane is displayed.
5. In the table on the Enterprise Applications > authnsvc_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 > authnsvc_ctges > Security role to user/group mapping > Lookup users or groups pane is displayed.
   - In the Enterprise Applications > authnsvc_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 > authnsvc_ctges > Security role to user/group mapping pane is displayed.
   - In the Enterprise Applications > authnsvc_ctges > Security role to user/group mapping pane, clear the Everyone check box.
   - Click OK. The Enterprise Applications > authnsvc_ctges pane is displayed.
   - Click Save to save the configuration. The Enterprise Applications pane is displayed.
   - Click OK. The Enterprise Applications > authnsvc_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 60.
  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 61.

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

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

**Procedure**

1. Go to the cd `$COLLATION_HOME/etc` directory and set useful parameters for `keytool` and TADDM `sslpassphrase` in the following manner:

   keytool=../external/jdk-Linux-x86_64/bin/keytool
   pass=XXXXXXXX30374

2. Remove self-signed certificate and key from TADDM by running the following commands:

   $keytool -delete -alias collation -noprompt -keystore jsseacerts.cert
   -storepass $pass

   $keytool -delete -alias collation -noprompt -keystore serverkeys -storepass $pass

3. Generate SSL key with the required CN, validity, algorithm..., and save it to the `serverkeys` file. You can run the following command:

   $keytool -genkey -alias collation -keystore serverkeys -validity 3650 -keyAlg RSA
   -sigalg SHA256WithRSA -keypass $pass -storepass $pass -dname "CN=sth-useful-here-eg-FQDN,
   OU=Engineering, OU=NA, o=Citigroup QACA1, L=WARREN, S=NJ, c=US"

4. Generate the certificate signing request (CSR file) by running the following command:

   Administering 67
5. Use the CSR file to get the SSL certificate from official certificate authority (.cert file).

6. Import generated certificate to TADDM, to both key and jsscacerts files by running the following commands:

   $keytool -import -trustcacerts -alias collation -noprompt -keystore serverkeys
   -storepass $pass -keypass $pass -file /tmp/cert.crt

   $keytool -import -trustcacerts -alias collation -noprompt -keystore
   jssecacerts.cert -storepass $pass -keypass $pass -file /tmp/cert.crt

7. Restart the TADDM server.

What to do next

If as the TADDM administrator you have backup of serverkeys file from step 3 and corresponding SSL certificate from step 5, complete the following steps:

1. Repeat the steps 1 and 2.
2. Restore the serverkeys file.
3. Repeat the steps 6 and 7.

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, Linux on System z, and Solaris 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
  - DbInit: Started
  - GigaSpaces: Started
  - Tomcat: Started
  - EcmdbCore: Started
  - TADDM: Running

* domain server
  - Discover: Started
  - GigaSpaces: Started
  - DbInit: Started
  - Tomcat: Started
  - Topology: Started
  - DiscoverAdmin: Started
  - Proxy: Started
  - EventsCore: Started
  - TADDM: Running

**streaming server deployment**

* storage server
  - TADDM: Starting
  - EtaddmCore: Started
  - GigaSpaces: Started
  - DbInit: Started
  - Tomcat: Started
  - TADDM: Running

* discovery server
  - Discover: Started
  - GigaSpaces: Started
  - Tomcat: Started
  - DiscoverAdmin: Started
  - ProxyLite: Started
  - EventsCore: Started
  - TADDM: Running

**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, Solaris, AIX, and Linux on System z operating systems:
     
     ```
     $COLLATION_HOME/bin/control start
     ```
   - For Windows operating systems:
     
     ```
     %COLLATION_HOME%\bin\startServer.bat
     ```

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, Solaris, 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:

```bash
% kill -9 23751
```

After running the command, verify that the process stopped by running the following command:

```bash
% 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:

Save all the files in the directory where you installed the TADDM server.

- For Linux, Solaris, 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.

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:
   ```
   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.
   ```

   Only user created profiles and templates are 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, refer to 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 the correct version of Java installed on the system that they are using to view the Discovery Management Console.
5. Refer users to the TADDM User’s Guide for information about how to install and start the Discovery Management Console.

Configuring TADDM communication

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

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.

Configuring firewalls

To ensure that appropriate communication is established between the Discovery Management Console and the TADDM server and between TADDM servers, you must configure the necessary firewalls.
Configuring a firewall between the Discovery Management Console and the TADDM server:

To have policies that enable communication, you must configure a firewall located between the Discovery Management Console and the TADDM server.

Confirm that the computer running the Discovery Management Console can establish connections to the TADDM server on the configured ports.

Table 11 lists the default ports. If you specified different ports during installation, you must open the ports that you specified.

Table 11. Port configuration

<table>
<thead>
<tr>
<th>Default port</th>
<th>Protocol</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>9430</td>
<td>TCP</td>
<td>Initial Web page and Administrator Console - Non-SSL</td>
</tr>
<tr>
<td>9431</td>
<td>TCP</td>
<td>Initial Web page and Administrator Console over SSL</td>
</tr>
<tr>
<td>9433</td>
<td>TCP</td>
<td>RMI Naming Service</td>
</tr>
<tr>
<td>9434</td>
<td>TCP</td>
<td>Required only for SSL communication</td>
</tr>
<tr>
<td>9435</td>
<td>TCP</td>
<td>Required only for non-SSL communication</td>
</tr>
</tbody>
</table>

If all the default ports and the ports that you specified during installation are open and the Discovery Management Console responds with the message of server is not running when you log in, the TADDM server might be requesting that the client connect to the wrong location. Complete the following steps:

1. Verify that the host identifies itself as its fully qualified host name.
2. Verify that the forward and reverse DNS mappings for the fully-qualified host name match.
   
   If this is not possible or practical, override the host name returned to the client with an IP address. To override the host name returned to the client, change the following property in the collation.properties file, for example:
   
   com.collation.clientproxy.rmi.server.hostname=192.168.253.128
   
   Use the appropriate IP address for your environment.

Configuring a firewall between the synchronization server and the domain server:

In a synchronization server deployment, to ensure that the computer that is running the synchronization server can establish connections to the domain server, you must configure the firewall so that specific ports are open for communication.

Table 12 on page 75 describes the firewall ports you must open on the synchronization server to enable communication between the synchronization server and the domain server.
Table 12. Communication ports used by the firewall

<table>
<thead>
<tr>
<th>Default port</th>
<th>Description</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>4160</td>
<td>The port that is used for communicating unicast discovery information. It is</td>
<td>Outgoing</td>
</tr>
<tr>
<td></td>
<td>the listening port of the TADDM database.</td>
<td></td>
</tr>
<tr>
<td>9430</td>
<td>The port that is used for communicating HTTP information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>9433</td>
<td>The port that is used for communicating naming service information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>9435</td>
<td>The port that is used for communicating RMI information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>9540</td>
<td>The port that is used for communicating security manager information in an</td>
<td>Outgoing</td>
</tr>
<tr>
<td></td>
<td>enterprise environment.</td>
<td></td>
</tr>
<tr>
<td>9550</td>
<td>The port that is used for communicating topology manager information in an</td>
<td>Outgoing</td>
</tr>
<tr>
<td></td>
<td>enterprise environment.</td>
<td></td>
</tr>
<tr>
<td>9560</td>
<td>The port that is used for communicating API server information to remote and</td>
<td>Outgoing</td>
</tr>
<tr>
<td></td>
<td>local services.</td>
<td></td>
</tr>
<tr>
<td>9570</td>
<td>The port that is used for communicating with the synchronization server about</td>
<td>Outgoing</td>
</tr>
<tr>
<td></td>
<td>API server information.</td>
<td></td>
</tr>
<tr>
<td>19430</td>
<td>The port that is used for communicating topology manager information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>19431</td>
<td>The port that is used for communicating change manager information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>19432</td>
<td>The port that is used for communicating API server information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>19433</td>
<td>The port that is used for communicating user registry information.</td>
<td>Incoming</td>
</tr>
<tr>
<td>19434</td>
<td>The port that is used for communicating reports server information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>19435</td>
<td>The port that is used for communicating with the synchronization server.</td>
<td>Outgoing</td>
</tr>
</tbody>
</table>

If you have changed any of the default ports set in `$COLLATION_HOME/etc/collation.properties`, you must ensure that you open ports that are appropriate to how your environment is configured.

The following default port values are set in `$COLLATION_HOME/etc/collation.properties`, on the TADDM server:

- `com.collation.jini.unicastdiscoveryport=4160`
- `com.collation.webport=9430`
- `com.collation.rmiport=9433`
- `com.collation.commport=9435`
- `com.collation.TopologyManager.port=19430`
- `com.collation.ChangeManager.port=19431`
- `com.collation.ApiServer.port=19432`
- `com.collation.SecurityManager.port=19433`
- `com.collation.ReportsServer.port=19434`

In addition, the default synchronization server port value of 19435 is set in `$COLLATION_HOME/external/gigaspaces-4.1/policy/reggie.config`.

