Copyright

Copyright © 2000 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

This software and documentation is subject to and made available only pursuant to the terms of the BEA Systems License Agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the agreement. This document may not, in whole or in part, be copied photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from BEA Systems, Inc.

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the BEA Systems License Agreement and in subparagraph (c)(1) of the Commercial Computer Software-Restricted Rights Clause at FAR 52.227-19; subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, subparagraph (d) of the Commercial Computer Software--Licensing clause at NASA FAR supplement 16-52.227-86; or their equivalent.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA Systems DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks or Service Marks


All other company names may be trademarks of the respective companies with which they are associated.

**BEA SNMP Agent MIB Reference for BEA Tuxedo and BEA WebLogic Enterprise**

<table>
<thead>
<tr>
<th>Document Edition</th>
<th>Date</th>
<th>Software Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>October 2000</td>
<td>BEA SNMP Agent 2.1</td>
</tr>
</tbody>
</table>
Contents

About This Document

What You Need to Know ................................................................. vi
e-docs Web Site ............................................................................... vi
How to Print the Document ............................................................ vi
Related Information ........................................................................ vii
Contact Us! .................................................................................... vii
Documentation Conventions ....................................................... viii

1. The Tuxedo MIB for SNMP

MIB Object Identifiers ................................................................. 1-1
ASN.1 File .................................................................................... 1-4
Relative and Absolute Object Identifiers .................................... 1-4
Specifying Object Identifiers ....................................................... 1-4
Updating MIB Objects ................................................................. 1-5
Supported MIB Objects ................................................................. 1-5
Managing Tuxedo and WLE Applications Using the Tuxedo MIB for SNMP 1-7
Querying Non-Existent MIB Objects ............................................ 1-7
Structure of the MIB Definitions .................................................. 1-8
MIB Event Trap Definitions Format .............................................. 1-8
Differences Between the Tuxedo MIB and the Tuxedo MIB for SNMP .... 1-9

2. Tuxedo Core MIB

tuxTserverCtxTbl ........................................................................ 2-159
3. Domains MIB

4. BEA Domain List MIB

5. WLE MIBs

6. Access Control List MIB

7. Workstation MIB
   tuxTwshTbl........................................................................................................ 7-2

8. Application Queue MIB

9. Event Broker MIB

10. Tuxedo Traps MIB
    Specific Trap Number ..................................................................................... 10-1
    Variable Bindings ............................................................................................ 10-2
    Trap Definitions............................................................................................... 10-5
    DOMAIN Traps................................................................................................. 10-6
    BRIDGE Traps ................................................................................................. 10-12
    SERVER Event Traps ...................................................................................... 10-14
    CLIENT Traps.................................................................................................... 10-18
    TRANSACTION Traps...................................................................................... 10-20
    EVENT Traps..................................................................................................... 10-21

Index
About This Document

The BEA SNMP Agent MIB Reference for BEA Tuxedo and BEA WebLogic Enterprise provides reference information about the MIBs shipped in the BEA SNMP Agent software. This guide is organized as follows:

- Chapter 1, “The Tuxedo MIB for SNMP,” describes MIB Object Identifiers, structure of the MIB definitions, the difference between the Tuxedo MIB and the SNMP Tuxedo MIB, how to manage Tuxedo and WLE applications using the SNMP MIB, and how to manage Tuxedo and WLE resources from a management framework.

- Chapter 2, “Tuxedo Core MIB,” describes the basic groups, objects, and attributes of the objects that form a Tuxedo application, and that are defined in the Core MIB.

- Chapter 3, “Domains MIB,” describes the interaction among domains using improved groups and attributes terminology.

- Chapter 4, “BEA Domain List MIB,” describes the BEA domain list.

- Chapter 5, “WLE MIBs,” describes the MIB groups and member objects specific to WLE applications, and lists the WLE-specific objects that are included as part of the Tuxedo Core MIB.

- Chapter 6, “Access Control List MIB,” describes the access control list (ACL) MIB groups.

- Chapter 7, “Workstation MIB,” describes the workstation MIB groups.

- Chapter 8, “Application Queue MIB,” describes the application queue MIB groups.

- Chapter 9, “Event Broker MIB,” describes the event broker MIB.

- Chapter 10, “Tuxedo Traps MIB,” describes the Tuxedo event traps MIB.
What You Need to Know

This document is intended for network or system administrators who are responsible for administering SNMP master agents and SMUX subagents.

e-docs Web Site

BEA product documentation is available on the BEA corporate Web site. From the BEA Home page, click on Product Documentation or go directly to the “e-docs” Product Documentation page at http://e-docs.beasys.com.

How to Print the Document

You can print a copy of this document from a Web browser, one file at a time, by using the File—>Print option on your Web browser.

A PDF version of this document is available on the BEA SNMP Agent documentation Home page on the e-docs Web site. You can open the PDF in Adobe Acrobat Reader and print the entire document (or a portion of it) in book format. To access the PDFs, open the BEA SNMP Agent documentation Home page, click the PDF files link and select the document you want to print.

If you do not have the Adobe Acrobat Reader, you can get it for free from the Adobe Web site at http://www.adobe.com/ or by contacting your local Adobe sales office.
Related Information

The following BEA SNMP Agent documents contain additional information that is relevant to using the BEA SNMP Agent MIB Reference:

- **BEA SNMP Agent Installation Guide for BEA Tuxedo and BEA WebLogic Enterprise**
- **BEA SNMP Agent Administrator's Guide for BEA Tuxedo and BEA WebLogic Enterprise**
- **BEA SNMP Agent Release Notes for BEA Tuxedo and BEA WebLogic Enterprise**

Contact Us!

Your feedback on the BEA BEA SNMP Agent documentation is important to us. Send us e-mail at docsupport@beasys.com if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the BEA SNMP Agent documentation.

In your e-mail message, please indicate that you are using the documentation for the BEA SNMP Agent 2.1 release.

If you have any questions about this version of BEA SNMP Agent, or if you have problems installing and running BEA SNMP Agent, contact BEA Customer Support through BEA WebSupport at www.beasys.com. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

## Documentation Conventions

The following documentation conventions are used throughout this document.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Item</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface text</strong></td>
<td>Indicates terms defined in the glossary.</td>
</tr>
<tr>
<td>Ctrl+Tab</td>
<td>Indicates that you must press two or more keys simultaneously.</td>
</tr>
<tr>
<td><em>italics</em></td>
<td>Indicates emphasis or book titles.</td>
</tr>
</tbody>
</table>
| **monospace text**| Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.  
**Examples:**  
#include <iostream.h> void main ( ) the pointer psz  
chmod u+w *  
\tux\data\ap  
.doc  
tux.doc  
BITMAP  
float |
| **monospace boldface text** | Identifies significant words in code.  
**Example:**  
void **commit** ( ) |
| **monospace italic text** | Identifies variables in code.  
**Example:**  
String **expr** |
<table>
<thead>
<tr>
<th>Convention</th>
<th>Item</th>
</tr>
</thead>
</table>
| **UPPERCASE TEXT** | Indicates device names, environment variables, and logical operators.  
*Examples:*  
LPT1  
SIGNON  
OR |
| { } | Indicates a set of choices in a syntax line. The braces themselves should never be typed. |
| [ ] | Indicates optional items in a syntax line. The brackets themselves should never be typed.  
*Example:*  
buildobjclient [-v] [-o name ] [-f file-list]...  
[-l file-list]... |
| | Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed. |
| ... | Indicates one of the following in a command line:  
- That an argument can be repeated several times in a command line  
- That the statement omits additional optional arguments  
- That you can enter additional parameters, values, or other information  
The ellipsis itself should never be typed.  
*Example:*  
buildobjclient [-v] [-o name ] [-f file-list]...  
[-l file-list]... |
| . | Indicates the omission of items from a code example or from a syntax line.  
The vertical ellipsis itself should never be typed. |
This chapter describes the Tuxedo MIB for SNMP. It includes the following sections:

- MIB Object Identifiers
- Managing Tuxedo and WLE Applications Using the Tuxedo MIB for SNMP
- Structure of the MIB Definitions
- Differences Between the Tuxedo MIB and the Tuxedo MIB for SNMP

**MIB Object Identifiers**

An SNMP management framework manages a collection of objects in a hierarchy of information known as a Management Information Base (MIB). Each object in the MIB has an object identifier (OID), which the manager uses to request the object’s value from the agent.

To use the BEA SNMP Agent Integrator polling feature described in the *BEA SNMP Agent Administrator’s Guide*, OIDs are used to identify the managed objects whose values are retrieved by the BEA SNMP Agent Integrator when it checks for the occurrence of events in the managed resource.
The Tuxedo MIB for SNMP

An OID is a sequence of integers that uniquely identifies a managed object by defining a path to that object through a tree-like structure called the OID tree or registration tree. When an SNMP agent needs to access a specific managed object, it traverses the OID tree to find the object. The MIB object identifier hierarchy and format is shown in Figure 1-1.
In this hierarchy, each BEA private MIB object that the BEA SNMP Agent software manages has a unique object identifier. The BEA Tuxedo object uses a prefix of .1.3.6.1.4.1.140 to identify it as an object in the BEA private MIB.

For a complete listing of objects in the BEA private MIB in ASN.1 notation, read the file bea.asn1 in the installed BEA SNMP Agent software.
ASN.1 File

An ASN.1 file is a standard SNMP file that defines the objects that make up an SNMP-compliant MIB. Each object in the file is defined in compliance with the SNMP standard. The BEA SNMP Agent provides the ASN.1 file `bea.asn1` for defining the Tuxedo MIB (with WebLogic Enterprise extensions) for SNMP.

**Note:** The Tuxedo and WLE MIB definitions are written in concise MIB format in accordance with RFC 1212, as required by the SNMP standard.

Relative and Absolute Object Identifiers

*Absolute OIDs* specify a path to an attribute from the root of the OID tree. Absolute OID names always begin with a dot and must specify every node of the OID tree from the top-most node to the specific managed object. For example:

```
.1.3.6.1.2.1.1.1
```

*Relative OIDs* specify a path to the attribute relative to some node in the OID tree. For example, `2.1.1.7` specifies the `sysContact` object in the `system` group, relative to the Internet node in the OID tree.

Specifying Object Identifiers

In addition to using the “dot-dot” notation, a series of integers separated by dots to describe OIDs, you can also express OIDs by using textual symbols instead of numbers to represent nodes in the path to the object, or by using a combination of both integers and textual symbols. A *symbolic* OID uses mnemonic keywords to specify the managed object. For example:

`mgmt.mib-2.system.sysDescr`

The following numeric OID uses integers to specify the same managed object:

```
2.1.1.1
```

Note that this example is a relative OID.
MIB Object Identifiers

An OID can combine both symbolic and numeric representations of individual nodes of the OID tree; for example:

```
mgmt.mib-2.1.sysDescr
```

**Note:** When using OIDs to specify objects whose values are checked using BEA SNMP Agent Integrator polling rules, only the numeric form of the OID can be used. For details, see Chapter 8 of the *BEA SNMP Agent Administrator’s Guide*.

### Updating MIB Objects

Some objects in the BEA SNMP MIB for Tuxedo systems can be set (updated) only under certain states of the Tuxedo system. If you get an error while trying to set read-write objects in this MIB, please refer to the ULOG file.

### Supported MIB Objects

To access MIB objects that are managed by agents or subagents, the scope of the OID tree for which each agent or subagent is responsible must be defined to the BEA SNMP Agent Integrator. For monolithic SNMP agents, and SMUX or DPI master agents, this is done by specifying an OID in one or more `NON_SMUX_PEER` entries in the `beamgr.conf` configuration file, as described in Chapter 5 of the *BEA SNMP Agent Administrator’s Guide*. The BEA SNMP Agent Integrator then knows to access the managed objects in that branch of the OID tree through the specified agent.

The BEA SNMP Agent Integrator *directly* accesses MIB objects in the SMUX MIB, the MIB II system and `snmp` groups, and the `beaIntAgtTable` MIB object in the BEA SNMP Agent MIB. The `beaIntAgtTable` MIB objects define the polling capability of the BEA SNMP Agent Integrator.

The `bea.asnl` file contains definitions for Tuxedo and WLE objects that are SNMP compliant. The BEA SNMP Agent supports the following MIBs:

- **Tuxedo Core MIB**—Supplies the definitions for controlling the operation and configuration of the Tuxedo system. This MIB contains the main information groups for Tuxedo applications, including domains, machines, queues, servers, routing, clients, and services. See Chapter 2, “Tuxedo Core MIB.”
The Tuxedo MIB for SNMP

- **Domains MIB**—Uses improved group and attribute terminology to describe the interaction between domains. This improved terminology has also been applied to DMCONFIG file syntax. See Chapter 3, “Domains MIB.”

- **BEA Domain List MIB**—Contains information about the WLE or Tuxedo domains that the agent is monitoring. See Chapter 4, “BEA Domain List MIB.”

- **WLE MIBs**—An extension of the core MIB that incorporates definitions for management of WLE CORBA and JAVA features. See Chapter 5, “WLE MIBs.”

- **Access Control List MIB**—Enables you to define and control your application security options. See Chapter 6, “Access Control List MIB.”

- **Workstation MIB**—Specifies information about Tuxedo client workstations including workstation listeners and handlers. See Chapter 7, “Workstation MIB.”

- **Application Queue MIB**—Provides the administrative control required for managing access to application queues. The objects in this MIB include items for managing queue spaces, queues, messages, and transactions. See Chapter 8, “Application Queue MIB.”

- **Event Broker MIB**—Represents event subscriptions registered with the Event Broker for receiving event notifications. See Chapter 9, “Event Broker MIB.”

- **Tuxedo Traps MIB**—Defines all the trap notifications that are generated by the BEA SNMP Agent and the objects passed in the variable bindings for these traps. See Chapter 10, “Tuxedo Traps MIB.”

Tuxedo managed object names within the MIBs for SNMP are usually prefixed with the letters **tux**. WLE managed object names are usually prefixed with **wle**. For example, the Tuxedo Core MIB contains a group termed Machine. Within this group are managed objects such as the following:

- **tuxTmachinePmid**
  - Represents a physical machine identifier

- **tuxTmachineLmid**
  - Represents the logical machine identifier

Most Tuxedo objects also apply to WLE applications.
Managing Tuxedo and WLE Applications Using the Tuxedo MIB for SNMP

The Tuxedo and WLE systems identify application items in a hierarchy of information known as Tuxedo Management Information Bases. These databases contain definitions that describe the components found in the Tuxedo or WLE application. Included with the BEA SNMP Agent is an SNMP version of the Tuxedo MIBs. The Tuxedo MIB for SNMP also includes MIB objects that represent attributes of Tuxedo and WLE resources.

To monitor or modify values of managed objects through your systems management platform, you need to know which MIB objects represent the features of Tuxedo or WLE resources that are relevant to your management goals. You also need to know the data types, default values, and access permissions for these MIB objects.

Querying Non-Existent MIB Objects

If you attempt to retrieve the value of a MIB object, and that object does not exist, either no value is returned, or one of the following values is returned:

- -1 if the object is numeric
- A dash (-) if the object data type is DisplayString

Using the Tuxedo MIB for SNMP, the BEA SNMP Agent Integrator can be configured to perform local polling and generate SNMP trap events, or execute a system command when certain conditions are met. The same effect can be achieved by defining a RULE_ACTION entry in the beamgr.conf file. This configuration file is described in Chapter 8, “Configuration Files,” in the BEA SNMP Agent Administrator’s Guide.
The following keywords are used to define MIB managed objects:

Syntax

Defines the abstract data structure corresponding to that object type. The ASN.1 language is used for this purpose. However, the SMI purposely restricts the ASN.1 constructs that can be used. These restrictions are made expressly for simplicity.

Access

Defines whether the object value can only be retrieved but not modified (read-only) or whether it can also be modified (read-write).

Note: For tabular objects, in some cases a read-write object can only be set during creation of a new row. Where this is true, it is noted in the Description section for that MIB object.

Description

Contains a textual definition of that object type that provides all semantic definitions necessary for interpretation. This clause typically contains information of the sort that would be communicated in any ASN.1 commentary annotations associated with the object.

Note: Each row in a table is an instance of the Entry object under that table. The Description section for the Entry object under a table (such as tuxTmachineTable) contains information on the columnar values that are minimally necessary for creation of a row—how a new row is created, whether the values pertain only to the local machine, and other pertinent information about the table objects.

MIB Event Trap Definitions Format

In addition to defining MIBs, the bea.asn1 file also defines traps. These traps are defined in accordance with RFC 1215, Trap definitions. Chapter 10, “Tuxedo Traps MIB,” specifies the list of traps generated by the BEA SNMP Agent. The following keywords are used to define a trap:
Differences Between the Tuxedo MIB and the Tuxedo MIB for SNMP

Enterprise

An object identifier that specifies the management enterprise under whose registration authority this trap is defined. All the traps generated by the BEA SNMP Agent have an enterprise field set to the Tuxedo object identifier. The Tuxedo object identifier is .1.3.6.1.4.1.140.300. This value is passed in the enterprise field of the trap packet (Protocol Data Unit — PDU).

Variables

Defines the ordered sequence of MIB objects that are contained in each instance of the trap type. Each variable is placed, in order, inside the variable-bindings field of the SNMP trap packet (PDU).

Description

Contains a textual definition of the trap type.

Trap ID

Specifies the enterprise-specific trap ID for the trap definition. This is passed in the specific trap ID field of the trap packet (PDU).

Note: The value of the generic trap ID field in traps is always set to 6, indicating an enterprise-specific trap.

Differences Between the Tuxedo MIB and the Tuxedo MIB for SNMP

If you are familiar with the Tuxedo MIB, the primary differences to note when using the Tuxedo MIB for SNMP are a difference in terms and a few additional MIB items in the SNMP-based MIB.

The Tuxedo MIB identifies an abstract structure for Tuxedo resources. In a Tuxedo framework, a MIB is the classification of information in a Tuxedo application. However, instead of referring to groups and managed objects, as is common in SNMP terminology, the Tuxedo MIB defines application resources as classes and attributes.
The Tuxedo MIB for SNMP

Classes are the administrative class definitions that make up the Tuxedo MIB. Each class has a set of attributes that identify individual items in the class. Examples of Tuxedo classes are:

T_Machine

The class definition for a machine

T_Service

The class definition for Tuxedo services

Attributes for these classes are identified by the prefix TA_ followed by the attribute name. A few examples for the T_Machine class are:

TA_Pmid

Represents a physical machine name

TA_Lmid

Represents the logical machine name

For more information about the standard Tuxedo MIB, refer to the BEA Tuxedo Reference Manual.

In contrast, the features of manageable resources in SNMP are called objects rather than attributes, and objects fall under MIB groups rather than classes.
The Tuxedo system Core MIB defines the set of groups through which the fundamental aspects of an application can be configured and managed. These include management of machines, servers, networking, and load balancing.

The Tuxedo Core MIB defines the basic objects that form a Tuxedo or WLE application. The Core MIB is the main information repository for controlling the operation and configuration of the application. When an application is active, the Core MIB contains groups related to the runtime activity of your application. You can use this information to monitor the behavior of your application. The Core MIB consists of the following groups.

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTBridgeTbl</td>
<td>Network connection</td>
</tr>
<tr>
<td>tuxTclientTbl</td>
<td>Client</td>
</tr>
<tr>
<td>tuxtconnTable</td>
<td>Conversation</td>
</tr>
<tr>
<td>tuxTdevice</td>
<td>Device</td>
</tr>
<tr>
<td>tuxTdomain</td>
<td>Domain information</td>
</tr>
<tr>
<td>tuxTgroupTable</td>
<td>Server group</td>
</tr>
<tr>
<td>tuxTmachineTable</td>
<td>Machine configuration attributes</td>
</tr>
<tr>
<td>tuxTmachineActive</td>
<td>Runtime machine characteristics</td>
</tr>
<tr>
<td>tuxTmsgTable</td>
<td>Message queue</td>
</tr>
<tr>
<td>tuxTqueueTable</td>
<td>Server queue</td>
</tr>
<tr>
<td>tuxTroutingTable</td>
<td>Routing criteria</td>
</tr>
<tr>
<td>Group Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>tuxTsrvrTbl</td>
<td>Server configuration attributes</td>
</tr>
<tr>
<td>tuxTsrvrTblExt</td>
<td>Server runtime characteristics</td>
</tr>
<tr>
<td>tuxTsvcTbl</td>
<td>Service</td>
</tr>
<tr>
<td>tuxTsvcGrp</td>
<td>Service-group configuration attributes</td>
</tr>
<tr>
<td>tuxTlistenTbl</td>
<td>/T listeners</td>
</tr>
<tr>
<td>tuxTranTbl</td>
<td>Transaction</td>
</tr>
<tr>
<td>tuxTulogTable</td>
<td>Userlog</td>
</tr>
<tr>
<td>tuxTulogCtrl</td>
<td>Control filter MIB for tuxTulogTable</td>
</tr>
<tr>
<td>tuxTnetMapTbl</td>
<td>Maps logical machine IDs to network groups</td>
</tr>
<tr>
<td>tuxTnetGrpTbl</td>
<td>Application attributes of network groups</td>
</tr>
<tr>
<td>tuxTserverCtxtTbl</td>
<td>Configuration and runtime attributes of individual server dispatch contexts.</td>
</tr>
<tr>
<td>beaEventFilters</td>
<td>You can use these filters to define a subset of Tuxedo event notifications</td>
</tr>
</tbody>
</table>
This group represents runtime attributes pertaining to connectivity between logical machines that make up an application. These attribute values represent connection status and statistics. Objects in this table are accessible either through a Tuxedo SNMP agent installed on the local machine or using the `-c` option on the master machine. The index into the table consists of `tuxTBridgeLmid` and `tuxTBridgeNetworkGrpNo`. In Tuxedo 6.4, SET requests are allowed only for the DEFAULTNET network group, so all SET requests should use 0 for `tuxTBridgeNetworkGrpNo` in the SNMP index.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTBridgeLmid</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.1</code></td>
</tr>
<tr>
<td><code>tuxTBridgeState</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.2</code></td>
</tr>
<tr>
<td><code>tuxTBridgeCurTime</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.3</code></td>
</tr>
<tr>
<td><code>tuxTBridgeConTime</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.4</code></td>
</tr>
<tr>
<td><code>tuxTBridgeSuspTime</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.5</code></td>
</tr>
<tr>
<td><code>tuxTBridgeRcvdByte</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.6</code></td>
</tr>
<tr>
<td><code>tuxTBridgeSentByte</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.7</code></td>
</tr>
<tr>
<td><code>tuxTBridgeRcvdNum</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.8</code></td>
</tr>
<tr>
<td><code>tuxTBridgeSentNum</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.9</code></td>
</tr>
<tr>
<td><code>tuxTBridgeFlowCnt</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.10</code></td>
</tr>
<tr>
<td><code>tuxTBridgeCurEncryptBits</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.11</code></td>
</tr>
<tr>
<td><code>tuxTBridgeNetworkGrpNo</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.12</code></td>
</tr>
<tr>
<td><code>tuxTBridgeNetworkGrpName</code></td>
<td><code>.1.3.6.1.4.1.140.300.16.1.1.13</code></td>
</tr>
</tbody>
</table>
tuxTBridgeLmid

Syntax  DisplayString(SIZE(1..61))
Access  read-only
Description  DisplayString is of the format: LMID1[,LMID2]

LMID1
Is the logical machine identifier for network connection and is in the range from one to sixty-one characters.

LMID2
Is the destination logical machine identifier for network connection and is in the range from one to sixty-one characters.

tuxTBridgeState

Syntax  INTEGER {active(1), inactive(2), suspended(3), pending(4)}
Access  read-write
Description  The values for GET and SET operations are as follows:

GET:{active(1)|inactive(2)|suspended(3)|pending(4)}

A GET operation retrieves runtime information for the selected tuxTBridgeTbl instance(s). A tuxTBridgeLmid attribute value with only one logical machine identifier matches all active connections from LMID1 to other machines in the application. In this case, each retrieved record contains an expanded tuxTBridgeLmid attribute value with the destination LMID filled in. The following states indicate the meaning of a tuxTBridgeState returned in response to a GET request. States not listed are not returned.

active(1)
The connection is established and active.

inactive(2)
The connection is inactive. This state is only returned when status is requested on a particular connection, that is, both LMIDs are specified in the tuxTBridgeLmid attribute and the source logical machine is reachable.

suspended(3)
An established connection was terminated due to an error condition, and reconnection has been suspended for at least the amount of time indicated in the `tuxTBridgeSuspTime` attribute value.

**pending(4)**

An asynchronous connection has been requested but has not yet completed. The final outcome of the connection request has not been determined. This state is only supported on Tuxedo 6.4 or later.

SET: `{active(1) | inactive(2) | suspended(3) | pending(4)}`

A SET operation updates runtime information for the selected `tuxTBridgeTbl` object. The following states indicate the meaning of a `tuxTBridgeState` set in a SET request. States not listed cannot be set.

**active(1)**

**Tuxedo 6.3 and earlier:** Activate the `tuxTBridgeTbl` object by establishing a connection between the indicated logical machines. This operation fails if only one logical machine is specified, if either of the two machines is not active, or if the source logical machine is not reachable. State change allowed in the `inactive(2)` and `suspended(3)` states. Successful return leaves the object in the `active(1)` state. **Tuxedo 6.4 and later:** Activate the `tuxTBridgeTbl` instance by establishing an asynchronous connection between the indicated logical machines. This operation fails if only one machine is specified, if either of the machines is not active, or if the source machine is not reachable. When in the `pending(4)` state, the success or failure of the connection has not yet been determined. The BRIDGE can continue to process other events and data while the connection is outstanding. This state change is allowed in the `inactive(2)` and `suspended(3)` states. Successful return leaves the instance in the `active(1)` or `pending(4)` state.

**inactive(2)**

Deactivate the `tuxTBridgeTbl` object by closing the connection between the indicated logical machines. This operation fails if only one logical machine is specified or if the two machines are not connected. State change allowed only when in the `active(1)` state. Successful return leaves the object in the `inactive(2)` state.

**suspended(3)**

Suspend the `tuxTBridgeTbl` object by closing the connection between the indicated logical machines and by setting the `tuxTBridgeSuspTime`
parameter as indicated. State change allowed only when in the active(1)
state. Successful return leaves the object in the suspended(3) state.

Note: Since the statistics reported are from the source logical machine, resetting
those statistics causes them to be out of sync with the statistics reported by
the destination logical machine for the same connection.

pending(4)

Activate the tuxTBridgeTbl instance by establishing an asynchronous
connection between the indicated logical machines. This operation fails if
only one logical machine is specified, if either of the two machines is
inactive, or if the source logical machine is not reachable. When in the
pending(4) state, the success or failure of the connection request has not
yet been determined. However, the BRIDGE can continue to process other
events and data while the connection request is outstanding. State change
allowed in inactive(2) and suspended(3) states. Successful return leaves
the instance in the pending(4) state. This state is supported only on 6.4 and
later versions of Tuxedo systems.

tuxTBridgeCurTime

Syntax INTEGER
Access read-only
Description Current time, in seconds, since 00:00:00 UTC, January 1, 1970, as returned by the
time(2) system call on tuxTBridgeLmid. This attribute can be used to compute
elapsed time from the following attribute values.

tuxTBridgeConTime

Syntax INTEGER
Access read-only
Description Time, in seconds, that this connection has been active.

tuxTBridgeSuspTime

Syntax INTEGER
Access read-write
| Description | Time, in seconds, remaining in the suspension of this connection. After this amount of time, the connection automatically changes to a `tuxTBridgeState` of `inactive(2)` and can be activated by normal application traffic. |

**tuxTBridgeRcvdByte**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Number of bytes sent from the destination logical machine to the source logical machine.

**tuxTBridgeSentByte**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Number of bytes sent from the source logical machine to the destination logical machine.

**tuxTBridgeRcvdNum**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Number of messages sent from the destination logical machine to the source logical machine.

**tuxTBridgeSentNum**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Number of messages sent from the source logical machine to the destination logical machine.
tuxTBridgeFlowCnt

Syntax  INTEGER
Access   read-only
Description  Number of times flow control has been encountered over this connection.

tuxTBridgeCurEncryptBits

Syntax  Integer {none(1), 40-bit(2), 128-bit(3), not-available(4)}
Access   read-only
Description  The current level of encryption for this link. This is negotiated between the machines when the link is established. The number specifies the encryption key length (in bits). This object is only supported on Tuxedo 6.4 and later.

tuxTBridgeNetworkGrpNo

Syntax  Integer
Access   read-only
Description  Logical network group number. When both the source and destination tuxTBridgeLmid machine identifiers are in the same network group, tuxTBridgeTbl presents all instances of related fields per network group. This object is supported only on Tuxedo 6.4 and later.


tuxTBridgeNetworkGrpName

Syntax  DisplayString
Access   read-only
Description  Logical network group name. This object is only supported on Tuxedo 6.4 or later.
This group represents runtime attributes of active clients within an application. These attribute values identify and track the activity of clients within a running application. Objects in this table are only accessible through a Tuxedo SNMP agent installed on the local machine.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTclientState</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.1</td>
</tr>
<tr>
<td>tuxTclientBirthTime</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.2</td>
</tr>
<tr>
<td>tuxTclientMachineId</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.3</td>
</tr>
<tr>
<td>tuxTclientReg</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.4</td>
</tr>
<tr>
<td>tuxTclientClntName</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.5</td>
</tr>
<tr>
<td>tuxTclientIdleTime</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.6</td>
</tr>
<tr>
<td>tuxTclientPid</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.7</td>
</tr>
<tr>
<td>tuxTclientSrvGrp</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.8</td>
</tr>
<tr>
<td>tuxTclientUsrName</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.9</td>
</tr>
<tr>
<td>tuxTclientWsc</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.10</td>
</tr>
<tr>
<td>tuxTclientWsh</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.11</td>
</tr>
<tr>
<td>tuxTclientWshClientId</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.12</td>
</tr>
<tr>
<td>tuxTclientRelease</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.13</td>
</tr>
<tr>
<td>tuxTclientWsProto</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.14</td>
</tr>
<tr>
<td>tuxTclientNumConv</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.15</td>
</tr>
<tr>
<td>tuxTclientNumDeque</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.16</td>
</tr>
<tr>
<td>tuxTclientNumEnque</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.17</td>
</tr>
</tbody>
</table>
## Tuxedo Core MIB

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTclientNumPost</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.18</td>
</tr>
<tr>
<td>tuxTclientNumReq</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.19</td>
</tr>
<tr>
<td>tuxTclientNumSubscribe</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.20</td>
</tr>
<tr>
<td>tuxTclientNumTran</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.21</td>
</tr>
<tr>
<td>tuxTclientNumTranAbt</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.22</td>
</tr>
<tr>
<td>tuxTclientNumTranCmt</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.23</td>
</tr>
<tr>
<td>tuxTclientCmtRet</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.24</td>
</tr>
<tr>
<td>tuxTclientCurConv</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.26</td>
</tr>
<tr>
<td>tuxTclientCurReq</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.27</td>
</tr>
<tr>
<td>tuxTclientCurTime</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.28</td>
</tr>
<tr>
<td>tuxTclientLastGrp</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.29</td>
</tr>
<tr>
<td>tuxTclientNaddr</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.30</td>
</tr>
<tr>
<td>tuxTclientNotify</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.31</td>
</tr>
<tr>
<td>tuxTclientNumUnSol</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.32</td>
</tr>
<tr>
<td>tuxTclientRpid</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.33</td>
</tr>
<tr>
<td>tuxTclientTimeLeft</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.34</td>
</tr>
<tr>
<td>tuxTclientTimeStart</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.36</td>
</tr>
<tr>
<td>tuxTclientTranLev</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.37</td>
</tr>
<tr>
<td>tuxTclientId</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.38</td>
</tr>
<tr>
<td>tuxTclientContextID</td>
<td>.1.3.6.1.4.1.140.300.17.1.1.50</td>
</tr>
</tbody>
</table>
tuxTclientState

Syntax  INTEGER { active(1), suspended(2), dead(3) }

Access  read-write

Description  The values for GET and SET operations are as follows:

GET: {active(1)|suspended(2)|dead(3)}

A GET operation retrieves runtime information for the selected
tuxTclientTbl instance(s). Note that client information is kept in local
bulletin board tables only. Therefore, for maximum performance, inquiries on
client status should be restricted, using key fields as much as possible. The
following states indicate the meaning of a tuxTclientState returned in
response to a GET request. States not listed are not returned.

active(1)

tuxTclientTbl instance active. This is not an indication of whether the
client is idle or busy. A non-0 value retrieved for either the
tuxTclientCurConv attribute or the tuxTclientCurReq attribute indicates
a busy client.

suspended(2)

tuxTclientTbl instance active and suspended from making further service
requests (tpcall(3) or tpacall(3)) and from initiating further conversations
(tpconnect(3)). See SET suspended(2) below for details.

dead(3)

tuxTclientTbl instance identified as active in the bulletin board but
currently not running due to an abnormal death. This state exists only until
the BBL local to the client notices the death and takes action to clean up the
client’s bulletin board resources.

SET: {active(1)|suspended(2)|dead(3)}

A SET operation updates runtime information for the selected
tuxTclientTbl object. The following states indicate the meaning of a
tuxTclientState set in a SET request. States not listed cannot be set.

active(1)
Activate a suspended(2) tuxTclientTbl instance. State change allowed only when in the suspended(2) state. Successful return leaves the object in the active(1) state.

suspended(2)

Suspend the tuxTclientTbl instance from making service requests (tpcall(3) or tpacall(3)), initiating conversations (tpconnect(3)), beginning transactions (tpbegin(3)), and enqueuing new requests (tpenqueue(3)). Clients within a transaction are permitted to make these calls until they abort or commit the current transaction, at which time the clients become suspended. Invocations of these routines result in a TPESYSTEM error return and a system log message being generated that indicates the situation. State change is allowed only when the object is in the active(1) state. Successful return leaves the object in the suspended(2) state.

dead(3)

Abortively deactivate the tuxTclientTbl instance. State change is allowed only when the object is in the active(1) or suspended(2) state. The recommended method for deactivating clients is to first suspend them, and then to abortively deactivate them by setting the state to dead(3). Successful return leaves the object in the dead(3) state.

Note: Workstation handlers (tuxTclientWsh == yes(1)) cannot be set to a state of dead(3). The system might not be able to kill the client, due to platform or signaling restrictions. In this case, a native client is abortively terminated at its next access to ATMI, and a workstation client’s connection to a WSH is preemptively torn down.

**tuxTclientBirthTime**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>Client identifier. The data in this field should not be interpreted directly by the end user except for equality comparison.</td>
</tr>
</tbody>
</table>

**tuxTclientMachineId**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
</table>
tuxTclientTbl

<table>
<thead>
<tr>
<th>Access</th>
<th>read-only</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Client identifier. The data in this field should not be interpreted directly by the end user except for equality comparison.</td>
</tr>
</tbody>
</table>

**tuxTclientReg**

Syntax  INTEGER
Access   read-only
Description Client identifier. The data in this field should not be interpreted directly by the end user except for equality comparison.

**tuxTclientClntName**

Syntax  DisplayString(SIZE(0..30))
Access   read-only
Description Client name associated with client at tpinit(3) time through the cltname element of the TPINIT structure.

**tuxTclientIdleTime**

Syntax  INTEGER
Access   read-only
Description Approximate amount of time, in seconds, since this client last interacted with the system through an ATMI call. This value is accurate to within tuxTdomainScanUnit (see the tuxTdomain group) seconds. When specified as a key field, a positive value indicates that all clients with idle times of at least the indicated value match, a negative value indicates that all clients with no more than the indicated value match, and a 0 value matches all clients.

**tuxTclientPid**

Syntax  INTEGER
Access   read-only
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Syntax</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTclientSrvGrp</td>
<td>DisplayString(SIZE(0..30))</td>
<td>read-only</td>
<td>Server group with which the client is associated. This information is set through the grpname element of the TPINIT structure at <code>tpinit(3)</code> time.</td>
</tr>
<tr>
<td>tuxTclientUsrName</td>
<td>DisplayString(SIZE(0..30))</td>
<td>read-only</td>
<td>User name associated with client at <code>tpinit(3)</code> time through the usrename element of the TPINIT structure.</td>
</tr>
<tr>
<td>tuxTclientWsc</td>
<td>INTEGER { yes(1), no(2) }</td>
<td>read-only</td>
<td>If this attribute is set to <code>yes(1)</code>, the indicated client is logged in to the application from a remote workstation.</td>
</tr>
<tr>
<td>tuxTclientWsh</td>
<td>INTEGER { yes(1), no(2) }</td>
<td>read-only</td>
<td>Workstation handler. If this attribute is set to <code>yes(1)</code>, the indicated client is a workstation handler process.</td>
</tr>
</tbody>
</table>
**tuxTclientWshClientId**

**Syntax**  
DisplayString(SIZE(1..78))

**Access**  
read-only

**Description**  
Client identifier for the associated workstation handler (WSH) if this client is a workstation client (tuxTclientWsc == yes(1)); otherwise, this attribute is returned as a 0-length string.

**tuxTclientRelease**

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
The Tuxedo System/T major protocol release number for the machine where the client is running. This can be different from the tuxTmachineSWrelease for the same machine. Note that for /WS clients (tuxTclientWsc == yes(1)), this value can be different from the major release associated with the application administered machine through which the /WS client accesses the application.

**tuxTclientWsProto**

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
The Tuxedo System/T /WS protocol version number for a workstation client. This value is changed with each update to the /WS protocol. A value of 0 is returned for this attribute when it is associated with non-/WS clients (tuxTclientWsc == no(2)).

**tuxTclientNumConv**

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Number of conversations initiated by this client through tpconnect(3).
**Tuxedo Core MIB**

### `tuxTclientNumDeque`

**Syntax** INTEGER  
**Access** read-only  
**Description** Number of dequeue operations initiated by this client through `tpdequeue(3)`.

### `tuxTclientNumEnque`

**Syntax** INTEGER  
**Access** read-only  
**Description** Number of enqueue operations initiated by this client through `tpenqueue(3)`.

### `tuxTclientNumPost`

**Syntax** INTEGER  
**Access** read-only  
**Description** Number of postings initiated by this client through `tppost(3)`.

### `tuxTclientNumReq`

**Syntax** INTEGER  
**Access** read-only  
**Description** Number of requests made by this client through `tpcall(3)` or `tpacall(3)`.

### `tuxTclientNumSubscribe`

**Syntax** INTEGER  
**Access** read-only  
**Description** Number of subscriptions made by this client through `tpssubscribe(3)`. 
tuxTclientTbl

**tuxTclientNumTran**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Number of transactions begun by this client.

**tuxTclientNumTranAbt**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Number of transactions aborted by this client.

**tuxTclientNumTranCmt**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Number of transactions committed by this client.

**tuxTclientCmtRet**

- **Syntax**: INTEGER { complete(1) | logged(2) }
- **Access**: read-only
- **Description**: Setting of the TP_COMMIT_CONTROL characteristic for this client. See the description of the System/T ATMI function tpscmt(3) for details on this characteristic.

**tuxTclientCurConv**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Number of conversations initiated by this client through tpconnect(3) that are still active.
tuxTclientCurReq

Syntax   INTEGER
Access   read-only
Description   Number of requests initiated by this client through tpcall(3) or tpacall(3) that are still active.


tuxTclientCurTime

Syntax   INTEGER
Access   read-only
Description   Current time, in seconds, since 00:00:00 UTC, January 1, 1970, as returned by the time(2) system call on the local host. This attribute can be used to compute elapsed time from the tuxTclientTimeStart attribute value.


tuxTclientLastGrp

Syntax   INTEGER
Access   read-only
Description   Server group number of the last service request made or conversation initiated from this client.


tuxTclientNaddr

Syntax   DisplayString(SIZE(1..78))
Access   read-only
Description   For workstation clients, this attribute indicates the network address of the client. Network addresses with unprintable characters are converted to the “0x...” network address format as described in the tuxTmachineNaddr attribute. Non-workstation clients have a 0-length string associated with them for this attribute value.
Note: The ability of the system to provide this information is determined by the transport provider in use. In some cases, workstation clients cannot have addresses associated with them if the provider does not make this information available.

**tuxTclientNotify**

Syntax: INTEGER { dipin(1) | signal(2) | ignore(3) }

Access: read-only

Description: Setting of the notification characteristic for this client. See the `tuxTdomain` group description of this attribute for more details.

**tuxTclientNumUnSol**

Syntax: INTEGER

Access: read-only

Description: Number of unsolicited messages queued for this client that are awaiting processing.

**tuxTclientRpid**

Syntax: INTEGER

Access: read-only

Description: UNIX system message queue identifier for the client's reply queue.

Note: This is a UNIX system specific attribute that might not be returned if the platform on which the application is being run is not UNIX-based.

**tuxTclientTimeLeft**

Syntax: INTEGER

Access: read-only

Description: Time left, in seconds, for this client to receive the reply for which it is currently waiting before it times out. This timeout can be a transactional timeout or a blocking timeout.
**tuxTclientTimeStart**

**Syntax** INTEGER

**Access** read-only

**Description** Time, in seconds, since 00:00:00 UTC, January 1, 1970, as returned by the `time(2)` system call on local host, since the client joined the application.

**tuxTclientTranLev**

**Syntax** INTEGER

**Access** read-only

**Description** Current transaction level for this client. 0 indicates that the client is not currently involved in a transaction.

**tuxTclientId**

**Syntax** DisplayString(Size(1..78))

**Access** read-only

**Description** Client Identifier.

**tuxTclientContextID**

**Syntax** INTEGER (-2..29999)

**Access** read-only

**Status** mandatory

**Description** Identifier for this particular application association.
This group represents runtime attributes of active conversations within an application. Objects in this table are only accessible through a Tuxedo SNMP agent installed on the local machine. All objects in this MIB group are local attributes, that is, values for these objects correspond to the local host only where the Tuxedo agent is running. Thus, the user needs to run an instance of the Tuxedo agent on every node for which these values are of interest. The index into this table is `tuxTconnSerNo`.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTconnSerNo</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.1</td>
</tr>
<tr>
<td><code>tuxTconnState</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.2</td>
</tr>
<tr>
<td><code>tuxTconnSvcName</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.3</td>
</tr>
<tr>
<td><code>tuxTconnClientId</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.4</td>
</tr>
<tr>
<td><code>tuxTconnOgrpNo</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.5</td>
</tr>
<tr>
<td><code>tuxTconnOlmid</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.6</td>
</tr>
<tr>
<td><code>tuxTconnOpid</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.7</td>
</tr>
<tr>
<td><code>tuxTconnOsndcnt</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.8</td>
</tr>
<tr>
<td><code>tuxTconnOsrvId</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.9</td>
</tr>
<tr>
<td><code>tuxTconnSgrpNo</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.10</td>
</tr>
<tr>
<td><code>tuxTconnSlmid</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.11</td>
</tr>
<tr>
<td><code>tuxTconnSpid</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.12</td>
</tr>
<tr>
<td><code>tuxTconnSsndcnt</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.13</td>
</tr>
<tr>
<td><code>tuxTconnSsrvId</code></td>
<td>.1.3.6.1.4.1.140.300.18.1.1.14</td>
</tr>
</tbody>
</table>
### tuxTconnSerNo

**Syntax**  INTEGER

**Access**  read-only

**Description**  A running number as an index for tuxTconnTable.

### tuxTconnState

**Syntax**  INTEGER { active(1) }

**Access**  read-only

**Description**  The values for GET and SET operations are as follows:

**GET:**

A GET operation retrieves runtime information for the selected tuxTconnTable instance(s). The following state indicates the meaning of a tuxTconnState returned in response to a GET request. States not listed are not returned.

- active(1)

  The object returned reflects one or both sides of an active conversation within the application.

**SET:**

  SET operations are not permitted for this class.

### tuxTconnSvcName

**Syntax**  DisplayString(SIZE(1..15))

**Access**  read-only

**Description**  Service name of the conversational service invoked by the originator and processed by the subordinate.
tuxTconnClientId

Syntax: DisplayString(SIZE(1..78))
Access: read-only
Description: Client identifier. The data in this field should not be interpreted directly by the end user except for equality comparison.

tuxTconnOgrpNo

Syntax: INTEGER (1..30001)
Access: read-only
Description: Server group number for the originator of the conversation. If the originator is a client, then 30,000 is returned as the value for this attribute.

tuxTconnOlmid

Syntax: DisplayString(SIZE(1..30))
Access: read-only
Description: Logical machine identifier that indicates where the originator is running, or (in the case of /WS clients) is accessing the application.

tuxTconnOpid

Syntax: INTEGER
Access: read-only
Description: Process identifier for the originator of the conversation.

tuxTconnOsndcnt

Syntax: INTEGER
Access: read-only
Description: Number of tpsend(3) calls made by the originator.
**tuxTconnOsrvld**

Syntax: INTEGER (1..30001)
Access: read-only
Description: Server identifier for the originator of the conversation.

**tuxTconnSgrpNo**

Syntax: INTEGER (1..30001)
Access: read-only
Description: Server group number for the subordinate of the conversation. If the originator is a client, then 30,000 is returned as the value for this attribute.

**tuxTconnSlmid**

Syntax: DisplayString(SIZE(1..30))
Access: read-only
Description: Logical machine identifier that indicates where the subordinate is running or, (in the case of /WS clients) is accessing the application.

**tuxTconnSpid**

Syntax: INTEGER
Access: read-only
Description: Process identifier for the subordinate in the conversation.

**tuxTconnSsndcnt**

Syntax: INTEGER
Access: read-only
Description: Number of tpsend(3) calls made by the subordinate.
tuxTconnTable

**tuxTconnSrvId**

- **Syntax**: INTEGER (1..30001)
- **Access**: read-only
- **Description**: Server identifier for the subordinate in the conversation.
### tuxTdevice

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTdeviceTbl</td>
<td>.1.3.6.1.4.1.140.300.19.1</td>
</tr>
<tr>
<td>tuxTwhichCfgDev</td>
<td>.1.3.6.1.4.1.140.300.19.2</td>
</tr>
</tbody>
</table>
This group represents configuration and runtime attributes of raw disk slices or UNIX system files being used to store Tuxedo System/T device lists. This class allows for the creation and deletion of device list entries within a raw disk slice or UNIX system file. Objects in this table are only accessible through a Tuxedo SNMP agent installed on the local machine. To create a new row in this table, the user needs to send a SET request, with at least a value for `tuxTdevSize`. The index into this table is `tuxTdevCfgDev` and `tuxTdevIndex`.

### Variable Name | Object ID
--- | ---
`tuxTdevLmid` | .1.3.6.1.4.1.140.300.19.1.1.1
`tuxTdevCfgDev` | .1.3.6.1.4.1.140.300.19.1.1.2
`tuxTdeviceName` | .1.3.6.1.4.1.140.300.19.1.1.3
`tuxTdevOffset` | .1.3.6.1.4.1.140.300.19.1.1.4
`tuxTdevSize` | .1.3.6.1.4.1.140.300.19.1.1.5
`tuxTdevIndex` | .1.3.6.1.4.1.140.300.19.1.1.6
`tuxTdevState` | .1.3.6.1.4.1.140.300.19.1.1.7

### `tuxTdevLmid`

**Syntax**

`DisplayString` *(SIZE(1..30))*

**Access**

read-write

**Description**

Logical machine identifier where the device is located. Note that this attribute can be used as a key field in both unbooted and booted applications as long as they are already configured (that is, at least one `tuxTmachineTable` instance exists). It is required as a key field on SET operations when they are accessing a booted application. If specified when accessing the `tuxTdeviceTbl` table in an unconfigured application, this attribute is ignored.

**Note:** This object can be set only during row creation.
**tuxTdevCfgDev**

**Syntax**  
DisplayString(SIZE(2..64))

**Access**  
read-write

**Description**  
Absolute pathname of the file or device where the Tuxedo System/T filesystem is stored or is to be stored.

**Note:** This object can be set only during row creation.

**tuxTdeviceName**

**Syntax**  
DisplayString(SIZE(2..64))

**Access**  
read-write

**Description**  
Absolute pathname of the device list entry.

**Note:** This object can be set only during row creation.

**tuxTdevOffset**

**Syntax**  
INTEGER

**Access**  
read-write

**Description**  
The offset, in blocks, at which space on this tuxTdevice begins for use within the Tuxedo System/T VTOC specified by tuxTdevCfgDev.

**Note:** This object can be set only during row creation.

**tuxTdevSize**

**Syntax**  
INTEGER

**Access**  
read-write

**Description**  
The size in pages of the disk area to be used for the device list entry.

**Note:** This attribute can be set only in conjunction with row creation.

**Note:** This object can be set only during row creation.
**tuxTdevIndex**

Syntax: INTEGER

Access: read-only

Description: Device index for tuxTdevice within the device list addressed by tuxTdevCfgDev. This attribute value is used for identification purposes only in getting and setting attribute values that relate to particular devices within a Tuxedo System/T filesystem.

**tuxTdevState**

Syntax: INTEGER { valid(1) | invalid(2) | re-init(3) }

Access: read-write

Description: The values for GET and SET operations are as follows:

GET: {valid(1)}

A GET operation retrieves runtime information for the selected tuxTdeviceTbl instance(s). The following state indicates the meaning of a tuxTdevState returned in response to a GET request. States not listed are not returned.

valid(1)

The Tuxedo System/T filesystem indicated by tuxTdevCfgDev exists and contains a valid device list. tuxTdevice is a valid device within that filesystem with the device index tuxTdevIndex.

SET: {invalid(2) | re-init(3)}

A SET operation updates information for the selected tuxTdeviceTbl instance or adds the indicated object. The following states indicate the meaning of a tuxTdevState set in a SET request. States not listed cannot be set.

invalid(2)

Delete tuxTdeviceTbl instance for application. State change is allowed only when the object is in the valid(1) state. Successful return leaves the object in the invalid(2) state. Note that tuxTdevIndex 0 is special and must be deleted last.

re-init(3)

The Tuxedo System/T filesystem indicated by tuxTdevCfgDev is initialized with the tuxTdevice instance(s).
Re-initializes a valid device.

**tuxTwhichCfgDev**

**Syntax**
DisplayString(SIZE(2..64))

**Access**
read-write

**Description**
The value of this object determines the device for which `tuxTdeviceTbl` returns configuration and runtime information.

The default value of this object is the TUXCONFIG file for the current domain.
The following objects of \texttt{tuxTdomain} represent global application attributes for the domain to which the Tuxedo SNMP Agent is currently connected. These object values serve to identify, customize, size, secure, and tune a Tuxedo System/T application. Many of the object values represented here serve as application defaults for other groups represented in this MIB.

There is exactly one instance of the \texttt{tuxTdomain} group for each application.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{tuxTdomainKey}</td>
<td>.1.3.6.1.4.1.140.300.3.1</td>
</tr>
<tr>
<td>\texttt{tuxTdomainMaster}</td>
<td>.1.3.6.1.4.1.140.300.3.2</td>
</tr>
<tr>
<td>\texttt{tuxTdomainModel}</td>
<td>.1.3.6.1.4.1.140.300.3.3</td>
</tr>
<tr>
<td>\texttt{tuxTdomainState}</td>
<td>.1.3.6.1.4.1.140.300.3.4</td>
</tr>
<tr>
<td>\texttt{tuxTdomainID}</td>
<td>.1.3.6.1.4.1.140.300.3.5</td>
</tr>
<tr>
<td>\texttt{tuxTdomainUID}</td>
<td>.1.3.6.1.4.1.140.300.3.7</td>
</tr>
<tr>
<td>\texttt{tuxTdomainGID}</td>
<td>.1.3.6.1.4.1.140.300.3.8</td>
</tr>
<tr>
<td>\texttt{tuxTdomainPerm}</td>
<td>.1.3.6.1.4.1.140.300.3.9</td>
</tr>
<tr>
<td>\texttt{tuxTdomainMask}</td>
<td>.1.3.6.1.4.1.140.300.3.10</td>
</tr>
<tr>
<td>\texttt{tuxTdomainMaxAccessers}</td>
<td>.1.3.6.1.4.1.140.300.3.11</td>
</tr>
<tr>
<td>\texttt{tuxTdomainMaxConv}</td>
<td>.1.3.6.1.4.1.140.300.3.12</td>
</tr>
<tr>
<td>\texttt{tuxTdomainMaxGTT}</td>
<td>.1.3.6.1.4.1.140.300.3.13</td>
</tr>
<tr>
<td>\texttt{tuxTdomainMaxBufsType}</td>
<td>.1.3.6.1.4.1.140.300.3.14</td>
</tr>
<tr>
<td>\texttt{tuxTdomainMaxBufType}</td>
<td>.1.3.6.1.4.1.140.300.3.15</td>
</tr>
<tr>
<td>\texttt{tuxTdomainMaxDRT}</td>
<td>.1.3.6.1.4.1.140.300.3.16</td>
</tr>
<tr>
<td>\texttt{tuxTdomainMaxGroups}</td>
<td>.1.3.6.1.4.1.140.300.3.17</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Object ID</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>tuxtTdomainMaxMachines</td>
<td>.1.3.6.1.4.1.140.300.3.18</td>
</tr>
<tr>
<td>tuxtTdomainMaxQueues</td>
<td>.1.3.6.1.4.1.140.300.3.19</td>
</tr>
<tr>
<td>tuxtTdomainMaxRFT</td>
<td>.1.3.6.1.4.1.140.300.3.20</td>
</tr>
<tr>
<td>tuxtTdomainMaxRTData</td>
<td>.1.3.6.1.4.1.140.300.3.21</td>
</tr>
<tr>
<td>tuxtTdomainMaxServers</td>
<td>.1.3.6.1.4.1.140.300.3.22</td>
</tr>
<tr>
<td>tuxtTdomainMaxServices</td>
<td>.1.3.6.1.4.1.140.300.3.23</td>
</tr>
<tr>
<td>tuxtTdomainMaxACLgroups</td>
<td>.1.3.6.1.4.1.140.300.3.24</td>
</tr>
<tr>
<td>tuxtTdomainCMTRET</td>
<td>.1.3.6.1.4.1.140.300.3.25</td>
</tr>
<tr>
<td>tuxtTdomainLoadBalance</td>
<td>.1.3.6.1.4.1.140.300.3.26</td>
</tr>
<tr>
<td>tuxtTdomainNotify</td>
<td>.1.3.6.1.4.1.140.300.3.27</td>
</tr>
<tr>
<td>tuxtTdomainSystemAccess</td>
<td>.1.3.6.1.4.1.140.300.3.28</td>
</tr>
<tr>
<td>tuxtTdomainOptions</td>
<td>.1.3.6.1.4.1.140.300.3.29</td>
</tr>
<tr>
<td>tuxtTdomainSignal</td>
<td>.1.3.6.1.4.1.140.300.3.30</td>
</tr>
<tr>
<td>tuxtTdomainSecurity</td>
<td>.1.3.6.1.4.1.140.300.3.31</td>
</tr>
<tr>
<td>tuxtTdomainAuthsvc</td>
<td>.1.3.6.1.4.1.140.300.3.33</td>
</tr>
<tr>
<td>tuxtTdomainScanUnit</td>
<td>.1.3.6.1.4.1.140.300.3.34</td>
</tr>
<tr>
<td>tuxtTdomainBBLQuery</td>
<td>.1.3.6.1.4.1.140.300.3.35</td>
</tr>
<tr>
<td>tuxtTdomainBlockTime</td>
<td>.1.3.6.1.4.1.140.300.3.36</td>
</tr>
<tr>
<td>tuxtTdomainDBBLWait</td>
<td>.1.3.6.1.4.1.140.300.3.37</td>
</tr>
<tr>
<td>tuxtTdomainSanityScan</td>
<td>.1.3.6.1.4.1.140.300.3.38</td>
</tr>
<tr>
<td>tuxtTdomainCurDRT</td>
<td>.1.3.6.1.4.1.140.300.3.39</td>
</tr>
<tr>
<td>tuxtTdomainCurGroups</td>
<td>.1.3.6.1.4.1.140.300.3.40</td>
</tr>
<tr>
<td>tuxtTdomainCurMachines</td>
<td>.1.3.6.1.4.1.140.300.3.41</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Object ID</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>tuxTdomainCurQueues</td>
<td>.1.3.6.1.4.1.140.300.3.42</td>
</tr>
<tr>
<td>tuxTdomainCurRFT</td>
<td>.1.3.6.1.4.1.140.300.3.43</td>
</tr>
<tr>
<td>tuxTdomainCurRTdata</td>
<td>.1.3.6.1.4.1.140.300.3.44</td>
</tr>
<tr>
<td>tuxTdomainCurServers</td>
<td>.1.3.6.1.4.1.140.300.3.45</td>
</tr>
<tr>
<td>tuxTdomainCurServices</td>
<td>.1.3.6.1.4.1.140.300.3.46</td>
</tr>
<tr>
<td>tuxTdomainCursType</td>
<td>.1.3.6.1.4.1.140.300.3.47</td>
</tr>
<tr>
<td>tuxTdomainCurType</td>
<td>.1.3.6.1.4.1.140.300.3.48</td>
</tr>
<tr>
<td>tuxTdomainHwDRT</td>
<td>.1.3.6.1.4.1.140.300.3.49</td>
</tr>
<tr>
<td>tuxTdomainHwGroups</td>
<td>.1.3.6.1.4.1.140.300.3.50</td>
</tr>
<tr>
<td>tuxTdomainHwMachines</td>
<td>.1.3.6.1.4.1.140.300.3.51</td>
</tr>
<tr>
<td>tuxTdomainHwQueues</td>
<td>.1.3.6.1.4.1.140.300.3.52</td>
</tr>
<tr>
<td>tuxTdomainHwRFT</td>
<td>.1.3.6.1.4.1.140.300.3.53</td>
</tr>
<tr>
<td>tuxTdomainHwRTdata</td>
<td>.1.3.6.1.4.1.140.300.3.54</td>
</tr>
<tr>
<td>tuxTdomainHwServers</td>
<td>.1.3.6.1.4.1.140.300.3.55</td>
</tr>
<tr>
<td>tuxTdomainHwServices</td>
<td>.1.3.6.1.4.1.140.300.3.56</td>
</tr>
<tr>
<td>tuxTdomainMaxNetGroups</td>
<td>.1.3.6.1.4.1.140.300.3.58</td>
</tr>
<tr>
<td>wleMaxObjects</td>
<td>.1.3.6.1.4.1.140.300.3.63</td>
</tr>
<tr>
<td>wleMaxInterfaces</td>
<td>.1.3.6.1.4.1.140.300.3.68</td>
</tr>
<tr>
<td>tuxTdomainSignatureAhead</td>
<td>.1.3.6.1.4.1.140.300.3.70</td>
</tr>
<tr>
<td>wleCurInterfaces</td>
<td>.1.3.6.1.4.1.140.300.3.73</td>
</tr>
<tr>
<td>wleHwInterfaces</td>
<td>.1.3.6.1.4.1.140.300.3.78</td>
</tr>
<tr>
<td>tuxTdomainSignatureBehind</td>
<td>.1.3.6.1.4.1.140.300.3.80</td>
</tr>
<tr>
<td>tuxTdomainEncryptionRequired</td>
<td>.1.3.6.1.4.1.140.300.3.90</td>
</tr>
</tbody>
</table>
Tuxedo Core MIB

### tuxTdomainKey

**Syntax**
INTEGER (32769..262143)

**Access**
read-write

**Description**
Numeric key for the well-known address in a Tuxedo System/T bulletin board. In a single processor environment, this key “names” the bulletin board. In a multiple processor or LAN environment, this key names the message queue of the DBBL. In addition, this key is used as a basis for deriving the names of resources other than the well-known address, such as the names for bulletin boards throughout the application.

### tuxTdomainMaster

**Syntax**
DisplayString(SIZE(1..30))

**Access**
read-write

**Description**
DisplayString is in format: LMID1[,LMID2]

- **LMID1**
  Is the master logical machine identifier and is in the range from one to thirty characters.

- **LMID2**
  Is the backup logical machine identifier and is in the range from one to thirty characters.

The master identifier (LMID1) must correspond to the local machine for inactive applications. single-machine(1) mode applications (see tuxTdomainModel below) can set only the master logical machine identifier. Modifications to this attribute value in an active multi-machine(2) application (see tuxTdomainModel below) have the following semantics.
Assuming current active master LMID A, current backup master LMID B, and secondary LMIDs C, D, ...., the following scenarios define the semantics of permitted changes to the tuxTdomainMaster attribute in a running multi-machine(2) mode application.

A,B -> B,A — Master migration from A to B. A,B -> A,C — Change backup master LMID designation to C.

Note that master migration can be either orderly or partitioned. Orderly migration takes place when the master machine is active and reachable. Otherwise, partitioned migration takes place. All newly established or re-established network connections verify that the two sites connecting share a common view of where the master machine is located. Otherwise, the connection is refused and an appropriate log message is generated.

The master and backup machines in an active application must always have a Tuxedo System/T release number greater than or equal to all other machines active in the application. The master and backup machines must be of the same release. Modifications to the tuxTdomainMaster attribute must preserve this relationship.

**tuxTdomainModel**

| Syntax         | INTEGER { single-machine(1) | multi-machine(2) } |
|----------------|-----------------------|
| Access         | read-write            |
| Description    | The configuration type. |
|                | single-machine(1)     |
|                | Specifies a single machine configuration; only one tuxTmachineTable object can be specified. |
|                | multi-machine(2)      |
|                | Specifies a multi-machine or network configuration; this must be specified if a networked application is being defined. |

**tuxTdomainState**

| Syntax         | INTEGER { active(1) | inactive(2) | forcible-inactive(3) } |
|----------------|----------------------|
| Access         | read-write           |
Description

The values for GET and SET operations are as follows:

GET: \{active(1) | inactive(2)\}

A GET operation retrieves configuration and runtime information for the `tuxTdomain` group. The following states indicate the meaning of a `tuxTdomainState` returned in response to a GET request. States not listed are not returned.

active(1)

`tuxTdomain` group is defined and the master machine is active.

inactive(2)

`tuxTdomain` group is defined and application is inactive.

SET: \{active(1) | inactive(2) | forcible-inactive(3)\}

A SET operation updates configuration and runtime information for the `tuxTdomain` group. The following states indicate the meaning of a `tuxTdomainState` set in a SET request. States not listed cannot be set.

active(1)

Activate administrative processes (DBBL, BBL, and so on) on the master machine. A state change is allowed only when the object is in the `inactive(2)` state. Successful return leaves the object in the `active(1)` state.

inactive(2)

Deactivate administrative processes (DBBL, BBL, and so on) on the master machine. A state change is allowed only when the object is in the `active(1)` state. Successful return leaves the object in the `inactive(2)` state. To do a complete shutdown of the application, you must first make all groups inactive. (See `tuxTgroupState`.) This state transition fails if any application servers or clients are still attached to the domain. To ignore any running clients or application servers, set to `forcible-inactive(3)` as explained below.

forcible-inactive(3)

Forcibly deactivate administrative processes (DBBL, BBL, and so on) on the master machine. Attached clients are ignored for the purpose of determining if shutdown should be allowed. State change is allowed only when the object is in the `active(1)` state. Successful return leaves the object in the
inactive(2) state. You need to restart any clients before they can be used to process services after this state transition.

**tuxTdomainID**

- **Syntax**: DisplayString(SIZE(0..30))
- **Access**: read-write
- **Description**: Domain identification string.

**tuxTdomainUID**

- **Syntax**: INTEGER
- **Access**: read-write
- **Description**: Default attribute setting for newly configured objects in the tuxTmachineTable group.
  
  **Note**: Changes to this attribute do not affect active or already configured tuxTmachineTable instances.

**tuxTdomainGID**

- **Syntax**: INTEGER
- **Access**: read-write
- **Description**: Default attribute setting for newly configured objects in the tuxTmachineTable group.
  
  **Note**: Changes to this attribute do not affect active or already configured tuxTmachineTable instances.

**tuxTdomainPerm**

- **Syntax**: DisplayString(SIZE(1..9))
- **Access**: read-write
Tuxedo Core MIB

Tuxedo Core MIB

Description
Default attribute setting for newly configured objects in the tuxTmachineTable group.

Note: Changes to this attribute do not affect active or already configured tuxTmachineTable instances.

tuxTdomainMask

Syntax 
DisplayString(SIZE(1..9))

Access 
read-write

Description 
Attribute access mask. User type/access mode combinations specified by this attribute value are no longer allowed for all class/attribute combinations defined in TM_MIB(5). For example, a setting of 0003 disallows all updates to users other than the administrator or the operator. The value of this object should be provided as an octal number — 0 through 0777.

tuxTdomainMaxAccessers

Syntax 
INTEGER (1..32767)

Access 
read-write

Description 
Default attribute setting for newly configured objects in the tuxTmachineTable group.

Note: Changes to this attribute do not affect active or already configured tuxTmachineTable instances.

tuxTdomainMaxConv

Syntax 
INTEGER (0..32767)

Access 
read-write

Description 
Default attribute setting for newly configured objects in the tuxTmachineTable group.

Note: Changes to this attribute do not affect active or already configured tuxTmachineTable instances.
tuxTdomainMaxGTT

**Syntax**
INTEGER (0..32767)

**Access**
read-write

**Description**
Default attribute setting for newly configured objects in the tuxTmachineTable group.

**Note:** Changes to this attribute do not affect active or already configured tuxTmachineTable instances.

---

tuxTdomainMaxBufsType

**Syntax**
INTEGER (1..32767)

**Access**
read-write

**Description**
Maximum number of buffer subtypes that can be accommodated in the bulletin board buffer subtype table.

---

tuxTdomainMaxBufType

**Syntax**
INTEGER (1..32767)

**Access**
read-write

**Description**
Maximum number of buffer types that can be accommodated in the bulletin board buffer type table.

---

tuxTdomainMaxDRT

**Syntax**
INTEGER (0..32767)

**Access**
read-write

**Description**
Maximum number of routing table entries that can be accommodated in the bulletin board routing table. One entry per tuxTroutingTable group object is required. Additional entries should be allocated to allow for runtime growth.
**tuxTdomainMaxGroups**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER (100..32767)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>Maximum number of server groups that can be accommodated in the bulletin board server group table.</td>
</tr>
</tbody>
</table>

**Note:** Tuxedo System/T Release 4.2.2 and earlier sites have a fixed setting of 100 for this attribute. Interoperability with these sites requires that no more than 100 server group entries be in use at any time. Release 4.2.2 and earlier sites are not allowed to join an application that has more than 100 defined server groups. Additionally, applications that already include Release 4.2.2 or earlier sites are not allowed to add server groups beyond 100.

**tuxTdomainMaxMachines**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER (256..8190)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>Maximum number of machines that can be accommodated in the bulletin board machine table.</td>
</tr>
</tbody>
</table>

**Note:** Tuxedo System/T Release 4.2.2 has a fixed setting of 256 for this attribute. Releases prior to Release 4.2.2 have a fixed setting of 50 for this attribute. Interoperability with Release 4.2.2 and earlier sites requires that no more than the lowest fixed setting number of machine table entries be in use at any time. Release 4.2.2 sites are not allowed to join an application that has more than 256 defined machines. Pre-Release 4.2.2 sites are not allowed to join an application that has more than 50 defined machines. Additionally, applications that already include active Release 4.2.2 or earlier sites are not allowed to add machines beyond the lowest applicable limit.
**tuxTdomainMaxQueues**

Syntax: INTEGER (1..8191)

Access: read-write

Description: Maximum number of queues to be accommodated in the bulletin board queue table.

**Note:** Release 4.2.2 and earlier sites can join an active application only if the setting for `tuxTdomainMaxQueues` is equal to the setting for `tuxTdomainMaxServers`.

**tuxTdomainMaxRFT**

Syntax: INTEGER (0..32767)

Access: read-write

Description: Maximum number of routing criteria range table entries to be accommodated in the bulletin board range criteria table. One entry per individual range within a `tuxTroutingRanges` specification is required plus one additional entry per `tuxTroutingTable` class object. Additional entries should be allocated to allow for runtime growth.

**tuxTdomainMaxRTData**

Syntax: INTEGER (0..32760)

Access: read-write

Description: Maximum string pool space to be accommodated in the bulletin board string pool table. Strings and carrays specified within `tuxTroutingRanges` values are stored in the string pool. Additional space should be allocated to allow for runtime growth.
### `tuxTdomainMaxServers`

**Syntax**

`INTEGER (1..8191)`

**Access**

`read-write`

**Description**

Maximum number of servers to be accommodated in the bulletin board server table. Allowances should be made in setting this attribute for system supplied administrative servers. Administration of each System/T site adds approximately one server. Additionally, if TMSs are specified for any server groups (see `tuxTgroupTMSname`), they are booted along with their server group and should be accounted for in setting `tuxTdomainMaxServers`.

### `tuxTdomainMaxServices`

**Syntax**

`INTEGER (1..32767)`

**Access**

`read-write`

**Description**

Maximum number of services to be accommodated in the bulletin board service table. Allowances should be made in setting this attribute for system supplied servers that offer services for administrative purposes. Administration of each System/T site adds approximately five services. Other administrative components such as `/WS`, `/Q`, and `/DM` can also add administrative services that should be accounted for.

### `tuxTdomainMaxACLgroups`

**Syntax**

`INTEGER (1..16384)`

**Access**

`read-write`

**Description**

Maximum number of group identifiers that can be used for checking ACL permissions. The maximum group identifier that can be defined is `tuxTdomainMaxACLgroups - 1`.
tuxTdomainCMTRET

Syntax: INTEGER { complete(1) | logged(2) }

Access: read-write

Description: Initial setting of the TP_COMMIT_CONTROL characteristic for all client and server processes in a System/T application. logged(2) initializes the TP_COMMIT_CONTROL characteristic to TP_CMT_LOGGED; otherwise, it is initialized to TP_CMT_COMPLETE. See the description of the System/T ATMI function tpscm(3) for details on the setting of this characteristic.

Note: Runtime modifications to this attribute do not affect active clients and servers.

---

tuxTdomainLoadBalance

Syntax: INTEGER { yes(1) | no(2) }

Access: read-write

Description: yes(1)

- Load balancing is on.

no(2)

- Load balancing is off.

Note: Runtime modifications to this attribute do not affect active clients and servers.

---

tuxTdomainNotify

Syntax: INTEGER { dipin(1) | signal(2) | ignore(3) }

Access: read-write

Description: Default notification detection method used by the system for unsolicited messages sent to client processes. This default value can be overridden on a per-client basis using the appropriate tpinit(3) flag value. Note that once unsolicited messages are detected, they are made available to the application through the application defined unsolicited message handling routine identified through the tpsetunsol(3) function.

dipin(1)
The value dipin(1) specifies that dip-in-based notification detection should be used. This means that the system only detects notification messages on behalf of a client process while within ATMI calls. The point of detection within any particular ATMI call is not defined by the system, and dip-in detection does not interrupt blocking system calls. dipin(1) is the default notification detection method.

signal(2)

The value signal(2) specifies that signal-based notification detection should be used. This means that the system sends a signal to the target client process after the notification message has been made available. The system installs a signal catching routine on behalf of clients that select this method of notification.

ignore(3)

The value ignore(3) specifies that by default, notification messages are to be ignored by application clients. This would be appropriate in applications where only clients that request notification at tpinit(3) time should receive unsolicited messages.

Note: Runtime modifications to this attribute do not affect active clients. All signaling of client processes is done by administrative system processes and not by application processes. Therefore, only clients running with the same UNIX system user identifier can be notified by use of the signal(2) method.

tuxTdomainSystemAccess

Syntax

```
INTEGER { fastpath(1) | protected(2) | fastpath-no-override(3) | protected-no-override(4) }
```

Access read-write

Description Default mode used by System/T libraries within application processes to gain access to System/T’s internal tables.

fastpath(1)

Specifies that System/T’s internal tables are accessible by System/T libraries via unprotected shared memory for fast access.

protected(2)
Specifies that System/T’s internal tables are accessible by System/T libraries through protected shared memory for safety against corruption by application code.

`fastpath-no-override(3)` or `protected-no-override(4)`

These can be specified to indicate that the mode selected cannot be overridden by an application process that uses flags available for use with `tpinit(3)`.

**Note:** Updates to this attribute value in a running application affect only newly started clients and newly configured `tuxtSrvrTbl` objects.

### tuxTdomainOptions

**Syntax**
```
INTEGER { lan(1) | migrate(2) | accstats(3) | lan-migrate(4) |
lan-accstats(5) | migrate-accstats(6) | lan-migrate-accstats(7) |
one(8) }
```

**Access** read-write

**Description** Comma separated list of application options in effect. Valid options are defined as follows:

- `lan(1)`
  - Networked application.

- `migrate(2)`
  - Allow server group migration.

- `accstats(3)`
  - Exact statistics (single-machine(1) mode only).

**Note:** Only the `accstats(3)` can be set or reset in an active application.

### tuxTdomainSignal

**Syntax**
```
INTEGER { sigusr1(1) | sigusr2(2) }
```

**Access** read-write

**Description** Signal to be used for signal-based notification (see `tuxtDomainNotify` above).
### txTdomainSecurity

**Syntax**
- INTEGER DisplayString

**Access**
- read-write

**Description**

Type of application security. The format is:

```
security_mode[/app_password]
```

where `security_mode` can have the following values:

- NONE
- APP_PW
- USER_AUTH
- ACL
- MANDATORY_ACL

**app_password**

A string value of NONE for this attribute indicates that security is/will be turned off.

**APP_PW**

The value APP_PW/app_password indicates that application password security is enforced. Clients must provide the application password during initialization.

**USER_AUTH**

The value USER_AUTH is similar to APP_PW, but indicates also that per-user authentication is done during client initialization.

**ACL**

The value ACL is similar to USER_AUTH, but also indicates that access control checks are done on service names, queue names, and event names. If an associated ACL is not found for a name, it is assumed that permission is granted.

**MANDATORY_ACL**

The value MANDATORY_ACL is similar to ACL, but permission is denied if an associated ACL is not found for the name.
This value is needed whenever security_mode is being set to anything but NONE. To change the value of app_password, SET this object to:

```
current_security_mode/new_password
```

On a GET operation, this object only returns the security mode; the password is not returned.

**tuxTdomainAuthsvc**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>DisplayString (SIZE (1..15))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>Application authentication service invoked by the system for each client that joins the system. This attribute is ignored if the tuxTdomainSecurity attribute is set to NONE or to APP-PW.</td>
</tr>
</tbody>
</table>

**tuxTdomainScanUnit**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER (0..60)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>Interval of time (in seconds) between periodic scans by the system. Periodic scans are used to detect old transactions and timed-out blocking calls within service requests. The tuxTdomainBBLQuery, tuxTdomainBlockTime, tuxTdomainDBBWait, and tuxTdomainSanityScan objects are multipliers of this value. Passing a value of 0 for this attribute on a SET operation causes the attribute to be reset to its default value.</td>
</tr>
</tbody>
</table>

**tuxTdomainBBLQuery**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER (0..32767)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>Multiplier of the tuxTdomainScanUnit object that indicates time between DBBL status checks on registered BBLs. The DBBL checks to ensure that all BBLs have reported within the tuxTdomainBBLQuery cycle. If a BBL has not been heard from, the DBBL sends a message to that BBL asking for status. If no reply is received, the BBL is partitioned. Passing a value of 0 for this attribute on a SET operation causes the attribute to be reset to its default value. This attribute value should be set to at least twice the value set for tuxTdomainSanityScan.</td>
</tr>
</tbody>
</table>
**tuxTdomainBlockTime**

**Syntax**  INTEGER (0..32767)

**Access**  read-write

**Description**  Multiplier of the `tuxTdomainScanUnit` object that indicates the minimum amount of time a blocking ATMI call blocks before timing out. Passing a value of 0 for this attribute on a `SET` operation causes the attribute to be reset to its default value.

**tuxTdomainDBBLWait**

**Syntax**  INTEGER (0..32767)

**Access**  read-write

**Description**  Multiplier of the `tuxTdomainScanUnit` object that indicates the maximum amount of time a DBBL should wait for replies from its BBLs before timing out. Passing a value of 0 for this attribute on a `SET` operation causes the attribute to be reset to its default value.

**tuxTdomainSanityScan**

**Syntax**  INTEGER (0..32767)

**Access**  read-write

**Description**  Multiplier of the `tuxTdomainScanUnit` object that indicates the time interval between basic sanity checks of the system. Sanity checking includes client/server viability checks done by each BBL for clients/servers running on the local machine as well as BBL status check-ins (`multi-machine(2)` mode only). Passing a value of 0 for this attribute on a `SET` operation causes the attribute to be reset to its default value.

**tuxTdomainCurDRT**

**Syntax**  INTEGER (0..32767)

**Access**  read-only

**Description**  Current number of bulletin board routing table entries in use.
tuxTdomainCurGroups
Syntax INTEGER (0..32767)
Access read-only
Description Current number of bulletin board server group table entries that are in use.

tuxTdomainCurMachines
Syntax INTEGER (0..32767)
Access read-only
Description Current number of configured machines.

tuxTdomainCurQueues
Syntax INTEGER (0..32767)
Access read-only
Description Current number of bulletin board queue table entries that are in use.

tuxTdomainCurRFT
Syntax INTEGER (0..32767)
Access read-only
Description Current number of bulletin board routing criteria range table entries that are in use.

tuxTdomainCurRTdata
Syntax INTEGER (0..32767)
Access read-only
Description Current size of routing table string pool.
**tuxTdomainCurServers**

Syntax: INTEGER (0..32767)
Access: read-only
Description: Current number of bulletin board server table entries that are in use.

**tuxTdomainCurServices**

Syntax: INTEGER (0..32767)
Access: read-only
Description: Current number of bulletin board service table entries that are in use.

**tuxTdomainCursType**

Syntax: INTEGER (0..32767)
Access: read-only
Description: Current number of bulletin board subtype table entries that are in use.

**tuxTdomainCurType**

Syntax: INTEGER (0..32767)
Access: read-only
Description: Current number of bulletin board type table entries that are in use.

**tuxTdomainHwDRT**

Syntax: INTEGER (0..32767)
Access: read-only
Description: High water number of bulletin board routing table entries that are in use.
### `tuxTdomainHwGroups`

**Syntax**
INTEGER (0..32767)

**Access**
read-only

**Description**
High water number of bulletin board server group table entries that are in use.

### `tuxTdomainHwMachines`

**Syntax**
INTEGER (0..32767)

**Access**
read-only

**Description**
High water number of configured machines.

### `tuxTdomainHwQueues`

**Syntax**
INTEGER (0..32767)

**Access**
read-only

**Description**
High water number of bulletin board queue table entries that are in use.

### `tuxTdomainHwRFT`

**Syntax**
INTEGER (0..32767)

**Access**
read-only

**Description**
High water number of bulletin board routing criteria range table entries that are in use.

### `tuxTdomainHwRTdata`

**Syntax**
INTEGER (0..32767)

**Access**
read-only

**Description**
High water size of routing table string pool.
tuxTdomainHwServers

Syntax INTEGER (0..32767)
Access read-only
Description High water number of bulletin board server table entries that are in use.

tuxTdomainHwServices

Syntax INTEGER (0..32767)
Access read-only
Description High water number of bulletin board service table entries that are in use.

tuxTdomainMaxNetGroups

Syntax INTEGER (1..8191)
Access read-write
Description The maximum number of groups that can be configured. This object is only supported on Tuxedo 6.4 or later.

wleMaxObjects

Syntax INTEGER
Access read-write
Description The default maximum number of active objects that can be accommodated in the Active Object Map tables in the WLE bulletin board.

Note: This object is supported for WLE applications only.
wleMaxInterfaces

Syntax  INTEGER (1..32765)
Access  read-write
Description  Specifies the maximum number of interfaces that can be accommodated in the interface table of the bulletin board. If not specified, the default is 100.