```java
import net.jini.jeri.tcp.TcpServerEndpoint;
import net.jini.jeri.BasicJeriExporter;
import net.jini.jeri.BasicILFactory;
```
Configuring a firewall between the primary storage server and the discovery server in a streaming server deployment:

In a streaming server deployment, to ensure that the computer that is running the primary storage server can establish connections to the discovery server, you must configure all firewalls so that specific ports are open for communication.

If you have changed any of the default ports set in $COLLATION_HOME/etc/collation.properties, you must ensure that you open ports that are appropriate to how your environment is configured.

The following default port values are set in $COLLATION_HOME/etc/collation.properties, on the discovery server:

<table>
<thead>
<tr>
<th>Default port</th>
<th>Description</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>4160</td>
<td>The port used for communicating unicast discovery information. It is the listening port of the TADDM database.</td>
<td>Incoming and outgoing</td>
</tr>
<tr>
<td>9430</td>
<td>The port used for communicating HTTP information.</td>
<td>Incoming and outgoing</td>
</tr>
<tr>
<td>9433</td>
<td>The port used for communicating naming service information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>9435</td>
<td>The port used for communicating RMI information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>9540</td>
<td>The port used for communicating security manager information in an enterprise environment.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>9550</td>
<td>The port used for communicating topology manager information in an enterprise environment.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>19430</td>
<td>The port used for communicating topology manager information.</td>
<td>Incoming and outgoing</td>
</tr>
</tbody>
</table>

Configuring a firewall for the primary storage server

If you have a firewall for the primary storage server, Table 13 describes the firewall ports you need to open on the primary storage server to enable communication between the primary storage server and the discovery server.

<table>
<thead>
<tr>
<th>Default port</th>
<th>Description</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>4160</td>
<td>The port used for communicating unicast discovery information. It is the listening port of the TADDM database.</td>
<td>Incoming and outgoing</td>
</tr>
<tr>
<td>9430</td>
<td>The port used for communicating HTTP information.</td>
<td>Incoming and outgoing</td>
</tr>
<tr>
<td>9433</td>
<td>The port used for communicating naming service information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>9435</td>
<td>The port used for communicating RMI information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>9540</td>
<td>The port used for communicating security manager information in an enterprise environment.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>9550</td>
<td>The port used for communicating topology manager information in an enterprise environment.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>19430</td>
<td>The port used for communicating topology manager information.</td>
<td>Incoming and outgoing</td>
</tr>
</tbody>
</table>
Table 13. Communication ports on the primary storage server used by the firewall (continued)

<table>
<thead>
<tr>
<th>Default port</th>
<th>Description</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>19431</td>
<td>The port used for communicating change manager information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>19432</td>
<td>The port used for communicating API server information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>19433</td>
<td>The port used for communicating user registry information.</td>
<td>Incoming and outgoing</td>
</tr>
<tr>
<td>19434</td>
<td>The port used for communicating reports server information.</td>
<td>Outgoing</td>
</tr>
<tr>
<td>19435</td>
<td>The port used for communicating user registry information.</td>
<td>Incoming and outgoing</td>
</tr>
</tbody>
</table>

Configuring a firewall for the discovery server

If you have a firewall for the discovery server, Table 14 describes the firewall ports you need to open on the discovery server to enable communication between the primary storage server and the discovery server.

Table 14. Communication ports on the discovery server used by the firewall

<table>
<thead>
<tr>
<th>Default port</th>
<th>Description</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>4160</td>
<td>The port used for communicating unicast discovery information. It is the listening port of the TADDM database.</td>
<td>Incoming and outgoing</td>
</tr>
<tr>
<td>9430</td>
<td>The port used for communicating HTTP information.</td>
<td>Incoming and outgoing</td>
</tr>
<tr>
<td>9432</td>
<td>The port used for communicating HTTP information.</td>
<td>Incoming and outgoing</td>
</tr>
<tr>
<td>5637</td>
<td>The port used for communicating HTTP information.</td>
<td>Incoming and outgoing</td>
</tr>
</tbody>
</table>

Tracking TADDM server performance and availability

The TADDM self-monitoring tool provides detailed tracking of performance and availability of the TADDM server and its component processes. You can view errors that TADDM has discovered by using the tool with the summaries of configuration item data that is stored in the database. Before you can use this tool, you must have IBM Tivoli Monitoring 6.1, or later, installed in your environment.

The self-monitoring tool is instrumented with IBM Tivoli Monitoring 6.1, which provides visualization of collected availability and performance data. Based on the data provided by the self-monitoring tool, administrators can create reflexive actions to counteract availability degradation.

Viewing availability data
The self-monitoring tool provides information about availability data.
Procedure

To view the availability data provided by the self-monitoring tool, complete the following steps:

1. Start the Tivoli Enterprise Portal Server user interface.
2. Go to the IBM Tivoli Monitoring Navigator Physical view. This is the initial view of the Navigator Physical view:
   - Enterprise
   - Operating Platform
   The Operating Platform is one or more of the following systems:
   - AIX systems
   - Linux systems
   - Solaris systems
   - Windows systems
3. Select the Enterprise icon.
4. Right-click Enterprise, then click Workspace > Tivoli Application Dependency Discovery Manager Availability. The corresponding workspace opens. The workspace includes the following information:
   - A table that provides a time stamp and message abstract for error messages.
   - A table of services with corresponding states.
   - A table of processes that are not operational.
   - A table that provides database status. The table lists the IBM Tivoli Monitoring Version 6.x, database agents that monitor the TADDM database in that host.
5. Click the workspace icon, next to the Status cell, to load the default workspace for the database agent. When you load the workspace, you can see performance metrics for the database.
6. Optional: To sort the columns of the tables in the workspace, click the column header. For example, in the Error Messages table, click the LocalTimeStamp column header to sort the messages by the time stamp. If the messages are already sorted by time stamp, the sort order (ascending or descending) is reversed.

Viewing error messages:

The self-monitoring tool provides error messages that are about system availability.

The table of error messages includes two columns of information:

LocalTimeStamp
   The time at the portal client when the status of a situation changes.

ErrorMessage
   The text of the error message that includes the date and the time associated with the message.

Viewing services:

The self-monitoring tool provides services that are related to system availability.

The services table includes two columns of information:
ServiceName
The name of the service.

State
The state of the service.

Working with non-operational processes:

The self-monitoring tool provides information about non-operational processes.

The table of non-operational processes includes nine columns of information:

- **Status**: The status for the situation:
  - **Acknowledged**: The event is acknowledged.
  - **Deleted**: The situation is deleted.
  - **Problem**: The situation is in a problem state.
  - **Expired**: The acknowledgement of the event is expired and the event remains open.
  - **Opened**: The situation is running and is now true, which opens an event.
  - **Closed**: The situation is running, was true, but is now false.
  - **Reopened**: The acknowledgement was removed before it expired, and the event is still open (reopened).
  - **Started**: The situation is started.
  - **Stopped**: The situation is stopped.

- **Situation name**: The name of the situation or policy.
- **Display item**: If the situation was defined with a display item, it is displayed here. Otherwise, this cell is empty. (A display item is an attribute designated to further qualify a situation. With a display item set for a multiple-row attribute group, the situation continues to look at the other rows in the sampling and opens more events if other rows qualify. The value is displayed in the event workspace and in the message log and situation event console views. You can select a display item when building a situation with a multiple row attribute group.)
- **Source**: The source of the situation.
- **Impact**: The impact of the situation.
- **Opened**: The time stamp indicating when the situation event was opened.
- **Age**: The length of time the situation event has existed.
- **LocalTimeStamp**: The time at the portal client when the status of the situation changes.
- **Type**: The type of event. The possible values are sampled and pure.
Sampled
Sampled events occur when a situation becomes true. Situations sample data at regular intervals. When the situation is true, it causes an event, which gets closed automatically when the situation become false again. You can also close the event manually.

Pure
Pure events are unsolicited notifications. Examples of pure events are an out-of-paper condition on a printer and an occurrence that adds an entry to a log. Some monitoring agents have attributes that report pure events, such as the Windows Event Log and Windows File Change attribute groups. A situation using one of these attributes can monitor for pure events. Because of the nature of pure events, they are not closed automatically like sampled events; you must close the pure event manually. Alternatively, you can create the situation with an UNTIL clause.

The table of non-operational processes is preceded by a row of icons. The following list shows these icons and information about each of them:

- **Filter Critical**
  Click the icon to display only Critical events.

- **Filter Warning**
  Click the icon to display only Warning events.

- **Filter Informational**
  Click the icon to display only Information events.

- **Filter Open**
  Click the icon to display only Open events, including any events that were reopened (acknowledgement removed) or whose acknowledgement expired.

- **Filter Acknowledged**
  Click the icon to display only Acknowledged events.

- **Filter Stopped**
  Click the icon to display only Stopped situations.

- **Filter Problem**
  Click the icon to display only Problem situations. These are situations that are in error for some reason.

- **Console Pause**
  A workspace control you can use to pause the automatic refresh. Click the icon to stop automatic refresh temporarily. You can manually refresh the workspace if you want. Click the Resume Refresh button (.groupby) to refresh the workspace and resume automatic refresh.

Following the list of icons, there is text that provides details about filters that are applied to the contents of the table. For the latest information about changing the filters, refer to the *IBM Tivoli Monitoring User’s Guide*.

**Viewing errors**
The self-monitoring tool provides information about errors that hinder availability and threaten the system health.
About this task

When viewing availability data, there is a table of error messages. To restore the health of the systems, you need to resolve these errors.

Procedure

To view the errors that need to be resolved for TADDM, complete the following steps:

1. Start the Tivoli Enterprise Portal Server user interface.
2. Go to the IBM Tivoli Monitoring Navigator Physical view. This is the initial view of the Navigator Physical view:
   - Enterprise
   - Operating Platform
   The Operating Platform is one or more of the following systems:
   - AIX systems
   - Linux systems
   - Solaris systems
   - Windows systems
3. Select Enterprise.
4. Right-click Enterprise, then click Workspace > Tivoli Application Dependency Discovery Manager Health. The corresponding workspace opens and includes a table of error messages. The table includes the following two columns of information:

   - LocalTimeStamp
     The time at the portal client when the status of the situation changes.

   - Error_message
     The text of the error message that includes the date and the time associated with the message.

Viewing performance data

The self-monitoring tool provides charts and tables with statistics related to resource performance.

Procedure

You can view performance data by performing the following steps:

1. Start the Tivoli Enterprise Portal user interface.
2. Go to the IBM Tivoli Monitoring Navigator Physical view. This is the initial view of the Navigator Physical view:
   - Enterprise
   - Operating Platform
   The Operating Platform is one or more of the following systems:
   - AIX systems
   - Linux systems
   - Solaris systems
   - Windows systems
3. Select Enterprise.
4. Right-click the **Enterprise** item, then click **Workspace > Tivoli Application Dependency Discovery Manager Performance**. The corresponding workspace opens. The workspace includes the following information:
   - A table with a summary of response times
   - A bar chart that graphs current response times
   - A plot chart with response times
   - A table with a summary of situation events

5. Optional: To sort the columns of the tables in the workspace, click the column header. For example, in the response time table, click the **LocalTimeStamp** column header to sort the messages by the time stamp. If the messages are already sorted by time stamp, the sort order (ascending or descending) is reversed.

**Viewing the table summary of response times:**

You can view the table of response times that is provided by the self-monitoring tool.

The table of response times that is provided by the self-monitoring tool includes four columns of information:

- **LocalTimeStamp**
  - The time at the portal client when the status of the situation changes.

- **RealTime**
  - The total amount of time, in seconds, used to complete the transaction.

- **UserTime**
  - The amount of user processing time, in seconds, used to complete the transaction.

- **SystemTime**
  - The amount of system processing time, in seconds, used to complete the transaction.

**Viewing the bar chart of current response times:**

You can view the bar chart of response times that is provided by the self-monitoring tool.

The bar chart with current response time graphs includes the following information:

- **RealTime**
  - The total amount of time, in seconds, used to complete the transaction.

- **UserTime**
  - The amount of user processing time, in seconds, used to complete the transaction.

- **SystemTime**
  - The amount of system processing time, in seconds, used to complete the transaction.

The y-axis of the chart indicates the time in seconds.
Viewing the plot chart of response times:

You can view the plot chart of response times that is provided by the self-monitoring tool.

The plot chart of response time graph includes the following information:

**RealTime**
The total amount of time, in seconds, used to complete the transaction.

**UserTime**
The amount of user processing time, in seconds, used to complete the transaction.

**SystemTime**
The amount of system processing time, in seconds, used to complete the transaction.

You can set the refresh interval of the graph to configure how often the graph refreshes. To set the refresh interval, click View > Refresh, and type the refresh interval, in seconds. The default value for the refresh interval is “On demand”. In this case, you must press F5 to display the most recent changes on the graph.

The x-axis of the chart graphs the time in 10-second increments. For example, if data collection begins at 10:22:35 (hh:mm:ss), there are entries along the x-axis for 10:22:35, 10:22:45, and 10:22:55.

The y-axis of the chart indicates the time in seconds.

Working with the table summary of situation events:

The table summary of situation events provided by the self-monitoring tool includes nine columns of information.

The following list provides details for the columns:

**Status**
The current status for the situation. The following are the possible values:
- Acknowledged: The event is acknowledged.
- Deleted: The situation is deleted.
- Problem: The situation is in a problem state.
- Expired: The acknowledgement of the event is expired and the event remains open.
- Opened: The situation is running and is now true, which opens an event.
- Closed: The situation is running, was true, but is now false.
- Reopened: The acknowledgement was removed before it expired, and the event is still open (reopened).
- Started: The situation is started.
- Stopped: The situation is stopped.

**Situation name**
The name of situation or policy.

**Display item**
If the situation was defined with a display item, it is displayed here. Otherwise, this cell is empty. (A display item is an attribute designated to further qualify a situation. With a display item set for a multiple-row
attribute group, the situation continues to look at the other rows in the sampling and opens more events if other rows qualify. The value is displayed in the event workspace and in the message log and situation event console views. You can select a display item when building a situation with a multiple row attribute group.)

Source
The source of the situation.

Impact
The impact of the situation.

Opened
The time stamp indicating when the situation event was opened.

Age
The length of time the situation event has existed.

LocalTimeStamp
The time at the portal client when the status of the situation changes.

Type
The type of event. The possible values are sampled and pure.

- Sampled
Sampled events occur when a situation becomes true. Situations sample data at regular intervals. When the situation is true, it causes an event, which gets closed automatically when the situation becomes false again. You can also close the event manually.

- Pure
Pure events are unsolicited notifications. Examples of pure events are an out-of-paper condition on a printer and an occurrence that adds an entry to a log. Some monitoring agents have attributes that report pure events, such as the Windows Event Log and Windows File Change attribute groups. A situation using one of these attributes can monitor for pure events. Because of the nature of pure events, they are not closed automatically like sampled events; you must close the pure event manually. Alternatively, you can create the situation with an UNTIL clause.

The table summary of situation events is preceded by a row of icons. The following list shows these icons and information about each of them:

- Filter Critical
  Click the icon to display only Critical events.

- Filter Warning
  Click the icon to display only Warning events.

- Filter Informational
  Click the icon to display only Information events.

- Filter Open
  Click the icon to display only Open events, including any events that were reopened (acknowledgement removed) or whose acknowledgement expired.

- Filter Acknowledged
  Click the icon to display only Acknowledged events.

- Filter Stopped
  Click the icon to display only Stopped situations.
- **Filter Problem**
  Click the icon to display only Problem situations. These are situations that are in error for some reason.

- **Console Pause**
  A workspace control that you can use to pause the automatic refresh. Click the icon to stop automatic refresh temporarily. You can manually refresh the workspace if you want. Click the Resume Refresh button (✓) to refresh the workspace and to resume automatic refresh.

Following the list of icons, there is text that provides details about filters that are applied to the contents of the table. For the latest information about changing the filters, refer to the *IBM Tivoli Monitoring User’s Guide*.

**Viewing the infrastructure data**
You can use the self-monitoring tool to view infrastructure data and statistics.

**About this task**
The self-monitoring tool provides charts and tables with statistics related to the resource infrastructure.

**Procedure**
To view infrastructure data, complete the following steps:
1. Start the Tivoli Enterprise Portal Server user interface.
2. Go to the IBM Tivoli Monitoring Navigator Physical view. This is the initial view of the Navigator Physical view:
   - **Enterprise**
   - **Operating Platform**
   The Operating Platform is one or more of the following systems:
   - AIX systems
   - Linux systems
   - Solaris systems
   - Windows systems
3. Select **Enterprise**.
4. Right-click the **Enterprise** icon, then click one of the following options:
   - For Linux servers: **Workspace > Tivoli Application Dependency Discovery Manager Infrastructure - Linux**
   - For AIX and Solaris servers: **Workspace > Tivoli Application Dependency Discovery Manager Infrastructure - UNIX**

The selected workspace opens. The workspace includes the following information:
- A bar chart that graphs a summary of system memory usage
- Bar charts that graph the memory usage for individual services
- A circular gauge chart that graphs the processing usage for the system
- Circular gauge charts that graph the processing usage for individual services
- A table summary of availability
There are two versions of the TADDM Infrastructure workspace. One version provides information for Linux servers. The other version provides information for AIX and Solaris servers.

5. Optional: To sort the columns of the tables in the workspace, click the column header. For example, in the system availability table, click the **ServiceName** column header to sort the contents by the name of the service. If the messages are already sorted by service name, the sort order (ascending or descending) is reversed.

**Viewing the bar chart that graphs an information summary:**

You can view the bar chart, provided by the self-monitoring tool, that graphs an information summary.

The self-monitoring tool provides a bar chart that graphs a summary of system information. The bar chart with the summary of system information graphs the following items:

**Total Memory (MB)**
- The total amount of memory available for the resource.

**Memory Used (MB)**
- The amount of memory currently used by the resource.

**Memory Free (MB)**
- The amount of memory not currently used and available for use by the resource.

The y-axis of the chart indicates the memory quantity in megabytes.

**Viewing bar charts that graph memory usage for individual services:**

The self-monitoring tool provides a bar chart that graphs the memory usage for services. The bar chart indicates the amount of memory used by each service.

The x-axis of the chart indicates the memory quantity in kilobytes.

**Viewing the circular gauge chart that graphs processor usage:**

The self-monitoring tool provides circular gauge charts that graph the processing usage for the system.

The circular gauge chart shows the proportional amount of processing that is being used. The number displayed below the circular gauge indicates the exact percentage displayed in the chart.

**Viewing circular gauge charts that graph processor usage for individual services:**

The self-monitoring tool provides circular gauge charts that graph the processing usage for individual services.

The circular gauge chart shows the proportional amount of processing that is being used. The number displayed below the circular gauge indicates the exact percentage displayed in the chart.
**Viewing table summary of system availability:**

You can view the table of system availability that is provided by the self-monitoring tool.

The table summary of system availability includes two columns of information:

- **ServiceName**
  - The name of the service.
- **State**
  - The state of the service.

**Viewing configuration items**

You can view the configuration items tracked by the self-monitoring tool.

**Procedure**

To view the configuration items tracked by the self-monitoring tool, complete the following steps:

1. Start the Tivoli Enterprise Portal Server user interface.
2. Go to the IBM Tivoli Monitoring Navigator Physical view. This is the initial view of the Navigator Physical view:
   - Enterprise
   - Operating Platform
   The Operating Platform is one or more of the following systems:
   - AIX systems
   - Linux systems
   - Solaris systems
   - Windows systems
3. Click the **Enterprise** icon.
4. Right-click the **Enterprise** icon, then click **Workspace > Tivoli Application Dependency Discovery Manager Configuration Items** The workspace opens.
   - The workspace includes the following information:
     - A bar chart that provides the total number of configuration item changes in the past week
     - A bar chart that provides the totals for major configuration items, including, system items, application items, network items, and storage items
     - A plot chart that displays trends over time and among configuration items
     - A table with a summary of situation events
5. Optional: To sort the columns of the tables in the workspace, click the column header. For example, in the Situation Events Console table, click the **LocalTimestamp** column header to sort the messages by the time stamp. If the messages are already sorted by time stamp, the sort order (ascending or descending) is reversed.

**Viewing total number of configuration item changes in the past week:**

You can view the total number of configuration item changes in the past week.

The bar chart indicates the total number of configuration item changes over the last seven days. The x-axis of the chart indicates the quantity.
**Viewing totals for configuration items:**

The bar chart provides a summary of configuration items totals.

The bar chart graphs the following items:
- Total items
- System items
- Application items
- Network items
- Storage items

The y-axis of the chart indicates the quantity of items.

**Viewing plot chart of configuration items:**

The x-axis of the chart indicates the time in 10-second increments.

You can set the refresh interval of the graph to configure how often the graph refreshes. To set the refresh interval, click **View > Refresh**, and type the refresh interval, in seconds. The default value for the refresh interval is “On demand”. In this case, you must press F5 to display the most recent changes on the graph.

For example, if data collection begins at 10:22:35 (hh:mm:ss), there are entries along the x-axis for 10:22:35, 10:22:45, and 10:22:55. The y-axis of the chart indicates the quantity of items.

**Working with the table summary of situation events:**

The table summary of situation events provided by the self-monitoring tool includes nine columns of information.

The following list provides details for the columns:

**Status**  
The current status for the situation. The following are the possible values:
- Acknowledged: The event is acknowledged.
- Deleted: The situation is deleted.
- Problem: The situation is in a problem state.
- Expired: The acknowledgement of the event is expired and the event remains open.
- Opened: The situation is running and is now true, which opens an event.
- Closed: The situation is running, was true, but is now false.
- Reopened: The acknowledgement was removed before it expired, and the event is still open (reopened).
- Started: The situation is started.
- Stopped: The situation is stopped.

**Situation name**  
The name of situation or policy.

**Display item**  
If the situation was defined with a display item, it is displayed here. Otherwise, this cell is empty. (A display item is an attribute designated to further qualify a situation. With a display item set for a multiple-row
attribute group, the situation continues to look at the other rows in the sampling and opens more events if other rows qualify. The value is displayed in the event workspace and in the message log and situation event console views. You can select a display item when building a situation with a multiple row attribute group.)

Source
The source of the situation.

Impact
The impact of the situation.

Opened
The time stamp indicating when the situation event was opened.

Age
The length of time the situation event has existed.

LocalTimeStamp
The time at the portal client when the status of the situation changes.

Type
The type of event. The possible values are sampled and pure.

Sampled
Sampled events occur when a situation becomes true. Situations sample data at regular intervals. When the situation is true, it causes an event, which gets closed automatically when the situation becomes false again. You can also close the event manually.

Pure
Pure events are unsolicited notifications. Examples of pure events are an out-of-paper condition on a printer and an occurrence that adds an entry to a log. Some monitoring agents have attributes that report pure events, such as the Windows Event Log and Windows File Change attribute groups. A situation using one of these attributes can monitor for pure events. Because of the nature of pure events, they are not closed automatically like sampled events; you must close the pure event manually. Alternatively, you can create the situation with an UNTIL clause.

The table summary of situation events is preceded by a row of icons. The following list shows these icons and information about each of them:

- Filter Critical
  Click the icon to display only Critical events.

- Filter Warning
  Click the icon to display only Warning events.

- Filter Informational
  Click the icon to display only Information events.

- Filter Open
  Click the icon to display only Open events, including any events that were reopened (acknowledgement removed) or whose acknowledgement expired.

- Filter Acknowledged
  Click the icon to display only Acknowledged events.

- Filter Stopped
  Click the icon to display only Stopped situations.
- **Filter Problem**
  Click the icon to display only Problem situations. These are situations that are in error for some reason.

- **Console Pause**
  A workspace control that you can use to pause the automatic refresh. Click the icon to stop automatic refresh temporarily. You can manually refresh the workspace if you want. Click the Resume Refresh button (-modal) to refresh the workspace and to resume automatic refresh.

Following the list of icons, there is text that provides details about filters that are applied to the contents of the table. For the latest information about changing the filters, refer to the *IBM Tivoli Monitoring User’s Guide*.

**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.

**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.ignorepropertyscopes`
- `com.collation.discover.agent.command.lsof.Linux`

However, the `com.collation.discover.agent.command.lsof.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.lsof.Linux.129.42.56.212=sudo lsof
- Example of appending a scope set named “scope1”:
  com.collation.discover.agent.command.lsof.Linux.scope1=sudo lsof

**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=true
  The default value is true.
  This property determines whether the auth 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:
  com.ibm.cdb.security.auth.cache.fallback.failed=false
  com.ibm.cdb.security.auth.cache.fallback.failed.10.160.160.11=true
  com.ibm.cdb.security.auth.cache.fallback.failed.ScopeA=true
  com.ibm.cdb.security.auth.cache.fallback.failed.GroupA=true
  com.ibm.cdb.security.auth.cache.fallback.failed.Level_2_Discovery=false
  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.

### Related concepts

<table>
<thead>
<tr>
<th>“Last successful credentials caching” on page 13</th>
</tr>
</thead>
<tbody>
<tr>
<td>TADDM can cache last working access credentials. They can be reused in the next (Level 2) discovery.</td>
</tr>
</tbody>
</table>

### API port properties

These properties apply to API ports.

- **com.collation.api.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.
  - This property is different from the `com.collation.ApiServer.port` property, which is the firewall port that is used by the API server.

- **com.collation.api.ssl.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.

#### 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.topomgr.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

**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:

- `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: 100.
  - 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.

**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=sudo /optVRTSvcs/bin/hastatus
com.collation.discover.agent.command.haclus.Linux=sudo /optVRTSvcs/bin/haclus
com.collation.discover.agent.command.hasys.Linux=sudo /optVRTSvcs/bin/hasys
com.collation.discover.agent.command.hares.Linux=sudo /optVRTSvcs/bin/hares
com.collation.discover.agent.command.hagrp.Linux=sudo /optVRTSvcs/bin/hagrp
com.collation.discover.agent.command.hatype.Linux=sudo /optVRTSvcs/bin/hatype
com.collation.discover.agent.command.hauser.Linux=sudo /optVRTSvcs/bin/hauser
```