All instances of an interface occupy and re-use the same slot in the interface table in the bulletin board. For example, if server SVR1 advertises interfaces IF1 and IF2, SVR2 advertises IF2 and IF3, and SVR3 advertises IF3 and IF4, the interface count is 4, not 6, when calculating wleMaxInterfaces.

Note: This object is supported for WLE applications only.

tuxTdomainSignatureAhead

Syntax  INTEGER (1..2147483647)
Access  read-write
Description  Number of seconds a valid signature’s timestamp can be ahead of the local machine’s clock.

wleCurInterfaces

Syntax  INTEGER
Access  read-only
Description  The current number of interface entries used in the bulletin board interface tables.

Note: This object is supported for WLE applications only.

wleHwInterfaces

Syntax  INTEGER
Access  read-only
Description  The high water mark for the number of interface entries used in the bulletin board interface tables.
**Note:** This object is supported for WLE applications only.

**tuxTdomainSignatureBehind**

- **Syntax:** INTEGER (1..2147483647)
- **Access:** read-write
- **Description:** Number of seconds a valid signature’s timestamp can be behind the local machine’s clock.

**tuxTdomainEncryptionRequired**

- **Syntax:** INTEGER {yes (1) | no (2)}
- **Access:** read-write
- **Description:** If set to “yes,” every application service in this domain requires an encrypted input message buffer.

**tuxTdomainSignatureRequired**

- **Syntax:** INTEGER {yes (1) | no (2)}
- **Access:** read-write
- **Description:** If set to “yes,” every application service in this domain requires a valid digital signature on its input message buffer.
tuxtTgroupTable

The tuxtTgroupTable group represents application attributes pertaining to a particular server group. These attribute values represent group identification, location, and DTP information. The index for this table is tuxtTgroupNo. To create a new row, it is necessary to issue a SET request for a non-existing instance that at least specifies values for tuxtTgroupName and tuxtTgroupLMID.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxtTgroupName</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.1</td>
</tr>
<tr>
<td>tuxtTgroupNo</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.2</td>
</tr>
<tr>
<td>tuxtTgroupLMID</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.3</td>
</tr>
<tr>
<td>tuxtTgroupState</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.4</td>
</tr>
<tr>
<td>tuxtTgroupCurLMID</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.5</td>
</tr>
<tr>
<td>tuxtTgroupCloseInfo</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.6</td>
</tr>
<tr>
<td>tuxtTgroupOpenInfo</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.7</td>
</tr>
<tr>
<td>tuxtTgroupTMScount</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.8</td>
</tr>
<tr>
<td>tuxtTgroupTMSname</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.9</td>
</tr>
<tr>
<td>tuxtTgroupEncryptionRequired</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.20</td>
</tr>
<tr>
<td>tuxtTgroupSignatureRequired</td>
<td>.1.3.6.1.4.1.140.300.4.1.1.30</td>
</tr>
</tbody>
</table>

tuxtTgroupName

Syntax: DisplayString(SIZE(1..30))

Access: read-write
Description Logical name of the server group. The group name must be unique within all group names in the `tuxTgroupTable` class and `tuxTgroupLMID` values in the `tuxTmachineTable` class. Server group names cannot contain an asterisk (*), comma, or colon.

**Note:** This object can be set only during row creation.

### `tuxTgroupNo`

**Syntax** INTEGER (1..29999)

**Access** read-write

**Description** Group number associated with this server group.

**Note:** This object can be set only during row creation.

### `tuxTgroupLMID`

**Syntax** DisplayString (SIZE (1..61))

**Access** read-write

**Description** `DisplayString` is in the format: `LMID1[,LMID2]`

- **LMID1**
  - Is the primary machine logical machine identifier for this server group and is in the range from one to sixty-one characters.

- **LMID2**
  - Is the optional secondary logical machine identifier and is in the range from one to sixty-one characters.

The secondary LMID indicates the machine to which the server group can be migrated (if the MIGRATE option is specified in the `tuxTdomainOptions` attribute). A single LMID specified on a GET operation matches either the primary or secondary LMID. Note that the location of an active group is available in the `tuxTgroupCurlMID` object. Logical machine identifiers specified with the `tuxTgroupLMID` object must already be configured.

**Note:** Modifications to this attribute for an active object can only change the backup LMID designation for the group.
**tuxTgroupTable**

**tuxTgroupState**

**Syntax**

INTEGER { active(1) \| inactive(2) \| migrating(3) \| invalid(4) \| re-active(5) \| suspend-services(6) \| resume-services(7) }

**Access**

read-write

**Description**

The values for **GET** and **SET** operations are as follows:

**GET**: \{active(1)\|inactive(2)\|migrating(3)\}

A **GET** operation retrieves configuration and runtime information for the selected **tuxTgroupTable** object(s). The following states indicate the meaning of a **tuxTgroupState** returned in response to a **GET** request. States not listed are not returned.

- **active(1)**
  
  **tuxTgroupTable** object defined and active (TMS and/or application servers). Server groups with non-0 length values for the **tuxTgroupTMSname** attribute are considered active if the TMSs associated with the group are active. Otherwise, a group is considered active if any server in the group is active.

- **inactive(2)**
  
  **tuxTgroupTable** object defined and inactive.

- **migrating(3)**
  
  **tuxTgroupTable** object defined and currently in a state of migration to the secondary logical machine. The secondary logical machine is the one listed in **tuxTgroupLMID** that does not match **tuxTgroupCurLMID**.

**SET**: \{active(1)\|inactive(2)\|migrating(3)\|invalid(4)\|re-active(5)\|suspend-services(6)\|resume-services(7)\}

A **SET** operation updates configuration and runtime information for the selected **tuxTgroupTable** object. The following states indicate the meaning of a **tuxTgroupState** set in a **SET** request. States not listed cannot be set.

- **active(1)**
  
  Activate the **tuxTgroupTable** object. State change is allowed only when the group is in the **inactive(2)** or **migrating(3)** state. If the group is currently in the **inactive(2)** state and the primary logical machine is active, then TMS and application servers are started on the primary logical machine; otherwise, if the secondary logical machine is active, the TMS and
application servers are started on the secondary logical machine. If neither machine is active, then the request fails. If the group is currently in the migrating(3) state, then the active secondary logical machine (identified as the alternate to tuxTgroupCurLMID in the tuxTgroupLMID list), if it is active, is used to start TMS and application servers. Otherwise, the request fails. Successful return leaves the object in the active(1) state.

inactive(2)

Deactivate the tuxTgroupTable instance. TMS and application servers are deactivated. State change is allowed only when the group is in the active(1) or migrating(3) state. Successful return leaves the object in the inactive(2) state.

migrating(3)

Deactivate the tuxTgroupTable object on its active primary logical machine (tuxTgroupCurLMID) and prepare the group to be migrated to the secondary logical machine. State change is allowed only when the group is in the active(1) state. Successful return leaves the object in the migrating(3) state.

invalid(4)

Delete tuxTgroupTable object for application. State change is allowed only when the group is in the inactive(2) state. Successful return leaves the object in the invalid(4) state.

re-active(5)

Identical to a transition to the active(1) state except that this state change is also allowed in the active(1) state as well as the inactive(2) and migrating(3) states.

suspend-services(6)

Suspend the application services in the group. A SET operation to this state is allowed only when the group is in the active(1) state. The operation leaves the group in active(1) state but with all its application services in a suspended state. This state is only available in Tuxedo 6.4 and later.

resume-services(7)

Unsuspend and resume all application services that are marked suspended in the group. This operation is allowed only when the group is in the active(1) state. The operation leaves the group in the active(1) state.
Note that his operation fails in an application environment that includes any machine where Tuxedo 6.3 or earlier applications are active.

**tuxTgroupCurLMID**

**Syntax**  
`DisplayString (SIZE (1..30))`

**Access**  
read-only

**Description**  
Current logical machine on which the server group is running. This attribute is not returned for server groups that are not active.

**tuxTgroupCloseInfo**

**Syntax**  
`DisplayString (SIZE (0..256))`

**Access**  
read-write

**Description**  
If a non-0 length value other than TMS is specified for the `tuxTgroupTMSname` object, then this object value indicates the resource manager-dependent information needed to terminate access to the resource manager. Otherwise, this attribute value is ignored.

The format for this object value is dependent on the requirements of the vendor providing the underlying resource manager. The information required by the vendor must be prefixed with `rm_name:`, which is the published name of the vendor’s transaction (XA) interface followed immediately by a colon (`:`).

A 0-length string value for this attribute means that the resource manager for this group (if specified) does not require any application-specific information to close access to the resource.

**Note:**  
Runtime modifications to this attribute do not affect active servers in the group.

**tuxTgroupOpenInfo**

**Syntax**  
`DisplayString (SIZE (0..256))`

**Access**  
read-write

**Description**  
If a non-0 length value other than TMS is specified for the `tuxTgroupTMSname` object, this object value indicates the resource manager-dependent information needed to initiate access to the resource manager. Otherwise, this object value is ignored.
The format for this object value is dependent on the requirements of the vendor that provides the underlying resource manager. The information required by the vendor must be prefixed with \texttt{rm\_name:}, which is the published name of the vendor’s transaction (XA) interface followed immediately by a colon (:).

A 0-length string value for this attribute means that the resource manager for this group (if specified) does not require any application-specific information to open access to the resource.

\textbf{Note:} Runtime modifications to this attribute do not affect active servers in the group.

\texttt{tuxTgroupTMScount}

\begin{itemize}
  \item \textbf{Syntax:} \texttt{INTEGER (0..11)}
  \item \textbf{Access:} read-write
  \item \textbf{Description:} If a non-0 length value is specified for the \texttt{tuxTgroupTMSname} object, this object value indicates the number of transaction manager servers to start for the associated group. Otherwise, this object value is ignored.
\end{itemize}

\texttt{tuxTgroupTMSname}

\begin{itemize}
  \item \textbf{Syntax:} \texttt{DisplayString (SIZE (0..78))}
  \item \textbf{Access:} read-write
  \item \textbf{Description:} Transaction manager server \texttt{a.out} associated with this group. This parameter must be specified for any group entry whose servers participate in distributed transactions (transactions across multiple resource managers and possibly machines that are started with \texttt{tpbegin(3)} and ended with \texttt{tpcommit(3)} or \texttt{tpabort(3)}). The value \texttt{TMS} is reserved to indicate use of the null XA interface. If a non-empty value other than \texttt{TMS} is specified, a \texttt{tuxTmachineTlogDevice} must be specified for the machine(s) associated with the primary and secondary logical machines for this object.

  A unique server identifier is selected automatically for each TM server, and the servers are restartable an unlimited number of times.
\end{itemize}
### tuxTgroupEncryptionRequired

| Syntax         | Integer { yes (1) | no (2) } |
|----------------|-------------------|
| Access         | read-write        |
| Description    | If set to "yes," every application service in this group requires an encrypted message buffer. |

### tuxTgroupSignatureRequired

| Syntax         | Integer { yes (1) | no (2) } |
|----------------|-------------------|
| Access         | read-write        |
| Description    | If set to "yes," every application service in this group requires a valid digital signature on its input message buffer. |
The `tuxTmachineTable` group represents application attributes pertaining to a particular machine. These attribute values represent machine characteristics, per-machine sizing, statistics, customization options, and UNIX system filenames. This group is available for configured-inactive as well as configured-active machines in the application. The index into this table is `tuxTmachinePmid`. To create a new row, issue a SET request for a non-existing row that specifies at least the values for `tuxTmachineLmid`, `tuxTmachineTuxDir`, `tuxTmachineTuxConfig`, and `tuxTmachineAppDir`. For a multi-machine Tuxedo application, `tuxTmachineNaddr`, `tuxTmachineNlsAddr`, and `tuxTmachineBridge` must also be specified.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTmachinePmid</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.1</td>
</tr>
<tr>
<td><code>tuxTmachineLmid</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.2</td>
</tr>
<tr>
<td><code>tuxTmachineTuxConfig</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.3</td>
</tr>
<tr>
<td><code>tuxTmachineTuxDir</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.4</td>
</tr>
<tr>
<td><code>tuxTmachineAppDir</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.5</td>
</tr>
<tr>
<td><code>tuxTmachineState</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.6</td>
</tr>
<tr>
<td><code>tuxTmachineUid</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.7</td>
</tr>
<tr>
<td><code>tuxTmachineGid</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.8</td>
</tr>
<tr>
<td><code>tuxTmachineEnvFile</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.9</td>
</tr>
<tr>
<td><code>tuxTmachinePerm</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.10</td>
</tr>
<tr>
<td><code>tuxTmachineUlogPfx</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.11</td>
</tr>
<tr>
<td><code>tuxTmachineType</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.12</td>
</tr>
<tr>
<td><code>tuxTmachineMaxAccessers</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.13</td>
</tr>
<tr>
<td><code>tuxTmachineMaxConv</code></td>
<td>.1.3.6.1.4.1.140.300.5.1.1.14</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Object ID</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>tuxTmachineMaxGtt</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.15</td>
</tr>
<tr>
<td>tuxTmachineMaxWsClients</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.16</td>
</tr>
<tr>
<td>tuxTmachineMaxAclCache</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.17</td>
</tr>
<tr>
<td>tuxTmachineTlogDevice</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.18</td>
</tr>
<tr>
<td>tuxTmachineTlogName</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.19</td>
</tr>
<tr>
<td>tuxTmachineTlogSize</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.20</td>
</tr>
<tr>
<td>tuxTmachineBridge</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.21</td>
</tr>
<tr>
<td>tuxTmachineNaddr</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.22</td>
</tr>
<tr>
<td>tuxTmachineNlsaddr</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.23</td>
</tr>
<tr>
<td>tuxTmachineCmpLimit</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.24</td>
</tr>
<tr>
<td>tuxTmachineTmNetLoad</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.25</td>
</tr>
<tr>
<td>tuxTmachineSpinCount</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.26</td>
</tr>
<tr>
<td>tuxTmachineRole</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.27</td>
</tr>
<tr>
<td>tuxTmachineMinor</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.28</td>
</tr>
<tr>
<td>tuxTmachineRelease</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.29</td>
</tr>
<tr>
<td>tuxTmachineMaxPendingBytes</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.30</td>
</tr>
<tr>
<td>wleMaxMachineObjects</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.35</td>
</tr>
<tr>
<td>tuxTmachineEncryptionRequired</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.50</td>
</tr>
<tr>
<td>tuxTmachineSignatureRequired</td>
<td>.1.3.6.1.4.1.140.300.5.1.1.60</td>
</tr>
</tbody>
</table>
**tuxTmachinePmid**

Syntax: DisplayString (SIZE (1..30))

Access: read-write

Description: Physical machine identifier. This identifier should match the UNIX system nodename returned by the `uname -n` command when run on the identified system. For a Windows NT system, this identifier should match the computer name and the name configured with the name server.

**Note:** This object can be set only during row creation.

**tuxTmachineLmid**

Syntax: DisplayString (SIZE (1..30))

Access: read-write

Description: Logical machine identifier.

**Note:** This object can be set only during row creation.

**tuxTmachineTuxConfig**

Syntax: DisplayString (SIZE (2..78))

Access: read-write

Description: Absolute pathname of the file or device where the binary Tuxedo System/T configuration file is found on this machine. The administrator need only maintain one such file, namely the one identified by the `tuxTmachineTuxConfig` attribute value on the master machine. The information contained in this file is automatically propagated to all other `tuxTmachineTable` objects as they are activated. See `tuxTmachineEnvFile` for a discussion of how this attribute value is used in the environment.

**tuxTmachineTuxDir**

Syntax: DisplayString (SIZE (2..78))

Access: read-write
### tuxTmachineTable

**Description**
Absolute pathname of the directory where the Tuxedo System/T software is found on this machine. See tuxTmachineEnvFile below for a discussion of how this attribute value is used in the environment.

### tuxTmachineAppDir

**Syntax**
\[\text{DisplayString (SIZE (2..78))}\]

**Access**
read-write

**Description**
Colon-separated list of application directory absolute pathnames. The first directory serves as the current directory for all application and administrative servers booted on this machine. All directories in the list are searched when application servers are started. See tuxTmachineEnvFile for a discussion of how this attribute value is used in the environment.

### tuxTmachineState

**Syntax**
\[\text{INTEGER \{ active(1) | inactive(2) | partitioned(3) | invalid(4) | re-activate(5) | cleaning(7) \}}\]

**Access**
read-write

**Description**
The values for GET and SET operations are as follows:

GET: \{active(1)|inactive(2)|partitioned(3)\}

A GET operation retrieves configuration and runtime information for the selected tuxTmachineTable instance(s). The following states indicate the meaning of a tuxTmachineState returned in response to a GET request.

States not listed are not returned

- **active(1)**
  tuxTmachineTable instance defined and active (administrative servers, that is, DBBL, BBL, and BRIDGE).

- **inactive(2)**
  tuxTmachineTable instance defined and inactive.

- **partitioned(3)**
  tuxTmachineTable instance defined, listed in accessible bulletin boards as active, but currently unreachable.
SET: {active(1)|inactive(2)|invalid(4)|re-activate(5)|cleaning(7)}

A SET operation updates configuration and runtime information for the selected tuxTmachineTable instance. The following states indicate the meaning of a tuxTmachineState set in a SET request. States not listed cannot be set.

active(1)
Activate the tuxTmachineTable instance. Necessary administrative servers such as the DBBL, BBL, and BRIDGE are started on the indicated site as well as application servers configured to run on that site. State change is allowed only when the machine is in the inactive(2) state. Successful return leaves the object in the active(1) state.

inactive(2)
Deactivate the tuxTmachineTable instance. Necessary administrative servers such as the BBL and BRIDGE are stopped on the indicated site as well as application servers running on that site. State change allowed only when the machine is in the active(1) state and when no other application resources are active on the indicated machine. Successful return leaves the object in the inactive(2) state.

invalid(4)
Delete tuxTmachineTable instance for application. State change is allowed only when the machine is in the inactive(2) state. Successful return leaves the object in the invalid(4) state.

re-activate(5)
Activate the tuxTmachineTable instance. Necessary administrative servers such as the DBBL, BBL, and BRIDGE are started on the indicated site. State change is allowed only when the machine is in either the active(1) or inactive(2) state. Successful return leaves the object in the active(1) state.

cleaning(7)
Initiate cleanup/scanning activities on and relating to the indicated machine. If there are dead clients or servers on the machine, they are detected at this time. If the machine has been partitioned from the application master site, then global bulletin board entries for that machine are removed. This combination is allowed when the application is in the active(1) state and the tuxTmachineTable instance is in either the active(1) or
partitioned(3) state. Successful return for a non-partitioned machine leaves the state unchanged. Successful return for a partitioned machine leaves the object in the **inactive(2)** state.

**Note:** State change to **inactive(2)** is allowed only for non-master machines. The master site administrative processes are deactivated through the `tuxTdomain` class.

### tuxTmachineUid

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>UNIX system user-identifier for the Tuxedo System/T application administrator on this machine. Administrative commands such as <code>tmboot(1)</code>, <code>tmshutdown(1)</code>, and <code>tmadmin(1)</code> must run as the indicated user on this machine. Application and administrative servers on this machine are started as this user.</td>
</tr>
</tbody>
</table>

**Note:** This is a UNIX system-specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.

### tuxTmachineGid

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>UNIX system group identifier for the Tuxedo System/T application administrator on this machine. Administrative commands such as <code>tmboot(1)</code>, <code>tmshutdown(1)</code>, and <code>tmadmin(1)</code> must run as part of the indicated group on this machine. Application and administrative servers on this machine are started as part of this group.</td>
</tr>
</tbody>
</table>

**Note:** This is a UNIX system-specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.

### tuxTmachineEnvFile

<table>
<thead>
<tr>
<th>Syntax</th>
<th>DisplayString (SIZE (2..78))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>Environment file for clients and servers running on this machine.</td>
</tr>
</tbody>
</table>
tuxTmachinePerm

Syntax  
DisplayString (SIZE(1..9))

Access  
read-write

Description  
UNIX system permissions associated with the shared memory bulletin board created on this machine. Default UNIX system permissions for system and application message queues.

Note: Modifications to this attribute for an active object do not affect running servers or clients.

Note: This is a UNIX system-specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.

tuxTmachineUlogPfx

Syntax  
DisplayString (SIZE(0..78))

Access  
read-write

Description  
Absolute pathname prefix of the path for the userlog(3) file on this machine. The userlog(3) file name is formed by appending the string .mmddyy to the tuxTmachineUlogPfx attribute value. mmddyy represents the month, day, and year that the messages were generated. All application and system userlog(3) messages generated by clients and servers running on this machine are directed to this file.

Note: Modifications to this attribute for an active object do not affect running servers or clients.

tuxTmachineType

Syntax  
DisplayString (SIZE(1..15))

Access  
read-write

Description  
Machine type. This attribute is used to group machines into classes of like data representations. Data encoding is not performed when communicating between machines of identical types. This attribute can be given any string value; values are used only for comparison. Distinct tuxTmachineType attributes should be set when the application spans a heterogeneous network of machines or when compilers
generate dissimilar structure representations. The default value for this attribute, a 0-length string, matches any other machine with a 0-length string as its `tuxTmachineType` attribute value.

**tuxTmachineMaxAccessers**

- **Syntax**: INTEGER (1..32767)
- **Access**: read-write
- **Description**: Maximum number of clients and servers that can have access to the bulletin board on this machine at one time. System administration processes such as the BBL and tmadmin need not be accounted for in this figure, but all application servers and clients and TMS servers should be counted. If the application is booting workstation listeners on this site, then both the listeners and the potential number of workstation handlers that can be booted should be counted.

**tuxTmachineMaxConv**

- **Syntax**: INTEGER (0..32767)
- **Access**: read-write
- **Description**: Maximum number of simultaneous conversations in which clients and servers on this machine can be involved.

**tuxTmachineMaxGtt**

- **Syntax**: INTEGER (0..32767)
- **Access**: read-write
- **Description**: Maximum number of simultaneous global transactions in which this machine can be involved.
tuxTmachineMaxWsClients

**Syntax**
INTEGER (0..32767)

**Access**
read-write

**Description**
Number of entries for accessers on this machine to be reserved for workstation clients. The number specified here takes a portion of the total slots for accessers specified with the `tuxTmachineMaxAccessers` attribute. The appropriate setting of this parameter helps to conserve IPC resources because workstation client access to the system is multiplexed through a System/T supplied surrogate, the workstation handler. It is an error to set this number greater than `tuxTmachineMaxAccessers`.

tuxTmachineMaxAclCache

**Syntax**
INTEGER (10..32000)

**Access**
read-write

**Description**
Number of entries in the cache used for ACL entries when `tuxTdomainSecurity` is set to `acl(4)` or `mandatory-acl(5)`. The appropriate setting of this parameter helps to conserve shared memory resources and yet reduce the number of disk access to do ACL checking.

tuxTmachineTlogDevice

**Syntax**
DisplayString (SIZE (0..64))

**Access**
read-write

**Description**
The device (raw slice) or UNIX system file containing the Tuxedo System/T filesystem that holds the DTP transaction log for this machine. The DTP transaction log is stored as a Tuxedo System/T VTOC table on the device. This device or file can be the same as that specified for the `tuxTmachineTuxConfig` attribute for this machine.
**tuxTmachineTlogName**

**Syntax**  
DisplayString (SIZE (0..30))

**Access**  
read-write

**Description**  
The name of the DTP transaction log for this machine. If more than one DTP transaction log exists on the same tuxTmachineTlogDevice, they must have unique names. tuxTmachineTlogName must be different from the name of any other table on the tuxTmachineTlogDevice where the DTP transaction log table is created.

---

**tuxTmachineTlogSize**

**Syntax**  
INTEGER (1..2048)

**Access**  
read-write

**Description**  
The numeric size, in pages, of the DTP transaction log for this machine. The tuxTmachineTlogSize attribute value is subject to limits based on available space in the Tuxedo System/T filesystem identified by the tuxTmachineTlogDevice attribute.

---

**tuxTmachineBridge**

**Syntax**  
DisplayString (SIZE (0..78))

**Access**  
read-write

**Description**  
Device name to be used by the BRIDGE process placed on this logical machine to access the network. This is a required value for participation in a networked application through a TLI-based Tuxedo System/T binary. This attribute is not needed for sockets-based Tuxedo System/T binaries.

---

**tuxTmachineNaddr**

**Syntax**  
DisplayString (SIZE (0..78))

**Access**  
read-write

**Description**  
Specifies the complete network address to be used by the BRIDGE process placed on the logical machine as its listening address. The listening address for a BRIDGE is the means by which it is contacted by other BRIDGE processes participating in the...
application. This attribute must be set if the logical machine is to participate in a networked application, that is, if the LAN option is set in the \texttt{tuxTdomainOptions} attribute value.

If \textit{DisplayString} has the form 0\text{hex-digits} or \textbackslash 0\text{hex-digits}, it must contain an even number of valid hexadecimal digits. These forms are translated internally into a character array that contains the hexadecimal representations of the specified string.

\textbf{tuxTmachineNlsaddr}

\begin{itemize}
  \item **Syntax** \textit{DisplayString} (\texttt{SIZE (0..78)})
  \item **Access** read-write
  \item **Description** Network address used by the \texttt{tlisten(1)} process servicing the network on the node identified by this logical machine. This network address has the same format as that specified for the \texttt{tuxTmachineNaddr} attribute.

  This attribute must be set if the logical machine is to participate in a networked application, that is, if the LAN option is set in the \texttt{tuxTdomainOptions} attribute value.
\end{itemize}

\textbf{tuxTmachineCmpLimit}

\begin{itemize}
  \item **Syntax** \textit{DisplayString}
  \item **Access** read-write
  \item **Description** Threshold message size at which compression occurs for remote traffic and, optionally, local traffic. Remote and local can be either non-negative numeric values or the string \texttt{MAXLONG} that is dynamically translated to the maximum long setting for the machine. Setting only the remote value defaults local to \texttt{MAXLONG}.

  \textbf{Note:} This attribute value is not part of the \texttt{tuxTmachineTable} object for active sites running Tuxedo System/T Release 4.2.2 or earlier. However, site release identification is not determined until runtime, so this attribute can be set and accessed for any inactive object. When a Tuxedo System/T Release 4.2.2 or earlier site is activated, the configured value is not used.
tuxTmachineTable

**tuxTmachineTmNetLoad**

**Syntax**
INTEGER (0..32767)

**Access**
read-write

**Description**
Service load added to any remote service evaluated during load balancing on this machine.

**Note:** This attribute value is not part of the `tuxTmachineTable` object for active sites running Tuxedo System/T Release 4.2.2 or earlier. However, site release identification is not determined until runtime, so this attribute can be set and accessed for any inactive object. When a Tuxedo System/T Release 4.2.2 or earlier site is activated, the configured value is not used.

**tuxTmachineSpinCount**

**Syntax**
INTEGER

**Access**
read-write

**Description**
Spincount used on this machine for pre-ticket user-level semaphore access. Default values are built into the Tuxedo System/T binaries on each machine. For tuning purposes, these defaults can be overridden at runtime using this attribute. The spincount can be reset to the default built-in value for the site by resetting this attribute value to 0.

**Note:** This attribute value is not part of the `tuxTmachineTable` object for active sites running Tuxedo System/T Release 4.2.2 or earlier. However, site release identification is not determined until runtime, so this attribute can be set and accessed for any inactive object. When a Tuxedo System/T Release 4.2.2 or earlier site is activated, the configured value is not used.

**tuxTmachineRole**

**Syntax**
INTEGER { master(1), backup(2), other(3) }

**Access**
read-only

**Description**
The role of this machine in the application.

master(1)
Indicates that this machine is the master machine,
backup(2)
Indicates that it is the backup master machine, and
other(3)
Indicates that the machine is neither the master nor the backup master machine.

tuxTmachineMinor
Syntax INTEGER
Access read-only
Description The Tuxedo System/T minor protocol release number for this machine.

tuxTmachineRelease
Syntax INTEGER
Access read-only
Description The Tuxedo System/T major protocol release number for this machine. This can be different from the tuxTmachineSWrelease for the same machine.

tuxTmachineMaxPendingBytes
Syntax INTEGER
Access read-write
Description Specifies a limit for the amount of space that can be allocated for messages waiting to be transmitted by the BRIDGE process. The minimum value is 100000. This object is supported on Tuxedo 6.4 and later only.

wleMachineMaxObjects
Syntax INTEGER
Access read-write
tuxTmachineTable

Description  The maximum number of objects that can be accommodated in the Active Object Map tables in the bulletin board.

Note: This object is supported for WLE applications only.

tuxTmachineEncryptionRequired

Syntax  INTEGER { yes(1) | no (2) }

Access  read-write

Description  If set to “yes,” every application service on this machine requires an encrypted input message buffer.

tuxTmachineSignatureRequired

Syntax  INTEGER { yes(1) | no (2) }

Access  read-write

Description  If set to “yes,” every application service on this machine requires a valid digital signature on its input message buffer.
This group represents runtime statistics on the local machine if it is active (that is, some component of the application is active on the machine). Objects in this group are only accessible through a Tuxedo SNMP agent installed on the local machine.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTmachineCurAccessers</td>
<td>1.3.6.1.4.1.140.300.5.2.1</td>
</tr>
<tr>
<td>tuxTmachineCurClients</td>
<td>1.3.6.1.4.1.140.300.5.2.2</td>
</tr>
<tr>
<td>tuxTmachineCurConv</td>
<td>1.3.6.1.4.1.140.300.5.2.3</td>
</tr>
<tr>
<td>tuxTmachineCurGTT</td>
<td>1.3.6.1.4.1.140.300.5.2.4</td>
</tr>
<tr>
<td>tuxTmachineCurLoad</td>
<td>1.3.6.1.4.1.140.300.5.2.5</td>
</tr>
<tr>
<td>tuxTmachineCurWsClients</td>
<td>1.3.6.1.4.1.140.300.5.2.6</td>
</tr>
<tr>
<td>tuxTmachineHwAccessers</td>
<td>1.3.6.1.4.1.140.300.5.2.7</td>
</tr>
<tr>
<td>tuxTmachineHwClients</td>
<td>1.3.6.1.4.1.140.300.5.2.8</td>
</tr>
<tr>
<td>tuxTmachineHwConv</td>
<td>1.3.6.1.4.1.140.300.5.2.9</td>
</tr>
<tr>
<td>tuxTmachineHwGTT</td>
<td>1.3.6.1.4.1.140.300.5.2.10</td>
</tr>
<tr>
<td>tuxTmachineHwWsClients</td>
<td>1.3.6.1.4.1.140.300.5.2.11</td>
</tr>
<tr>
<td>tuxTmachineNumConv</td>
<td>1.3.6.1.4.1.140.300.5.2.12</td>
</tr>
<tr>
<td>tuxTmachineNumDequeue</td>
<td>1.3.6.1.4.1.140.300.5.2.13</td>
</tr>
<tr>
<td>tuxTmachineNumEnqueue</td>
<td>1.3.6.1.4.1.140.300.5.2.14</td>
</tr>
<tr>
<td>tuxTmachineNumPost</td>
<td>1.3.6.1.4.1.140.300.5.2.15</td>
</tr>
<tr>
<td>tuxTmachineNumReq</td>
<td>1.3.6.1.4.1.140.300.5.2.16</td>
</tr>
<tr>
<td>tuxTmachineNumSubscribe</td>
<td>1.3.6.1.4.1.140.300.5.2.17</td>
</tr>
<tr>
<td>tuxTmachineNumTran</td>
<td>1.3.6.1.4.1.140.300.5.2.18</td>
</tr>
</tbody>
</table>
### Variable Name Object ID

- `tuxTmachineNumTranAbt` .1.3.6.1.4.1.140.300.5.2.19
- `tuxTmachineNumTranCmt` .1.3.6.1.4.1.140.300.5.2.20
- `tuxTmachineLicExpires` .1.3.6.1.4.1.140.300.5.2.21
- `tuxTmachineLicMaxUsers` .1.3.6.1.4.1.140.300.5.2.22
- `tuxTmachineLicSerial` .1.3.6.1.4.1.140.300.5.2.23
- `tuxTmachinePageSize` .1.3.6.1.4.1.140.300.5.2.24
- `tuxTmachineSWrelease` .1.3.6.1.4.1.140.300.5.2.25
- `tuxTmachineHwAclCache` .1.3.6.1.4.1.140.300.5.2.26
- `tuxTmachineAclCacheHits` .1.3.6.1.4.1.140.300.5.2.27
- `tuxTmachineAclCacheAccess` .1.3.6.1.4.1.140.300.5.2.28
- `tuxTmachineAclFail` .1.3.6.1.4.1.140.300.5.2.29
- `tuxTmachineWkCompleted` .1.3.6.1.4.1.140.300.5.2.30
- `tuxTmachineWkInitiated` .1.3.6.1.4.1.140.300.5.2.31
- `wleMachineCurObjects` .1.3.6.1.4.1.140.300.5.2.36
- `wleMachineHwObjects` .1.3.6.1.4.1.140.300.5.2.41

### `tuxTmachineCurAccessers`

**Syntax**  
INTEGER (0..32767)

**Access**  
read-only

**Description**  
Number of clients and servers that currently access the application either directly on this machine or through a workstation handler on this machine.
**tuxTmachineCurClients**

- **Syntax**: INTEGER (0..32767)
- **Access**: read-only
- **Description**: Number of clients, both native and workstation, currently logged in to this machine.

**tuxTmachineCurConv**

- **Syntax**: INTEGER (0..32767)
- **Access**: read-only
- **Description**: Number of active conversations with participants on this machine.

**tuxTmachineCurGTT**

- **Syntax**: INTEGER (0..32767)
- **Access**: read-only
- **Description**: Number of in use transaction table entries on this machine.

**tuxTmachineCurLoad**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Current service load enqueued on this machine.

*Note: If the tuxTdomainLoadBalance attribute is no(2) or the tuxTdomainModel attribute is multi-machine(2), then an FML32 NULL value (0) is returned.*

**tuxTmachineCurWsClients**

- **Syntax**: INTEGER (0..32767)
- **Access**: read-only
- **Description**: Number of workstation clients currently logged in to this machine.
**tuxTmachineHwAccessers**

Syntax: INTEGER (0..32767)

Access: read-only

Description: High water number of clients and servers accessing the application either directly on this machine or through a workstation handler on this machine.

**tuxTmachineHwClients**

Syntax: INTEGER (0..32767)

Access: read-only

Description: High water number of clients, both native and workstation, logged in to this machine.

**tuxTmachineHwConv**

Syntax: INTEGER (0..32767)

Access: read-only

Description: High water number of active conversations with participants on this machine.

**tuxTmachineHwGTT**

Syntax: INTEGER (0..32767)

Access: read-only

Description: High water number of in use transaction table entries on this machine.

**tuxTmachineHwWsClients**

Syntax: INTEGER (0..32767)

Access: read-only

Description: High water number of workstation clients currently logged in to this machine.
tuxTmachineNumConv

Syntax INTEGER
Access read-only
Description Number of tpconnect(3) operations performed from this machine.

tuxTmachineNumDequeue

Syntax INTEGER
Access read-only
Description Number of tpdequeue(3) operations performed from this machine.

tuxTmachineNumEnqueue

Syntax INTEGER
Access read-only
Description Number of tpenqueue(3) operations performed from this machine.

tuxTmachineNumPost

Syntax INTEGER
Access read-only
Description Number of tppost(3) operations performed from this machine.

tuxTmachineNumReq

Syntax INTEGER
Access read-only
Description Number of tpacall(3) or tpcall(3) operations performed from this machine.
**tuxTmachineNumSubscribe**

Syntax: INTEGER  
Access: read-only  
Description: Number of tpsubscribe(3) operations performed from this machine.

**tuxTmachineNumTran**

Syntax: INTEGER  
Access: read-only  
Description: Number of transactions initiated (tpbegin(3)) from this machine.

**tuxTmachineNumTranAbt**

Syntax: INTEGER  
Access: read-only  
Description: Number of transactions aborted (tpabort(3)) from this machine.

**tuxTmachineNumTranCmt**

Syntax: INTEGER  
Access: read-only  
Description: Number of transactions committed (tpcommit(3)) from this machine.

**tuxTmachineLicExpires**

Syntax: DisplayString (SIZE(0..78))  
Access: read-only  
Description: Expiration date for the binary on the machine or a 0-length string if binary is not a Tuxedo System/T master binary.
tuxTmachineLicMaxUsers

Syntax  INTEGER (0..32767)
Access  read-only
Description  Maximum number of licensed users on that machine, or -1 if binary is not a Tuxedo System/T master binary.

tuxTmachineLicSerial

Syntax  DisplayString (SIZE(0..78))
Access  read-only
Description  Serial number for binary on the machine or a 0-length string if binary is not a Tuxedo System/T master binary.

tuxTmachinePageSize

Syntax  INTEGER
Access  read-only
Description  Disk pagesize used on this machine.

tuxTmachineSWrelease

Syntax  DisplayString (SIZE(0..78))
Access  read-only
Description  Software release for binary on that machine or a 0-length string if binary is not a Tuxedo System/T master binary.

tuxTmachineHwAclCache

Syntax  INTEGER
Access  read-only
Description  High water number of entries used in the ACL cache.
**tuxTmachineAclCacheHits**

**Syntax**
INTEGER

**Access**
read-only

**Description**
Number of accesses to the ACL cache that resulted in a “hit” (that is, the entry was already in the cache).

**tuxTmachineAclCacheAccess**

**Syntax**
INTEGER

**Access**
read-only

**Description**
Number of accesses to the ACL cache.

**tuxTmachineAclFail**

**Syntax**
INTEGER

**Access**
read-only

**Description**
Number of accesses to the ACL cache that resulted in a access control violation.

**tuxTmachineWkCompleted**

**Syntax**
INTEGER

**Access**
read-only

**Description**
Total service load dequeued and processed successfully by servers running on this machine. Note that for long running applications this attribute can wraparound, that is, exceed the maximum value for a long, and start back at 0 again.
tuxTmachineWkInitiated

Syntax  INTEGER
Access    read-only
Description  Total service load enqueued by clients/servers running on this machine. Note that for long running applications this attribute can wraparound, that is, exceed the maximum value for a long, and start back at 0 again.

wleMachineCurObjects

Syntax  INTEGER
Access    read-only
Description  The number of entries in use in the bulletin board object table for this machine.

Note:  This object is supported for WLE applications only.

wleMachineHwObjects

Syntax  INTEGER
Access    read-only
Description  The high water mark of entries used in the bulletin board object table for this machine.

Note:  This object is supported for WLE applications only.
The tuxTmsgTable class represents runtime attributes of the Tuxedo System/T managed UNIX system message queues. Objects in this table are only accessible through a Tuxedo SNMP agent installed on the local machine. tuxTmsgId is the index into this table.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTmsgId</td>
<td>1.3.6.1.4.1.140.300.6.1.1.1</td>
</tr>
<tr>
<td>tuxTmsgState</td>
<td>1.3.6.1.4.1.140.300.6.1.1.2</td>
</tr>
<tr>
<td>tuxTmsgCurTime</td>
<td>1.3.6.1.4.1.140.300.6.1.1.3</td>
</tr>
<tr>
<td>tuxTmsgCbytes</td>
<td>1.3.6.1.4.1.140.300.6.1.1.4</td>
</tr>
<tr>
<td>tuxTmsgCtime</td>
<td>1.3.6.1.4.1.140.300.6.1.1.5</td>
</tr>
<tr>
<td>tuxTmsgLrPid</td>
<td>1.3.6.1.4.1.140.300.6.1.1.6</td>
</tr>
<tr>
<td>tuxTmsgLsPid</td>
<td>1.3.6.1.4.1.140.300.6.1.1.7</td>
</tr>
<tr>
<td>tuxTmsgQbytes</td>
<td>1.3.6.1.4.1.140.300.6.1.1.8</td>
</tr>
<tr>
<td>tuxTmsgQnum</td>
<td>1.3.6.1.4.1.140.300.6.1.1.9</td>
</tr>
<tr>
<td>tuxTmsgRtime</td>
<td>1.3.6.1.4.1.140.300.6.1.1.10</td>
</tr>
<tr>
<td>tuxTmsgStime</td>
<td>1.3.6.1.4.1.140.300.6.1.1.11</td>
</tr>
</tbody>
</table>

**tuxTmsgId**

**Syntax:** INTEGER  
**Access:** read-only  
**Description:** UNIX system message queue identifier.  
**Note:** This is a UNIX system-specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.
**tuxTmsgState**

Syntax

```
INTEGER { active(1) }
```

Access

`read-only`

Description

The values for GET and SET operations are as follows:

**GET:**

```
active(1)
```

A GET operation retrieves runtime information for the selected `tuxTmsgTable` object(s). The following state indicates the meaning of a `tuxTmsgState` returned in response to a GET request. States not listed are not returned.

```
active(1)
```

`tuxTmsgTable` object active. This corresponds exactly to the related `tuxTmachineTable` object being active.

**SET:**

SET operations are not permitted on this class.

**tuxTmsgCurTime**

Syntax

`INTEGER`

Access

`read-only`

Description

Current time, in seconds, since 00:00:00 UTC, January 1, 1970, as returned by the `time(2)` system call on the local host.

**tuxTmsgCbytes**

Syntax

`INTEGER`

Access

`read-only`

Description

Current number of bytes on the queue.
## tuxTmsgCtime

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>Time of the last <code>msgctl(2)</code> operation that changed a member of the <code>msqid_ds</code> structure associated with the queue.</td>
</tr>
</tbody>
</table>

## tuxTmsgLrPid

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>Process identifier of the last process that read from the queue.</td>
</tr>
</tbody>
</table>

## tuxTmsgLsPid

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>Process identifier of the last process that wrote to the queue.</td>
</tr>
</tbody>
</table>

## tuxTmsgQbytes

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>Maximum number of bytes allowed on the queue.</td>
</tr>
</tbody>
</table>

## tuxTmsgQnum

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>Number of messages currently on the queue.</td>
</tr>
</tbody>
</table>
tuxTmsgRtime

Syntax  INTEGER
Access  read-only
Description  Time since the last read from the queue.

tuxTmsgStime

Syntax  INTEGER
Access  read-only
Description  Time since the last write to the queue.
The `tuxTqueueTable` group represents runtime attributes of queues in an application. These attribute values identify and characterize allocated Tuxedo System/T request queues associated with servers in a running application. They also track statistics related to application workloads associated with each queue object. The index into this table is `tuxTqueueRqAddr`. Objects in this table are only accessible through a Tuxedo SNMP agent installed on the local machine.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTqueueRqAddr</code></td>
<td>.1.3.6.1.4.1.140.300.7.1.1.1</td>
</tr>
<tr>
<td><code>tuxTqueueState</code></td>
<td>.1.3.6.1.4.1.140.300.7.1.1.2</td>
</tr>
<tr>
<td><code>tuxTqueueRqId</code></td>
<td>.1.3.6.1.4.1.140.300.7.1.1.3</td>
</tr>
<tr>
<td><code>tuxTqueueSrvrCnt</code></td>
<td>.1.3.6.1.4.1.140.300.7.1.1.4</td>
</tr>
<tr>
<td><code>tuxTqueueTotNqueued</code></td>
<td>.1.3.6.1.4.1.140.300.7.1.1.5</td>
</tr>
<tr>
<td><code>tuxTqueueTotWkQueued</code></td>
<td>.1.3.6.1.4.1.140.300.7.1.1.6</td>
</tr>
<tr>
<td><code>tuxTqueueSource</code></td>
<td>.1.3.6.1.4.1.140.300.7.1.1.7</td>
</tr>
<tr>
<td><code>tuxTqueueNqueued</code></td>
<td>.1.3.6.1.4.1.140.300.7.1.1.8</td>
</tr>
<tr>
<td><code>tuxTqueueWkQueued</code></td>
<td>.1.3.6.1.4.1.140.300.7.1.1.9</td>
</tr>
</tbody>
</table>

**tuxTqueueRqAddr**

**Syntax**  
`DisplayString(SIZE(1..30))`

**Access**  
read-only

**Description**  
Symbolic address of the request queue. Servers with the same `tuxTsrvrRqAddr` attribute value are grouped into a Multiple Server Single Queue (MSSQ) set. Attribute values returned with a `tuxTqueueTable` object apply to all active servers associated with this symbolic queue address.
tuxTqueueState

Syntax    INTEGER { active(1), migrating(2), suspended(3), partitioned(4) }
Access    read-only
Description The values for GET and SET operations are as follows:

GET: {active(1), migrating(2), suspended(3), partitioned(4)}

A GET operation retrieves runtime information for the selected tuxTqueueTable instance(s). The tuxTqueueTable group does not address configuration information directly. Configuration-related attributes discussed here must be set as part of the related tuxTsrvrTbl instances. The following states indicate the meaning of a tuxTqueueState returned in response to a GET request. States not listed are not returned.

active(1)
At least one server associated with this tuxTqueueTable instance is active(1).

migrating(2)
The server(s) associated with this tuxTqueueTable instance is currently in the migrating(2) state. See the tuxTsrvrTbl group for more details on this state.

suspended(3)
The server(s) associated with this tuxTqueueTable instance is currently in the suspended(3) state. See the tuxTsrvrTbl group for more details on this state.

partitioned(4)
The server(s) associated with this tuxTqueueTable instance is currently in the partitioned(4) state. See the tuxTsrvrTbl group for more details on this state.

SET:
A SET operation updates runtime information for the selected tuxTqueueTable object. State changes are not allowed when updating tuxTqueueTable object information. Modification of an existing tuxTqueueTable object is allowed only when the object is in the active(1) state.
**tuxTqueueTable**

**tuxTqueueRqId**

**Syntax**  INTEGER  
**Access**  read-only  
**Description**  UNIX system message queue identifier.

**Note:**  This is a UNIX system specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.

**tuxTqueueSrvrCnt**

**Syntax**  INTEGER  
**Access**  read-only  
**Description**  Number of active servers associated with this queue.

**tuxTqueueTotNqueued**

**Syntax**  INTEGER  
**Access**  read-only  
**Description**  The sum of the queue lengths of this queue while it has been active. This sum includes requests enqueued to and processed by servers that are no longer active on the queue. Each time a new request is assigned to the queue, the sum is incremented by the length of the queue immediately before the new request is enqueued.

**Note:**  If the `tuxTdomainLoadBalance` attribute is `no` (2) or the `tuxTdomainModel` attribute is `multi-machine` (2), then `tuxTqueueTotNqueued` is not returned. In the same configuration, updates to this attribute are ignored. Consequently, when this attribute is returned `tuxTqueueSource` has the same value as the local host.
**tuxTqueueTotWkQueued**

Syntax: INTEGER  
Access: read-only  
Description: The sum of the workloads enqueued to this queue while it has been active. This sum includes requests enqueued to and processed by servers that are no longer active on the queue. Each time a new request is assigned to the queue, the sum is incremented by the workload on the queue immediately before the new request is enqueued.

**Note:** If the **tuxTdomainLoadBalance** attribute is no(2) or the **tuxTdomainModel** attribute is multi-machine(2), then **tuxTqueueTotWkQueued** is not returned. In the same configuration, updates to this attribute are ignored. Consequently, when this attribute is returned **tuxTqueueSource** has the same value as the local host.

**tuxTqueueSource**

Syntax: DisplayString(SIZE(1..30))  
Access: read-only  
Description: Logical machine from which local attribute values are retrieved.

**tuxTqueueNqueued**

Syntax: INTEGER  
Access: read-only  
Description: Number of requests currently enqueued to this queue from the **tuxTqueueSource** logical machine. This value is incremented at enqueue time and decremented when the server dequeues the request.

**Note:** If the **tuxTdomainLoadBalance** attribute is no(2) or the **tuxTdomainModel** attribute is multi-machine(2), then **tuxTqueueNqueued** is not returned. Consequently, when this attribute is returned **tuxTqueueSource** has the same value as the local host.
**tuxTqueueWkQueued**

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Workload currently enqueued to this queue from the `tuxTqueueSource` logical machine. If the `tuxTdomainModel` attribute is set to single-machine(1) and the `tuxTdomainLoadBalance` attribute is set to yes(1), then this attribute reflects the application-wide workload enqueued to this queue. However, if `tuxTdomainModel` is set to multi-machine(2) and `tuxTdomainLoadBalance` is set to yes(1), this attribute reflects the workload enqueued to this queue from the `tuxTqueueSource` logical machine during a recent timespan. This attribute is used for load balancing purposes. In order not discriminate against newly started servers, this attribute value is zeroed out on each machine periodically by the BBL.
The `tuxTroutingTable` group represents configuration objects of routing specifications for an application. These object values identify and characterize application data dependent routing criteria with respect to field names, buffer types, and routing definitions. This table also represents configuration objects for factory-based routing for WLE applications. `wleRoutingFieldType` is valid only for factory-based routing. `tuxTroutingBufType` is valid only for service-based routing.

The index into this table consists of the following attributes: `tuxTroutingName`, `tuxRoutingType`, and `tuxInternalIdx`.

`wleRoutingFieldType` is valid only for factory-based routing. This is supported only for WLE applications.

`tuxTroutingBufType` is valid only for service-based routing (either Tuxedo or WLE applications).

When specifying the index in SET requests, `tuxInternalIdx` is used as an index.

For factory-based routing, `tuxInternalIdx` must always have a value of `-`.

For service-based routing, `tuxInternalIdx` should equal the first 30 characters in `tuxTroutingBufType`.

To create a new row in the table, it is necessary to issue a SET request for a non-existing row specifying the values of all objects applicable to the `tuxRoutingType`.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTroutingName</code></td>
<td>.1.3.6.1.4.1.140.300.8.1.1.1</td>
</tr>
<tr>
<td><code>tuxTroutingBufType</code></td>
<td>.1.3.6.1.4.1.140.300.8.1.1.2</td>
</tr>
<tr>
<td><code>tuxTroutingField</code></td>
<td>.1.3.6.1.4.1.140.300.8.1.1.3</td>
</tr>
<tr>
<td><code>tuxTroutingRanges</code></td>
<td>.1.3.6.1.4.1.140.300.8.1.1.4</td>
</tr>
<tr>
<td><code>tuxTroutingState</code></td>
<td>.1.3.6.1.4.1.140.300.8.1.1.5</td>
</tr>
<tr>
<td><code>tuxRoutingType</code></td>
<td>.1.3.6.1.4.1.140.300.8.1.1.6</td>
</tr>
<tr>
<td><code>wleRoutingFieldType</code></td>
<td>.1.3.6.1.4.1.140.300.8.1.1.7</td>
</tr>
<tr>
<td><code>tuxInternalIdx</code></td>
<td>.1.3.6.1.4.1.140.300.8.1.1.8</td>
</tr>
</tbody>
</table>
**tuxTroutingName**

**Syntax**  
DisplayString (SIZE(1..15))

**Access**  
read-write

**Description**  
Routing criterion name.

**Note:** This object can be set only during row creation.

**tuxTroutingBufType**

**Syntax**  
DisplayString (SIZE(1..256))

**Access**  
read-write

**Description**  
List of types and subtypes of data buffers for which this routing entry is valid. A maximum of 32 type/subtype combinations are allowed. The types are restricted to one of FML, VIEW, X_C_TYPE, or X_COMMON. No subtype can be specified for type FML, and subtypes are required for types VIEW, X_C_TYPE, and X_COMMON (* is not allowed). Note that subtype names should not contain semicolon, colon, comma, or asterisk characters. Duplicate type/subtype pairs cannot be specified for the same routing criterion name. More than one routing entry can have the same criterion name as long as the type/subtype pairs are unique. If multiple buffer types are specified for a single routing entry, the data types of the routing field for each buffer type must be the same.

**Note:** This object is applicable only for service-based routing.

**Note:** This object can be set only during row creation.

**tuxTroutingField**

**Syntax**  
DisplayString (SIZE(1..30))

**Access**  
read-write

**Description**  
Routing field name.
For Service-based Routing: This field is assumed to be an FML buffer or view field name that is identified in an FML field table (using the FLDTBLDIR and FIELDTBLS environment) or an FML view table (using the VIEWDIR and VIEWFILES environment), respectively. This information is used to get the associated field value for data dependent routing during the sending of a message.

For factory-based routing: This is assumed to be a field that is specified in an NVList parameter to:

```
PortableServer::POA::create_reference_with_criteria
```

for an interface that has this factory routing criteria associated with it. See the WLE documentation for more details.

**tuxTroutingRanges**

**Syntax**

```
DisplayString(SIZE(1..2048))
```

**Access**

read-write

**Description**

The ranges and associated server groups for a routing criterion are as follows:

- **criterion**: range: group
- **range**: value | lower - upper | *
- **lower**: value
- **upper**: value
- **value**: MIN | MAX | numeric | string
- **group**: string | *
- **numeric**: [+ | -]digits[.digits][e | E[ | + | - ] digit
digit: 0-9
digits: digit[digit]

\ can be used to escape the single-quote character in strings.

**lower** must be less than **upper**. A group specified as a string must specify a valid **tuxTgroupName**.

**Note**: Attribute values greater than 256 bytes in length disable interoperability with Tuxedo System/T Release 4.2.2 and earlier.
tuxTroutingState

Syntax  INTEGER { valid(1) | unknown(2) | invalid(3) }
Access  read-write
Description  The values for GET and SET operations are as follows:

GET: valid(1)

A GET operation retrieves configuration information for the selected tuxTroutingTable instance(s). The following state indicates the meaning of a tuxTroutingState returned in response to a GET request. States not listed are not returned.

valid(1)

A tuxTroutingTable instance is defined. Note that this is the only valid state for this class. Routing criteria are never active; rather, they are associated through the configuration with service names and are acted upon at runtime to provide data dependent routing.

SET: invalid(3)

A SET operation updates configuration information for the selected tuxTroutingTable instance. The following state indicates the meaning of a tuxTroutingState set in a SET request. States not listed cannot be set.

invalid(3)

Delete tuxTroutingTable instance for application. State change allowed only when in the valid(1) state. Successful return leaves the object in the invalid(2) state.

tuxRoutingType

Syntax  INTEGER { service(1) | factory(2) }
Access  read-write
Description  service(1)

Specifies that routing criteria apply to data-dependent routing for a BEA Tuxedo service.

factory(2)
Specifies that the routing criterion applies to factory-based routing for a CORBA interface.

**Note:** The routing type affects the validity and possible values for other attributes defined for this table.

**Note:** This object can be set during row creation only.

**wleRoutingFieldType**

**Syntax**

| INTEGER | short(1) | long(2) | float(3) | double(4) | char(5) | string(6) |

**Access** read-write

**Description**

This object specifies the type of `tuxTroutingField` on which this routing criterion is defined. This is valid only for factory-based routing.

**Note:** This object is supported only for WLE.

**Note:** This object can be set only during row creation.

**tuxInternalIdx**

**Syntax**

`DisplayString(SIZE(1..30))`

**Access** read-write

**Description**

This object is used as an index of this table instead of `tuxTroutingBufType` (for service-based routing) or `tuxTroutingField` (for factory-based routing) to reduce the size of the index. Its value, for service-based routing (`tuxRoutingType = service(1)`), is equal to the first 30 characters in `tuxTroutingBufType`.

In case of entries for factory-based routing (`tuxRoutingType = factory(2)`), the value is always `tuxTroutingField`.

**Note:** This object can be set only during row creation.
This group represents configuration and runtime attributes of servers within an application. These attribute values identify and characterize configured servers as well as provide runtime tracking of statistics and resources associated with each server object. The index into this table is provided by the attributes tuxtsrvrGrpNo and tuxtsrvrId. To create a new row in the table, it is necessary to issue a SET request specifying the values of at least tuxtsrvrGrp and tuxtsrvrName.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxtsrvrGrp</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.1</td>
</tr>
<tr>
<td>tuxtsrvrId</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.2</td>
</tr>
<tr>
<td>tuxtsrvrName</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.3</td>
</tr>
<tr>
<td>tuxtsrvrGrpNo</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.4</td>
</tr>
<tr>
<td>tuxtsrvrState</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.5</td>
</tr>
<tr>
<td>tuxtsrvrBaseSrvId</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.6</td>
</tr>
<tr>
<td>tuxtsrvrClOpt</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.7</td>
</tr>
<tr>
<td>tuxtsrvrEnvFile</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.8</td>
</tr>
<tr>
<td>tuxtsrvrGrace</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.9</td>
</tr>
<tr>
<td>tuxtsrvrMaxgen</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.10</td>
</tr>
<tr>
<td>tuxtsrvrMax</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.11</td>
</tr>
<tr>
<td>tuxtsrvrMin</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.12</td>
</tr>
<tr>
<td>tuxtsrvrRcmd</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.13</td>
</tr>
<tr>
<td>tuxtsrvrRestart</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.14</td>
</tr>
<tr>
<td>tuxtsrvrSequence</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.15</td>
</tr>
<tr>
<td>tuxtsrvrSystemAccess</td>
<td>.1.3.6.1.4.1.140.300.20.1.1.16</td>
</tr>
</tbody>
</table>
### tuxTsrvrGrp

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTsrvrConv</td>
<td>1.3.6.1.4.1.140.300.20.1.1.17</td>
</tr>
<tr>
<td>tuxTsrvrReplyQ</td>
<td>1.3.6.1.4.1.140.300.20.1.1.18</td>
</tr>
<tr>
<td>tuxTsrvrRpPerm</td>
<td>1.3.6.1.4.1.140.300.20.1.1.19</td>
</tr>
<tr>
<td>tuxTsrvrRqAddr</td>
<td>1.3.6.1.4.1.140.300.20.1.1.20</td>
</tr>
<tr>
<td>tuxTsrvrRqPerm</td>
<td>1.3.6.1.4.1.140.300.20.1.1.21</td>
</tr>
<tr>
<td>tuxTsrvrGeneration</td>
<td>1.3.6.1.4.1.140.300.20.1.1.22</td>
</tr>
<tr>
<td>tuxTsrvrPid</td>
<td>1.3.6.1.4.1.140.300.20.1.1.23</td>
</tr>
<tr>
<td>tuxTsrvrRpid</td>
<td>1.3.6.1.4.1.140.300.20.1.1.24</td>
</tr>
<tr>
<td>tuxTsrvrRqid</td>
<td>1.3.6.1.4.1.140.300.20.1.1.25</td>
</tr>
<tr>
<td>tuxTsrvrTimeRestart</td>
<td>1.3.6.1.4.1.140.300.20.1.1.26</td>
</tr>
<tr>
<td>tuxTsrvrTimeStart</td>
<td>1.3.6.1.4.1.140.300.20.1.1.27</td>
</tr>
<tr>
<td>tuxTsrvrMinDispatchThreads</td>
<td>1.3.6.1.4.1.140.300.20.1.1.40</td>
</tr>
<tr>
<td>tuxTsrvrMaxDispatchThreads</td>
<td>1.3.6.1.4.1.140.300.20.1.1.50</td>
</tr>
<tr>
<td>tuxTsrvrThreadStackSize</td>
<td>1.3.6.1.4.1.140.300.20.1.1.60</td>
</tr>
<tr>
<td>wleSrvrSrvType</td>
<td>1.3.6.1.4.1.140.300.20.1.1.70</td>
</tr>
</tbody>
</table>

**Syntax**

DisplayString(SIZE(1..30))

**Access**

read-write

**Description**

Logical name of the server group. Server group names cannot contain an asterisk (*), comma, or colon.

**Note:** This object can be set only during row creation.
**tuxTsrvrId**

**Syntax**  INTEGER (1..30001)

**Access** read-write

**Description** Unique (within the server group) server identification number.

**Note:** This object can be set only during row creation.

**tuxTsrvrName**

**Syntax** DisplayString (SIZE(1..78))

**Access** read-write

**Description** Name of the server executable file. The server identified by `tuxTsrvrName` runs on the machine(s) identified by the `tuxTgroupLMID` object for this server’s server group. If a relative pathname is given, the search for the executable file is done first in `tuxTmachineAppDir`, then in `tuxTmachineTuxDir/bin`, then in `/bin` and `/usr/bin`, and then in `<path>`, where `<path>` is the value of the first `PATH=` line that appears in the machine environment file, if one exists. Note that the attribute value returned for an active server is always a full pathname.

**tuxTsrvrGrpNo**

**Syntax** INTEGER (1..30000)

**Access** read-only

**Description** Group number associated with this server’s group.

**tuxTsrvrState**

**Syntax** INTEGER { active(1), inactive(2), migrating(3), cleaning(4), restarting(5), suspended(6), partitioned(7), dead(8), invalid(10) }

**Access** read-write

**Description** The values for GET and SET operations are as follows:
GET: active(1) | inactive(2) | migrating(3) | cleaning(4) | restarting(5) | suspended(6) | partitioned(7) | dead(8)

A GET operation retrieves configuration and runtime information for the selected tuxTsrvrTbl instance(s). The following states indicate the meaning of a tuxTsrvrState returned in response to a GET request. States not listed are not returned.

active(1)

tuxTsrvrTbl instance is defined and active. This is not an indication of whether the server is idle or busy. An active server with a non-0 length tuxTsrvrCurService attribute should be interpreted as a busy server, that is, one that is processing a service request.

inactive(2)

tuxTsrvrTbl instance is defined and inactive.

migrating(3)

tuxTsrvrTbl instance is defined and currently in a state of migration to the server group’s secondary logical machine. The secondary logical machine is the one listed in tuxTgroupLMID attribute that does not match the tuxTgroupCurLMID object.

cleaning(4)

tuxTsrvrTbl instance is defined and currently being cleaned up after by the system due to an abnormal death. Note that restartable servers can enter this state if they exceed tuxTsrvrMaxgen starts/restarts within their tuxTsrvrGrace period.

restarting(5)

tuxTsrvrTbl instance is defined and currently being restarted by the system due to an abnormal death.

suspended(6)

tuxTsrvrTbl instance is defined and currently suspended, pending shutdown.

partitioned(7)

tuxTsrvrTbl instance is defined and active; however, the machine where the server is running is currently partitioned from the tuxTdomainMaster site.
dead(8)

A tuxTsrvrTbl instance is defined, identified as active in the bulletin board, but currently not running due to an abnormal death. This state exists only until the BBL local to the server notices the death and takes action (restarting(5)|cleaning(4)).

SET: {active(1)|inactive(2)|dead(8)|invalid(10)}

A SET operation updates configuration and runtime information for the selected tuxTsrvrTbl instance. The following states indicate the meaning of a tuxTsrvrState set in a SET request. States not listed cannot be set.

active(1)

Activate the tuxTsrvrTbl instance. State change is allowed only when the server is in the inactive(2) state. (Servers in the migrating(3) state must be restarted by setting the tuxTgroupState to active(1).) Successful return leaves the object in the active(1) state.

inactive(2)

Deactivate the tuxTsrvrTbl instance. State change is allowed only when the server is in the active(1) state. Successful return leaves the object in the inactive(2) state.

dead(8)

Deactivate the tuxTsrvrTbl instance by sending the server a SIGTERM signal followed by a SIGKILL signal if the server is still running after 20 seconds. Note that by default, a SIGTERM signal causes the server to initiate orderly shutdown and the server becomes inactive even if it is restartable. If a server is processing a long running service or has chosen to disable the SIGTERM signal, then SIGKILL can be used and is treated by the system as an abnormal termination. State change is allowed only when the server is in the active(1) or suspended(6) state. Successful return leaves the object in the inactive(2), cleaning(4), or restarting(5) state.

invalid(10)

Delete tuxTsrvrTbl instance for application. State change is allowed only when the server is in the inactive(2) state. Successful return leaves the object in the invalid(10) state.
**tuxTsrvrBaseSrvId**

**Syntax**

INTEGER (1..30001)

**Access**

read-only

**Description**

Base server identifier. For servers with a `tuxTsrvrMax` attribute value of 1, this attribute is always the same as `tuxTsrvrId`. However, for servers with a `tuxTsrvrMax` value of greater than 1, this attribute indicates the base server identifier for the set of servers configured identically.

**tuxTsrvrClOpt**

**Syntax**

DisplayString(SIZE(0..256))

**Access**

read-write

**Description**

Command line options to be passed to server when it is activated. See the `servopts(5)` manual page for details.

**Note:** Runtime modifications to this attribute do not affect a running server.

**tuxTsrvrEnvFile**

**Syntax**

DisplayString(SIZE(0..78))

**Access**

read-write

**Description**

Server specific environment file. See `tuxTmachineEnvFile` for a complete discussion of how this file is used to modify the environment.

**Note:** Runtime modifications to this attribute do not affect a running server.

**tuxTsrvrGrace**

**Syntax**

INTEGER

**Access**

read-write

**Description**

The period of time, in seconds, over which the `tuxTsrvrMaxgen` limit applies. This attribute is meaningful only for restartable servers, that is, if the `tuxTsrvrRestart` attribute is set to `yes (1)`. When a restarting server would exceed the `tuxTsrvrMaxgen` limit but the `tuxTsrvrGrace` period has expired, the system resets...
the current generation (tuxTsrvrGeneration) to 1 and resets the initial boot time (tuxTsrvrTimeStart) to the current time. A value of 0 for this attribute indicates that a server should always be restarted.

Note that servers sharing a request queue (that is, equal values for tuxTsrvrRqAddr) should have equal values for this attribute. If they do not, then the first server activated establishes the runtime value associated with all servers on the queue.

**Note:** Runtime modifications to this attribute affect a running server and all other active servers with which it is sharing a request queue. However, only the selected server’s configuration parameter is modified. Thus, the behavior of the application depends on the order of boot in subsequent activations unless the administrator ensures that all servers sharing a queue have the same value for this attribute.

### tuxTsrvrMaxgen

**Syntax**

INTEGER (0..256)

**Access**

read-write

**Description**

Number of generations allowed for a restartable server (tuxTsrvrRestart == yes(1)) over the specified grace period (tuxTsrvrGrace). The initial activation of the server counts as one generation and each restart also counts as one. Processing after the maximum generations is exceeded is discussed above with respect to tuxTsrvrGrace.

Note that servers sharing a request queue (that is, equal values for tuxTsrvrRqAddr) should have equal values for this attribute. If they do not, then the first server activated establishes the runtime value associated with all servers on the queue.

**Note:** Runtime modifications to this attribute affect a running server and all other active servers with which it is sharing a request queue. However, only the selected server’s configuration parameter is modified. Thus, the behavior of the application depends on the order of boot in subsequent activations unless the administrator ensures that all servers sharing a queue have the same value for this attribute.
tuxTsrvrMax

Syntax  INTEGER (1..1001)
Access  read-write
Description  Maximum number of occurrences of the server to be booted. Initially, tmboot(1) boots tuxTsrvrMin objects of the server, and additional objects can be started individually (by starting a particular server id) or through automatic spawning (conversational servers only). Runtime modifications to this attribute affect all running servers in the set of identically configured servers (see tuxTsrvrBaseSrvId above) as well as the configuration definition of the server.

Note:  Runtime modifications to this attribute affect all running servers in the set of identically configured servers (see tuxTsrvrBaseSrvId above) as well as the configuration definition of the server.

tuxTsrvrMin

Syntax  INTEGER (1..1001)
Access  read-write
Description  Minimum number of occurrences of the server to be booted by tmboot(1). If a tuxTsrvrRqAddr is specified and tuxTsrvrMin is greater than 1, then the servers form an MSSQ set. The server identifiers for the servers are tuxTsrvrId up to tuxTsrvrId + tuxTsrvrMax - 1. All occurrences of the server have the same sequence number, as well as any other server parameters.

Note:  Runtime modifications to this attribute do not affect a running server.

tuxTsrvrRcmd

Syntax  DisplayString(SIZE(0..78))
Access  read-write
Description  Application-specified command to be executed in parallel with the system restart of an application server. This command must be an executable file.

Note:  Runtime modifications to this attribute affect a running server and all other active servers with which it shares a request queue. However, only the selected server’s configuration parameter is modified. Thus, the behavior of the
application depends on the order of boot in subsequent activations unless the administrator ensures that all servers sharing a queue have the same value for this attribute.

### tuxTsrvrRestart

**Syntax**

INTEGER { yes(1) | no(2) }

**Access**

read-write

**Description**

Restartable yes(1) or non-restartable no(2) server. If server migration is specified for this server group (tuxTdomainOptions = migrate(2) and tuxTgroupLmid with alternate site), this attribute must be set to yes(1).

Note that servers sharing a request queue (that is, equal values for tuxTsrvrRqAddr) should have equal values for this attribute. If they do not, the first server activated establishes the runtime value associated with all servers on the queue.

**Note:** Runtime modifications to this attribute affect a running server and all other active servers with which it shares a request queue. However, only the selected server’s configuration parameter is modified. Thus, the behavior of the application depends on the order of boot in subsequent activations unless the administrator ensures that all servers sharing a queue have the same value for this attribute.

### tuxTsrvrSequence

**Syntax**

INTEGER (1..10000)

**Access**

read-write

**Description**

Specifies when this server should be booted (tmboot(1)) or shutdown (tmshutdown(1)) relative to other servers. If two servers are given the same sequence number, it is possible for tmboot(1) to boot them in parallel and for tmshutdown(1) to shut them down in parallel. tuxTsrvrTbl instances added without a tuxTsrvrSequence attribute specified or with an invalid value have one generated for them that is 10,000 or more and is higher than any other automatically selected default value. Servers are booted by tmboot(1) in increasing order of sequence number and shutdown by tmshutdown(1) in decreasing order. Runtime modifications to this attribute affect only tmboot(1) and tmshutdown(1) and affect the order in which running servers can be shutdown by a subsequent invocation of tmshutdown(1).
**tuxTsrvrSystemAccess**

**Syntax**

`INTEGER { fastpath(1) | protected(2) }`

**Access**

read-write

**Description**

Mode used by System/T libraries within this server process to gain access to System/T’s internal tables. See `tuxTdomainSystemAccess` for a complete discussion of this attribute.

**Note:** Runtime modifications to this attribute do not affect a running server.

**tuxTsrvrConv**

**Syntax**

`INTEGER { yes(1) | no(2) }`

**Access**

read-write

**Description**

Conversational server `yes(1)` or request/response server `no(2)`.

**tuxTsrvrReplyQ**

**Syntax**

`INTEGER { yes(1) | no(2) }`

**Access**

read-write

**Description**

Specifies whether to allocate a separate reply queue for the server (`tuxTsrvrReplyQ == yes(1)`). MSSQ servers that expect to receive replies should set this attribute to `yes(1)`.

**tuxTsrvrRpPerm**

**Syntax**

`DisplayString(SIZE(4))`

**Access**

read-write

**Description**

UNIX system permissions for the server’s reply queue. If a separate reply queue is not allocated (`tuxTsrvrReplyQ == no(2)`), this attribute is ignored. This is a string representation of octal numbers starting with a leading 0001 through 0777.

**Note:** This is a UNIX system-specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.
**tuxtsrvrRqAddr**

**Syntax**  
`DisplayString (SIZE(0..30))`

**Access**  
read-write

**Description**  
Symbolic address of the request queue for the server. Specifying the same `tuxtsrvrRqAddr` attribute value for more than one server is the way multiple server, single queue (MSSQ) sets are defined. Servers with the same `tuxtsrvrRqAddr` attribute value must be in the same server group.

**tuxtsrvrRqPerm**

**Syntax**  
`DisplayString (SIZE(4))`

**Access**  
read-write

**Description**  
UNIX system permissions for the server’s request queue. This is a string representation of octal numbers starting with a leading 0 0001 through 0777.

**Note:** This is a UNIX system specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.

**tuxtsrvrGeneration**

**Syntax**  
`INTEGER (1..32768)`

**Access**  
read-only

**Description**  
Generation of the server. When a server is initially booted via `tmboot(1)` or activated through the SNMP agent, its generation is set to 1. Each time the server dies abnormally and is restarted, its generation is incremented. Note that when `tuxtsrvrMaxgen` is exceeded and `tuxtsrvrGrace` has expired, the server is restarted with the generation reset to 1.

**tuxtsrvrPid**

**Syntax**  
`INTEGER`

**Access**  
read-only
Description UNIX system process identifier for the server. Note that this cannot be a unique attribute since servers can be located on different machines allowing for duplication of process identifiers.

Note: This is a UNIX system specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.

tuxTsrvrRpid
Syntax INTEGER
Access read-only
Description UNIX system message queue identifier for the server's reply queue. If a separate reply queue is not allocated (tuxTsrvrReplyQ == no(2)), this attribute value is the same as tuxTsrvrRqId.

Note: This is a UNIX system specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.

tuxTsrvrRqId
Syntax INTEGER
Access read-only
Description UNIX system message queue identifier for the server's request queue. If a separate reply queue is not allocated (tuxTsrvrReplyQ == no(2)), this attribute value is the same as tuxTsrvrRpid.

Note: This is a UNIX system specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.

tuxTsrvrTimeRestart
Syntax INTEGER
Access read-only
Description Time, in seconds, since 00:00:00 UTC, January 1, 1970, as returned by the time(2) system call on local host, when the server was last started or restarted.
### tuxTsrvrTimeStart

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Time, in seconds, since 00:00:00 UTC, January 1, 1970, as returned by the `time(2)` system call on local host, when the server was first started. Restarts of the server do not reset this value; however, if `tuxTsrvrMaxgen` is exceeded and `tuxTsrvrGrace` is expired, this attribute is reset to the time of the restart.

### tuxTsrvrMinDispatchThreads

**Syntax**  
INTEGER (1..999)

**Access**  
read-only

**Description**  
Specifies the number of server dispatch threads started on the initial server boot. This parameter is effective only if the server has been built with the `buildserver -t` command.

The separate dispatcher thread that is used when the value of `tuxTsrvrMaxDispatchThreads` is greater than one is not counted as part of the `tuxTsrvrMinDispatchThreads` value. The value of `tuxTsrvrMinDispatchThreads` must be less than the value of `tuxTsrvrMaxDispatchThreads`. If this parameter is not specified, the default is 0.

### tuxTsrvrMaxDispatchThreads

**Syntax**  
INTEGER (0..999)

**Access**  
read-only

**Description**  
Specifies the maximum number of concurrently dispatched threads that each server process can spawn. This parameter is effective only if the server has been built with the `buildserver -t` command.

If `tuxTsrvrMaxDispatchThreads` is greater than one, a separate dispatcher thread is used and does not count against this limit. The value of `tuxTsrvrMinDispatchThreads` must be less than the value of `tuxTsrvrMaxDispatchThreads`. If this parameter is not specified, the default is 1.
tuxTsrvrThreadStackSize

**Syntax**
INTEGER (0..2147483647)

**Access**
read-write

**Description**
If this parameter is not specified or if the value specified is 0, the operating system default is used. This option affects the server only when a value greater than 1 is specified for `tuxTsrvrMaxDispatchThreads`.

wleSrvrSrvType

**Syntax**
INTEGER {java (1)}

**Access**
read-only

**Description**
If `SRVTYPE` is set to `java`, it indicates that the server is a Java server. If at least one `MODULE` attribute is specified for this server or the name of the server is `JavaServer` (not case-sensitive), `java` becomes the default value.
An extension of the `tuxTsrvrTbl`. Objects in this table are only accessible through a Tuxedo SNMP agent installed on the local machine

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTsrvrIdExt</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.1</td>
</tr>
<tr>
<td><code>tuxTsrvrGrpNoExt</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.2</td>
</tr>
<tr>
<td><code>tuxTsrvrNumConv</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.3</td>
</tr>
<tr>
<td><code>tuxTsrvrNumDeque</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.4</td>
</tr>
<tr>
<td><code>tuxTsrvrNumEnque</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.5</td>
</tr>
<tr>
<td><code>tuxTsrvrNumPost</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.6</td>
</tr>
<tr>
<td><code>tuxTsrvrNumReq</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.7</td>
</tr>
<tr>
<td><code>tuxTsrvrNumSubscribe</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.8</td>
</tr>
<tr>
<td><code>tuxTsrvrNumTran</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.9</td>
</tr>
<tr>
<td><code>tuxTsrvrTranAbt</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.10</td>
</tr>
<tr>
<td><code>tuxTsrvrTranCmt</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.11</td>
</tr>
<tr>
<td><code>tuxTsrvrTotReqC</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.12</td>
</tr>
<tr>
<td><code>tuxTsrvrTotWorkL</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.13</td>
</tr>
<tr>
<td><code>tuxTsrvrCltLmid</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.14</td>
</tr>
<tr>
<td><code>tuxTsrvrCltPid</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.15</td>
</tr>
<tr>
<td><code>tuxTsrvrCltReply</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.16</td>
</tr>
<tr>
<td><code>tuxTsrvrCmtRet</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.17</td>
</tr>
<tr>
<td><code>tuxTsrvrCurConv</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.18</td>
</tr>
<tr>
<td><code>tuxTsrvrCurReq</code></td>
<td>.1.3.6.1.4.1.140.300.20.2.1.19</td>
</tr>
</tbody>
</table>
## Variable Name | Object ID
--- | ---
tuxTsrvrCurService | .1.3.6.1.4.1.140.300.20.2.1.20
tuxTsrvrCurTime | .1.3.6.1.4.1.140.300.20.2.1.21
tuxTsrvrLastGrp | .1.3.6.1.4.1.140.300.20.2.1.22
tuxTsrvrSvcTimeOut | .1.3.6.1.4.1.140.300.20.2.1.23
tuxTsrvrTimeLeft | .1.3.6.1.4.1.140.300.20.2.1.24
tuxTsrvrTranLev | .1.3.6.1.4.1.140.300.20.2.1.25
tuxTsrvrStateExt | .1.3.6.1.4.1.140.300.20.2.1.26
tuxTsrvrGrpExt | .1.3.6.1.4.1.140.300.20.2.1.27
wleSrvrCurObjsExt | .1.3.6.1.4.1.140.300.20.2.1.32
wleSrvrCurInterfaceExt | .1.3.6.1.4.1.140.300.20.2.1.37
wleSrvrClassPath | .1.3.6.1.4.1.140.300.20.2.1.50
wleSrvrjavaHeap | .1.3.6.1.4.1.140.300.20.2.1.60
wleSrvrjavaHeapuse | .1.3.6.1.4.1.140.300.20.2.1.70
wleSrvrjavaVendor | .1.3.6.1.4.1.140.300.20.2.1.80
wleSrvrjavaVersion | .1.3.6.1.4.1.140.300.20.2.1.90
tuxTsrvrCurDispatchThreads | .1.3.6.1.4.1.140.300.20.2.1.100
tuxTsrvrHwDispatchThreads | .1.3.6.1.4.1.140.300.20.2.1.110
tuxTsrvrNumDispatchThreads | .1.3.6.1.4.1.140.300.20.2.1.120

tuxTsrvrIdExt

Syntax: INTEGER (1..30001)
Access: read-only
Description: Unique (within the server group) server identification number.
tuxTsrvrGrpNoExt

Syntax: INTEGER (1..30000)
Access: read-only
Description: Group number associated with this server’s group.

TuxTsrvrNumConv

Syntax: INTEGER
Access: read-only
Description: Number of conversations initiated by this server through tpconnect(3).

TuxTsrvrNumDeque

Syntax: INTEGER
Access: read-only
Description: Number of dequeue operations initiated by this server through tpdequeue(3).

TuxTsrvrNumEnque

Syntax: INTEGER
Access: read-only
Description: Number of enqueue operations initiated by this server through tpenqueue(3).