• These properties are required to discover Veritas Cluster components.
• To execute these commands without 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.

```
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:

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

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

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 9:

These properties are required to discover manufacturer, model, and serial number on Linux systems.

This property can be used to discover a guest user ID on a target Linux virtual system running on a z/VM® operating system.

This property is required to discover configuration information for a Check Point firewall on Solaris systems.

These properties are required to discover advanced network interface information (interface speed, for example).

This property is required to discover processor information about HP systems.
This property is required to discover kernel modules on HP systems.

This property is required to discover partition table information about Solaris systems.

These properties are required to discover storage volume information.

This property is required to discover Emulex fibre channel HBA information on Solaris systems.

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:

These properties are required to discover file systems.

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.

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

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.

com.ibm.cdb.DisCmsIntegration.dbUser=user
This property specifies the database user ID for the Context Menu Service and Data Integration Service database.

com.ibm.cdb.DisCmsIntegration.dbPassword=password
This property specifies the password for the Context Menu Service and Data Integration Service database user.

com.ibm.cdb.DisCmsIntegration.dbUrl=url
This property specifies the database URL for the Context Menu Service and Data Integration Service database.

com.ibm.cdb.DisCmsIntegration.dbDriver=driver
This property specifies the database driver for the Context Menu Service and Data Integration Service.

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.

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

com.collation.db.archive.password=password
This property specifies the database password, which is stored on the TADDM server, for the database archive 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.bin 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.collation.ChangeManager.port=19431
The default value is 19431.
This property specifies the firewall port that is used by the change manager.

**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

**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 be 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 default value is 600000 (in milliseconds), which is 10 minutes.
This property specifies the default timeout for sensors in milliseconds. The default timeout can be changed, and it can be specified by individual sensors.
To override the timeout for a particular sensor, add the following line to the collation.properties file:
```properties
com.collation.discover.agent.sensorNameSensor.timeout= timeInMilliseconds
```
Here is an example:
```properties
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.

```java
com.collation.number.persist.discovery.run=10
```