TuxTsrvrNumPost

Syntax: INTEGER
Access: read-only
Description: Number of postings initiated by this server through tppost(3).
tuxTsrvrNumReq

Syntax INTEGER
Access read-only
Description Number of requests made by this server through tpcall(3) or tpacall(3).

tuxTsrvrNumSubscribe

Syntax INTEGER
Access read-only
Description Number of subscriptions made by this server through tpsubscribe(3).

tuxTsrvrNumTran

Syntax INTEGER
Access read-only
Description Number of transactions begun by this server since its last (re)start.

tuxTsrvrTranAbt

Syntax INTEGER
Access read-only
Description Number of transactions aborted by this server since its last (re)start.

tuxTsrvrTranCmt

Syntax INTEGER
Access read-only
Description Number of transactions committed by this server since its last (re)start.
tuxTsrvrTotReqC

Syntax: INTEGER
Access: read-only
Description: Total number of requests completed by this server. For conversational servers (tuxTsrvrConv == yes(1)), this attribute value indicates the number of completed incoming conversations. This is a runtime attribute that is kept across server restart but is lost at server shutdown.

tuxTsrvrTotWorkL

Syntax: INTEGER
Access: read-only
Description: Total workload completed by this server. For conversational servers (tuxTsrvrConv == yes(1)), this attribute value indicates the workload of completed incoming conversations. This is a runtime attribute that is kept across server restart but is lost at server shutdown.

tuxTsrvrCltLmid

Syntax: DisplayString (SIZE(1..30))
Access: read-only
Description: Logical machine for the initiating client or server. The initiating client or server is the process that made the service request on which the server is currently working.

tuxTsrvrCltPid

Syntax: INTEGER
Access: read-only
Description: UNIX system process identifier for the initiating client or server.

Note: This is a UNIX system specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.
tuxTsrvrCltReply

Syntax  INTEGER { yes(1) | no(2) | null(3) }
Access  read-only
Description  yes(1)

The initiating client or server expects a reply.

no(2).

The initiating client or server does not expect a reply.

tuxTsrvrCmtRet

Syntax  INTEGER { complete(1) | logged(2) }
Access  read-only
Description  The setting of the TP_COMMIT_CONTROL characteristic for this server. See the description of the System/T ATMI function tpScmt(3) for details on this characteristic.

tuxTsrvrCurConv

Syntax  INTEGER
Access  read-only
Description  Number of conversations initiated by this server through tpconnect(3) that are still active.

tuxTsrvrCurReq

Syntax  INTEGER
Access  read-only
Description  Number of requests initiated by this server through tpcall(3) or tpacall(3) that are still active.
**tuxTsrvrCurService**

**Syntax**  
DisplayString (SIZE (1..15))

**Access**  
read-only

**Description**  
Service name, if any, on which the server is currently working.

**tuxTsrvrCurTime**

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Current time, in seconds, since 00:00:00 UTC, January 1, 1970, as on the local host.  
This attribute can be used to compute elapsed time from the tuxTsrvrTimeStart and tuxTsrvrTimeRestart object values.

**tuxTsrvrLastGrp**

**Syntax**  
INTEGER (1..30000)

**Access**  
read-only

**Description**  
Server group number (tuxTgroupNo) of the last service request made or conversation initiated from this server outward.

**tuxTsrvrSvcTimeOut**

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Time left in seconds, if any, for this server to process the current service request.  
A value of 0 for an active service indicates that no timeout processing is being done.  
See tuxTsvcTimeOut for more information.

**tuxTsrvrTimeLeft**

**Syntax**  
INTEGER

**Access**  
read-only
2  Tuxedo Core MIB

Description  Time left, in seconds, for this server to receive the reply for which it is currently waiting before it times out. This timeout can be a transactional timeout or a blocking timeout.

tuxTsrvrTranLev

Syntax  INTEGER

Access  read-only

Description  Current transaction level for this server. 0 indicates that the server is not currently involved in a transaction.

tuxTsrvrStateExt

Syntax  INTEGER { active(1) | inactive(2) | migrating(3) | cleaning(4) | restarting(5) | suspended(6) | partitioned(7) | dead(8) }

Access  read-only

Description  Refer to description of tuxTsrvrState for details.

tuxTsrvrGrpExt

Syntax  DisplayString

Access  read-only

Description  Name of group to which this server belongs. This object is included for readability purposes only.

wleSrvrCurObjsExt

Syntax  INTEGER

Access  read-only

Description  The number of entries in use in the bulletin board object table for this server.

wleSrvrCurlInterfaceExt

Syntax  DisplayString {SIZE(1..128)}
Access: read-only
Description: The interface name of the interface currently active in this server.

**wleSrvrClassPath**

- **Syntax:** DisplayString (SIZE(0..2047))
- **Access:** read-only
- **Description:** The current CLASSPATH for the runtime.

**wleSrvrJavaHeap**

- **Syntax:** INTEGER
- **Access:** read-only
- **Description:** The heap size, as specified in the runtime options.

**wleSrvrJavaHeapuse**

- **Syntax:** INTEGER(1..100)
- **Access:** read-only
- **Description:** The percentage of heap space used.

**wleSrvrJavaVendor**

- **Syntax:** DisplayString (SIZE(0..30))
- **Access:** read-only
- **Description:**

**wleSrvrJavaVersion**

- **Syntax:** DisplayString (SIZE(0..30))
- **Access:** read-only
tuxTsrvrCurDispatchThreads

Syntax     INTEGER
Access     read-only
Description Current number of active service dispatch threads for this server.

tuxTsrvrHwDispatchThreads

Syntax     INTEGER
Access     read-only
Description Highest number of active service dispatch threads created for this server since its last restart. This number can differ from the number of service calls, because an administrator can specify parameters that control the caching of idle service threads.

tuxTsrvrNumDispatchThreads

Syntax     INTEGER
Access     read-only
Description Total number of active service dispatch threads created for this server since its last restart.
This represents configuration attributes of services within an application. These attribute values identify and characterize configured services. A `tuxTsvcTbl` object provides activation time configuration attributes for services not specifically configured as part of the `tuxTsvcGrp` group. The index into this table is `tuxTsvcName`. Objects in this group are only accessible through a Tuxedo SNMP agent installed on the local machine. To create a new row in the table, it is necessary to issue a SET request for a non-existing row in the table.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTsvcName</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.1</td>
</tr>
<tr>
<td><code>tuxTsvcType</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.2</td>
</tr>
<tr>
<td><code>tuxTsvcState</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.3</td>
</tr>
<tr>
<td><code>tuxTsvcAutoTran</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.4</td>
</tr>
<tr>
<td><code>tuxTsvcLoad</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.5</td>
</tr>
<tr>
<td><code>tuxTsvcPrio</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.6</td>
</tr>
<tr>
<td><code>tuxTsvcTimeOut</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.7</td>
</tr>
<tr>
<td><code>tuxTsvcTranTime</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.8</td>
</tr>
<tr>
<td><code>tuxTsvcBufType</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.9</td>
</tr>
<tr>
<td><code>tuxTsvcRoutingName</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.10</td>
</tr>
<tr>
<td><code>tuxTsvcEncryptionRequired</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.20</td>
</tr>
<tr>
<td><code>tuxTsvcSignatureRequired</code></td>
<td>.1.3.6.1.4.1.140.300.10.1.1.30</td>
</tr>
</tbody>
</table>
tuxTsvcName

Syntax      DisplayString(SIZE(1..15))
Access      read-write
Description Service name.

Note: This object can be set only during row creation.

tuxTsvcType

Syntax      INTEGER { app(1) | callable(2) | system(3) | unknown(4) }
Access      read-only
Description Type of service.

app(1)
Indicates an application-defined service name.

callable(2)
Indicates a system-provided callable service.

system(3)
Indicates a system-provided and system-callable service. system(3) services
are not available to application clients and servers for direct access.

2-124 BEA SNMP Agent MIB Reference
tuxTsvcTbl

  tuxTsvcTbl instance is defined and at least one tuxTsvcGrp object with a matching tuxTsvcName value is active.

inactive(2)
  tuxTsvcTbl instance is defined and no tuxTsvcGrp object with a matching tuxTsvcName value is active.

SET: invalid(3)
  A SET operation updates configuration information for the selected tuxTsvcTbl instance. The following state indicates the meaning of a tuxTsvcState set in a SET request. States not listed cannot be set.

invalid(3)
  Delete tuxTsvcTbl instance for application. State change is allowed only when the service is in the inactive(2) state. Successful return leaves the object in the invalid(3) state.

**tuxTsvcAutoTran**

- **Syntax**: INTEGER { yes(1) | no(2) }
- **Access**: read-write
- **Description**: Automatically begin a transaction.

(yes(1))
  If the request is not already in transaction mode when a service request message is received for this service, automatically begin a transaction.

(no(2))
  Do not automatically begin a transaction

**Note**: Runtime updates to this attribute are not reflected in active tuxTsvcGrp objects.

**tuxTsvcLoad**

- **Syntax**: INTEGER (1..32768)
- **Access**: read-write
Tuxedo Core MIB

**Description**
This `tuxTsvcTbl` object imposes the indicated load on the system. Service loads are used for load balancing purposes, that is, queues with higher enqueued workloads are less likely to be chosen for a new request. Service loads have meaning only if the `tuxTdomainLoadBalance` is set to `yes(1)`.

**Note:** Runtime updates to this attribute are not reflected in active `tuxTsvcGrp` objects.

### `tuxTsvcPrio`

**Syntax**
INTEGER (1..100)

**Access**
read-write

**Description**
This `tuxTsvcTbl` object has the indicated dequeuing priority. If multiple service requests are waiting on a queue for servicing, the higher priority requests are serviced first.

**Note:** Runtime updates to this attribute are not reflected in active `tuxTsvcGrp` objects.

### `tuxTsvcTimeOut`

**Syntax**
INTEGER

**Access**
read-write

**Description**
Time limit (in seconds) for processing requests for this service name. Servers processing service requests for this service are abortively terminated (`kill -9`) if they exceed the specified time limit in processing the request. A value of 0 for this attribute indicates that the service should not be abortively terminated.

**Note:** Runtime updates to this attribute are not reflected in active `tuxTsvcGrp` objects.

**Note:** This attribute value is not enforced on Tuxedo System/T Release 4.2.2 sites or earlier.

### `tuxTsvcTranTime`

**Syntax**
INTEGER
tuxtsvcTbl

Access read-write
Description Transaction timeout value (in seconds) for transactions automatically started for this tuxtsvcTbl object. Transactions are started automatically when a request not in transaction mode is received and the tuxtsvcAutoTran attribute value for the service is yes(1).

Note: Runtime updates to this attribute are not reflected in active tuxtsvcGrp objects.

tuxtsvcBufType

Syntax DisplayString (SIZE(1..256))
Access read-write
Description type1[:subtype1[,subtype2 ... ]][;type2[:subtype3[, ... ]]] ... List of types and subtypes of data buffers accepted by this service. A maximum of 32 type/subtype combinations are allowed. Types of data buffers provided with Tuxedo System/T are FML (for FML buffers), VIEW, X_C_TYPE, or X_COMMON (for FMLviews), STRING (for NULL terminated character arrays), and CARRAY or X_OCTET (for a character array that is neither encoded nor decoded during transmission). Of these types, only VIEW, X_C_TYPE, and X_COMMON have subtypes. A VIEW subtype gives the name of the particular VIEW expected by the service. Application types and subtypes can also be added (see tuxtypes(5)). For a buffer type that has subtypes, "*" can be specified for the subtype to indicate that the service accepts all subtypes for the associated buffer type.

A single service can only interpret a fixed number of buffer types, namely those found in its buffer type switch (see tuxtypes(5)). If the tuxtsvcBufType value is set to ALL, that service accepts all buffer types found in its buffer type switch.

A type name can be 8 characters or less in length and a subtype name can be 16 characters or less in length. Note that type and subtype names should not contain semicolon, colon, comma, or asterisk characters.

Note: This attribute value represents the buffer types that must be supported by each and every instance of an application service with this service name. Since this attribute value is processed at service activation time, updates to this attribute are allowed only when there are no active tuxtsvcGrp objects with matching service names.
tuxTsvcRoutingName

Syntax: `DisplayString(SIZE(0..15))`

Access: read-write

Description: This `tuxTsvcTbl` object has the indicated routing criteria name. Active updates to this attribute are reflected in all associated `tuxTsvcGrp` objects.

tuxTsvcEncryptionRequired

Syntax: `INTEGER { yes(1) | no(2) }`

Access: read-write

Description: If set to `yes`, every application service in this group requires an encrypted input message buffer.

tuxTsvcSignatureRequired

Syntax: `INTEGER { yes(1) | no(2) }`

Access: read-write

Description: If set to `yes`, every application service in this group requires a valid digital signature on its input message buffer.
The `tuxTsvcGrp` group represents configuration and runtime attributes of services/groups within an application. These attribute values identify and characterize configured services/groups as well as provide runtime tracking of statistics and resources associated with each object.

Both `tuxTsvcTbl` and `tuxTsvcGrp` define activation time attribute settings for service names within the application. When a new service is activated (advertised), either due to initial activation of a server or due to a call to `tpadvertise(3)`, the following hierarchy exists for determining the attribute values to be used at service startup time.

1. If a matching configured `tuxTsvcGrp` entry exists (matching service name and server group), the attributes defined in that object are used to initially configure the advertised service.

2. Otherwise, if a matching configured `tuxTsvcTbl` entry exists (matching service name), the attributes defined in that object are used to initially configure the advertised service.

3. Otherwise, if any configured `tuxTsvcGrp` entries are found with matching service name value, the first one found is used to initially configure the advertised service.

4. If none of the preceding cases is used, the system defaults for service attributes are used to initially configure the advertised service.

Objects in this group are only accessible through a Tuxedo SNMP agent installed on the local machine.

To create a new row in the table, it is necessary to issue a `SET` request that specifies at least `tuxTsvcGrpName`. The combination of values specified for `tuxTsvcGrpName` and `tuxTsvcGrpSvcName` in the `SET` request should not correspond to an existing row. If the value of `tuxTsvcSrvrId` is zero in the `SET` request, the service entry is configured but not activated (advertised). If `tuxTsvcSrvrId` is not set to zero, the service is activated using the value of `tuxTsvcSrvrId` to identify the server instance.
Tuxedo Core MIB

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTsvcGrpSvcName</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.1</td>
</tr>
<tr>
<td>tuxTsvcGrpName</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.2</td>
</tr>
<tr>
<td>tuxTsvcGrpNo</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.3</td>
</tr>
<tr>
<td>tuxTsvcGrpState</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.4</td>
</tr>
<tr>
<td>tuxTsvcGrpAutoTran</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.5</td>
</tr>
<tr>
<td>tuxTsvcGrpLoad</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.6</td>
</tr>
<tr>
<td>tuxTsvcGrpPrio</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.7</td>
</tr>
<tr>
<td>tuxTsvcGrpSvcTimeOut</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.8</td>
</tr>
<tr>
<td>tuxTsvcGrpTranTime</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.9</td>
</tr>
<tr>
<td>tuxTsvcSrvrLmid</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.10</td>
</tr>
<tr>
<td>tuxTsvcSrvrRqAddr</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.11</td>
</tr>
<tr>
<td>tuxTsvcSrvrId</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.12</td>
</tr>
<tr>
<td>tuxTvcrName</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.13</td>
</tr>
<tr>
<td>tuxTvscSrvrNcompleted</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.14</td>
</tr>
<tr>
<td>tuxTvscSrvrNqueued</td>
<td>.1.3.6.1.4.1.140.300.10.2.1.15</td>
</tr>
</tbody>
</table>

**tuxTsvcGrpSvcName**

Syntax: `DisplayString(SIZE(1..15))`

Access: read-only

Description: Service name.
**tuxTsvcGrpName**

Syntax:  
DisplayString (SIZE(1..30))

Access: read-only

Description: Server group name. Server group names cannot contain an asterisk.

**tuxTsvcGrpNo**

Syntax:  
INTEGER (1..29999)

Access: read-write

Description: Server group number.

**tuxTsvcGrpState**

Syntax:  
INTEGER { active(1) | inactive(2) | invalid(3) }

Access: read-write

Description: The values for GET and SET operations are as follows:

**GET:**

- active(1) | inactive(2)

  A GET operation retrieves configuration information for the selected `tuxTsvcGrpState` instance(s). The following states indicate the meaning of a `tuxTsvcGrpState` returned in response to a GET request. States not listed are not returned.

- active(1)
  
  At least one instance is active, suspended, or partitioned.

- inactive(2)
  
  `tuxTsvcGrp` instance defined and inactive.

**SET:**

- invalid(3)

  A SET operation removes the corresponding `tuxTsvcGrp` instance. When a `tuxTsvcGrp` instance is deleted it also removes the associated `tuxTsvcSrvr` instances that correspond to server instances that are a part of the group advertising this service. This transition is permissible only in `inactive(2)` state.
tuxTsvcGrpAutoTran

Syntax  INTEGER { yes(1) | no(2) }
Access  read-write
Description  Automatically begin a transaction (yes(1)) when a service request message is received for this service if the request is not already in transaction mode.


tuxTsvcGrpLoad

Syntax  INTEGER (1..32767)
Access  read-write
Description  This tuxTsvcGrp instance imposes the indicated load on the system. Service loads are used for load balancing purposes, that is, queues with higher enqueued workloads are less likely to be chosen for a new request.


tuxTsvcGrpPrio

Syntax  INTEGER (1..100)
Access  read-write
Description  This tuxTsvcGrp object has the indicated dequeuing priority. If multiple service requests are waiting on a queue for servicing, the higher priority requests are serviced first.


tuxTsvcGrpSvcTimeOut

Syntax  INTEGER
Access  read-write
Description  Time limit (in seconds) for processing requests for this service name. Servers processing service requests for this service are abortively terminated (kill -9) if they exceed the specified time limit in processing the request. A value of 0 for this attribute indicates that the service should not be abortively terminated.
**tuxTsvcGrpTranTime**

**Syntax**
INTEGER

**Access**
read-write

**Description**
Transaction timeout value (in seconds) for transactions automatically started for this tuxTsvcGrp instance. Transactions are started automatically when a request not in transaction mode is received and the tuxTsvcGrpAutoTran attribute value for the service is yes(1).

**tuxTsvcSrvrLmid**

**Syntax**
DisplayString (SIZE(1..30))

**Access**
read-only

**Description**
Current logical machine on which an active server that offers this service is running.

**tuxTsvcSrvrRqAddr**

**Syntax**
DisplayString (SIZE(1..30))

**Access**
read-only

**Description**
Symbolic address of the request queue for an active server that offers this service. See tuxTsrvrRqAddr for more information on this attribute.

**tuxTsvcSrvrId**

**Syntax**
INTEGER (1..30000)

**Access**
read-write

**Description**
Server ID of which the service is a part. The user can also set the value of this object to activate (advertise) one or more tuxTsvcGrp instances. The value provided to set this object is used to activate another instance of tuxTsvcGrp.
Tuxedo Core MIB

**tuxTsvcrName**

*Syntax*  
DisplayString(SIZE(1..15))

*Access*  
read-write

*Description*  
Function name within the associated server assigned to process requests for this service. When this object is specified, the `tuxTsvcGrp` instance is activated (advertised). The user needs to specify the server ID of the corresponding server instance (`tuxTsvcSrvrId`) in the SNMP index. This object can be updated only during row creation.

**tuxTsvcSrvrNcompleted**

*Syntax*  
INTEGER

*Access*  
read-only

*Description*  
Number of service requests completed with respect to the retrieved active or suspended instance since it was activated (advertised).

*Note*: This attribute is returned only when `tuxTdomainLoadBalance` is equal to `yes(1)`.

**tuxTsvcSrvrNqueued**

*Syntax*  
INTEGER (0..32767)

*Access*  
read-only

*Description*  
Number of requests currently enqueued to this service. This attribute is incremented at enqueue time and decremented when the server dequeues the request.

*Note*: This attribute is returned only when the `tuxTdomainModel` is set to `single-machine(1)` and the `tuxTdomainLoadBalance` attribute is set to `yes(1)`.
This group represents runtime attributes of /T listener processes for a distributed application.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTlistenLmid</td>
<td>.1.3.6.1.4.1.140.300.21.1.1.1</td>
</tr>
<tr>
<td>tuxTlistenState</td>
<td>.1.3.6.1.4.1.140.300.21.1.1.2</td>
</tr>
</tbody>
</table>

**tuxTlistenLmid**

Syntax: `DisplayString(SIZE(1..30))`

Access: read-only

Description: Logical machine identifier.

**tuxTlistenState**

Syntax: `INTEGER { inactive(2) | active(1) }

Access: read-only

Description: The values for GET and SET operations are as follows:

GET: `{active(1) | inactive(2)}`

A GET operation retrieves runtime information for the selected tuxTlistenTbl instance(s). The following states indicate the meaning of a tuxTlistenState returned in response to a GET request. States not listed are not returned.

- active(1)
  
  tuxTlistenTbl instance active.

- inactive(2)
  
  tuxTlistenTbl instance not active.
This table represents runtime attributes of active transactions within the application. The following objects comprise the index for rows in this table: `tuxTranIndx1`, `tuxTranIndx2`, `tuxTranIndx3`, `tuxTranIndx4`, `tuxTranIndx5`. Objects in this table are accessible only through a Tuxedo SNMP agent running on the local machine.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTranCoordLmid</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.1</td>
</tr>
<tr>
<td><code>tuxTpTranId</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.2</td>
</tr>
<tr>
<td><code>tuxTranXid</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.3</td>
</tr>
<tr>
<td><code>tuxTranIndx1</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.4</td>
</tr>
<tr>
<td><code>tuxTranIndx2</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.5</td>
</tr>
<tr>
<td><code>tuxTranIndx3</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.6</td>
</tr>
<tr>
<td><code>tuxTranIndx4</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.7</td>
</tr>
<tr>
<td><code>tuxTranIndx5</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.8</td>
</tr>
<tr>
<td><code>tuxTranState</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.9</td>
</tr>
<tr>
<td><code>tuxTranTimeOut</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.10</td>
</tr>
<tr>
<td><code>tuxTranGrpCnt</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.11</td>
</tr>
<tr>
<td><code>tuxTranGrpIndex</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.12</td>
</tr>
<tr>
<td><code>tuxTranGrpNo</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.13</td>
</tr>
<tr>
<td><code>tuxTranGstate</code></td>
<td>.1.3.6.1.4.1.140.300.23.1.1.14</td>
</tr>
</tbody>
</table>

**tuxTranCoordLmid**

**Syntax**  
`DisplayString(SIZE(1..30))`

**Access**  
read-only
Logical machine identifier of the server group responsible for coordinating the transaction.

**tuxTpTranId**

- **Syntax**: `DisplayString(SIZE(2..78))`
- **Access**: read-only
- **Description**: Transaction identifier as returned from `tpsuspend(3)` mapped to a string representation. The data in this field should not be interpreted directly by the user except for equality comparison.

**tuxTranXid**

- **Syntax**: `DisplayString(SIZE(2..78))`
- **Access**: read-only
- **Description**: Transaction identifier as returned from `tx_info(3)` mapped to a string representation. The data in this field should not be interpreted directly by the user except for equality comparison.

**tuxTranIndx1**

- **Syntax**: `INTEGER`
- **Access**: read-only
- **Description**: This number is purely for unique indexing of this table.

**tuxTranIndx2**

- **Syntax**: `INTEGER`
- **Access**: read-only
- **Description**: This number is purely for unique indexing of this table.

**tuxTranIndx3**

- **Syntax**: `INTEGER`
2 Tuxedo Core MIB

**tuxTranIndx4**

Syntax: INTEGER

Access: read-only

Description: This number is purely for unique indexing of this table.

**tuxTranIndx5**

Syntax: INTEGER

Access: read-only

Description: This number is purely for unique indexing of this table.

**tuxTranState**

Syntax: INTEGER { active(1) | abort-only(2) | aborted(3) | com-called(4) | ready(5) | decided(6) | suspended(7) }

Access: read-write

Description: The values for GET and SET operations are as follows:

GET: active(1)|abort-only(2)|aborted(3)|com-called(4)|ready(5)|decided(6)|suspended(7)

A GET operation retrieves runtime information for the selected tuxTranTbl instance(s). The following states indicate the meaning of a tuxTranState object. States not listed are not returned. Note that distinct objects pertaining to the same global transaction (equivalent transaction identifiers) can indicate differing states. In general, the state indicated on the coordinator’s site (tuxTranCoordLmid) indicates the true state of the transaction. The exception is when a noncoordinator site notices a condition that transitions the transaction state to abort-only(2). This transition is eventually propagated to the coordinator site and results in the rollback of the transaction, but this change cannot be immediately reflected on the coordinator site.
tuxTranTbl

active(1)
The transaction is active.

abort-only(2)
The transaction has been identified for rollback on the retrieval site.

aborted(3)
The transaction has been identified for rollback and rollback has been initiated on the retrieval site.

com-called(4)
The initiator of the transaction has called 
\texttt{tpcommit(3)} and the first phase of two-phase commit has begun on the retrieval site.

ready(5)
All of the participating groups on the retrieval site have successfully completed the first phase of two-phase commit and are ready to be committed.

decided(6)
The second phase of the two-phase commit has begun on the retrieval site.
suspended(7)
The initiator of the transaction has suspended processing on the transaction. Note that this state is returned from the initiator’s site only.

SET: aborted(3)
A \texttt{SET} operation updates runtime information for the selected tuxTranTbl instance. The following state indicates the meaning of a \texttt{tuxTranState set} in a \texttt{SET} request. States not listed cannot be set.

aborted(3)
Abort the tuxTranTbl instance for the application. State change is allowed only when the transaction is in the \texttt{active(1)}, \texttt{abort-only(2)}, or \texttt{com-called(4)} state. Cannot be accompanied by a change to \texttt{tuxTranGstate}. Successful return leaves the object in the \texttt{aborted(3)} state.
2 Tuxedo Core MIB

**tuxTranTimeOut**

Syntax: INTEGER  
Access: read-only  
Description: Time left (in seconds) before the transaction times out on the retrieval site. Note that this attribute value is returned only when the transaction state is `active(1)`.

**tuxTranGrpCnt**

Syntax: INTEGER  
Access: read-only  
Description: Number of groups identified as participants in the transaction by the information returned from the retrieval site.

**tuxTranGrpIndex**

Syntax: INTEGER  
Access: read-only  
Description: Index of the first group-specific attribute values (`tuxTranGrpNo` and `tuxTranGstate`) corresponding to this object.

**tuxTranGrpNo**

Syntax: INTEGER  
Access: read-only  
Description: Group number of the participating group.

**tuxTranGstate**

Syntax: INTEGER { active(1) | aborted(2) | rd-only(3) | ready(4) | hcommit(5) | habort(6) | done(7) | pre-prepare(8) | post-abort(9) | post-commit(10) | unknown(11) }  
Access: read-write
### Description

The values for **GET** and **SET** operations are as follows:

**GET:** active(1) | aborted(2) | rd-only(3) | ready(4) | hcommit(5) | habort(6) | done(7)

A **GET** operation retrieves runtime information for the selected **tuxTranTbl** instance(s) pertaining to the indicated group. The following states indicate the meaning of a **tuxTranGstate** returned in response to a **GET** request. States not listed are not returned. Note that distinct objects pertaining to the same global transaction (equivalent transaction identifiers) can indicate differing states for individual groups. In general, the state indicated on the group’s site indicates the true state of the group’s participation in the transaction. The exception is when the coordinator site determines that the transaction should abort and sets each participant group state to **aborted(2)**. This transition is propagated to the group’s site and results in the rollback of the group’s work in the transaction, but cannot be reflected immediately.

**active(1)**

The transaction is active in the indicated group.

**aborted(2)**

The transaction has been identified for rollback and rollback has been initiated for the indicated group.

**rd-only(3)**

The group has successfully completed the first phase of two-phase commit and has performed only read operations on the resource manager, thus making it unnecessary to perform the second phase of commit for this group.

**ready(4)**

The group has successfully completed the first phase of two-phase commit and is ready to be committed.

**hcommit(5)**

The group has been heuristically committed. This might or might not agree with the final resolution of the transaction.

**habort(6)**

The group has been heuristically rolled back. This might or might not agree with the final resolution of the transaction.

**done(7)**
This group has completed the second phase of the two-phase commit.

**pre-prepare(8)**

Indicates that the transaction group contains WLE servers that have called `xa_end (TMSUSPEND)` during the course of transactional work and that commit processing is beginning. This state exists until either (1) All servers that called `xa_end (TMSUSPEND)` have caused a call to `xa_end (TMSUCCESS)`, at which point the group state becomes ready, or (2) One of the target servers does a rollback of the transaction at which point the group state becomes either `post-abort(9)` or `aborted(2)`.

**Note:** This state is supported for WLE applications only.

**post-abort(9)**

Indicates that an WLE server called `xa_end (TPFAIL)` and that the TMS has not yet called `xa_rollback()`. In this case, other WLE servers that called `xa_end (TMSUSPEND)` are being notified by the TMS in order to clean up their associated CORBA objects.

**Note:** This state is supported for WLE applications only.

**post-commit(10)**

This state is not implemented yet.

**Note:** This state is supported for WLE applications only.

**SET: hcommit(5) | habort(6)**

A SET operation updates runtime information for the first group in the originating request within the selected `tuxTranTbl` instance. The following states indicate the meaning of `a tuxTranGstate set in a SET request. States not listed cannot be set. State transitions are allowed only when performed within the object representing the group’s site.

**hcommit(5)**

Heuristically commit the group’s work as part of the indicated transaction. State change is allowed only when `tuxTranGstate is ready, tuxTranState is ready, and the indicated group is not on the coordinator’s site. Successful return leaves the object in the `hcommit(5)` state.

**habort(6)**

Heuristically rollback the group’s work as part of the indicated transaction. State change is allowed only when `tuxTranGstate is active(1)` or
ready(4), tuxTranState is ready(4), and the indicated group is not on the coordinator’s site. Successful return leaves the object in the habort(6) state.
The `tuxTulogTable` group represents runtime attributes of userlog files within an application. The index into this table is `tuxTulogSerNo`. The values returned for objects in this table are controlled by the MIB control group `tuxTulogCtrl`.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTulogSerNo</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.1</td>
</tr>
<tr>
<td>tuxTulogLmid</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.2</td>
</tr>
<tr>
<td>tuxTulogPmid</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.3</td>
</tr>
<tr>
<td>tuxTulogMmDdYy</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.4</td>
</tr>
<tr>
<td>tuxTulogTime</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.5</td>
</tr>
<tr>
<td>tuxTulogLine</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.6</td>
</tr>
<tr>
<td>tuxTulogMsg</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.7</td>
</tr>
<tr>
<td>tuxTulogTpTranId</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.8</td>
</tr>
<tr>
<td>tuxTulogXid</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.9</td>
</tr>
<tr>
<td>tuxTulogPid</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.10</td>
</tr>
<tr>
<td>tuxTulogSeverity</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.11</td>
</tr>
<tr>
<td>tuxTulogCat</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.12</td>
</tr>
<tr>
<td>tuxTulogMsgNum</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.13</td>
</tr>
<tr>
<td>tuxTulogProcName</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.14</td>
</tr>
<tr>
<td>tuxTulogThreadID</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.20</td>
</tr>
<tr>
<td>tuxTulogContextID</td>
<td>.1.3.6.1.4.1.140.300.9.1.1.30</td>
</tr>
</tbody>
</table>
tuxTulogTable

**tuxTulogSerNo**
- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: A running serial number for the rows in tuxTulogTable.

**tuxTulogLmid**
- **Syntax**: DisplayString(SIZE(1..30))
- **Access**: read-only
- **Description**: Retrieval machine logical machine identifier.

**tuxTulogPmid**
- **Syntax**: DisplayString(SIZE(1..30))
- **Access**: read-only
- **Description**: Physical machine identifier.

**tuxTulogMmDdYy**
- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Month, day, and year of the log file.

**tuxTulogTime**
- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Time at which the message was generated.
**tuxTulogLine**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Line number of the message in the log file.

**tuxTulogMsg**

- **Syntax**: DisplayString(SIZE(1..256))
- **Access**: read-only
- **Description**: The entire text of the userlog message as it appears in the userlog file.

**tuxTulogTpTranId**

- **Syntax**: DisplayString(SIZE(1..78))
- **Access**: read-only
- **Description**: Transaction identifier as returned from `tpsuspend(3)`. The data in this field should not be interpreted directly by the user except for equality comparison. Messages not associated with transactions retrieve a 0-length string as the value for this attribute.

**tuxTulogXid**

- **Syntax**: DisplayString(SIZE(1..78))
- **Access**: read-only
- **Description**: Transaction identifier as returned from `tx_info(3)`. The data in this field should not be interpreted directly by the user except for equality comparison. Messages not associated with transactions retrieve a 0-length string as the value for this attribute.

**tuxTulogPid**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Process identifier of the client or server that generated the userlog message.
tuxTulogTable

**tuxTulogSeverity**

Syntax: `DisplayString(SIZE(1..30))`
Access: read-only
Description: Severity of message, if any.

**tuxTulogCat**

Syntax: `DisplayString(SIZE(1..30))`
Access: read-only
Description: Catalog name from which the message was derived, if any.

**tuxTulogMsgNum**

Syntax: `INTEGER`
Access: read-only
Description: Catalog message number, if the message was derived from a catalog.

**tuxTulogProcName**

Syntax: `DisplayString(SIZE(1..30))`
Access: read-only
Description: Process name of the client or server that generated the userlog message.

**tuxTulogThreadID**

Syntax: `INTEGER`
Access: read-only
Description: Identifier for the thread that wrote this userlog message.
tuxTulogContextID

Syntax  INTEGER
Access   read-only
Description  Identifier for this particular application association.
The values of objects in this group control the ulog messages returned by the `tuxTulogTable`.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTulogLmidCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.1</td>
</tr>
<tr>
<td><code>tuxTulogPmidCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.2</td>
</tr>
<tr>
<td><code>tuxTulogMmddyyCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.3</td>
</tr>
<tr>
<td><code>tuxTulogTimeCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.4</td>
</tr>
<tr>
<td><code>tuxTulogEndTimeCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.5</td>
</tr>
<tr>
<td><code>tuxTulogLineCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.6</td>
</tr>
<tr>
<td><code>tuxTulogMsgCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.7</td>
</tr>
<tr>
<td><code>tuxTulogTptranIdCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.8</td>
</tr>
<tr>
<td><code>tuxTulogXidCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.9</td>
</tr>
<tr>
<td><code>tuxTulogPidCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.10</td>
</tr>
<tr>
<td><code>tuxTulogSeverityCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.11</td>
</tr>
<tr>
<td><code>tuxTulogCatCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.12</td>
</tr>
<tr>
<td><code>tuxTulogMsgNumCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.13</td>
</tr>
<tr>
<td><code>tuxTulogProcNameCtrl</code></td>
<td>.1.3.6.1.4.1.140.300.9.2.14</td>
</tr>
</tbody>
</table>
tuxTulogLmidCtrl

Syntax  
DisplayString(SIZE(1..30))

Access  
read-write

Description  
Logical machine ID to qualify machine from where the userlog file is read for tuxTulogTable. By default, the ULOG files from the local host are returned, per the ULOGPFX. To revert to the default setting, set this object to null.

tuxTulogPmidCtrl

Syntax  
DisplayString(SIZE(1..30))

Access  
read-write

Description  
Physical machine name to qualify the source machine for userlog messages to be listed in tuxTulogTable. By default, messages from all hosts within ULOG files qualified by tuxTulogLmidCtrl are returned. To revert to the default setting, set this object to null.

tuxTulogMmddyyCtrl

Syntax  
INTEGER

Access  
read-write

Description  
Date value to qualify userlog messages listed in tuxTulogTable. Default value is current date. To reset the value of the qualifier to its default, set this object to 0.

tuxTulogTimeCtrl

Syntax  
INTEGER

Access  
read-write

Description  
Starting time of the time range for which the userlog messages are listed in tuxTulogTable. This number is calculated as under - “hrs*10000 + mins*100 + secs”. The default value is 0.
**tuxTulogEndTimeCtrl**

Syntax: INTEGER
Access: read-write
Description: Ending time of the time range for which the userlog messages are listed in tuxTulogTable. This number is calculated as under - “hrs*10000 + mins*100 + secs”. By default, the maximum value is considered. To revert to the default setting, set this object to 0.

**tuxTulogLineCtrl**

Syntax: INTEGER
Access: read-write
Description: Beginning line number from which the userlog messages are listed in tuxTulogTable. By default, all messages are returned. To revert to the default setting, set this object to 0.

**tuxTulogMsgCtrl**

Syntax: DisplayString (SIZE(1..30))
Access: read-write
Description: Regular expression to qualify userlog messages listed in tuxTulogTable on the basis of the message body. By default, all messages are listed. To revert to the default setting, set this object to null.

**tuxTulogTptranIdCtrl**

Syntax: DisplayString (SIZE(1..78))
Access: read-write
Description: Value of tuxTpTranId to qualify messages to be displayed in the in tuxTulogTable. By default, all messages are returned. To revert to the default setting, set it to null.
2 Tuxedo Core MIB

**tuxTulogXidCtrl**

Syntax: `DisplayString(SIZE(1..30))`

Access: read-write

Description: Value of `tuxTranXid` to qualify messages to be displayed in the `tuxTulogTable`. By default, all messages are returned. To revert to the default setting, set it to `null`.

**tuxTulogPidCtrl**

Syntax: `INTEGER`

Access: read-write

Description: Value of process Id of the source to qualify messages to be displayed in the `tuxTulogTable`. By default, messages with any pid are listed. To revert to the default setting, set this object to `0`.

**tuxTulogSeverityCtrl**

Syntax: `DisplayString(SIZE(1..30))`

Access: read-write

Description: Regular expression to qualify userlog messages to be listed in `tuxTulogTable` on the basis of message severity, if any. By default, messages with any severity are listed. To revert to the default setting, set this object to `null`.

**tuxTulogCatCtrl**

Syntax: `DisplayString(SIZE(1..30))`

Access: read-write

Description: Regular expression to qualify userlog messages to be listed in `tuxTulogTable` on the basis of the catalog name, if any. By default, messages from all catalogs are listed. To revert to the default setting, set this object to `null`.

2-152 BEA SNMP Agent MIB Reference
**tuxTulogMsgNumCtrl**

**Syntax**  INTEGER

**Access**  read-write

**Description**  Message number in catalog to qualify userlog messages to be listed in tuxTulogTable. By default, all message numbers are returned. To revert to the default setting, set this object to 0.

**tuxTulogProcNameCtrl**

**Syntax**  DisplayString (SIZE(1..30))

**Access**  read-write

**Description**  Regular expression to qualify userlog messages to be listed in tuxTulogTable on the basis of the process name that generated the message, if known. By default, all messages are returned. To revert to the default setting, set this object to null.
This table represents application attributes of network groups. Network groups are groups of logical machine IDs that can communicate over the network address defined in the `tuxTnetMapNaddr` object in the `tuxTnetMapTbl` table entry. For row creation, a SET request with `tuxTnetGrpName`, `tuxTnetGrpNo` and `tuxTnetGrpPrio` is required. `tuxTnetGrpNo` provides the index into this table.

**Note:** This table is supported only on Tuxedo 6.4 or later.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTnetGrpName</td>
<td>.1.3.6.1.4.1.140.300.28.1.1</td>
</tr>
<tr>
<td>tuxTnetGrpNo</td>
<td>.1.3.6.1.4.1.140.300.28.1.2</td>
</tr>
<tr>
<td>tuxTnetGrpState</td>
<td>.1.3.6.1.4.1.140.300.28.1.3</td>
</tr>
<tr>
<td>tuxTnetGrpPrio</td>
<td>.1.3.6.1.4.1.140.300.28.1.4</td>
</tr>
</tbody>
</table>

**tuxTnetGrpName**

- **Syntax**  
  `DisplayString (SIZE(1..30))`
- **Access**  
  read-write
- **Description**  
  Logical name of the network group. A group name is a string of printable characters and cannot contain a pound sign (#), comma (,), colon (:), or newline character. This object can be updated only during row creation.

**tuxTnetGrpNo**

- **Syntax**  
  `INTEGER (1..8191)`
- **Access**  
  read-write
- **Description**  
  Group identifier of the network group. This object can be updated only during row creation.
**tuxTnetGrpState**

**Syntax**  INTEGER { valid(1) | invalid(2) }

**Access**  read-write

**Description**  A GET request retrieves configuration information for the selected tuxTnetGrpTbl instance (or instances). The following states indicate the meaning of the value that is returned:

- **GET: valid(1)**
  
  The instance is defined. This is the only valid state for this object.

- **SET: invalid(2)**
  
  Delete the selected tuxTnetGrpTbl instance from the application.

States not listed are not returned.

**tuxTnetGrpPrio**

**Syntax**  INTEGER (1..8191)

**Access**  read-write

**Description**  The priority band for this network group. All network groups that have an equivalent band priority are used in parallel.
The instances in the `tuxTnetMapTbl` associate `tuxTmachineLmid`s to an instance in the `tuxTnetGrpTbl`. The rows in this table identify which logical machines belong to which network groups. For row creation, a `SET` request with at least `tuxTnetMapNaddr` is needed. The index into this table is provided by `tuxTnetMapGrpNo` and `tuxTnetMapLmid`.

**Note:** This table is supported only on Tuxedo 6.4 or later.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTnetMapGrpName</code></td>
<td>.1.3.6.1.4.1.140.300.33.1.1</td>
</tr>
<tr>
<td><code>tuxTnetMapGrpNo</code></td>
<td>.1.3.6.1.4.1.140.300.33.1.2</td>
</tr>
<tr>
<td><code>tuxTnetMapLmid</code></td>
<td>.1.3.6.1.4.1.140.300.33.1.3</td>
</tr>
<tr>
<td><code>tuxTnetMapState</code></td>
<td>.1.3.6.1.4.1.140.300.33.1.4</td>
</tr>
<tr>
<td><code>tuxTnetMapNaddr</code></td>
<td>.1.3.6.1.4.1.140.300.33.1.5</td>
</tr>
<tr>
<td><code>tuxTnetMapMinEncryptBit</code></td>
<td>.1.3.6.1.4.1.140.300.33.1.6</td>
</tr>
<tr>
<td><code>tuxTnetMapMaxEncryptBit</code></td>
<td>.1.3.6.1.4.1.140.300.33.1.7</td>
</tr>
</tbody>
</table>

**tuxTnetMapGrpName**

**Syntax**

`DisplayString(SIZE(1..30))`

**Access**

read-write

**Description**

The logical name of the network group. A group name is a string of printable characters and cannot contain a pound sign (#), comma (,), colon (:), or a newline character.
**tuxTnetMapGrpNo**

Syntax: INTEGER (1..8191)

Access: read-write

Description: Identifier for this logical network group. This object can be updated only during row creation.

**tuxTnetMapLmid**

Syntax: DisplayString (SIZE(1..30))

Access: read-write

Description: Logical machine name for this network mapping. This object can be updated only during row creation.

**tuxTnetMapState**

Syntax: Integer { valid(1) | invalid(2) }

Access: read-write

Description: A GET request retrieves configuration information for the selected tuxTnetMapTbl instance (or instances). The following states indicate the meaning of the value of tuxTnetMapState that is returned:

GET: valid(1)

The instance is defined. This is the only valid state for this object.

SET: invalid(2)

Delete the selected tuxTnetMapTbl instance from the application. If any network links are active as a result of the mapping, they are disconnected. This disconnection can cause a state change in tuxTBridgeTbl instances associated with the network links.

States not listed are not returned.
**tuxTnetMapNaddr**

**Syntax**  
DisplayString (SIZE (1..78))

**Access**  
read-write

**Description**  
Specifies the complete network address to be used by the BRIDGE process placed on the logical machine as its listening address. The listening address for a BRIDGE is the means by which it is contacted by other BRIDGE processes participating in a networked application, that is, if the value of `tuxTdomainOptions` is `lan(1)`. If the string is of the form `0xhex-digits` or `\xhex-digits`, it must contain an even number of valid hexadecimal digits. These forms are translated internally into a character array containing the hexadecimal representation of the string specified. For TCP/IP addresses, either the `//hostname:port` or `.#.#.#:port` format is used.

**tuxTnetMapMinEncryptBit**

**Syntax**  
INTEGER { none(1) | 40-bit(2) | 128-bit(3) | unknown(4) }

**Access**  
read-write

**Description**  
Specifies the required level of encryption when establishing a network link to this machine.

none(1)  
No encryption.

40-bit(2) and 128-bit(3)  
These values specify the encryption key length (in bits). If this minimum level of encryption cannot be met, the attempt to establish the link fails.

The default value is none(1). Modifications to this object do not affect network links that have already been established.

**tuxTnetMapMaxEncryptBit**

**Syntax**  
Integer { none(1) | 40-bit(2) | 128-bit(3) | unknown(4) }

**Access**  
read-write

**Description**  
Encryption can be negotiated up to the specified level when establishing a network link.
none(1)

No encryption.

40-bit(2) and 128-bit(3)

These values specify the encryption key length (in bits).

The default value is 128-bit(3). Modifications to this object do not affect network links that are already established.

tuxTserverCtxtTbl

This table represents configuration and runtime attributes of individual server dispatch contexts within an application. This class is defined for both single-context and multi-context servers. For single-context servers, the values in this class are repeated as part of the tuxTsrvrTbl class. The objects in this group are read-only.

These attribute values provide runtime tracking of statistics and resources associated with each server dispatch context.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTserverCtxtGrp</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.10</td>
</tr>
<tr>
<td>tuxTserverCtxtServerID</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.20</td>
</tr>
<tr>
<td>tuxTserverCtxtContextID</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.30</td>
</tr>
<tr>
<td>tuxTserverCtxtCltLmId</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.40</td>
</tr>
<tr>
<td>tuxTserverCtxtCltPid</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.50</td>
</tr>
<tr>
<td>tuxTserverCtxtCltReply</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.60</td>
</tr>
<tr>
<td>tuxTserverCtxtCmtRet</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.70</td>
</tr>
<tr>
<td>tuxTserverCtxtCurConv</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.80</td>
</tr>
<tr>
<td>tuxTserverCtxtCurReq</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.90</td>
</tr>
<tr>
<td>tuxTserverCtxtCurService</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.100</td>
</tr>
</tbody>
</table>
2  Tuxedo Core MIB

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTserverCtxtLastGrp</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.110</td>
</tr>
<tr>
<td>tuxTserverCtxtSvcTimeOut</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.120</td>
</tr>
<tr>
<td>tuxTserverCtxtTimeLeft</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.130</td>
</tr>
<tr>
<td>tuxTserverCtxtTranLev</td>
<td>.1.3.6.1.4.1.140.300.34.1.1.140</td>
</tr>
</tbody>
</table>

**tuxTserverCtxtGrp**

Syntax  
*DisplayString (SIZE (1..30))*

Access  
read-only

Description  
Logical name of the server group. Server group names cannot contain an asterisk (*), comma (,), or colon (:).

**tuxTserverCtxtServerID**

Syntax  
*INTEGER (SIZE (1..30000))*

Access  
read-only

Description  
Unique (within the server group) server identification number.

**tuxTserverCtxtContextID**

Syntax  
*INTEGER (SIZE (-2..29999))*

Access  
read-only

Description  
Identifier of this particular server context.

**tuxTserverCtxtCltLmId**

Syntax  
*INTEGER (SIZE (1..30))*

Access  
read-only

2-160  BEA SNMP Agent MIB Reference
**Description**  
Logical machine for the initiating client or server. The initiating client or server is the process that made the service request on which the server is currently working.

**tuxTserverCtxtCltPid**

Syntax: INTEGER

Access: read-only

Description: UNIX system process identifier for the initiating client or server.

Limitation: This is a UNIX system-specific attribute that cannot be returned if the platform on which the application is being run is not UNIX-based.

**tuxTserverCtxtCltReply**

Syntax: INTEGER { yes (1) | no (2) }

Access: read-only

Description: The initiating client or server is expecting a reply (yes (1)) or is not expecting a reply (no (2)).

**tuxTserverCtxtCmtRet**

Syntax: INTEGER { complete (1) | logged (2) }

Access: read-only

Description: This is the setting of the TP_COMMIT_CONTROL characteristic for this server.

See the description of the BEA Tuxedo ATMI function tpscmt (3c) for details on this characteristic.

**tuxTserverCtxtCurConv**

Syntax: INTEGER

Access: read-only

Description: Number of conversations initiated by this server through tpconnect() that are still active.
### tuxTserverCtxtCurReq

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Number of requests initiated by this server through tpcall() or t pacall() that are still active.

### tuxTserverCtxtCurService

**Syntax**  
DisplayString (SIZE (1..15))

**Access**  
read-only

**Description**  
Service name, if any, on which the server is currently working.

### tuxTserverCtxtLastGrp

**Syntax**  
INTEGER (1..29999)

**Access**  
read-only

**Description**  
Time left (in seconds), if any, for this server to process the current service request. A value of 0 for an active service indicates that no time out processing is being done.

See tuxTsvcTbl: tuxTsvcTimeOut for more information.

### tuxTserverCtxtSvcTimeOut

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Server group number (tuxTgroupTable: tuxTgroupNo) of the last service request made or conversation initiated from this server outward.

### tuxTserverCtxtTimeLeft

**Syntax**  
INTEGER

**Access**  
read-only
**tuxTserverCtxtTbl**

| Description | Time left (in seconds) for this server to receive the reply for which it is currently waiting before it will time out. This timeout can be a transactional timeout or a blocking timeout. |

**tuxTserverCtxtTranLev**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Current transaction level for this server. 0 indicates that the server is not currently involved in a transaction.
You can use the Tuxedo event filters to define a subset of Tuxedo event notifications to be generated for each Tuxedo or WLE domain being monitored. The columnar objects in the beaEvtFilterTable correspond to fields in TMEVENT_FILTER entries in the BEA Manager configuration file (beamgr.conf). Refer to the “Configuration Files” chapter in the Agent Integrator Reference Manual.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>beaEvtFilterTblStatus</td>
<td>.1.3.6.1.4.1.140.300.14.1</td>
</tr>
<tr>
<td>beaEvtFilterTable</td>
<td>.1.3.6.1.4.1.140.300.14.2</td>
</tr>
</tbody>
</table>

**beaEvtFilterTblStatus**

**Syntax**

INTEGER { sync(1) | dirty(2) }

**Access**

read-write

**Description**

When the agent starts, this value is always sync(1). If any change is done to beaEvtFilterTable through SET requests, the value of this object becomes dirty(2) and the changes made to beaEvtFilterTable do not take effect. The changes made to the beaEvtFilterTable take effect only when you set the value of this object to sync(1). When you set the value to sync(1), all changes since the last synchronization are applied to the event-processing module.
beaEventFilters

beaEvtFilterTable

This MIB group represents all the event filters defined for the SNMP Agent. These are used to determine the collection of events to be forwarded as SNMP trap notifications.

**Note:** Changes to this table are applied only once beaEvtFilterStatus is set to sync(1).

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>beaEvtFilterId</td>
<td>.1.3.6.1.4.1.140.300.14.1.1.1</td>
</tr>
<tr>
<td>beaEvtAgentName</td>
<td>.1.3.6.1.4.1.140.300.14.1.1.2</td>
</tr>
<tr>
<td>beaEvtExpr</td>
<td>.1.3.6.1.4.1.140.300.14.1.1.3</td>
</tr>
<tr>
<td>beaEvtFilter</td>
<td>.1.3.6.1.4.1.140.300.14.1.1.4</td>
</tr>
<tr>
<td>beaEvtFilterState</td>
<td>.1.3.6.1.4.1.140.300.14.1.1.5</td>
</tr>
</tbody>
</table>

**beaEvtFilterId**

- **Syntax**  
  `DisplayString (SIZE (1..16))`
- **Access**  
  read-write
- **Description**  
  A unique identifier for the event filter within the filter table.
  
  **Note:** This object can be **SET** only during row creation.

**beaEvtAgentName**

- **Syntax**  
  `DisplayString (SIZE (1..32))`
- **Access**  
  read-only
- **Description**  
  This logical agent name of the agent supporting this filter. This object is provided only for user convenience since the MIB only returns the event filters for the agent that was queried.
beaEvtExpr

Syntax  DisplayString (SIZE (1..255) )
Access  read-write
Description  An event name expression. Consult the BEA Tuxedo Reference Manual entry for recomp(3) for the format of this expression. For a Tuxedo system event to be forwarded as an SNMP trap, its name should match this expression. Consult the BEA Tuxedo Reference Manual (EVENTS(5)) for a list of Tuxedo event names. The default for this object is all system events.
Examples  \.Sys.*
matches all system events. (This is the default.)
\.SysServer.*
matches all system events related to servers.
A value of NONE blocks all events from being forwarded by the selected agent and overrides any other filter table entries for the same logical agent name.

beaEvtFilter

Syntax  DisplayString (SIZE (1..255) )
Access  read-write
Description  An event filter expression. Each Tuxedo event is accompanied by an FML buffer that contains pertinent information about the event. The buffer’s contents are evaluated with respect to this filter, if it is present. The filter must evaluate to TRUE or the event is not forwarded.

The SNMP Agent uses this attribute as an argument to tpsubscribe(). Please refer to the BEA Tuxedo Reference Manual for further information.

Example  TA_EVENT_SEVERITY=='ERROR' || TA_EVENT_SEVERITY=='WARN'
   TA_EVENT_SEVERITY!=='INFO'
   TA_EVENT_LMID=='SITE1'
This filter selects events with a severity of either ERROR or WARNING.
beaEvtFilterState

Syntax  INTEGER { active(1) | inactive(2) | invalid(3) }

Access  read-write

Description  This object denotes the current state of the event filter instance.

GET {active(1)|inactive(2)}

A GET operation retrieves configuration and runtime information for the selected beaEvtFilterTbl instance(s). The following states indicate the meaning of a beaEvtFilterState returned in response to a GET request. States not listed are not returned.

active(1)
This filter is being used.

inactive(2)
This filter is not being used.

SET {active(1)|inactive(2)|invalid(3)}

A SET operation updates configuration and runtime information for the selected beaEvtFilterTbl instance. The following states indicate the meaning of a beaEvtFilterState set in a SET request. States not listed cannot be set.

active(1)
 Activate the event filter. This can be done only when the filter is in the inactive(2) state.

inactive(2)
 Inactivate the event filter. This can be done only when the filter is in the active(1) state.

invalid(3)
 Inactivate (if active) and remove this event filter.
CHAPTER 3

Domains MIB

The Domains MIB uses improved group and attribute terminology to describe the interaction between domains. This improved terminology has also been applied to DMCONFIG file syntax.

These terminology improvements eliminate multiple uses of the term “domain” and introduce terms that more clearly describe the actions that occur. For example, the term access point defines an object through which you gain access to another object. Therefore, you access a remote domain through a remote domain access point, and remote domains gain access to a local domain through a local domain access point. The Domains MIB consists of the following groups.

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmAclTable</td>
<td>Domains Access Control List</td>
</tr>
<tr>
<td>tuxDmConnectionTable</td>
<td>Domain Access Points Connection Status</td>
</tr>
<tr>
<td>tuxDmExportTable</td>
<td>Resources for Exporting to Remote Domains</td>
</tr>
<tr>
<td>tuxDmImportTable</td>
<td>Resources imported through access points</td>
</tr>
<tr>
<td>tuxDmLocalTable</td>
<td>Defines a local domain access point</td>
</tr>
<tr>
<td>tuxDmOsitpTable</td>
<td>Defines the local or remote OSI TP protocol</td>
</tr>
<tr>
<td>tuxDmPasswordTable</td>
<td>Inter-domain authentication</td>
</tr>
<tr>
<td>tuxDmPrincipalMapTable</td>
<td>For mapping principal names</td>
</tr>
<tr>
<td>tuxDmRemoteTable</td>
<td>Remote domain configuration information</td>
</tr>
<tr>
<td>tuxDmResourcesTable</td>
<td>Domains-specific configuration information</td>
</tr>
<tr>
<td>tuxDmRoutingTable</td>
<td>Routing criteria information</td>
</tr>
</tbody>
</table>
### Domains MIB

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmrPrincipalTable</td>
<td>Configuration info for remote principal names</td>
</tr>
<tr>
<td>tuxDmSnaCRMTable</td>
<td>SNA-CRM-specific configuration info</td>
</tr>
<tr>
<td>tuxDmSnaLinkTable</td>
<td>snax-specific configuration info</td>
</tr>
<tr>
<td>tuxDmSnaStackTable</td>
<td>Defines SNA stack used by a specific SNA CRM</td>
</tr>
<tr>
<td>tuxDmTdomainTable</td>
<td>Defines the TDomain specific configuration</td>
</tr>
<tr>
<td>tuxDmTopendTable</td>
<td>BEA TOP END-Specific Configuration info</td>
</tr>
<tr>
<td>tuxDmTransactionTable</td>
<td>Info about transactions that span domains</td>
</tr>
</tbody>
</table>
**tuxDmAclTable**

This Tuxedo MIB group represents access control information for domains.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmAclName</td>
<td>.1.3.6.1.4.1.140.300.80.1.1.10</td>
</tr>
<tr>
<td>tuxDmrAccessPointList</td>
<td>.1.3.6.1.4.1.140.300.80.1.1.20</td>
</tr>
<tr>
<td>tuxDmAclState</td>
<td>.1.3.6.1.4.1.140.300.80.1.1.30</td>
</tr>
</tbody>
</table>

**tuxDmAclName**

Syntax: `DisplayString (SIZE(1..15))`

Access: read-only

Description: The access control list name.

**tuxDmrAccessPointList**

Syntax: `DisplayString (SIZE(1..1000))`

Access: read-write

Description: The list of remote domain access points associated with this access control list. `tuxDmrAccessPointList` is a comma-separated list of remote access point names (that is, the value of the `tuxDMRemoteDmAccessPoint` attribute of a valid `tuxDMRemote` object). The list can contain up to 50 remote access point identifier elements. Setting this attribute to "*" means that all the remote domains in the configuration are associated with this entry. Blank string means no remote access points are associated with this entry. The default is - (equivalent of NULL string).

**tuxDmAclState**

Syntax: `INTEGER { valid(1) | invalid (2) }`

Access: read-write

Description: This object denotes the current state of the `tuxDmAcl` instance.
GET requests:
  valid (1): tuxDmAcl object is defined and inactive. This is the only valid state for this class. ACL groups are never active.

SET requests:
  invalid (2): Delete.
tuxDmConnectionTable

This class represents the status of connections between domain access points.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmConDmlAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.90.1.1.10</td>
</tr>
<tr>
<td>tuxDmConDmrAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.90.1.1.20</td>
</tr>
<tr>
<td>tuxDmConDmType</td>
<td>.1.3.6.1.4.1.140.300.90.1.1.30</td>
</tr>
<tr>
<td>tuxDmConState</td>
<td>.1.3.6.1.4.1.140.300.90.1.1.40</td>
</tr>
<tr>
<td>tuxDmConDmCurEncryptBits</td>
<td>.1.3.6.1.4.1.140.300.90.1.1.50</td>
</tr>
</tbody>
</table>

**tuxDmConDmlAccessPoint**

- **Syntax**: DisplayString (SIZE(1..24))
- **Access**: read-only
- **Description**: The name of the local domain access point that identifies the connection between the domains.

**tuxDmConDmrAccessPoint**

- **Syntax**: DisplayString (SIZE(1..24))
- **Access**: read-only
- **Description**: The name of the remote domain access point that identifies the connection between the domains.

**tuxDmConDmType**

- **Syntax**: INTEGER { tdomain (1) | topend (2) }
- **Access**: read-only
- **Description**: The type of domain—either tdomain or topend.
3 Domains MIB

**tuxDmConState**

Syntax INTEGER { active (1) | suspended (2) | initializing (3) | inactive (4) | unknown (5)}

Access read-only

Description This object denotes the current state of the tuxDmConnection instance.

GET requests:
- active (1): The connection is active.
- suspended (2): The connection is awaiting retry.
- initializing (3): The connection is initializing.
- inactive (4): The specified domain access points are disconnected. (Returned in case of Tuxedo 7.1 and later only.)
- unknown (5): The state cannot be determined.

SET requests:
- active (1): Connect the specified domain access points. If the current state is suspended or inactive, SET:active places the connection into the state initializing, otherwise, there is no change.
- inactive (4): Disconnect the specified domain access points and destroy the tuxDmConnection object

**tuxDmConDmCurEncryptBits**

Syntax INTEGER { enc-0-bit (1) | enc-40-bits (2) | enc-56-bits (3) | enc-128-bits (4)}

Access read-only

Description This attribute is available when tuxDmConDmType=tdomain.

The level of encryption in use on this connection:
- enc-0-bit (1) means no encryption
- enc-40-bits (2) specifies the encryption length in bits
- enc-56-bits (3) specifies the encryption length in bits
- enc-128-bits (4) specifies the encryption length in bits.
tuxDmExportTable

This class represents local resources that are exported to one or more remote domains through a local access point.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmExpDmResourceName</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.10</td>
</tr>
<tr>
<td>tuxDmExpDmlAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.20</td>
</tr>
<tr>
<td>tuxDmExpState</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.30</td>
</tr>
<tr>
<td>tuxDmExpDmAc1Name</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.40</td>
</tr>
<tr>
<td>tuxDmExpDmConv</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.50</td>
</tr>
<tr>
<td>tuxDmExpDmResourceType</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.60</td>
</tr>
<tr>
<td>tuxDmExpDmRemoteName</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.70</td>
</tr>
<tr>
<td>tuxDmExpDmInBufType</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.80</td>
</tr>
<tr>
<td>tuxDmExpDmOutBufType</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.90</td>
</tr>
<tr>
<td>tuxDmExpDmTopendProduct</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.100</td>
</tr>
<tr>
<td>tuxDmExpDmTopendFunction</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.110</td>
</tr>
<tr>
<td>tuxDmExpDmTopendTarget</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.120</td>
</tr>
<tr>
<td>tuxDmExpDmTopendQualifier</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.130</td>
</tr>
<tr>
<td>tuxDmExpDmTopendRtqGroup</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.140</td>
</tr>
<tr>
<td>tuxDmExpDmTopendRtqName</td>
<td>.1.3.6.1.4.1.140.300.100.1.1.150</td>
</tr>
</tbody>
</table>

**tuxDmExpDmResourceName**

- **Syntax**: `DisplayString (SIZE(1..15))`
- **Access**: read-only
Description
The local resource name for entries of resource type service (the service name), qspace (the queue space name), and qname (the queue name). For a service entry, the value of this attribute corresponds to the value of an active tuxTSrvGrp:tuxTsvcName object. This resource is exported to other domains with the same name or with the alias defined in the tuxDmExpDmRemoteName or tuxDmExpDmTopend* variables.

tuxDmExpDmlAccessPoint

Syntax
DisplayString (SIZE(1..30))

Access
read-only

Description
The local access point name. Setting this attribute to "*" means the resource is available at all local access points.

tuxDmExpState

Syntax
INTEGER { valid (1) | invalid (2) }

Access
read-write

Description
This object denotes the current state of the tuxDmExport instance.

GET requests:
valid (1): The object exists.

SET requests:
invalid (2): Delete object.

tuxDmExpDmAclName

Syntax
DisplayString (SIZE(1..15))

Access
read-write

Description
The name of a tuxDmAcl object to use for security on this local service. When access is permitted from topend remote access points, this optional attribute can be specified if tuxDmExpDmResourceType=service or qspace. This attribute is not permitted if tuxDmExpDmResourceType=qname.
**tuxDmExpDmConv**

Syntax  
INTEGER { yes (1) | no (2) }

Access  
read-only

Description  
Specifies whether this local service is conversational or not. When access is permitted from topend remote access points, this attribute must be set to no (2) for entries of tuxDmExpDmResourceType=qspace or qname.

**tuxDmExpDmResourceType**

Syntax  
INTEGER { service (1) | qspace (2) | qname (3) }

Access  
read-only

Description  
Specifies whether this entry is for a service, qspace or qname. The default is service.

**tuxDmExpDmRemoteName**

Syntax  
DisplayString (SIZE(1..30))

Access  
read-write

Description  
For entries of type service or qspace, this attribute specifies the name exported through non-topend remote access points.

**tuxDmExpDmInBufType**

Syntax  
DisplayString (SIZE(1..513))

Access  
read-write

Description  
Attributes available from remote access points of tuxDmExpDmResourceType=snax|ositp|topend:

type[::subtype] - Input buffer type, optionally followed by subtype.

If this attribute is present, it defines the buffer type (and subtype) accepted. This attribute should be defined for entries of tuxDmExpDmResourceType=service when access is permitted from remote access points using ositp with the UDT application context, or when using snax. When access is permitted from topend remote access
For BEA TopEnd service and queue name entries, the valid values for type are: FML32, CARRAY, and X_OCTET.

### **tuxDmExpDmOutBufType**

**Syntax**  
DisplayString (SIZE(1..513))

**Access**  
read-write

**Description**  
Attributes available from remote access points of tuxDmExpDmResourceType=snax|ositp|topend:

type[:subtype] -Output buffer type, optionally followed by subtype.

If this attribute is present, it defines the buffer type (and subtype) output by the service. This attribute should be defined for entries of tuxDmExpDmResourceType=service when access is permitted from remote access points using ositp with the UDT application context, or when using snax. When access is permitted from topend remote access points, this optional attribute can be specified if tuxDmExpDmResourceType=service. This attribute is not permitted if tuxDmExpDmResourceType=qspace and qname.

For BEA TopEnd service and queue name entries, the valid values for type are FML32, CARRAY, and X_OCTET.

### **tuxDmExpDmTopendProduct**

**Syntax**  
DisplayString (SIZE(1..32))

**Access**  
read-write

**Description**  
Attributes available from remote access points of tuxDMRemoteDmType=topend:

The BEA TOP END product name. When access is permitted from topend remote access points, this attribute must be specified if tuxDmExpDmResourceType=service. This attribute is not permitted if tuxDmExpDmResourceType=qspace or qname.
**tuxDmExpDmTopendFunction**

- **Syntax**: DisplayString (SIZE(1..8))
- **Access**: read-write
- **Description**: Attributes available from remote access points of `tuxDmRemoteDmType=topend`:
  
  The BEA TOP END function name. When access is permitted from topend remote access points, this attribute must be specified if `tuxDmExpDmResourceType=service` or `qname`. This attribute is not permitted if `tuxDmExpDmResourceType=qspace`.

**tuxDmExpDmTopendTarget**

- **Syntax**: DisplayString (SIZE(1..8))
- **Access**: read-write
- **Description**: Attributes available from remote access points of `tuxDmRemoteDmType=topend`:
  
  The BEA TOP END Message Sensitive Routing (MSR) target. This attribute is optional for entries of `tuxDmExpDmResourceType=service`, `qspace`, and `qname` when access is permitted from topend remote access points.

**tuxDmExpDmTopendQualifier**

- **Syntax**: INTEGER
- **Access**: read-write
- **Description**: Attributes available from remote access points of `tuxDmRemoteDmType=topend`:
  
  This attribute is optional for entries of `tuxDmExpDmResourceType=service` or `qname` when access is permitted from topend remote access points. This attribute is not permitted if `tuxDmExpDmResourceType=qspace`.

**tuxDmExpDmTopendRtgGroup**

- **Syntax**: DisplayString (SIZE(1..32))
- **Access**: read-write
- **Description**: Attributes available from remote access points of `tuxDmRemoteDmType=topend`:
The BEA TOP END Recoverable Transaction Queuing (RTQ) queue group name. This attribute must be specified for `tuxDmRemoteDmType=qspace` when access is permitted from topend remote access points. This attribute is not permitted if `tuxDmRemoteDmType=service` or `qname`.

**tuxDmExpDmTopendRtqName**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>DisplayString (SIZE(1..8))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
</tbody>
</table>
| Description   | Attributes available from remote access points of `tuxDmRemoteDmType=topend`:

The BEA TOP End RTQ queue name. This attribute must be specified for `tuxDmExpDmResourceType=qspace` and access is permitted from topend remote access points. This attribute is not permitted if `tuxDmExpDmResourceType=service` or `qname`.
tuxDmImportTable

This group represents remote resources that are imported through one or more remote domain access points and made available to the local domain through one or more local domain access points.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmImpDmResourceName</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.10</td>
</tr>
<tr>
<td>tuxDmImpDmrAccessPointList</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.20</td>
</tr>
<tr>
<td>tuxDmImpDmlAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.30</td>
</tr>
<tr>
<td>tuxDmImpState</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.40</td>
</tr>
<tr>
<td>tuxDmImpDmAutoTran</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.50</td>
</tr>
<tr>
<td>tuxDmImpDmConv</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.60</td>
</tr>
<tr>
<td>tuxDmImpDmLoad</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.70</td>
</tr>
<tr>
<td>tuxDmImpDmPrio</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.80</td>
</tr>
<tr>
<td>tuxDmImpDmResourceType</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.90</td>
</tr>
<tr>
<td>tuxDmImpDmRemoteName</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.100</td>
</tr>
<tr>
<td>tuxDmImpDmRoutingName</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.110</td>
</tr>
<tr>
<td>tuxDmImpDmTranTime</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.120</td>
</tr>
<tr>
<td>tuxDmImpDmInBufType</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.130</td>
</tr>
<tr>
<td>tuxDmImpDmOutBufType</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.140</td>
</tr>
<tr>
<td>tuxDmImpDmteProduct</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.150</td>
</tr>
<tr>
<td>tuxDmImpDmteFunction</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.160</td>
</tr>
<tr>
<td>tuxDmImpDmteTarget</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.170</td>
</tr>
<tr>
<td>tuxDmImpDmteQualifier</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.180</td>
</tr>
<tr>
<td>tuxDmImpDmteRtqGroup</td>
<td>.1.3.6.1.4.1.140.300.110.1.1.190</td>
</tr>
</tbody>
</table>
3 Domains MIB

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmImpDmResourceName</td>
<td>1.3.6.1.4.1.140.300.110.1.1.200</td>
</tr>
</tbody>
</table>

**tuxDmImpDmResourceName**

Syntax: DisplayString (SIZE(1..15))

Access: read-only

Description: The remote resource name used for entries of resource type service (the service name, qspace (the queue space name), and qname (the queue name). This resource is imported from remote domains with the same name or with the alias defined in the tuxDmImpDmRemoteName or tuxDmImpDmte* variables.

**tuxDmImpDmrAccessPointList**

Syntax: DisplayString (SIZE(1..15))

Access: read-only

Description: Identifies the remote domain access point through which this resource should be imported. This is a comma-separated failover domain list; it can contain up to three remote domain access points. If this attribute is set to “*”, the resource can be imported from all remote access points.

**tuxDmImpDmlAccessPoint**

Syntax: DisplayString (SIZE(1..15))

Access: read-only

Description: The name of the local domain access point through which this imported resource should be made available. If set to the null string, the resource is made available through all local domain access points.

**tuxDmImpState**

Syntax: INTEGER { valid (1) | invalid (2) }

Access: read-write
Description  This object denotes the current state of the tuxDmImport instance.

GET requests:
valid: The object exists.

SET requests:
invalid: The object is deleted. A state change is allowed in the active or suspended state and results in the invalid state.

tuxDmImpDmAutoTran

Syntax  INTEGER { yes (1) | no (2) }
Access  read-write
Description  When a request is received for a resource that is not already within a transaction, this attribute automatically starts a transaction for the resource. The default is no (2).

tuxDmImpDmConv

Syntax  INTEGER { yes (1) | no (2) }
Access  read-write
Description  A boolean value (yes or no) specifies whether the service is conversational. When access is permitted from topend remote access points, this attribute must be set to no (2) for entries of tuxDmImpDmResourceType=qspace and qname.

tuxDmImpDmLoad

Syntax  INTEGER (1..32767)
Access  read-write
Description  The service load.

tuxDmImpDmPrio

Syntax  INTEGER (1..100)
Access  read-write
3 Domains MIB

**Description**
The dequeuing priority. Service requests with a higher priority are serviced first.

**tuxDmImpDmResourceType**

**Syntax**  INTEGER { service (1) | qspace (2) | qname (3) }

**Access**  read-write

**Description**  Specifies whether this entry is for a service, qspace, or qname. The default is service.

**tuxDmImpDmRemoteName**

**Syntax**  DisplayString (SIZE(1..30))

**Access**  read-write

**Description**  For entries of type service or qspace, this attribute specifies the name imported through non-topend remote access points.

**tuxDmImpDmRoutingName**

**Syntax**  DisplayString (SIZE(1..15))

**Access**  read-write

**Description**  The name of a tuxDmRoutingTable object to use for routing criteria for this service or qspace.

**tuxDmImpDmTranTime**

**Syntax**  INTEGER (1..32767)

**Access**  read-write

**Description**  Transaction time value (in seconds) of transactions automatically started for this service or qspace. Transactions are started automatically when a request not in transaction mode is received and the tuxDmImpDmAutoTran attribute is set to yes.

Limitation: Runtime updates to this attribute are not reflected in active requests.
**tuxDmImpDmInBufType**

- **Syntax**: DisplayString (SIZE(0..256))
- **Access**: read-write

**Description**: Attributes available from remote access points of tuxDmRemoteDmType=snax|ositp|topend:

- `type[:subtype]` - Input buffer type, optionally followed by subtype. If this attribute is present, it defines the buffer type (and subtype) accepted. This attribute should be defined for entries of DMRESOURCETYPE=service when access is permitted to remote access points that use ositp with the UDT application context, or that use snax. When access is permitted from topend remote access points, this optional attribute can be specified if `tuxDmImpDmResourceType=service` and qname. This attribute is not permitted if `tuxDmImpDmResourceType=qspace`. For BEA TOP END service and queue name entries, the valid values for type are: FML32, CARRAY, AND X_OCTET.

**tuxDmImpDmOutBufType**

- **Syntax**: DisplayString (SIZE(0..256))
- **Access**: read-write

**Description**: Attributes available from remote access points of tuxDmRemoteDmType=snax|ositp|topend:

**tuxDmImpDmteProduct**

- **Syntax**: DisplayString (SIZE(1..32))
- **Access**: read-write

**Description**: Attributes available from remote access points of tuxDmRemoteDmType=topend:

The BEA TOP END product name. This attribute must be specified if `tuxDmImpDmResourceType=service` or qname. It is not permitted if `tuxDmImpDmResourceType=qspace`.

**tuxDmImpDmteFunction**

- **Syntax**: DisplayString (SIZE(1..8))
Domains MIB

Access  read-write

Description  Attributes available from remote access points of \texttt{tuxDmRemoteDmType=topend}:

The BEA TOP END product name. This attribute must be specified if \texttt{tuxDmImpDmResourceType=service} or \texttt{qname}. It is not permitted if \texttt{tuxDmImpDmResourceType=qspace}.

\textbf{tuxDmImpDmteTarget}

Syntax  \texttt{DisplayString \{SIZE(1..8)\}}

Access  read-write

Description  Attributes available from remote access points of \texttt{tuxDmRemoteDmType=topend}:

The BEA TOP END Message Sensitive Routing (MSR) target. This attribute is optional for entries of \texttt{tuxDmImpDmResourceType=service}, \texttt{qspace}, and \texttt{qname}.

\textbf{tuxDmImpDmteQualifier}

Syntax  \texttt{INTEGER}

Access  read-write

Description  Attributes available from remote access points of \texttt{tuxDmRemoteDmType=topend}:

The BEA TOP END function qualifier. This attribute is optional for entries of \texttt{tuxDmImpDmResourceType=service} or \texttt{qname}. It is not permitted for entries of \texttt{tuxDmImpDmResourceType=qspace}.

\textbf{tuxDmImpDmteRtqGroup}

Syntax  \texttt{DisplayString \{SIZE(1..32)\}}

Access  read-write

Description  Attributes available from remote access points of \texttt{tuxDmRemoteDmType=topend}:

The BEA TOP END Recoverable Transaction Queuing (RTQ) queue group name. This attribute must be specified if \texttt{tuxDmImpDmResourceType=qspace}. It is not permitted if \texttt{tuxDmImpDmResourceType=service} or \texttt{qname}.
**tuxDmImpDmteRtgName**

**Syntax**
DisplayString (SIZE(1..8))

**Access**
read-write

**Description**
Attributes available from remote access points of `tuxDmRemoteDmType=topend`:

The BEA TOP END RTQ queue name. This attribute must be specified if `tuxDmImpDmResourceType=qspace`. It is not permitted if `tuxDmImpDmResourceType=service` or `qname`.
tuxDmLocalTable

This group defines a local domain access point. A local domain access point is used to control access to local services exported to remote domains and to control access to remote services imported from remote domains.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmLclDmAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.10</td>
</tr>
<tr>
<td>tuxDmLclDmAccessPointID</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.20</td>
</tr>
<tr>
<td>tuxDmLclDmSrvGroup</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.30</td>
</tr>
<tr>
<td>tuxDmLclDmType</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.40</td>
</tr>
<tr>
<td>tuxDmLclDmState</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.50</td>
</tr>
<tr>
<td>tuxDmLclDmAuditLog</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.60</td>
</tr>
<tr>
<td>tuxDmLclDmBlockTime</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.70</td>
</tr>
<tr>
<td>tuxDmLclDmMaxRapTran</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.80</td>
</tr>
<tr>
<td>tuxDmLclDmMaxTran</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.90</td>
</tr>
<tr>
<td>tuxDmLclDmSecurity</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.100</td>
</tr>
<tr>
<td>tuxDmLclDmTlogDev</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.110</td>
</tr>
<tr>
<td>tuxDmLclDmTlogName</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.120</td>
</tr>
<tr>
<td>tuxDmLclDmTlogSize</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.130</td>
</tr>
<tr>
<td>tuxDmLclDmConnectionPolicy</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.140</td>
</tr>
<tr>
<td>tuxDmLclDmRetryInterval</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.150</td>
</tr>
<tr>
<td>tuxDmLclDmMaxRetry</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.160</td>
</tr>
<tr>
<td>tuxDmLclDmConnPrincipalName</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.170</td>
</tr>
<tr>
<td>tuxDmLclDmMachineType</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.180</td>
</tr>
<tr>
<td>tuxDmLclDmBlobShmSize</td>
<td>.1.3.6.1.4.1.140.300.120.1.1.190</td>
</tr>
</tbody>
</table>
tuxDmLclDmAccessPoint

Syntax  DisplayString (SIZE(1..30))
Access  read-only
Description  This is an identifier unique within the scope of tuxDmLocal and tuxDmRemote entry names in the domain configuration.

TuxDmLclDmAccessPoint

Syntax  DisplayString (SIZE(1..30))
Access  read-write
Description  The domain access point identifier. This identifier is unique across all local and remote domain access points.

TuxDmLclDmSrvGroup

Syntax  DisplayString (SIZE(1..30))
Access  read-write
Description  The group in which the administrative servers and gateway processes of the local domain reside.

TuxDmLclDmType

Syntax  INTEGER { tdomain(1) | ositp(2) | snax(3) | topend(4) }
Access  read-write
Description  The type of domain: tdomain for a BEA Tuxedo system domain, ositp for an OSI domain, snax for an SNA domain, or topend for a BEA TOP END domain. The presence or absence of other attributes depends on the value of this attribute.

TuxDmLclState

Syntax  INTEGER { valid(1) | invalid(2) }
Access  read-write
3 Domains MIB

**tuxDmLclDmAuditLog**

**Syntax**  
DisplayString (SIZE(1..78))

**Access**  
read-write

**Description**  
The name of the audit log file for this local domain.

**tuxDmLclDmBlockTime**

**Syntax**  
INTEGER (0..32767)

**Access**  
read-write

**Description**  
Specifies the maximum wait time allowed for a blocking call. The value sets a multiplier of the SCANUNIT parameters specified in the tuxTdomain group. The value SCANUNIT * tuxDmLclDmBlockTime must be greater than or equal to SCANUNIT and less than 32,768 seconds. If this attribute is not specified, the default is set to the value of the tuxDmLclDmBlockTime attribute specified in the tuxTdomain object. A timeout always implies a failure of the affected request. Notice that the timeout specified for transactions in the tuxTdomain is always used when the request is issued within a transaction.

**tuxDmLclDmMaxRapTran**

**Syntax**  
INTEGER (0..32767)

**Access**  
read-write

**Description**  
The maximum number of remote domain access points that can be involved in a single transaction.
**tuxDmLclDmMaxTran**

**Syntax**  
INTEGER (0..32767)

**Access**  
read-write

**Description**  
The maximum number of simultaneous transactions allowed on this local domain access point. This number must be greater than or equal to the `tuxTdomainMaxGTT` object in the `tuxTdomain` group.