The default value is 10.

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.

```java
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
Specifications 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.

Limitation: Because script-based discovery uses only a Level 3 discovery profile, rediscovery does not support script-based discovery.

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, the storage server, or both of these servers.

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.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=\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\tar
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.
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 TADDM server, depending on the operating system:
For AIX:
  com.ibm.cdb.tarpath=\t\t\t\t\t\t\t\t\t\t\tar
For Solaris:
  com.ibm.cdb.tarpath=/usr/sfw/bin/gtar

com.ibm.cdb.targettarpath=\t\t\t\t\t\t\t\t\t\tar
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:

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

For Solaris:

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

Script-based discovery properties:

These properties apply to script-based discovery.

**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.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.

Fix Pack 3

**com.ibm.cdb.taddm.asd.prefix=**

With this property, you can add a prefix to a script that is used by the script-based discovery. For example, you can specify sudo.

Properties for discovery using IBM Tivoli Monitoring:

These properties apply to discovery using IBM Tivoli Monitoring.

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 and Level 3 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 IBM Tivoli Monitoring endpoints.

To connect directly to an IBM 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:

```plaintext
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 IBM 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 IBM Tivoli Monitoring endpoints.

This property specifies whether TADDM uses IBM Tivoli Monitoring as the preferred method for discovering IBM 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 IBM Tivoli Monitoring endpoints:

```
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 IBM 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 IBM 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 IBM 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 IBM 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.

**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, name lookups (for example, JAVA and DNS) 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.cacheTTLSuccessfulNameLookups=60**
  The default value is 60.
  This command is specified in the java.security file to indicate the caching policy for successful name lookups from the name service. The value is specified as an integer to indicate the number of seconds to cache the successful lookup. A value of 0 means "never cache". A value of -1 means "cache forever".

- **com.collation.platform.os.cacheTTLUnsuccessfulNameLookups=60**
  The default value is 60.
  This command is specified in the java.security file to indicate the caching policy for unsuccessful name lookups from the name service. The value is
specified as an integer to indicate the number of seconds to cache the failure for an unsuccessful lookup. A value of 0 means "never cache". A value of -1 means "cache forever".

```java
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, enable 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, Linux, and SunOS operating systems are supported. This property is not supported on a Windows TADDM server.

### GUI JVM memory properties

These properties apply to GUI JVM memory.

```java
com.collation.gui.initial.heap.size=128m
```

The default value is `128m`. Initial heap size for the TADDM user interface.

```java
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:

```java
com.collation.gui.initial.heap.size=256m
```

The default value is `256m`.

```java
com.collation.gui.max.heap.size=768m
```

The default value is `768m`.

For a large environment:

```java
com.collation.gui.initial.heap.size=512m
```
The default value is 512m.

```java
com.collation.gui.max.heap.size=1024m
```

The default value is 1024m.

**GUI port properties**

These properties apply to GUI ports.

```java
com.collation.tomcatshutdownport=9436
```

The default value is 9436.

This port is used for the Tomcat shutdown command.

```java
com.collation.webport=9430
```

The default value is 9430.

The HTTP port is used without SSL.

```java
com.collation.websslport=9431
```

The default value is 9431.

The HTTPS port is used with SSL.

```java
com.collation.commport=9435
```

The default value is 9435.

The RMI data port to use without SSL.

```java
com.collation.commsslport=9434
```

The default value is 9434.

The RMI data port to use with SSL.

```java
com.collation.rmiport=9433
```

The default value is 9433.

The naming service RMI registry port.

**Jini properties**

These properties apply to Jini.

```java
com.collation.jini.rmidport=1098
```

The default value is 1098. The value must be an integer.

```java
com.collation.jini.unicastdiscoveryport=4160
```

The default value is 4160. The value must be an integer.

**Fix Pack 3**

```java
com.collation.jini.service.call.timeout=
```

Setting the custom timeout for the communication between TADDM internal workstations (Proxy, Topology). Specify the time value in milliseconds. By default, the value is 2100000 milliseconds (35 minutes).

**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.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.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. This is the default value.
- ldap for an LDAP user registry
- vmm for a user registry that uses the federated repositories of WebSphere Application Server

**com.collation.security.auth.ldapAuthenticationEnabled=true**

The default value is true.

This property defines whether LDAP authentication has been enabled.

**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.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.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.ldapUIDNamingAttribute=cn**

The default value is cn.

This property defines the name of the attribute used for naming users in LDAP.
com.collation.security.auth.ldapGroupObjectClass=groupofuniquenames
  The default value is groupofuniquenames.
  This property defines the class used to represent user groups 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.ldapGroupMemberAttribute=uniqueMember
  The default value is uniquemember.
  This property defines the name of the attribute used to contain the members of a group in LDAP.

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.

Logging properties
These properties apply to logging.

com.collation.log.filesize=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
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
- EcmdbCore
- 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 performance.

Modify the thread count value

This product is tuned for a 2 processor, 4 GB system. If you have more processors or memory, you can change the thread properties in the collation.properties file. No formula exists for determining the optimal values for these thread counts. Optimal values depend on how sparse your subnets are and how the firewalls are configured. Here are a few general guidelines:

1. If all of the processors are not saturated during a discovery run, you can increase the dwcount by a few threads and try again. If you increase this thread count when your processors are saturated, the sensors start to timeout and you do not receive complete discover results.

2. If you increase the dwcount by a few threads and continue not to get increased processor usage, bump the sccount by 1 and try again.

3. Do not change the ascount.

The following list identifies extra details for the properties for performance:

com.collation.discover.dwcount=16
The default value is 16. The value must be an integer.
A discover worker thread is a thread that runs sensors. This property defines how many discover worker threads can be running simultaneously.
com.collation.discover.observer.topopumpcount=16
   The default is value 16. The value must be an integer.
   This property specifies the number of database "writer" threads that are 
   created. These threads are used for persisting discovery results into the 
   TADDM database.

Modify the refresh interval

The following entry can be added to the collation.properties file to alter the 
refresh interval when carrying out analytical tasks.
com.ibm.cdb.typesServiceRefreshInterval=120
   The default value is 120.
   This property specifies, in seconds, the refresh interval to update 
   component types when creating a custom query, displaying a change 
   history, or displaying component comparison information. The minimum 
   value is 30 and the maximum value is 1800 seconds. Add this entry to the 
collation.properties file to change the default value.

Reporting properties
These properties apply to reporting.
com.collation.ReportsServer.port=19434
   The default value is 19434.
   Specifies the firewall port used by the reports server.

Secure Shell (SSH) properties
These properties apply to the Secure Shell (SSH).
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 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 should 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 have 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.
**Security properties**
These properties apply to security.

**com.collation.security.privatetruststore=true**  
The default value is true.  
Valid values are true or false.

**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 a lot of 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 hosting 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.collation.SecurityManager.port=19433**  
The default value is 19433.  
Specifies the firewall port used by the security manager.
The Analytics panel can be restricted to a specific role. By default, this property is not defined in the `collation.properties` file and the Analytics panel is available for everyone. The value of the property must be the name of the role that is allowed to access the panel.

The access to the following areas of the Analytics panel can be subject to the specified role:
- Inventory Summary
- Application Summary
- Service Summary
- System Inventory
- Software Server Inventory
- BIRT Reports

**Startup properties**
These properties apply to the startup process.

You can use these properties to determine whether TADDM monitors server startup.

If enabled, the restartwatcher will monitor the TADDM startup process so that the TADDM server will restart if it does not start within a certain defined time frame. The restartwatcher checks for the following conditions:
- All but one of the services has reached started state
- One or more of the services is in stopped state and at least one service has started

If these conditions persist for a configurable period of time, a restart is requested. The restartwatcher turns off after the server has started so that it will not interfere with the service restart configuration options.