**tuxDmLclDmSecurity**

**Syntax**  
INTEGER { none(1) | app-pw(2) | dm-pw(3) | dm-user-pw(4) | te-clear(5) | te-safe(6) | te-private(7) }

**Access**  
read-write

**Description**  
The type of security enabled on this domain. This attribute must be set to one of the following:

- **none:** No security is enabled.

- **app-pw:**
  Valid only when `tuxDmRemoteDmType=tdomain`. Application password-based security is enabled.

- **dm-pw:**
  Valid only when `tuxDmRemoteDmType=tdomain`. Domain password-based security is enabled.

- **dm-user-pw:**
  Valid only when `tuxDmRemoteDmType=snax`. Translation of principal names is enabled.

- **te-clear:**
  Valid only when `tuxDmRemoteDmType=topend`. BEA TOP END security is enabled. Between the local domain and the BEA TOP END system. Network messages are sent in plain text.

- **te-safe:**
Valid only when `tuxDmRemoteDmType=topend`. BEA TOP END security is enabled between the local domain and the BEA TOP END system. Network messages are protected by a checksum.

**te-private:**

Valid only when `tuxDmRemoteDmType=topend`. BEA TOP END security is enabled between the local domain and the BEA TOP END system. Network messages are encrypted.

### `tuxDmLclDmTlogDev`

**Syntax**

DisplayString (SIZE(1..78))

**Access**

read-write

**Description**

The device (raw slice) or file that contains the domain TLOG for this local domain access point. The TLOG is stored as a BEA Tuxedo System VTOC table on the device. For reliability, the use of a device (raw slice) is recommended.

### `tuxDmLclDmTlogName`

**Syntax**

DisplayString (SIZE(1..30))

**Access**

read-write

**Description**

The domain TLOG name for this local domain access point. If more than one TLOG exists on the same device, each TLOG must have a unique name.

### `tuxDmLclDmTlogSize`

**Syntax**

INTEGER

**Access**

read-write

**Description**

The size in pages of the TLOG for this local domain access point. This size is constrained by the amount of space available on the device identified in `tuxDmLclTlogDev`.

### `tuxDmLclDmConnectionPolicy`

**Syntax**

INTEGER { on-demand(1) | on-startup(2) | incoming-only(3)}
Specifies the conditions under which a local domain gateway tries to establish a connection to a remote domain. Supported values are:

- **on-demand**
  Means that a connection is attempted only when requested by either a client request to a remote service or an administrative “connect” command. The default setting for this variable is on-demand. The on-demand policy provides the equivalent behavior to previous releases, in which this variable was not explicitly available.

- **on-startup**
  Means that a domain gateway attempts to establish a connection with its remote domain access points at gateway server initialization time. Remote services, (that is, services advertised by the domain gateway for this local access point) are advertised only if a connection is successfully established to that remote domain access point. Therefore, if there is no active connection to a remote domain access point, the remote services are suspended. By default, this connection policy retries failed connections every 60 seconds; however, you can specify a different value for this interval using the `tuxDmLclDmMaxRetry` and `tuxDmLclDmRetryInterval` attributes.

**tuxDmLclDmRetryInterval**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>The number of seconds between automatic attempts to establish a connection to remote domain access points. The minimum value is 0 and the maximum value is 2147483647. The default setting is 60. If <code>tuxDmLclDmMaxRetry</code> is set to 0, setting <code>tuxDmLclDmRetryInterval</code> is not allowed. This attribute is valid only when the <code>tuxDmLclDmConnectionPolicy</code> attribute is set to on-startup. For other connection policies, automatic retries are disabled.</td>
</tr>
</tbody>
</table>

**tuxDmLclDmMaxRetry**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
</table>
3 Domains MIB

Access read-write

Description The number of times that a domain gateway tries to establish connections to remote domain access points. The minimum value is 0 and the maximum is MAXLONG. MAXLONG indicates that retry processing is repeated indefinitely, or until a connection is established. For a connection policy of on-startup, the default setting for tuxDmLclMaxRetry is MAXLONG. Setting this attribute to 0 turns off the auto retry mechanism. For other connection policies, auto retries are disabled.

The tuxDmLclMaxRetry attribute is valid only when the connection policy is on-startup.

tuxDmLclDmConnPrincipalName

Syntax DisplayString {SIZE(0..511)}
Access read-write

Description The connection principal name identifier, which is the principal name used for verifying the identity of this local domain access point when establishing a connection to a remote domain access point. This attribute only applies to domains of type TDOMAIN that are running BEA Tuxedo 7.1 or later software.

This variable can contain a maximum of 511 characters (excluding the terminating null character). If this attribute is not specified, the connection principal name defaults to the tuxDmLclLclDmAccessPointId string for this local domain access point.

For default authentication plug-ins, if a value is assigned to this variable for this local domain access point, it must be the same as the value assigned to the tuxDmLclDmAccessPointId attribute for this local domain access point. If these values do not match, the local domain gateway process does not boot and the system generates the following userlog(3c) message: “ERROR: Unable to acquire credentials”.

tuxDmLclDmMachineType

Syntax DisplayString {SIZE(0..15)}
Access read-write

Description Used for grouping domains so that encoding/decoding of messages between domains can be bypassed. If it is not specified, the default is to turn encoding/decoding on. If the value set for this field is the same in both the DM_LOCAL and the DM_REMOTE section.
of a domain configuration file, data encoding/decoding is bypassed. The value set for this variable can be any string value up to 15 characters in length. It is used only for comparison.

This attribute is valid only when tuxDmRemoteDmType=tdomain.

**tuxDmLclDmBlobShmSize**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>This attribute is relevant only to local domain access point entries. It specifies the amount of shared memory allocated to storing binary large object log information specific to ositp or topend.</td>
</tr>
</tbody>
</table>
tuxDmOsitpTable

This group defines the OSI TP protocol-related configuration information for a specific local or remote domain access point.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmOsidmAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.130.1.1.10</td>
</tr>
<tr>
<td>tuxDmOsidmState</td>
<td>.1.3.6.1.4.1.140.300.130.1.1.20</td>
</tr>
<tr>
<td>tuxDmOsidmApt</td>
<td>.1.3.6.1.4.1.140.300.130.1.1.30</td>
</tr>
<tr>
<td>tuxDmOsidmAeq</td>
<td>.1.3.6.1.4.1.140.300.130.1.1.40</td>
</tr>
<tr>
<td>tuxDmOsidmNwDevice</td>
<td>.1.3.6.1.4.1.140.300.130.1.1.50</td>
</tr>
<tr>
<td>tuxDmOsidmApid</td>
<td>.1.3.6.1.4.1.140.300.130.1.1.60</td>
</tr>
<tr>
<td>tuxDmOsidmAeid</td>
<td>.1.3.6.1.4.1.140.300.130.1.1.70</td>
</tr>
<tr>
<td>tuxDmOsidmUrch</td>
<td>.1.3.6.1.4.1.140.300.130.1.1.80</td>
</tr>
<tr>
<td>tuxDmOsidmMaxListeningEp</td>
<td>.1.3.6.1.4.1.140.300.130.1.1.90</td>
</tr>
<tr>
<td>tuxDmOsidmXatmiEncoding</td>
<td>.1.3.6.1.4.1.140.300.130.1.1.100</td>
</tr>
</tbody>
</table>

**tuxDmOsidmAccessPoint**

Syntax: DisplayString (SIZE(0..30))

Access: read-only

Description: The domain access point name for which this entry provides the protocol-specific configuration information. This field matches the domain access point name given in the tuxDmLocal or tuxDmRemote entry that defines the protocol-independent configuration of the domain access point.

**tuxDmOsidmState**

Syntax: INTEGER { valid(1) | invalid(2) }
**tuxDmOsiDmApt**

**Syntax**  
DisplayString (SIZE(1..78))

**Access**  
read-write

**Description**  
The application process title of the domain access point in object identifier form.

**tuxDmOsiDmAeq**

**Syntax**  
DisplayString (SIZE(1..78))

**Access**  
read-write

**Description**  
The application entity qualifier of the domain access point in integer form.

**tuxDmOsiDmNwDevice**

**Syntax**  
DisplayString (SIZE(1..78))

**Access**  
read-write

**Description**  
This attribute, which specifies the network device to be used, is relevant when it defines a local domain access point and ignored for a remote domain access point.

**tuxDmOsiDmAcn**

**Syntax**  
INTEGER { atmi(1) | udt(2) }

**Access**  
read-write

**Description**  
This object denotes the current state of the tuxDmOsitp instance.

GET requests:

valid: The object exists.

SET requests:

invalid: The object is deleted.
3 Domains MIB

**tuxDmOsiDmApid**

Syntax: INTEGER  
Access: read-write  
Description: This optional attribute defines the application process-invocation identifier to be used on this domain access point.

**tuxDmOsiDmAeid**

Syntax: INTEGER  
Access: read-write  
Description: This optional attribute defines the application entity-invocation identifier to be used on this domain access point.

**tuxDmOsiDmUrch**

Syntax: DisplayString (SIZE(0..30))  
Access: read-write  
Description: This attribute specifies the user portion of the OSI TP recovery context handle. It can be required by an OSI TP provider in order to perform recovery of distributed transactions after a communication line or system failure. This attribute is relevant for defining a local domain access point and ignored for a remote domain access point.

**tuxDmOsiDmMaxListeningEp**

Syntax: INTEGER (1..32767)
<table>
<thead>
<tr>
<th><strong>BEA SNMP Agent MIB Reference</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Access</strong></td>
</tr>
<tr>
<td><strong>Description</strong></td>
</tr>
</tbody>
</table>

**tuxDmOsiDmXatmiEncoding**

| **Syntax** | INTEGER { cae(1) | preliminary(2) | oltp-tm2200(3) } |
| **Access** | read-write |
| **Description** | This attribute specifies the version of the XATMI protocol used to communicate with a remote system. Valid values are: cae, preliminary, and oltp-tm2200. This attribute is relevant for remote domain access points and ignored for local domain access points. |
tuxDmPasswordTable

This group represents configuration information for inter-domain authentication through access points of type t.domain.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmPasswdDmlAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.140.1.1.10</td>
</tr>
<tr>
<td>tuxDmPasswdDmrAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.140.1.1.20</td>
</tr>
<tr>
<td>tuxDmPasswdDmlPWD</td>
<td>.1.3.6.1.4.1.140.300.140.1.1.30</td>
</tr>
<tr>
<td>tuxDmPasswdDmrPWD</td>
<td>.1.3.6.1.4.1.140.300.140.1.1.40</td>
</tr>
<tr>
<td>tuxDmPasswdState</td>
<td>.1.3.6.1.4.1.140.300.140.1.1.50</td>
</tr>
</tbody>
</table>

**tuxDmPasswdDmlAccessPoint**

**Syntax**  DisplayString (SIZE(1..24))

**Access**  read-only

**Description**  The name of the local domain access point to which the password applies.

**tuxDmPasswdDmrAccessPoint**

**Syntax**  DisplayString (SIZE(1..24))

**Access**  read-only

**Description**  The name of the remote domain access point to which the password applies.

**tuxDmPasswdDmlPWD**

**Syntax**  DisplayString (SIZE(1..30))

**Access**  write-only

**Description**  The local password used to authenticate connections between the local domain access point identified by tuxDmPasswdDmlAccessPoint and the remote domain access point identified by tuxDmPasswdDmrAccessPoint.
tuxDmPasswdDmrPWD

**Syntax**
DisplayString (SIZE(1..30))

**Access**
write-only

**Description**
The remote password used to authenticate connections between the local domain
access point identified by `tuxDmPasswdDmlAccessPoint` and the remote domain
access point identified by `tuxDmPasswdDmrAccessPoint`.

---

tuxDmPasswdState

**Syntax**
INTEGER { valid(1) | invalid(2) | recrypt(3) }

**Access**
read-write

**Description**
This object denotes the current state of the tuxDmPassword instance.

**GET requests:**
valid: The object exists.

**SET requests:**
invalid: The object is deleted.
recrypt: Re-encrypt all passwords using a new encryption key.
3  Domains MIB

**tuxDmPrincipalMapTable**

This group represents configuration information for mapping principal names to and from external principal names across access point of type `snax`.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmPrinMapDmlAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.150.1.1.10</td>
</tr>
<tr>
<td>tuxDmPrinMapDmrAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.150.1.1.20</td>
</tr>
<tr>
<td>tuxDmPrinMapDmlPrinName</td>
<td>.1.3.6.1.4.1.140.300.150.1.1.30</td>
</tr>
<tr>
<td>tuxDmPrinMapDmrPrinName</td>
<td>.1.3.6.1.4.1.140.300.150.1.1.40</td>
</tr>
<tr>
<td>tuxDmPrinMapDirection</td>
<td>.1.3.6.1.4.1.140.300.150.1.1.50</td>
</tr>
<tr>
<td>tuxDmPrinMapState</td>
<td>.1.3.6.1.4.1.140.300.150.1.1.60</td>
</tr>
</tbody>
</table>

**tuxDmPrinMapDmlAccessPoint**

- Syntax: `DisplayString (SIZE(1..12))`
- Access: read-only
- Description: The local domain access point to which the principal mapping applies.

**tuxDmPrinMapDmrAccessPoint**

- Syntax: `DisplayString (SIZE(1..12))`
- Access: read-only
- Description: The remote domain access point to which the principal mapping applies.

**tuxDmPrinMapDmlPrinName**

- Syntax: `DisplayString (SIZE(1..12))`
- Access: read-only
- Description: The local principal name in the principal mapping.
**tuxDmPrinMapDmrPrinName**

**Syntax**
DisplayString (SIZE(1..12))

**Access**
read-only

**Description**
The remote principal name in the principal mapping.

**tuxDmPrinMapDirection**

**Syntax**
INTEGER { in(1) | out(2) | both(3) }

**Access**
read-write

**Description**
The direction to which the principal mapping applies.

*in:*
Is incoming to this BEA Tuxedo domain through the given remote domain access point and local domain access point.

*out:*
Is outgoing from this BEA Tuxedo domain through the given local domain access point and remote domain access point.

*both:*
Applies to both incoming to and outgoing from this BEA Tuxedo domain through the given local domain access point and remote domain access point.

**tuxDmPrinMapState**

**Syntax**
INTEGER { valid(1) | invalid(2) }

**Access**
read-write

**Description**
This object denotes the current state of the tuxDmPrincipalMap instance.

**GET requests:**
valid: The object exists.

**SET requests:**
invalid: The object is deleted.
3 Domains MIB

**tuxDmRemoteTable**

The tuxDmRemote class represents remote domain access point configuration information. Local resources that can be exported through one or more local domain access points are made accessible to a remote domain through a remote domain access point. Similarly, remote resources are imported from a remote domain through a remote domain access point.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmRemoteDmAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.160.1.1.10</td>
</tr>
<tr>
<td>tuxDmRemoteDmAccessPointID</td>
<td>.1.3.6.1.4.1.140.300.160.1.1.20</td>
</tr>
<tr>
<td>tuxDmRemoteType</td>
<td>.1.3.6.1.4.1.140.300.160.1.1.30</td>
</tr>
<tr>
<td>tuxDmRemoteState</td>
<td>.1.3.6.1.4.1.140.300.160.1.1.40</td>
</tr>
<tr>
<td>tuxDmRemoteDmCodePage</td>
<td>.1.3.6.1.4.1.140.300.160.1.1.50</td>
</tr>
<tr>
<td>tuxDmRemoteDmMachineType</td>
<td>.1.3.6.1.4.1.140.300.160.1.1.90</td>
</tr>
</tbody>
</table>

**tuxDmRemoteDmAccessPoint**

- **Syntax**: DisplayString {SIZE(1..30)}
- **Access**: read-only
- **Description**: The name of this tuxDmRemote entry. This is an identifier unique within the scope of tuxDmLocal and tuxDmRemote entry names in the domain configuration.

**tuxDmRemoteDmAccessPointID**

- **Syntax**: DisplayString {SIZE(1..30)}
- **Access**: read-write
- **Description**: The access point identifier. This identifier is unique across all local and remote domain access points.
**tuxDmRemoteType**

**Syntax**
INTEGER { tdomain(1) | ositp(2) | snax(3) | topend(4)}

**Access**
read-write

**Description**
The type of domain:

- tdomain:
  A BEA Tuxedo system domain.
- ositp:
  An OSI domain.
- snax:
  An SNA domain.
- topend:
  A BEA TOP END domain.

The presence or absence of other attributes depends on the value of this attribute.

**tuxDmRemoteState**

**Syntax**
INTEGER { valid(1) | invalid(2)}

**Access**
read-write

**Description**
This object denotes the current state of the tuxDmRemote instance.

GET requests:
- valid: The object exists.

SET requests:
- invalid: The object is deleted.

**tuxDmRemoteDmCodePage**

**Syntax**
DisplayString (SIZE(1..20))

**Access**
read-write
### 3 Domains MIB

<table>
<thead>
<tr>
<th>Description</th>
<th>Attributes available when <code>tuxDmRemoteDmType=snax</code>. The name of the default translation tables to use when translating requests and replies that are sent through this access point.</th>
</tr>
</thead>
</table>

#### tuxDmRemoteDmMachineType

**Syntax**

DisplayString (SIZE(0..15))

**Access**

read-write

**Description**

Attributes available when `tuxDmRemoteDmType=tdomain`.

These attributes are used for grouping domains so that encoding/decoding of messages between domains can be bypassed. If it is not specified, the default is to turn encoding/decoding on. If the value set for this field is the same in both the **DM_LOCAL** and the **DM_REMOTE** sections of a domain configuration file, data encoding/decoding is bypassed. The value set for this variable can be any string value up to 15 characters in length. It is used only for comparison.
tuxDmResourcesTable

This group represents Domains-specific configuration information.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmResourcesDmVersion</td>
<td>.1.3.6.1.4.1.140.300.170.1.1.10</td>
</tr>
</tbody>
</table>

**tuxDmResourcesDmVersion**

- **Syntax**: DisplayString (SIZE(1..30))
- **Access**: read-only
- **Description**: A user-supplied identifier for the Domains configuration.
tuxDmRoutingTable

The tuxDmRoutingTable class represents routing criteria information for routing requests to a domain through a remote domain access point.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmRoutingDmRoutingName</td>
<td>.1.3.6.1.4.1.140.300.180.1.1.10</td>
</tr>
<tr>
<td>tuxDmRoutingDmBufType</td>
<td>.1.3.6.1.4.1.140.300.180.1.1.20</td>
</tr>
<tr>
<td>tuxDmRoutingDmField</td>
<td>.1.3.6.1.4.1.140.300.180.1.1.30</td>
</tr>
<tr>
<td>tuxDmRoutingDmFieldType</td>
<td>.1.3.6.1.4.1.140.300.180.1.1.40</td>
</tr>
<tr>
<td>tuxDmRoutingDmRanges</td>
<td>.1.3.6.1.4.1.140.300.180.1.1.50</td>
</tr>
<tr>
<td>tuxDmRoutingDmState</td>
<td>.1.3.6.1.4.1.140.300.180.1.1.60</td>
</tr>
</tbody>
</table>

**tuxDmRoutingDmRoutingName**

- **Syntax**: DisplayString (SIZE(1..15))
- **Access**: read-only
- **Description**: The name of the routing criteria table entry.

**tuxDmRoutingDmBufType**

- **Syntax**: DisplayString (SIZE(1..30))
- **Access**: read-only
- **Description**: List of types and subtypes of data buffers for which this routing entry is valid. A maximum of 32 type/subtype combinations is allowed. The types are restricted to the following: FML, XML, VIEW, X-C-TYPE, or X_COMMON. No subtype can be specified for FML or XML; subtypes are required for types VIEW, X_C_TYPE, and X_COMMON ("*" is not allowed).
Note that subtype names should not contain semicolon (;), colon (:), comma (,), or asterisk (*) characters. Duplicate type/subtype pairs cannot be specified for the same routing criterion name; more than one routing entry can have the same criterion name as long as the type/subtype pairs are unique. If multiple buffer types are specified for a single routing entry, the data types of the routing field for each buffer type must be the same.

**tuxDmRoutingDmField**

**Syntax**
DisplayString (SIZE(1..30))

**Access**
read-write

**Description**
This attribute is the routing field name. This field is assumed to be an FML buffer, XML buffer, or VIEW field name that is identified in an FML field table (using the FLDTBLDIR and FIELDTBLS environment variables), or an FML VIEW table (using the VIEWDIR and VIEWFILES environment variables), respectively. This information is used to get the associated field value for data-dependent routing to an access point of a remote domain.

For an XML buffer type, this field contains either a routing element type (or name) or a routing element attribute name.

The syntax of this variable for an XML buffer type is as follows:

```xml
```

The element is assumed to be an XML document or datagram element type. Indexing is not supported. Therefore, the BEA Tuxedo system recognizes only the first occurrence of a given element type when it processes an XML buffer for data-dependent routing. This information is used to get the associated element content for data-dependent routing while sending a message. The content must be a string encoded in UTF-8.

The attribute is assumed to be an XML document or datagram attribute of the defined element. This information is used to get the associated attribute value for data-dependent routing while sending a message. The value must be a string encoded in UTF-8.

The combination of element name and attribute name can contain up to 30 characters.
### tuxDmRoutingDmFieldType

**Syntax**  
INTEGER { char(1) | short(2) | long(3) | float(4) | double(5) | string (6) }

**Access**  
read-write

**Description**  
The type can be char, short, long, float, double, or string. Only one type is allowed. This attribute is used only for routing XML buffers.

### tuxDmRoutingDmRanges

**Syntax**  
DisplayString (SIZE(1..1000))

**Access**  
read-write

**Description**  
This attribute includes the ranges and associated server groups for the tuxDmRoutingFieldDmType routing field. The format of the string is a comma-separated, ordered list of range/group name pairs. A range/group pair has the following format:

lower[-upper]:raccesspoint

where lower and upper are assigned numeric values or character strings in single quotes. lower must be less than or equal to upper. To embed a single quote in a character string value, the quote must be preceded by two backslashes (for example, ‘O\’Brien’). The value MIN can be used to indicate the minimum value for the data type of the associated field on the machine. The value MAX can be used to indicate the maximum value for the data type of the associated field on the machine. Thus, “MIN-5” is all numbers less than or equal to -5, and “6-MAX” is all numbers greater than or equal to 6.

The meta-character “*” (wild-card) in the position of a range indicates any values not covered by the other ranges previously seen in the entry. Only one wild-card range is allowed per entry and it should be the last range (ranges that follow it are ignored).

The routing field can be of any data type supported in FML. A numeric routing field must have numeric range values and a string routing field must have string range values.

String range values for string, carray, and character field types must be placed inside a pair of single quotes, and cannot be preceded by a sign. The short and long integer values are a string of digits, optionally preceded by a plus or minus sign.
Floating point numbers are of the form accepted by the C compiler or `atof(3)`: an optional sign, then a string of digits (that optionally contains a decimal point), then an optional e or E followed by an optional sign or space, followed by an integer.

The `raccesspoint` parameter indicates the remote domain access point to which the request is routed if the field matches the range. A `raccesspoint` of “*” indicates that the request can go to any remote domain access point that imports the desired service.

**tuxDmRoutingState**

| Syntax       | INTEGER { valid(1) | invalid(2)} |
|--------------|-----------------|
| Access       | read-write      |
| Description  | This object denotes the current state of the tuxDmRouting instance. |
| GET requests | valid: The object exists. |
|              | invalid: The object is deleted. |
| SET requests | invalid: The object is deleted. |
## tuxDmrPrincipalTable

This group represents password configuration information for remote principal names.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmrPrincipalDmrAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.190.1.1.10</td>
</tr>
<tr>
<td>tuxDmrPrincipalDmrPrinName</td>
<td>.1.3.6.1.4.1.140.300.190.1.1.20</td>
</tr>
<tr>
<td>tuxDmrPrincipalDmrPrinPasswd</td>
<td>.1.3.6.1.4.1.140.300.190.1.1.30</td>
</tr>
<tr>
<td>tuxDmrPrincipalState</td>
<td>.1.3.6.1.4.1.140.300.190.1.1.40</td>
</tr>
</tbody>
</table>

### tuxDmrPrincipalDmrAccessPoint

- **Syntax**: DisplayString (SIZE(1..24))
- **Access**: read-only
- **Description**: The remote domain access point to which the principal is applicable.

### tuxDmrPrincipalDmrPrinName

- **Syntax**: DisplayString (SIZE(1..24))
- **Access**: read-only
- **Description**: The remote principal name.

### tuxDmrPrincipalDmrPrinPasswd

- **Syntax**: DisplayString (SIZE(0..8))
- **Access**: write-only
- **Description**: The remote password used for the principal name when communicating through the remote domain access point identified in `tuxDmrPrincipalDmrAccessPoint`.
**tuxDmrPrincipalState**

**Syntax**  INTEGER { valid(1) | invalid(2)}

**Access**  read-write

**Description**  This object denotes the current state of the tuxDmrPrincipal instance.

- **GET requests:**
  - valid: The object exists.

- **SET requests:**
  - invalid: The object is deleted.


3 Domains MIB

**tuxDmSnaCRMTable**

This group defines the SNM-CRM-specific configuration information for the named local domain access point.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmSnaCRMDmSNACRM</td>
<td>.1.3.6.1.4.1.140.300.200.1.1.10</td>
</tr>
<tr>
<td>tuxDmSnaCRMDmAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.200.1.1.20</td>
</tr>
<tr>
<td>tuxDmSnaCRMState</td>
<td>.1.3.6.1.4.1.140.300.200.1.1.30</td>
</tr>
<tr>
<td>tuxDmSnaCRMDmNWAddr</td>
<td>.1.3.6.1.4.1.140.300.200.1.1.40</td>
</tr>
<tr>
<td>tuxDmSnaCRMDmNWDevice</td>
<td>.1.3.6.1.4.1.140.300.200.1.1.50</td>
</tr>
</tbody>
</table>

**tuxDMSnaCRMDmSNACRM**

Syntax  
DisplayString (SIZE(1..30))

Access  
read-only

Description  
This is an identifier, unique within the scope of the SNA CRM entries in the domain configuration, used to identify this SNA CRM entry.

**tuxDMSnaCRMDmAccessPoint**

Syntax  
DisplayString (SIZE(1..30))

Access  
write-only

Description  
The name of the local domain access point entry with which this SNA CRM is used.

**tuxDMSnaCRMState**

Syntax  
INTEGER { valid(1) | invalid(2)}

Access  
read-write

Description  
This object denotes the current state of the tuxDmSnaCRM instance.
GET requests:
  valid: The object exists.

SET requests:
  invalid: The object is deleted.

**tuxDMSnaCRMDmNWAddr**

**Syntax**
DisplayString (SIZE(1..78))

**Access**
read-write

**Description**
Specifies the network address for communication between the domain gateway of the local domain access point and the SNA CRM.

**tuxDMSnaCRMDmNWDevice**

**Syntax**
DisplayString (SIZE(1..78))

**Access**
read-write

**Description**
Specifies the network device to be used for communication between the domain gateway of the local domain access point and the SNA CRM.
## tuxDmSnaLinkTable

This group represents snx-specific configuration information for a remote domain access point.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmSnaLinkDmSNALink</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.10</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmSNAStack</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.20</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmrAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.30</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmlSysID</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.40</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmrSysID</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.50</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmlUname</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.60</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmMinWin</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.70</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmModeName</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.80</td>
</tr>
<tr>
<td>tuxDmSnaLinkState</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.90</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmSecType</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.100</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmStartType</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.110</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmMaxSNASess</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.120</td>
</tr>
<tr>
<td>tuxDmSnaLinkDmMaxSyncLvl</td>
<td>.1.3.6.1.4.1.140.300.210.1.1.130</td>
</tr>
</tbody>
</table>

### tuxDmSnaLinkDmSNALink

**Syntax**
DisplayString (SIZE(1..30))

**Access**
read-only

**Description**
This is an identifier, unique within the scope of the SNA LINK entries within the domain configuration, used to identify rows in this table.
**tuxDmSnaLinkDmSNAStack**

**Syntax**  DisplayString (SIZE(1..30))  
**Access**  read-write  
**Description**  The name of the snax stack entry to be used to reach this remote domain access point.

**tuxDmSnaLinkDmrAccessPoint**

**Syntax**  DisplayString (SIZE(1..30))  
**Access**  read-write  
**Description**  Identifies the remote domain access point name for which this entry provides the snax configuration data.

**tuxDmSnaLinkDmlSysID**

**Syntax**  DisplayString (SIZE(1..4))  
**Access**  read-write  
**Description**  The local SYSID used when establishing an SNA link to the remote logical unit (LU).

**tuxDmSnaLinkDmrSysID**

**Syntax**  DisplayString (SIZE(1..4))  
**Access**  read-write  
**Description**  The remote SYSID used when establishing an SNA link to the remote logical unit (LU).

**tuxDmSnaLinkDmlUname**

**Syntax**  DisplayString (SIZE(1..8))  
**Access**  read-write  
**Description**  Specifies the logical unit (LU) name associated with the remote domain access point.
3 Domains MIB

**tuxDmSnaLinkDmMinWin**

Syntax: INTEGER

Access: read-write

Description: The minimum number of winner sessions to the remote LU.

**tuxDmSnaLinkDmModeName**

Syntax: DisplayString (SIZE(1..8))

Access: read-write

Description: Specifies the name associated with the session characteristics for sessions to the remote LU.

**tuxDmSnaLinkDmState**

Syntax: INTEGER { valid(1) | invalid(2)}

Access: read-write

Description: This object denotes the current state of the tuxDmSnaLink instance.

GET requests:
- valid: The object exists.

SET requests:
- invalid: The object is deleted.

**tuxDmSnaLinkDmSecType**

Syntax: INTEGER { local(1) | identify(2) | verify(3) | persistent(4) | mixidpe(5) }

Access: read-write

Description: Specifies the type of SNA security to be used on sessions to the remote logical unit. Valid values for this attribute are local, identify, verify, persistent, and mixidpe.
tuxDmSnaLinkDmStartType
Syntax   INTEGER { auto(1) | cold(2) }
Access   read-write
Description  Specifies the type of session start-up for the destination logical unit (LU).
cold
    Forces a COLDSTART with the LU.
auto
    The SNACRM, in conjunction with the domain gateway, chooses whether to COLDSTART or WARMSTART the LU.

---
tuxDmSnaLinkDmSNAsess
Syntax   INTEGER (0..32767)
Access   read-write
Description  Specifies maximum number of sessions to establish with the remote LU.

---
tuxDmSnaLinkDmMaxSyncLvl
Syntax   INTEGER (0..2)
Access   read-only
Description  The maximum SYNC LEVEL that can be support to this remote LU.
3 Domains MIB

**tuxDmSnaStackTable**

This group defines an SNA stack to be used by a specific SNA CRM.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmSnaStackDmSnaStack</td>
<td>1.3.6.1.4.1.140.300.220.1.1.10</td>
</tr>
<tr>
<td>tuxDmSnaStackDmSnaCRM</td>
<td>1.3.6.1.4.1.140.300.220.1.1.20</td>
</tr>
<tr>
<td>tuxDmSnaStackDmStackType</td>
<td>1.3.6.1.4.1.140.300.220.1.1.30</td>
</tr>
<tr>
<td>tuxDmSnaStackDmlUname</td>
<td>1.3.6.1.4.1.140.300.220.1.1.40</td>
</tr>
<tr>
<td>tuxDmSnaStackDmTpName</td>
<td>1.3.6.1.4.1.140.300.220.1.1.50</td>
</tr>
<tr>
<td>tuxDmSnaStackDmStackParams</td>
<td>1.3.6.1.4.1.140.300.220.1.1.60</td>
</tr>
<tr>
<td>tuxDmSnaStackState</td>
<td>1.3.6.1.4.1.140.300.220.1.1.70</td>
</tr>
</tbody>
</table>

**tuxDmSnaStackDmSnaStack**

Syntax: DisplayString (SIZE(1..30))

Access: read-only

Description: The name of this tuxDmSnaStack entry. This is an identifier, unique within the scope of the tuxDmSnaStack table in the domain configuration.

**tuxDmSnaStackDmSnaCRM**

Syntax: DisplayString (SIZE(1..30))

Access: read-write

Description: Identifies the tuxDmSnaCRM table entry of the SNA CRM in which this SNA protocol stack definition is used.

**tuxDmSnaStackDmStackType**

Syntax: DisplayString (SIZE(1..30))
**tuxDmSnaStackDmlUname**

- **Syntax**: `DisplayString (SIZE(1..8))`
- **Access**: read-write
- **Description**: Specifies the LU name to be used on sessions established using this stack definition.

**tuxDmSnaStackDmTpName**

- **Syntax**: `DisplayString (SIZE(1..8))`
- **Access**: read-write
- **Description**: Specifies the TP name associated with the SNA stack. A value of “*” means accept any TP name.

**tuxDmSnaStackDmStackParams**

- **Syntax**: `DisplayString (SIZE(1..128))`
- **Access**: read-write
- **Description**: Provides protocol stack specific parameters.

**tuxDmSnaStackState**

- **Syntax**: `INTEGER { valid(1) | invalid(2)}`
- **Access**: read-write
- **Description**: This object denotes the current state of the tuxDmSnaStack instance.
  
  **GET requests:**
  
  - **valid**: The object exists.
  
  **SET requests:**
  
  - **invalid**: The object is deleted.
tuxDmTdomainTable

This group defines the domain-specific configuration for a local or remote domain access point.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmTdomainDmAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.240.1.1.10</td>
</tr>
<tr>
<td>tuxDmTdomainDmNwAddr</td>
<td>.1.3.6.1.4.1.140.300.240.1.1.20</td>
</tr>
<tr>
<td>tuxDmTdomainState</td>
<td>.1.3.6.1.4.1.140.300.240.1.1.30</td>
</tr>
<tr>
<td>tuxDmTdomainDmNwDevice</td>
<td>.1.3.6.1.4.1.140.300.240.1.1.40</td>
</tr>
<tr>
<td>tuxDmTdomainDmCmpLimit</td>
<td>.1.3.6.1.4.1.140.300.240.1.1.50</td>
</tr>
<tr>
<td>tuxDmTdomainDmFailOverSeq</td>
<td>.1.3.6.1.4.1.140.300.240.1.1.60</td>
</tr>
<tr>
<td>tuxDmTdomainDmMinEncryptBits</td>
<td>.1.3.6.1.4.1.140.300.240.1.1.70</td>
</tr>
<tr>
<td>tuxDmTdomainDmMaxEncryptBits</td>
<td>.1.3.6.1.4.1.140.300.240.1.1.80</td>
</tr>
</tbody>
</table>

**tuxDmTdomainDmAccessPoint**

Syntax  
DisplayString (SIZE(1..24))

Access  
read-only

Description  
The local or remote domain access point name for which this entry provides the TDomain-specific configuration data.

When domain-level failover is in use, more than one tuxDmTdomain Table entry can be defined with the same tuxTDmTdomainDmAccessPoint.

**tuxDmTdomainDmNwAddr**

Syntax  
DisplayString (SIZE(1..24))

Access  
read-only

Description  
Specifies the network address associated with the access point.
For a local domain access point, this attribute supplies the address used to listen for incoming connections.

For a remote domain access point, this attribute supplies the destination used when you connect to a remote domain access point.

The value of this field must be unique across all tuxDmTdomainTable entries.

### tuxDmTdomainState

**Syntax**

INTEGER { valid(1) | invalid(2) }

**Access**

read-write

**Description**

This object denotes the current state of the tuxDmTdomain instance.

GET requests:

valid: The object exists.

SET requests:

invalid: Delete the object.

### tuxDmTdomainDmNwDevice

**Syntax**

DisplayString (SIZE(1..78))

**Access**

read-write

**Description**

Specifies the network device used.

For a local domain access point, this attribute specifies the device used for listening.

For a remote domain access point, this attribute specifies the device used to connect to the remote domain access point.

### tuxDmTdomainDmCmpLimit

**Syntax**

INTEGER

**Access**

read-write

**Description**

This attribute is relevant to remote domain access points only. It specifies the threshold over which compression occurs for traffic on connections to this access point.
tuxDmTdomainDmFailOverSeq

Syntax
INTEGER (0..32767)

Access
read-write

Description
This attribute is relevant to remote domain access points only. It specifies the position of this set of addressing in the failover sequence for this remote domain access point. If no failover sequence number is supplied, the first entry for this remote domain access point is allocated the number 10 greater than the highest failover sequence number known for the remote domain access point. Thus, the first entry gets 10, the second, 20, and so on.

tuxDmTdomainDmMinEncryptBits

Syntax
INTEGER { enc-0-bit(1) | enc-40-bits(2) | enc-56-bits (3) | enc-128-bits (4) }

Access
read-write

Description
This attribute is relevant to remote domain access points only. When establishing a network link to this access point, this attribute specifies the minimum level of encryption required.

enc-0-bit (1): No encryption
enc-40-bits (2), enc-56-bits (3), and enc-128-bits (4): These specify the encryption length (in bits).

If this minimum level of encryption is not met, link establishment fails. The default value is enc-0-bit.

tuxDmTdomainDmMaxEncryptBits

Syntax
INTEGER { enc-0-bit(1) | enc-40-bits(2) | enc-56-bits (3) | enc-128-bits (4) }

Access
read-write

Description
This attribute is relevant to remote domain access points only. When establishing a network link to this access point, this attribute specifies the maximum level of encryption required.

enc-0-bit (1): No encryption
enc-40-bits (2), enc-56-bits (3), and enc-128-bits (4): These specify the encryption length (in bits).

The default value is enc-128-bits.

**Note:** Modifications to this attribute do not affect established connections.
3 Domains MIB

**tuxDmTopendTable**

This group defines the configuration for a local or remote domain access point specific to a BEA TOP END system.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmTopendDmAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.250.1.1.10</td>
</tr>
<tr>
<td>tuxDmTopendDmNwAddr</td>
<td>.1.3.6.1.4.1.140.300.250.1.1.20</td>
</tr>
<tr>
<td>tuxDmTopendDmtetpSystem</td>
<td>.1.3.6.1.4.1.140.300.250.1.1.30</td>
</tr>
<tr>
<td>tuxDmTopendState</td>
<td>.1.3.6.1.4.1.140.300.250.1.1.40</td>
</tr>
<tr>
<td>tuxDmTopendDmNwDevice</td>
<td>.1.3.6.1.4.1.140.300.250.1.1.50</td>
</tr>
<tr>
<td>tuxDmTopendDmtePwd</td>
<td>.1.3.6.1.4.1.140.300.250.1.1.60</td>
</tr>
<tr>
<td>tuxDmTopendDmFailoverSeq</td>
<td>.1.3.6.1.4.1.140.300.250.1.1.70</td>
</tr>
</tbody>
</table>

**tuxDmTopendDmAccessPoint**

**Syntax**  
DisplayString (SIZE(1..24))

**Access**  
read-only

**Description**  
Specifies the local or remote domain access point name for which this entry provides the BEA TOP END system-specific configuration data.

**tuxDmTopendDmNwAddr**

**Syntax**  
DisplayString (SIZE(1..24))

**Access**  
read-only

**Description**  
Specifies the network address associated with the local or remote domain access point.

**tuxDmTopendDmtetpSystem**

**Syntax**  
DisplayString (SIZE(1..8))

3-58  
BEA SNMP Agent MIB Reference
Access          read-write
Description  Specifies the name of the BEA TOP END system.

**Note:** All remote domain access points accessible through a local domain access point must have the same BEA TOP END system name.

**tuxDmTopendState**

Syntax       INTEGER { valid(1) | invalid(2) | recrypt(3) }
Access        read-write
Description   This object denotes the current state of the tuxDmTopend instance.

GET requests:
valid: The object exists.

SET requests:
invalid: Delete the object.
recrypt: Re-encrypt all passwords that use a new encryption key.

**tuxDmTopendDmNwDevice**

Syntax       DisplayString (SIZE(1..78))
Access        read-write
Description   Specifies the network device associated with the local or remote domain access point.