```java
com.collation.platform.jini.restartwatcher.enabled=false
```

The default value is `false`. This is a boolean property that determines whether TADDM will monitor server startup.

```java
com.collation.platform.jini.restartwatcher.delay=240
```

The default value is 240. This property has an integer value that determines the number of seconds to wait after a restart condition is detected before restarting the server. The minimum value is 60. Slower systems should use longer delay times to allow for longer service start times.

Fix Pack 3

```java
com.collation.Proxy.jvmargs.ibm=-Xss8192k
com.collation.StorageService.jvmargs.ibm=-Xss8192k
```

Fix Pack 3

```java
com.collation.Proxy.jvmargs.sun=-Xss8192k
com.collation.StorageService.jvmargs.sun=-Xss8192k
```

TADDM Java virtual machines use 8192k for the stack for proxy machine and storage service.

**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, the name of the scope, or the 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\%

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 . . .

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.

**Fix Pack 3**

`com.collation.topobuilder.RuntimeGcThreadCount=`
The default value is 4.
This property adds parallelism to the RuntimeGC agent which can improve performance

**Topology manager properties**
These properties apply to the topology manager.

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.collation.TopologyManager.port=19430`
The default value is 19430.
Specifies the firewall port used by the topology manager.

**View manager properties**
These properties apply to the view manager.

`com.collation.gui.doNotShowGraphs=Application Infrastructure, Physical Infrastructure`
The default value is Application Infrastructure, Physical Infrastructure, which means that only the Business Applications graph is shown.

This property applies to the display of graphs for business applications, application infrastructure, and physical infrastructure. The following information describes the default behavior for this property in TADDM 7.2.1:
- This property is deprecated. Therefore, by default, it is not shown in the `collation.properties` file.
- The default behavior is that the Business Applications graph is shown, but the Application Infrastructure and Physical Infrastructure graphs are not shown.
- The Application Infrastructure and Physical Infrastructure graphs are deprecated.

Any combination of the following values are valid, but if you use more than one value, use a comma to separate the entries:
- Business Applications
- Application Infrastructure
- Physical Infrastructure

The following examples further illustrate the use of this property:

`com.collation.gui.doNotShowGraphs=Application Infrastructure`
The Business Applications and Physical Infrastructure graphs are shown, but the Application Infrastructure graph is not shown.
com.collation.gui.doNotShowGraphs=
All three graphs are shown.

Because graphs are not pregenerated, they might take time to be generated.

com.collation.view.maxnodes=500
The default value is 500. The value must be an integer.

This property specifies the maximum number of nodes that can be viewed in a topology graph in the Data Management Portal. If the value is greater than 500, the graphs for business applications, application infrastructure, and physical infrastructure are not supported.

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.

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.

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 TADDDM 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:

```properties
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:

```properties
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:SI` 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 credentials 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: 3B954CE4CFBF346CBDF538F09F1F7FFD
  lastModified: Thursday, 5 September 2013 11:00:38
  Authorization: invalid. Error message: CTJTP1190E The server did not complete
  the authorization process.
  CachedAuthEntry
  guid: ACC2F35A66D3379BAC13FC606C5A0BA3
  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:

  cachemgr.sh -u user -p password -r invalid -s 9.123.149.10-9.123.149.12
  -c 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.
4         Some error occurred but it is unknown. Go to the log directory and open
          the cachemgr.log to look for more information.
5         The provided user does not have enough privileges (discovery) to perform
          operation.
6         There were no entries in database that match provided criteria.
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 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.

**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.

**Procedure**

1. To generate a discovery script package, enter the following command from the $COLLATION_HOME/bin directory:

   ```
   makeSSDScriptPackage 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, or SunOS.

   - **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.

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 examples show how to specify the path of the tar command on the TADDM server, depending on the operating system:

For AIX:
    com.ibm.cdb.tarpath=tar

For Solaris:
    com.ibm.cdb.tarpath=/usr/sfw/bin/gtar

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

For AIX:
    com.ibm.cdb.targettarpath.AIX=tar

For Solaris:
    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:
    ./makeASDScriptPackage /tmp AIX

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 root user, grant execute privileges to all script files. If the discovery script is run as a non-root 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.

5. Move the resulting archive file (for example, /tmp/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 sensor (ASDSensor) is enabled in your discovery profile. By default, the sensor 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.

**Restriction:** Your tar program must support long file paths. GNU Tar 1.13 is not supported because it might truncate long file names.

**Configuring for script-based discovery**
To run a script-based discovery, you must first configure the discovery.

**Procedure**

Configure the sensor in one of the following ways:
To globally enable all sensors that support script-based discovery, set the value of the `com.ibm.cdb.discover.PreferScriptDiscovery` property to `true`.

For a specific discovery profile, to enable all sensors that support script-based discovery, set the value of the `com.ibm.cdb.discover.PreferScriptDiscovery` property to `true` in the platform properties for that 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.

**Configuring for 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.

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.

**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” on page 102.

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:

```java
com.ibm.cdb.session.allow.ITYM.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:

```java
com.ibm.cdb.session.allow.ITYM.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.discover.DefaultAgentTimeout=3600000
  com.collation.platform.session.ITMSessionNumProgressChecks=3600
  ```

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” on page 39
- "Configuring for discovery using IBM Tivoli Monitoring” on page 123
- 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 136.

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 http://www.ibm.com/developerworks/wikis/display/tivoliaddm/Home.

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 15](#).

<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>
<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:
   ```bash
   $ 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 **1ssyscfg** commands

- **Logical Partition**
  - Needed to use the **lshwres**, **1ssyscfg**, 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 Sun 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 TADDM Sensor Reference in the “Sensor overview” topic for the IBM® Lotus® Domino® server sensor.

*Enabling a JBoss system for discovery:*

You must complete two steps to enable a JBoss system for discovery.

**Procedure**

To enable a JBoss system for discovery, complete the following steps:

1. Copy the **jbossall-client.jar** and **jboss-jmx.jar** files from a JBoss distribution to the \$COLLATION_HOME\lib\jboss\402 directory.

2. From the Discovery Management Console, create a JBoss 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 **Application Servers**.
   d. In the **Vendor** field, click **JBoss**.
e. Type the credentials for the JBoss Server. The credentials (ID and password) must match the ID and password used to access JBoss on the appropriate target being discovered.

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 server
# 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 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 the Microsoft Exchange server:**

You must configure the Microsoft Exchange server that you want the TADDM server to discover.

**About this task**

To discover the Microsoft Exchange Server, the Microsoft Exchange Management service must be running on the target Windows system. The Windows service ID for the TADDM service account must be created on the Windows system on which the Microsoft Exchange server is running. The Windows service ID must have full permission (Execute Methods, Full Write, Partial Write, Provider Write, Enable Account, Remote Enable, Read Security, and Edit Security) to the following WMI namespaces:

- Root\CIMV2
- Root\CIMV2\Applications\Exchange
- Root\MicrosoftExchangeV2

If the Windows service ID for the TADDM service account has sufficient permissions to discover a Microsoft Exchange server, the sensor uses the Windows service ID and a separate Microsoft Exchange server access list entry is not required.

If the Windows service ID for the TADDM service account does not have sufficient permissions to discover a Microsoft Exchange server, you must create a separate Microsoft Exchange server access list.

**Procedure**

To configure the access list, complete the following steps:

1. From the Discovery Management Console, create a discovery scope set that contains your Microsoft Exchange Server, or use an existing scope that contains your Microsoft Exchange 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 **Messaging Servers**.
5. In the **Vendor** field, click **Microsoft Exchange Server**.
6. Type the credentials for the Microsoft Exchange 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 http://communities.vmware.com/home.jspa

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. 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:
   
   ```
   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.
   
   ```
   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.

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.

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 are easier than configuring SSH. WMI is available by default on all Windows target systems that are supported by TADDM. 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, direct discovery using SSH requires the .NET 2.0 Framework on each Windows target. .NET Framework 2.0 is not installed by default on Windows Server 2000.

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`

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. Requires a dedicated Windows Server computer system to serve as the gateway. Operating system requirements for gateway servers are the same as Windows operating system requirements for TADDM servers.
See the Installation Guide, “TADDM server software requirement” for details about supported Windows operating systems.

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 discovery 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 SSH-based discovery**

1. You must install a supported version of an SSH server on each Windows target computer system.

2. You must install .NET Framework 2.0 or higher on each Windows Server target computer system.

**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 [Windows gateways](http://www.bitvise.com/).

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.
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:

```
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:

```
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, Microsoft Exchange 2007 Server sensor. Complete the following steps:

- Configure the domain user account by running the following commands:
  ```
  mkpasswd -u [domain_user] -d [domain] >> /etc/passwd
  mkgroup -d [domain] >> /etc/group
  ```
- 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.
- 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:
  ```
  $ chown [domain_user] /var/log/sshd.log
  $ chown [-R [domain_user]] /var/empty
  $ chown [domain_user] /etc/ssh*
  ```
- 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

**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 Windows gateways.

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/)

**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`

See the TADDM Sensor Reference for more information about the TADDM server properties that the Windows computer system sensor uses.

Windows-based discoveries might fail due to unclear reasons that are related to WMI. You can use the WMI Tester, also known as WBEMTest, to navigate the WMI classes on the command line and determine any possible user permission or WMI corruption issues. The `wbemtest.exe` file is in the WBEM directory of your Windows system directory. For more information, see WMI Tester overview.

### 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 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.
- 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 to true:

```
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.
**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:

```properties
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:

```
anchor_host_1=192.168.1.13
anchor_scope_1=FIRST_SCOPE
anchor_zone_1=FIRST_ZONE
anchor_location_1=FIRST_LOCATION
anchor_host_2=192.168.2.22
anchor_scope_2=SECOND_SCOPE
anchor_location_2=SECOND_LOCATION
```

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:

```
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:

```
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.

**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.

**Tuning**

To maximize TADDM performance, you might want to complete additional configuration steps.

**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=800

  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 800, and the maximum cache size value is 2000.
    com.ibm.cdb.bulk.allocpoolsiz=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.

  You can collect thread dumps of TADDM processes and review them to make sure that a Java virtual machine is not running out of memory. If necessary, increase the memory size to 2 GB (2048).

**Database tuning**

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.


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 (this is the Java time stamp)
```

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);
            System.out.println("Milliseconds is " + msec);
        } catch (ParseException e)
    }
```
Run the code as follows:
```java
java DateToString 1/11/2008
Date is Sat Nov 01 00:00:00 EST 2008
Milliseconds 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:
```sql
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 IBM 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 tuning**

These tuning guidelines apply to IBM DB2 databases. You should perform tuning and maintenance tasks when there is little or no database activity expected.
**REORG and RUNSTATS utilities**

Run the REORG and RUNSTATS utilities, in that order, on the database tables. After the database is populated, do this task on a regularly scheduled basis (for example, weekly).

**REORG**

After many changes to table data caused by the insertion, deletion, and updating of variable length columns activity, logically sequential data might be on non-sequential physical data pages. This result in the database manager performing additional read operations to access data. Reorganize DB2 tables to eliminate fragmentation and reclaim space by using the REORG command.

To generate all of the REORG TABLE commands that you must run, run the following SQL statement on the DB2 database server, where `dbuser` is the value from `com.collation.db.user`:

```sql
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;
```

To run this procedure, complete the following steps:
1. Copy the preceding SQL statement to a file, for example, `temp.sql`.
2. On the database server, on a DB2 command line, connect to the DB and run the following commands:
   ```
db2 –x –tf temp.sql > cmdbreorg.sql
   db2 –tvf cmdbreorg.sql > cmdbreorg.out
   ```

**RUNSTATS**

The DB2 optimizer uses information and statistics in the DB2 catalog to determine the best access to the database, 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, run the RUNSTATS command again to update the statistics.

Ensure that your TADDM database tables are populated before running the RUNSTATS command on the database. This can occur by way of discovery, bulk load, or by using the API. Running the RUNSTATS command on your database tables before there is data in them results in the catalog statistics reflecting 0 rows in the tables. This generally causes the DB2 optimizer to perform table scans when accessing the tables, and to not use the available indexes, resulting in poor performance.

The DB2 product provides functions to automate database maintenance by using database configuration parameters. You must evaluate the use of these parameters in your environment to determine if they fit into your database maintenance process. In a typical production environment, you want to control when database maintenance activities occur. Database maintenance activities are typically performed during off-peak hours to prevent major problems with the database.

The following list describes some of the database configuration parameters:

**Automatic maintenance (AUTO_MAINT)**

This parameter is the parent of all the other automatic maintenance database configuration parameters (`auto_db_backup`).
auto_tbl_maint, auto_runstats, auto_stats_prof, auto_prof_upd, and auto_reorg). When this parameter is disabled, all of its child parameters are also disabled, but their settings, as recorded in the database configuration file, do not change. When this parent parameter is enabled, recorded values for its child parameters take effect. In this way, automatic maintenance can be enabled or disabled globally.

The default value for DB2 V8 is OFF.

The default value for DB2 V9 is ON.

To set this parameter to OFF, use the following command:

```
UPDATE db cfg for dbname using AUTO_MAINT OFF
```

**Automatic table maintenance (AUTO_TBL_MAINT)**

This parameter is the parent of all table maintenance parameters (auto_runstats, auto_stats_prof, auto_prof_upd, and auto_reorg). When this parameter is disabled, all of its child parameters are also disabled, but their settings, as recorded in the database configuration file, do not change. When this parent parameter is enabled, recorded values for its child parameters take effect. In this way, table maintenance can be enabled or disabled globally.

To set this parameter to OFF, use the following command:

```
UPDATE db cfg for dbname using AUTO_TBL_MAINT OFF
```

**Automatic runstats (AUTO_RUNSTATS)**

This automated table maintenance parameter enables or disables automatic table runstats operations for a database. A runstats policy (a defined set of rules or guidelines) can be used to specify the automated behavior. To be enabled, this parameter must be set to ON, and its parent parameters must also be enabled.

To set this parameter to OFF, use the following command:

```
UPDATE db cfg for dbname using AUTO_RUNSTATS OFF
```

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. You do not have to run the program every time you execute the `RUNSTATS` command. It is enough to run it after you install a new fix pack or a new release that changes the schema of the TADDM database. The following example shows how the program is used:

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 and run the following command:
   ```
   db2 -tvf tmpdir/TADDM_table_stats.sql > tmpdir/TADDM_table_stats.out
   ```

There is an additional performance fix that is used to modify some of the statistics that are generated by the `RUNSTATS` command. There is a program in the `$COLLATION_HOME/bin` directory called `db2updatestats.sh` (for UNIX
and Linux systems), or $db2updatestats.bat (for Windows systems). The following example shows how the program is used:

1. Run the following command:
   
   ```
cd $COLLATION_HOME/bin
   ```

2. Run the following command:
   
   ```
./db2updatestats.sh
   ```

You must run $db2updatestats.sh immediately after running the RUNSTATS command. If TADDM queries are run in the time after the RUNSTATS command completes, and before $db2updatestats.sh starts, the queries might choose an access path that is not efficient. This might cause an increase in locking.

**Buffer pool**

A buffer pool is memory used to cache table and index data pages as they are being read from disk, or being modified. The buffer pool improves database system performance by allowing data to be accessed from memory instead of from disk. Memory access is much faster than disk access. The less often the database manager must read from or write to a disk, the better the performance. Because most data manipulation takes place in buffer pools, configuring buffer pools is the single most important tuning area. Only large objects and long field data are not manipulated in a buffer pool.

Modify the buffer pool sizes based on the amount of available system memory that you have and the amount of data that is in your database. The default buffer pool sizes provided with the TADDM database are generally not large enough for production environments. There is no definitive answer to the question of how much memory you can dedicate to the buffer pool. Generally, more memory is better. Because it is a memory resource, its use must be considered along with all other applications and processes that are running on a server. Use the DB2 SNAPSHOT monitor to determine buffer pool usage and hit ratios. If an increase to the size of the buffer pools causes system paging, lower the size to eliminate paging.

The appropriate buffer pool size depends on the number of configuration items (CIs) in your database. To count the number of CIs, in your database, run the Healthcheck program with the getPersObjTableInfo module. For more information about using the Healthcheck program, see the “Using the Healthcheck program” topic in the Troubleshooting Guide.

<table>
<thead>
<tr>
<th>Number of CIs</th>
<th>Available system memory</th>
<th>Guideline buffer pool size</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 500,000</td>
<td>4K</td>
<td>50,000</td>
</tr>
<tr>
<td></td>
<td>8K</td>
<td>5,000</td>
</tr>
<tr>
<td></td>
<td>32K</td>
<td>1,000</td>
</tr>
<tr>
<td>500,000 - 1,000,000</td>
<td>4K</td>
<td>90,000</td>
</tr>
<tr>
<td></td>
<td>8K</td>
<td>12,000</td>
</tr>
<tr>
<td></td>
<td>32K</td>
<td>1,500</td>
</tr>
<tr>
<td>&gt; 1,000,000</td>
<td>4K</td>
<td>150,000</td>
</tr>
<tr>
<td></td>
<td>8K</td>
<td>24,000</td>
</tr>
<tr>
<td></td>
<td>32K</td>
<td>2,500</td>
</tr>
</tbody>
</table>
For example, if your database has more than one million CIs, you can implement the following buffer pool changes:

- ALTER BUFFERPOOL IBMDEFAULTBP SIZE 150000
- ALTER BUFFERPOOL BUF8K SIZE 24000
- ALTER BUFFERPOOL BUF32K SIZE 2500

This change might require a database restart.

**Database configuration parameters**

The following list includes important DB2 database configuration parameters that you can adjust, depending on data volumes, usage, and deployment configuration:

- **DBHEAP**
- **NUM_IOCLEANERS**
- **NUM_IOSERVERS**
- **LOCKLIST**
- **MAXAPPLS** This parameter specifies the maximum number of concurrent applications that can be connected (both local and remote) to a database. Set this parameter to automatic and DB2 dynamically allocates the resources required to support new applications.

**Database manager parameters**

The following list includes important DB2 database manager parameters that you can adjust, depending on data volumes, usage, and deployment configuration:

- **ASLHEAPSZ**
- **INTRA_PARALLEL**
- **QUERY_HEAP_SZ**
- **RQRIOBLK**

**Database registry variables**

Set the following DB2 registry variables:

- **DB2_PARALLEL_IO** This property enables parallel I/O operations. This property is applicable only if your table space containers and hardware are configured appropriately.
- **DB2NTNOCACHE=ON** This property is applicable on Windows systems only.
- **DB2_USE_ALTERNATE_PAGE_CLEANING**
- **DB2_EVALUNCOMMITTED=YES**

**Database logs**

You can optimize database logging by completing the following activities:

- Tune the log file size (`logfilsiz`) database configuration parameter so that you are not creating excessive log files.
- Use Log Retain logging to ensure recoverability of your database.
- Mirror your log files to ensure availability of your database system.
- Modify the size of the database configuration Log Buffer parameter (`logbufsz`) based on the volume of activity. This parameter specifies the amount of the database heap to use as a buffer for log records before writing these records to
disk. Buffering the log records results in more efficient logging file I/O. The log records are written to disk less frequently, and more log records are written at a time.

**Database prefetch size**

Modify the PREFETCHSIZE on the table spaces based on the following formula. An ideal size is a multiple of the extent size, the number of physical disks under each container (if a RAID device is used) and the number of table space containers. The extent size can be fairly small, with a good value being in the range of 8 - 32 pages. For example, for a table space on a RAID device with five physical disks, one container (suggested for RAID devices) and an EXTENTSIZE of 32, the PREFETCHSIZE can be set to 160 (32 x 5 x 1).

**DB2 for z/OS database tuning**

These tuning guidelines apply to IBM DB2 for z/OS® databases.

**Procedure**

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 RUNSTATS control statement for each new database. The following example assumes that the databases are named CMDBA and CMDBB:

   ```sql
   SELECT DISTINCT 'RUNSTATS TABLESPACE '||DBNAME||'.'||TSNAME||' INDEX(ALL)
   SHRLEVEL REFERENCE' FROM SYSIBM.SYSTABLES
   WHERE DBNAME IN ('CMDBA', 'CMDBB') ORDER BY 1;
   ```

4. Immediately after RUNSTATS is complete, generate and run the UPDATE control statement for each new database. Run the following statements only from a DB2 command-line window for the schemas corresponding to both the primary user ID and the archive user ID:

   ```sql
   select 'UPDATE SYSIBM.SYSINDEXES SET FIRSTKEYCARDF=FULLKEYCARDF WHERE'
   ||chr(10)||
   ' NAME ='''||CAST(RTRIM(name) AS VARCHAR(40))||''''
   ||chr(10)||
   ' AND CREATOR ='''||CAST(RTRIM(creator) AS VARCHAR(40))||''''
   ||chr(10)||
   ' AND TBNAME ='''||CAST(RTRIM(tbname) AS VARCHAR(40))||''''
   ||chr(10)||
   ' AND TBCREATOR ='''||CAST(RTRIM(tbcreator) AS VARCHAR(40))||'''';
   from sysibm.sysindexes a
   where tbcreator = 'SYSADM'
   AND NAME IN
   (SELECT IXNAME
    FROM SYSIBM.SYSKEYS B
    WHERE A.CREATOR = B.IXCREATOR
    AND A.NAME = B.IXNAME
    AND COLNAME = 'PK__JDOIDX');

   select 'UPDATE SYSIBM.SYSCOLUMNS SET COLCARDF=(SELECT FULLKEYCARDF FROM'
   ||chr(10)||
   ' SYSIBM.SYSINDEXES WHERE')||chr(10)||
   ' NAME ='''||CAST(RTRIM(name) AS VARCHAR(40))||''''
   ||chr(10)||
   ' AND CREATOR ='''||CAST(RTRIM(creator) AS VARCHAR(40))||''''
   ||chr(10)||
   ' AND TBNAME ='''||CAST(RTRIM(tbname) AS VARCHAR(40))||''''
   ||chr(10)||
   ' AND TBCREATOR ='''||CAST(RTRIM(tbcreator) AS VARCHAR(40))||'''')
   ||chr(10)||
WHERE NAME = 'PK__JDOIDX'
  AND TBNAME = CAST(RTRIM(tbname) AS VARCHAR(40))
  AND TBCREATOR = CAST(RTRIM(tbcreator) AS VARCHAR(40))
from sysibm.sysindexes a
where tbcreator = 'SYSADM'
AND NAME IN
(SELECT IXNAME
FROM SYSIBM.SYSKEYS B
WHERE A.CREATOR = B.IXCREATOR
  AND A.NAME = B.IXNAME
  AND COLNAME = 'PK__JDOIDX');

where SYSADM is the schema name corresponding to the primary or archive user ID. Then run the resulting UPDATE SYSIBM.SYSINDEXES and UPDATE SYSIBM.SYSCOLUMNS statements for each schema.

5. Regularly monitor the size of the TADDM database tables, and adjust their storage attributes if necessary. In particular, monitor the size of the following database tables, which can become very large:
   - CHANGE_HISTORY_TABLE
   - CMDB_GUID_ALIAS
   - PERSOBJ
   - RELATION
   - SFTCMP
   - MEDACDEV
   - WINSVC
   - MSSOBJLINK
   - BINDADDR
   - OPSYS
   - OPERATINGSYSENTS_FD67DE48X
   - COMPSYS
   - COMPOSITE
   - MSSOBJLINK_REL
   - SOFTMODL
   - RUNTIMEPROCESSJDO_PORTS_X
   - COMPUTERSYSTICES_E032D816X
   - APPSRVR
   - IPINTRFC
   - ORCLINITV
   - RUNTIMEPROCEORTS_13B7EE75X
   - IPROUTE
   - IPADDR

Use ALTER statements to modify the PRIQTY and SECQTY attributes according to the needs of your environment. If appropriate, consider moving tables to separate tablespaces.

6. Use the REBIND command on the following packages with the KEEP_DYNAMIC(YES) option:
   - SYSLH200
   - SYSLH201
   - SYSLH202

**Oracle database tuning**

These 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:

  ```bash
  ./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:

  - To execute a script file in SQLPlus, type `@` and then the file name: SQL
    ```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 17. Buffer pool size guidelines (db_cache_size)

<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>
<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>

Discovery performance tuning

You can update the com.collation.discover.dwcount and com.collation.discover.observer.topopumpcount properties in the collation.properties file to influence the discovery rate and the rate at which discovery results are stored in the TADDM database.

For details about these properties, see “Performance properties” on page 109.

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:
- 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 153.


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:

```
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:
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:
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:
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 be editing the following property:
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. Storage of the discovery results is the
performance bottleneck if the number of sensors in the storing stage is
approximately the value of the property that specifies the number of parallel
storage threads.

The following property specifies the number of parallel storage threads. It is one of
the main settings for controlling the discovery storage performance. You must,
however, adjust it carefully.
com.collation.discover.observer.topopumpcount

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
- Discover: -Xk3700
- Gigaspaces: -Xk3000

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, the $COLLATION_HOME/deploy-tomcat/ROOT/WEB-INF/cmdb-context.xml file.
- For a synchronization server, the $COLLATION_HOME/deploy-tomcat/ROOT/WEB-INF/ecmdb-context.xml file.
- For a discovery server, the $COLLATION_HOME/deploy-tomcat/ROOT/WEB-INF/discovery-server-context.xml file.
- For a storage server, the $COLLATION_HOME/deploy-tomcat/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.jini.ServiceLifecycle"initmethod="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>

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

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 CMDB USING
  UTIL_HEAP_SZ 5000
  LOGBUFSZ 1024
  LOCKLIST 20000
  SORTHEAP 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.
**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.

© Copyright IBM Corp. 2006, 2014
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.