**tuxDmTopendDmtePwd**

Syntax       DisplayString (SIZE(1..12))
Access        read-write
Description   Specifies the password to use when sending messages to the BEA TOP END system. This attribute is relevant only to local domain access point entries.
3 Domains MIB

tuxDmTopendDmFailoverSeq

Syntax  INTEGER (0..32767)
Access   read-write
Description  This attribute is relevant to remote domain access points only. It specifies the position of this set of addressing in the failover sequence for this remote domain access point. If no failover sequence number is supplied, the first entry for this remote domain access point is allocated the number 10 greater than the highest failover sequence number known for the remote domain access point. Thus, the first entry gets 10, the second, 20, and so on.

The domain gateway uses the tuxDmTopend addressing entries for a particular remote domain access point strictly in the order of its failover sequence numbers—lowest to highest.
**tuxDmTransactionTable**

This group represents information about transactions that span domains. This object can be used to find out what remote domain access points are involved in the transaction, the parent domain access point, the transaction state, and other information.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmTransactionDmIAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.10</td>
</tr>
<tr>
<td>tuxDmTransactionDmTpTranID</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.20</td>
</tr>
<tr>
<td>tuxDmTransactionState</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.30</td>
</tr>
<tr>
<td>tuxDmTransactionDmTxAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.40</td>
</tr>
<tr>
<td>tuxDmTransactionDmTxNetTranID</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.50</td>
</tr>
<tr>
<td>tuxDmTransactionDmBranchCount</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.60</td>
</tr>
<tr>
<td>tuxDmTransactionDmBranchIndex</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.70</td>
</tr>
<tr>
<td>tuxDmTransactionDmBranchNo</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.80</td>
</tr>
<tr>
<td>tuxDmTransactionDmRAccessPoint</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.90</td>
</tr>
<tr>
<td>tuxDmTransactionDmNetTranID</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.100</td>
</tr>
<tr>
<td>tuxDmTransactionDmBranchState</td>
<td>.1.3.6.1.4.1.140.300.260.1.1.110</td>
</tr>
</tbody>
</table>

**tuxDmTransactionDmIAccessPoint**

*Syntax*  
DisplayString (SIZE(1..30))

*Access*  
read-only

*Description*  
The name of the local domain access point with which the transaction is associated. This is a required field for GET operations. For SET operations, this variable must be specified.
3 Domains MIB

tuxDmTransactionDmTPTranID

Syntax: DisplayString (SIZE(1..24))
Access: read-write
Description: The transaction identifier returned from tpsuspend(3c) mapped to a string representation. The data in this field should not be interpreted directly by the user, except for equality comparison. For SET operations, this variable must be specified.

tuxDmTransactionState

Syntax: INTEGER { aborted(1) | abortonly(2) | active(3) | comcalled(4) | decided(5) | done(6) | habort(7) | hcommit(8) | heuristic(9) | ready(10) | unknown(11) | invalid(12) }
Access: read-write
Description: This object denotes the current state of the tuxDmTransaction instance.

GET requests:
- aborted(1): The transaction is being rolled back.
- abortonly(2): The transaction has been identified for rollback.
- active(3): The transaction is active.
- comcalled(4): The transaction has initiated the first phase of commitment.
- decided(5): The transaction has initiated the second phase of commitment.
- done(6): The transaction has completed the second phase of commitment.
- habort(7): The transaction has been heuristically rolled back.
- hcommit(8): The transaction has been heuristically committed.
- heuristic(9): The transaction commitment or rollback has completed heuristically.
- ready(10): The transaction has completed the first phase of a two-phase commit. All the participating groups and remote domains have completed the first phase of commitment and are ready to be committed.
- unknown(11): It was not possible to determine the state of the transaction.
SET requests:
invalid: Forget the specified table entry. This state change is only valid in states HCommit and HAbort.

**tuxDmTransactionDmTxAccessPoint**

- **Syntax**: DisplayString (SIZE(1..30))
- **Access**: read-only
- **Description**: If the transaction originated from a remote domain, this is the name of the remote domain access point through which it originated. If the transaction originated within this domain, this is the name of the local domain access point.

**tuxDmTransactionDmTxNetTranID**

- **Syntax**: DisplayString (SIZE(1..78))
- **Access**: read-only
- **Description**: If the transaction originated from a remote domain, this is the external transaction identifier received from the remote domain access point through which the transaction originated. If the transaction originated within this domain, it contains the same value as the

**tuxDmTransactionDmBranchCount**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: The number of branches to remote domain access points involved in the transaction. For a domain gateway that does not make branch information available, this value is zero.

**tuxDmTransactionDmBranchIndex**

- **Syntax**: INTEGER
- **Access**: read-only
**Domains MIB**

**Description**

The index of the first branch-specific attribute values 
(tuxDmTransactionDmBranchNo, tuxDmTransactionDmrAccessPoint, 
tuxDmTransactionDmNetTranID, and tuxDmTransactionDmBranchState) 
corresponding to this object.

**tuxDmTransactionDmBranchNo**

**Syntax**

INTEGER

**Access**

read-only

**Description**

The branch number of the participating branch (numbered from zero).

**tuxDmTransactionDmrAccessPoint**

**Syntax**

DisplayString (SIZE(1..30))

**Access**

read-only

**Description**

The name of the remote domain access point for this branch.

**tuxDmTransactionDmNetTranID**

**Syntax**

DisplayString (SIZE(1..78))

**Access**

read-only

**Description**

The external transaction identifier used with the remote domain access point for this branch. Some types of domain gateways do not return this information; in this scenario, this attribute is set to the empty string. For example, TDomains uses the local transaction identifier in tuxDmTransactionDmTpTranID for branches to remote domain access points and sets this value to the empty string.

**tuxDmTransactionDmBranchState**

**Syntax**

INTEGER { aborted(1) | abortonly(2) | active(3) | comcalled(4) | 
decided(5) | done(6) | habort(7) | hcommit(8) | heuristic-hazard(9) | 
heuristic-mixed(10 | ready(11) | unknown(12) }

**Access**

read-write
<table>
<thead>
<tr>
<th>Description</th>
<th>A GET operation retrieves runtime information for the transaction branch (when the information is available for a particular domain gateway type).</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET requests:</td>
<td></td>
</tr>
<tr>
<td>A\BorteD(1)</td>
<td>The transaction branch is being rolled back.</td>
</tr>
<tr>
<td>A\BortonlY(2)</td>
<td>The transaction branch has been identified for rollback.</td>
</tr>
<tr>
<td>ACTive(3)</td>
<td>The transaction branch is active.</td>
</tr>
<tr>
<td>COMcalled(4)</td>
<td>The transaction branch has initiated the first phase of commitment.</td>
</tr>
<tr>
<td>DECided(5)</td>
<td>The transaction branch has initiated second phase of commitment.</td>
</tr>
<tr>
<td>DONe(6)</td>
<td>The transaction branch has completed the second phase of commitment.</td>
</tr>
<tr>
<td>H\ABort(7)</td>
<td>The transaction has been heuristically rolled back.</td>
</tr>
<tr>
<td>H\COmmit(8)</td>
<td>The transaction has been heuristically committed.</td>
</tr>
<tr>
<td>Heuristic Ha\Zard(9)</td>
<td>Communications for the transaction branch failed and it has not been determined if rollback completed successfully.</td>
</tr>
<tr>
<td>Heuristic M\I\ixed(10)</td>
<td>The transaction commitment or rollback for the transaction branch has completed and the remote domain has reported that the state of some of the resources used for the commitment or rollback is not consistent with the outcome of the transaction.</td>
</tr>
<tr>
<td>REA\dy(11)</td>
<td>The transaction has completed the first phase of a two-phase commit. All the participating groups and remote domains have completed the first phase of commitment and are ready to be committed.</td>
</tr>
<tr>
<td>UN\Known(12)</td>
<td>The state of the transaction could not be determined.</td>
</tr>
</tbody>
</table>
The MIB group `beaDomainList` represents information about the Tuxedo or WLE domain the agent is monitoring, as specified at startup.

**Note:** Row creation is not allowed in this MIB group. A minimal `tuxconfig` file must exist before you can start the agent.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>beaDomainKey</td>
<td>.1.3.6.1.4.1.140.305.1.1</td>
</tr>
<tr>
<td>beaLogicalAgentName</td>
<td>.1.3.6.1.4.1.140.305.1.2</td>
</tr>
<tr>
<td>beaDomainID</td>
<td>.1.3.6.1.4.1.140.305.1.3</td>
</tr>
<tr>
<td>beaDomainTuxdir</td>
<td>.1.3.6.1.4.1.140.305.1.4</td>
</tr>
<tr>
<td>beaDomainTuxconfig</td>
<td>.1.3.6.1.4.1.140.305.1.5</td>
</tr>
<tr>
<td>beaDomainStatus</td>
<td>.1.3.6.1.4.1.140.305.1.6</td>
</tr>
</tbody>
</table>

**beaDomainKey**

**Syntax** INTEGER (32769..262143)

**Access** read-only

**Description** Numeric key for the well-known address in a Tuxedo System/T bulletin board. In a single-processor environment, this key names the bulletin board. In a multi-processor environment, this key names the message queue of the DBBL. This key is used as the basis for deriving the names of resources other than the well-known address, such as the names for the bulletin boards throughout the application.
beaLogicalAgentName

Syntax  
DisplayString(SIZE(1..32))

Access  
read-only

Description  
The logical agent name of the agent as specified in the -l option when the agent was started (UNIX systems). On Windows NT systems, the logical agent name is the name of the Windows NT service used to start the agent. This is the agent that monitors the domain. If there are multiple SNMP agents running on a managed node, this name needs to be appended to the community string with an @ sign when sending an SNMP request to the agent. For example, if there are two logical agents simp_snmpd and bank_snmpd, the default communities used to query values from these agents would be public@simp_snmpd and public@bank_snmpd, respectively. To run multiple agents on the same managed node, they must be run as subagents (without the -s option) with the agent integrator.

beaDomainId

Syntax  
DisplayString(SIZE(1..30))

Access  
read-only

Description  
This is the BEA domain identifier of the domain being managed by this agent. This object is optional.

beaDomainTuxdir

Syntax  
DisplayString(SIZE(1..78))

Access  
read-only

Description  
The tuxdir value for the domain being managed by this agent. tuxdir is the absolute path name to the directory where the Tuxedo software is found on the master machine.
**beaDomainTuxconfig**

**Syntax**  
`DisplayString(SIZE(1..64))`

**Access**  
read-only

**Description**  
The absolute location, including file name, for the configuration file of the domain being managed by this agent.

**beaDomainStatus**

**Syntax**  
`INTEGER { active(1), inactive(2) }`

**Access**  
read-only

**Description**  
This object represents the current state of the domain being managed. The values and their interpretation are the same as for `tuxTdomainState`. 
This chapter describes five MIB tables that are specific to WLE. To access the objects in these MIB tables, the WLE version of the BEA SNMP Agent should be running on the machine where WLE application resources are accessible. Table 5-1 lists the WLE groups.

In addition to the objects in these WLE specific groups, the Tuxedo Core MIB contains the following WLE specific objects:

- wleMaxObjects
- wleMaxInterfaces
- wleCurInterfaces
- wleHwInterfaces
- wleMachineMaxObjects
- wleMachineCurObjects
- wleMachineHwObjects
- wleSrvrCurObjsExt
- wleSrvrCurInterfaceExt
- wleSrvrSrvType
- wleSrvrClassPath
- wleSrvrjavaHeap
- wleSrvrjavaHeapuse
- wleSrvrjavaVendor
- wleSrvrjavaVersion

The object tuxTranGstate also has WLE specific states. For more information on these objects, refer to Chapter 2, “Tuxedo Core MIB.”
### Table 5-1 WLE Specific MIB Groups

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wleInterfaceTable</td>
<td>This group represents the configuration and runtime attributes of CORBA interfaces at both the domain and server-group levels.</td>
</tr>
<tr>
<td>wleLclInterfaceTable</td>
<td>The instances in this table return local wleInterfaceTable attributes for the local host on which the BEA SNMP Agent is running. Adam metric.</td>
</tr>
<tr>
<td>wleIfQueueTable</td>
<td>The instances in this table represent the runtime attributes of interface as it pertains to a particular server queue (tuxTqueue). Adam metric.</td>
</tr>
<tr>
<td>wleLclIfQueueTable</td>
<td>The instances in this table represent the local attributes of wleIfQueueTable instances. These values are specific to the host on which the BEA SNMP Agent is running.</td>
</tr>
<tr>
<td>wleJdbcConPoolTable</td>
<td>This class represents the configuration and runtime attributes of JDBC connection pools on a Java server.</td>
</tr>
<tr>
<td>wleJdbcConPoolExtnTable</td>
<td>This class represents the extensions to the configuration and runtime attributes of JDBC connection pools on a Java server. Adam metric.</td>
</tr>
<tr>
<td>wleModuleTable</td>
<td>This group represents the modules installed on a Java server. Adam metric.</td>
</tr>
</tbody>
</table>

This table represents occurrences of factories registered with the FactoryFinder.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>wleFactorySerNo</td>
<td>.1.3.6.1.4.1.140.300.48.1.1</td>
</tr>
</tbody>
</table>
### wleFactorySerNo

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
This object is the running number. This is used as the index to instances in this table.

### wleFactoryId

**Syntax**  
`DisplayString(SIZE(1..256))`

**Access**  
read-only

**Description**  
The registered ID for the factory.

### wleFactoryIfName

**Syntax**  
`DisplayString(SIZE(1..128))`

**Access**  
read-only

**Description**  
The fully qualified interface name for the factory. This is the interface repository ID for the factory. The format of this name is dependent on the options specified in the IDL that generates the interface implementation. Consult the CORBA 2.1 specification, section 7.6, for details.

### wleFactoryState

**Syntax**  
`INTEGER { active(1) }`

**Access**  
read-only
Description

A GET operation retrieves runtime information for the selected `wleFactoryTable` instance or instances. The returned value is 1 (active) if the instance is registered with the FactoryFinder.
The `wleInterfaceTable` represents configuration and runtime attributes of CORBA interfaces at both the domain and server-group levels.

There are certain semantic differences in the objects of the `wleInterfaceTable` between server-group and domain level instances.

A domain-level `wleInterfaceTable` instance is not associated with a Server group. In this case, its `wleIfSrvGrp` attribute has the invalid value `*`.

A server-group level instance has an associated Server group. In this case, its `wleIfSrvGrp` attribute has a valid server group name for the domain. This server-group level representation of an interface also provides a container for managing the interface state (the `wleIfState` object) and for collecting accumulated statistics.

Every CORBA interface that is activated in a server must have a server-group level `wleInterfaceTable` instance. The activation of interfaces in a server is controlled by the state of a `wleIfQueue` instance for the interface. Activation of a `wleIfQueue` instance causes its attributes to be initialized with values specified for the associated server-group level `wleInterfaceTable` instance. If such an instance does not already exist, then one is dynamically created. This dynamically created server-group level `wleInterfaceTable` instance is initialized with the attributes of the domain-level `wleInterfaceTable` instance for the interface, if one exists. If an associated domain-level instance does not exist, system-specified default configuration values are used. After they are activated, interfaces are always associated with a server-group level `wleInterfaceTable` instance.

The specification of configuration attributes for interfaces at any level is optional. Interfaces offered by a server are identified through the ICF file used for generating skeletons. The interfaces are advertised automatically by the system when the server is activated.

The following table lists the objects within the `wleInterfaceTable`.

---

**wleInterfaceTable**

The `wleInterfaceTable` represents configuration and runtime attributes of CORBA interfaces at both the domain and server-group levels.

There are certain semantic differences in the objects of the `wleInterfaceTable` between server-group and domain level instances.

A domain-level `wleInterfaceTable` instance is not associated with a Server group. In this case, its `wleIfSrvGrp` attribute has the invalid value `*`.

A server-group level instance has an associated Server group. In this case, its `wleIfSrvGrp` attribute has a valid server group name for the domain. This server-group level representation of an interface also provides a container for managing the interface state (the `wleIfState` object) and for collecting accumulated statistics.

Every CORBA interface that is activated in a server must have a server-group level `wleInterfaceTable` instance. The activation of interfaces in a server is controlled by the state of a `wleIfQueue` instance for the interface. Activation of a `wleIfQueue` instance causes its attributes to be initialized with values specified for the associated server-group level `wleInterfaceTable` instance. If such an instance does not already exist, then one is dynamically created. This dynamically created server-group level `wleInterfaceTable` instance is initialized with the attributes of the domain-level `wleInterfaceTable` instance for the interface, if one exists. If an associated domain-level instance does not exist, system-specified default configuration values are used. After they are activated, interfaces are always associated with a server-group level `wleInterfaceTable` instance.

The specification of configuration attributes for interfaces at any level is optional. Interfaces offered by a server are identified through the ICF file used for generating skeletons. The interfaces are advertised automatically by the system when the server is activated.

The following table lists the objects within the `wleInterfaceTable`.

---

**wleInterfaceTable**
### Variable Name | Object ID
--- | ---
wlIfSerNo | .1.3.6.1.4.1.140.300.53.1.1.1
wlIfName | .1.3.6.1.4.1.140.300.53.1.1.2
wlIfSrvGrp | .1.3.6.1.4.1.140.300.53.1.1.3
wlIfState | .1.3.6.1.4.1.140.300.53.1.1.4
wlIfAutoTran | .1.3.6.1.4.1.140.300.53.1.1.5
wlIfLoad | .1.3.6.1.4.1.140.300.53.1.1.6
wlIfPrio | .1.3.6.1.4.1.140.300.53.1.1.7
wlIfTimeout | .1.3.6.1.4.1.140.300.53.1.1.8
wlIfTranTime | .1.3.6.1.4.1.140.300.53.1.1.9
wlIfFbRoutingName | .1.3.6.1.4.1.140.300.53.1.1.10
wlIfLmid | .1.3.6.1.4.1.140.300.53.1.1.11
wlIfNumServers | .1.3.6.1.4.1.140.300.53.1.1.12
wlIfTpPolicy | .1.3.6.1.4.1.140.300.53.1.1.13
wlIfTxPolicy | .1.3.6.1.4.1.140.300.53.1.1.14

### wleIfSerNo

**Syntax** INTEGER  
**Access** read-only  
**Description** This is the running number. This object is used as an index to instances in this table.
### wleInterfaceTable

#### wleIfName

**Syntax**  
DisplayString(SIZE(1..128))

**Access**  
read-only

**Description**  
The fully qualified interface name. This is the interface ID. The format of this name is one of the options specified in the IDL that generates the interface implementation. Consult the CORBA 2.1 specification, Section 7.6, for details.

#### wleIfSrvGrp

**Syntax**  
DisplayString(SIZE(1..30))

**Access**  
read-write

**Description**  
The server group name. Server group names cannot contain an asterisk, comma, or colon. An asterisk (*) specified as a value for this object specifies a domain level instance.

**Note:** This object can be SET only during creation of a new row.

#### wleIfState

**Syntax**  
INTEGER { active(1), inactive(2), suspended(3), partitioned(4), invalid(5), reactivate(6) }

**Access**  
read-write

**Description**  
The semantics for GET and SET requests differ between server-group and domain level instances as noted in the following list.

**GET:** {active(1) | inactive(2) | suspended(3) | partitioned(4)}

A GET request retrieves configuration information for the selected wleInterfaceTable instance or instances. The only states that can be returned are: active, inactive, suspended, partitioned.

**active(1)**

The wleInterfaceTable instance is defined and at least one corresponding wleIfQueueTable instance is in the active state. For a server-group level wleInterfaceTable instance, corresponding wleIfQueueTable instances are those with matching wleIfName and wleIfSrvGrp objects. For a
domain-level wleInterfaceTable instance, corresponding 
wleIfQueueTable instances are those with matching wleIfName value 
regardless of their wleIfSrvGrp value.

inactive(2)

The wleInterfaceTable instance is defined and there are no corresponding 
wleIfQueueTable instances in any active state.

suspended(3)

The wleInterfaceTable instance is defined and among all corresponding 
wleIfQueueTable instances, there are none in the active state and at least 
one in the suspended state.

partitioned(4)

The wleInterfaceTable instance is defined and among all the 
corresponding wleIfQueueTable instances, there are none in the active 
state, none in the suspended state, and at least one in the partitioned state.

SET: {invalid(5)|active(1)|inactive(2)|reactivate(6)|suspended(3)}

A SET request updates runtime and configuration information for the selected 
wleInterfaceTable instance. Modifications can affect more than one 
server group when domain-level changes are made, and runtime 
modifications can affect more than one server if multiple servers are 
currently offering an interface. Only the following values can be used in a 
SET request: invalid, active, reactivate, or suspended.

invalid(5)

Delete the wleInterfaceTable instance for the application. This state 
change is allowed only when the instance is in the inactive state.

active(1)

Activate the wleInterfaceTable instance for the application. Setting this 
state on a domain level instance has the effect of activating all corresponding 
wleIfQueueTable instances that are currently suspended throughout the 
domain. Setting this state on a server-group level instance affects only 
servers within the group offering the interface. This state change is allowed 
only when the instance is in the suspended state. A successful return leaves 
the object in the active(1) state.

reactivate(6)
wleInterfaceTable

Reactivates the wleInterfaceTable instance. Setting this state on a domain level instance has the effect of activating all corresponding wleIfQueueTable instances that are currently suspended throughout the domain. Setting this state on a server-group level instance affects only servers within the group offering the interface. This state change is allowed only when the instance is in the active(1) or suspended(3) state. Successful return leaves the instance in the active(1) state. Setting this state permits global activation of wleIfQueueTable instances suspended at the server-group level without having to individually activate each server-group level wleInterfaceTable instance.

suspended(3)

Suspend the wleInterfaceTable instance. Setting this state on the domain-level object has the effect of suspending all corresponding wleIfQueueTable instances that are currently active throughout the domain. Setting this state on a server-group level instance affects only servers within the group offering the interface. This state change is permitted only in the active(1) state. Successful return leaves the object in the suspended(3) state.

Note: Dynamic advertisement of interfaces (that is, state change from inactive(2) or invalid(5) to active(1)) is not supported, nor is removal of advertisement (that is, state change from active(1) to inactive(2)).

wleIfAutoTran

Syntax

INTEGER { yes(1), no(2) }

Access

read-write

Description

Signifies whether a transaction is automatically started for invocations made outside a transaction context.

This object has the following limitations:

- Runtime updates to this attribute are not reflected in active equivalent wleInterfaceTable instances.
- The wleIfTxPolicy object can override the value specified for this attribute in the ubbconfig file. If wleIfTxPolicy is always(1), a wleIfAutoTran value of no(2) has no effect at runtime. Behavior is as though the setting were yes(1). If wleIfTxPolicy is never(2), an wleIfAutoTran value of yes(1) has no effect. The interface is never involved in a transaction. If
wleIfTxPolicy is ignore(4), a wleIfAutoTran value of yes(1) has no effect. The interface is never involved in a transaction.

wleIfLoad

Syntax INTEGER (1..32767)
Access read-write
Description This object imposes the indicated load on the system. Interface loads are used for load-balancing. That is, queues with higher enqueued workloads are less likely to be chosen for a new request.

Note: Runtime updates to this attribute for domain level instances do not affect corresponding server-group level instances for the same interface.

wleIfPrio

Syntax INTEGER (1..100)
Access read-write
Description Dequeueing priority. If multiple interface requests are waiting on a queue for servicing, the higher priority requests are handled first.

Note: Runtime updates to this attribute for domain-level instances do not affect corresponding server-group level instances for the same interface.

wleIfTimeout

Syntax INTEGER
Access read-write
Description The time limit (in seconds) for processing individual method invocations for this interface. Servers that process method invocations for this interface are terminated abortively if they exceed the specified time limit in processing the request. A value of 0 for this attribute indicates that the server should not be terminated abortively.

Note: Runtime updates to this attribute for domain-level instances do not affect corresponding server-group level instances for the same interface.
## wleIfTranTime

**Syntax**
INTEGER

**Access**
read-write

**Description**
Transaction timeout value in seconds for transactions automatically started for this `wleInterfaceTable` instance. Transactions are started automatically when a request not in transaction mode is received and the `wleIfAutoTran` object value for the interface is `yes(1)`.

**Note:** Runtime updates to this attribute for domain-level instances do not affect corresponding server-group level instances for the same interface.

## wleIfFbRoutingName

**Syntax**
`DisplayString (SIZE(1..15))`

**Access**
read-write

**Description**
The factory-based routing criteria associated with this interface.

**Note:** This attribute can be set only for a domain-level `wleInterfaceTable` instance, that is, only if `wleIfSrvGrp` is `*`.

## wleIfLmid

**Syntax**
`DisplayString (SIZE(1..30))`

**Access**
read-only

**Description**
Current logical machine with which the active equivalent server-group level `wleInterfaceTable` instance is associated. This attribute is `NULL` for domain-level instances.

## wleIfNumServers

**Syntax**
INTEGER

**Access**
read-only

**Description**
The number of corresponding servers that offer this interface.
5 WLE MIBs

**wleIfTpPolicy**

**Syntax**
INTEGER { method(1), transaction(2), process(3) }

**Access**
read-only

**Description**
The TP framework deactivation policy. This reflects the policy registered with the framework at server startup. The first server to register with the interface sets the value in wleInterfaceTable. This cannot be changed.

**wleIfTxPolicy**

**Syntax**
INTEGER { always(1), never(2), optional(3), ignore(4) }

**Access**
read-only

**Description**
The transaction policy for the interface. The setting in this attribute affects the wleIfAutoTran object. This policy is set by the application developer and is registered when the server starts.
The table returns values for the local host on which the Agent Connection is running. The following table lists the columnar objects that comprise each row (instance) in the table.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>wleLclIfSerNo</td>
<td>.1.3.6.1.4.1.140.300.53.2.1.1</td>
</tr>
<tr>
<td>wleLclIfName</td>
<td>.1.3.6.1.4.1.140.300.53.2.1.2</td>
</tr>
<tr>
<td>wleLclSrvGrp</td>
<td>.1.3.6.1.4.1.140.300.53.2.1.3</td>
</tr>
<tr>
<td>wleLclIfNcompleted</td>
<td>.1.3.6.1.4.1.140.300.53.2.1.4</td>
</tr>
<tr>
<td>wleLclIfNqueued</td>
<td>.1.3.6.1.4.1.140.300.53.2.1.5</td>
</tr>
</tbody>
</table>

**wleLclIfSerNo**

**Syntax**  INTEGER

**Access**  read-only

**Description**  This is the running number. This number is used as an index into the table.

**wleLclIfName**

**Syntax**  DisplayString (SIZE(1..128))

**Access**  read-only

**Description**  The fully qualified interface name. The interface repository ID for the interface. The format of this name is dependent on the options specified in the IDL that generates the interface implementation. See the CORBA 2.1 Specification Section 7.6 [CORBA] for details.
**wleLclIfSrvGrp**

**Syntax**
DisplayString (SIZE(1..30))

**Access**
read-only

**Description**
The server group name. Server group names cannot contain an asterisk, comma, or colon. A value of * for this object indicates a domain-level interface.

**wleLclIfNcompleted**

**Syntax**
INTEGER

**Access**
read-only

**Description**
The number of method invocations completed for the corresponding wleIfQueueTable instances since they were initially offered. The values returned are for the indicated interface on the local host where the BEA SNMP Agent is running.

**Note:** This attribute is returned only when tuxTdomainLoadBalance is yes(1).

**wleLclIfNqueued**

**Syntax**
INTEGER

**Access**
read-only

**Description**
The number of requests currently enqueued for this interface. The values returned are for the indicated interface on the local host where the BEA SNMP Agent is running.

**Note:** This attribute is returned only when tuxTdomainLoadBalance is yes(1).
This table represents the runtime attributes of an interface for a particular server queue. This group provides access to the inherited configuration attributes of an interface as well as statistics relating to the interface on the queue. This class gives administrators finer granularity in suspending and activating interfaces. This group provides the link between the interface name and the server processes capable of processing method invocations on the interface. For example, `wleIfQueueAddr` can be used to identify the corresponding server in the `tuxTsrvrTbl` and `tuxTsrvrTblExt` groups.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>wleIfQueueSerNo</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.1</td>
</tr>
<tr>
<td><code>wleIfQueueName</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.2</td>
</tr>
<tr>
<td><code>wleIfQueueSrvGrp</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.3</td>
</tr>
<tr>
<td><code>wleIfQueueRqAddr</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.4</td>
</tr>
<tr>
<td><code>wleIfQueueState</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.5</td>
</tr>
<tr>
<td><code>wleIfQueueAutoTran</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.6</td>
</tr>
<tr>
<td><code>wleIfQueueLoad</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.7</td>
</tr>
<tr>
<td><code>wleIfQueuePrio</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.8</td>
</tr>
<tr>
<td><code>wleIfQueueTimeout</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.9</td>
</tr>
<tr>
<td><code>wleIfQueueTranTime</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.10</td>
</tr>
<tr>
<td><code>wleIfQueueFbRoutingName</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.11</td>
</tr>
<tr>
<td><code>wleIfQueueLmid</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.12</td>
</tr>
<tr>
<td><code>wleIfQueueNumServers</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.13</td>
</tr>
<tr>
<td><code>wleIfQueueTpPolicy</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.14</td>
</tr>
<tr>
<td><code>wleIfQueueTxPolicy</code></td>
<td>.1.3.6.1.4.1.140.300.53.3.1.15</td>
</tr>
</tbody>
</table>
**wleIfQueueSerNo**

*Syntax*: INTEGER  
*Access*: read-only  
*Description*: The running number used as an index into this table.

**wleIfQueueName**

*Syntax*: DisplayString (SIZE(1..128))  
*Access*: read-only  
*Description*: The fully qualified interface name. The interface repository ID for the interface. The format of this name is dependent on the options specified in the IDL that generates the interface implementation. See the CORBA 2.1 specification Section 7.6 for details.

**wleIfQueueSrvGrp**

*Syntax*: DisplayString (SIZE(1..30))  
*Access*: read-only  
*Description*: The server group name. Server group names cannot contain an asterisk, comma, or colon.

**wleIfQueueRqAddr**

*Syntax*: DisplayString (SIZE(1..30))  
*Access*: read-only  
*Description*: The symbolic address of the request queue for an active server offering this interface. See `tuxTsrvrRqAddr` for more information about this attribute.
**wleIfQueueState**

**Syntax**  
INTEGER { active(1), suspended(2), partitioned(3), unknown(4) }  

**Access**  
read-write  

**Description**  
The values for **GET** and **SET** operations are as follows:

**GET:** \{active(1) | suspended(2) | partitioned(3)\}

A **GET** request retrieves configuration information for the selected `wleIfQueueTable` instances. The meaning of the possible return values are as follows:

- **active(1)**  
  Represents an available interface in the running system.

- **suspended(2)**  
  Represents a currently suspended interface in the running system.

- **partitioned(3)**  
  Represents a currently partitioned interface in the running system.

**SET:** \{active(1) | suspended(2)\}

The values for **SET** are:

- **active(1)**  
  Activates the `wleIfQueueTable` instance. This state change is allowed only in the **suspended(2)** state. A successful return leaves instances in the **active(1)** state.

- **suspended(2)**  
  Suspends the `wleIfQueueTable` instance. This state change is allowed only in the **active(1)** state. A successful return leaves the object in the **suspended(2)** state.

**Note:** Dynamic advertisement of interfaces (that is, a state change from inactive or invalid to active) is not supported, nor is unadvertisement (that is, a state change from active to inactive).
**wleIfQueueAutoTran**

**Syntax**
INTEGER { yes(1), no(2) }

**Access**
read-only

**Description**
Signifies whether a transaction is automatically started for invocations made outside a transaction context.

This object has the following limitations:

- Runtime updates to this attribute are not reflected in active equivalent `wleInterfaceTable` instances.

- The `wleIfTxPolicy` object can override the value specified for this attribute in the `ubbconfig` file. If `wleIfTxPolicy` is `always(1)`, an `wleIfQueueAutoTran` value of `no(2)` has no effect at runtime. Behavior is as though the setting were `yes(1)`. If `wleIfTxPolicy` is `never(2)`, an `wleIfQueueAutoTran` value of `yes(1)` has no effect. The interface is never involved in a transaction. If `wleIfTxPolicy` is `ignore(4)`, an `wleIfQueueAutoTran` value of `yes(1)` has no effect. The interface is never involved in a transaction.

**wleIfQueueLoad**

**Syntax**
INTEGER (1..32767)

**Access**
read-only

**Description**
Load imposed on the system by this instance. Interface loads are used for load-balancing. Queues with higher enqueued workloads are less likely to be chosen for a new request.

**wleIfQueuePrio**

**Syntax**
INTEGER (1..101)

**Access**
read-only

**Description**
Dequeueing priority. If multiple interface requests are waiting on a queue for servicing, the higher priority requests are handled first.
**wleIfQueueTable**

**wleIfQueueTimeout**

Syntax: INTEGER  
Access: read-only  
Description: The time limit (in seconds) for processing individual method invocations for this interface. Servers processing method invocations for this interface are abortively terminated if they exceed the specified time limit in processing the request. A value of 0 for this attribute indicates that the server should not be abortively terminated.

**wleIfQueueTranTime**

Syntax: INTEGER  
Access: read-only  
Description: The transaction timeout value in seconds for transactions automatically started for this instance. Transactions are started automatically when a request not in transaction mode is received and the wleIfAutoTran attribute value for the interface is yes(1).

**wleIfQueueFbRoutingName**

Syntax: DisplayString (SIZE(1..15))  
Access: read-only  
Description: The factory-based routing criterion associated with this interface.

**wleIfQueueLmid**

Syntax: DisplayString (SIZE(1..30))  
Access: read-only  
Description: The current logical machine on which this queue is offering this interface.
wleIfQueueNumServers

Syntax INTEGER
Access read-only
Description The number of corresponding servers that offer this interface on this queue.

wleIfQueueTpPolicy

Syntax INTEGER { method(1), transaction(2), process(3) }
Access read-only
Description The TP framework deactivation policy. This reflects the policy registered with the framework at the server startup. The first server to register the interface sets the value. This value cannot be changed.

wleIfQueueTxPolicy

Syntax INTEGER { always(1), never(2), optional(3), ignore(4) }
Access read-only
Description The transaction policy for the interface. The setting in this attribute affects the effect of the wleIfQueueAutoTran object. See wleIfQueueAutoTran for further explanation. This attribute is always read-only. It is set by the developer when the server is built and registered at server startup.
This table represents the local attributes of the `wleIfQueueTable`. These values are specific to the host on which the BEA SNMP Agent is running.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>wleLclIfQueueSerNo</code></td>
<td><code>.1.3.6.1.4.1.140.300.53.4.1.1</code></td>
</tr>
<tr>
<td><code>wleLclIfQueueName</code></td>
<td><code>.1.3.6.1.4.1.140.300.53.4.1.2</code></td>
</tr>
<tr>
<td><code>wleLclIfQueueSrvGrp</code></td>
<td><code>.1.3.6.1.4.1.140.300.53.4.1.3</code></td>
</tr>
<tr>
<td><code>wleLclIfQueueRqAddr</code></td>
<td><code>.1.3.6.1.4.1.140.300.53.4.1.4</code></td>
</tr>
<tr>
<td><code>wleLclIfQueueNcompleted</code></td>
<td><code>.1.3.6.1.4.1.140.300.53.4.1.5</code></td>
</tr>
<tr>
<td><code>wleLclIfQueueNqueued</code></td>
<td><code>.1.3.6.1.4.1.140.300.53.4.1.6</code></td>
</tr>
<tr>
<td><code>wleLclIfQueueCurObjs</code></td>
<td><code>.1.3.6.1.4.1.140.300.53.4.1.7</code></td>
</tr>
<tr>
<td><code>wleLclIfQueueCurTrans</code></td>
<td><code>.1.3.6.1.4.1.140.300.53.4.1.8</code></td>
</tr>
</tbody>
</table>

**wleLclIfQueueSerNo**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: The running number used as an index into this table.

**wleLclIfQueueName**

- **Syntax**: `DisplayString(SIZE(1..128))`
- **Access**: read-only
- **Description**: The fully qualified interface name. The interface repository ID for this interface. The format of this name is dependent on the options specified in the IDL that generates the interface implementation. See the CORBA 2.1 specification Section 7.6 for details.
wleLclIfQueueSrvGrp
Syntax: DisplayString(SIZE(1..30))
Access: read-only
Description: The server group name. Server group names cannot contain an asterisk, comma, or colon.

wleLclIfQueueRqAddr
Syntax: DisplayString(SIZE(1..30))
Access: read-only
Description: The symbolic address of the request queue for an active server that offers this interface. See tuxTsrvrRqAddr for more information about this attribute.

wleLclIfQueueNcompleted
Syntax: INTEGER
Access: read-only
Description: The number of interface method invocations completed since the interface was initially offered.

Note: This attribute is returned only when tuxTdomainLoadBalance is equal to yes(1).

wleLclIfQueueNqueued
Syntax: INTEGER
Access: read-only
Description: The number of requests currently enqueued for this interface.

Note: This attribute is returned only when tuxTdomainLoadBalance is equal to yes(1).
### wleLclIfQueueCurObjs

**Syntax**
INTEGER

**Access**
read-only

**Description**
The number of active objects for this interface for the associated queue. This number represents the number of entries in the active object table for this queue on the associated machine. This number includes objects that are not in memory but were invoked within an active transaction.

### wleLclIfQueueCurTrans

**Syntax**
INTEGER

**Access**
read-only

**Description**
The number of active global transactions associated with this interface for its associated queue.
This class represents the configuration and runtime attributes of JDBC connection pools on a Java server. The attributes consist of statistics or values associated with each connection pool. Except for `tuxJdbcConPoolState`, attributes are persistent in TUXCONFIG. Local attributes are local to the memory allocated to a Java server.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>wleJdbcConPoolSrvId</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.10</td>
</tr>
<tr>
<td>wleJdbcConPoolSrvGrp</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.20</td>
</tr>
<tr>
<td>wleJdbcConPoolDsName</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.30</td>
</tr>
<tr>
<td>wleJdbcConPoolDriver</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.40</td>
</tr>
<tr>
<td>wleJdbcConPoolUrl</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.50</td>
</tr>
<tr>
<td>wleJdbcConPoolState</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.60</td>
</tr>
<tr>
<td>wleJdbcConPoolDbName</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.70</td>
</tr>
<tr>
<td>wleJdbcConPoolDbUser</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.80</td>
</tr>
<tr>
<td>wleJdbcConPoolDbPassword</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.90</td>
</tr>
<tr>
<td>wleJdbcConPoolUserRole</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.100</td>
</tr>
<tr>
<td>wleJdbcConPoolDbHost</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.110</td>
</tr>
<tr>
<td>wleJdbcConPoolDbNetProtocol</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.120</td>
</tr>
<tr>
<td>wleJdbcConPoolDbPort</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.130</td>
</tr>
<tr>
<td>wleJdbcConPoolProps</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.140</td>
</tr>
<tr>
<td>wleJdbcConPoolEnableXa</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.150</td>
</tr>
<tr>
<td>wleJdbcConPoolCreateOnStartUp</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.160</td>
</tr>
<tr>
<td>wleJdbcConPoolLoginDelay</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.170</td>
</tr>
<tr>
<td>wleJdbcConPoolInitCapacity</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.180</td>
</tr>
<tr>
<td>wleJdbcConPoolMaxCapacity</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.190</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Object ID</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>wleJdbcConPoolCapacityIncr</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.200</td>
</tr>
<tr>
<td>wleJdbcConPoolAllowShrinking</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.210</td>
</tr>
<tr>
<td>wleJdbcConPoolShrinkPeriod</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.220</td>
</tr>
<tr>
<td>wleJdbcConPoolTestTable</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.230</td>
</tr>
<tr>
<td>wleJdbcConPoolRefresh</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.240</td>
</tr>
<tr>
<td>wleJdbcConPoolTestOnReserve</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.250</td>
</tr>
<tr>
<td>wleJdbcConPoolTestOnRelease</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.260</td>
</tr>
<tr>
<td>wleJdbcConPoolWaitForConn</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.270</td>
</tr>
<tr>
<td>wleJdbcConPoolWaitTimeOut</td>
<td>.1.3.6.1.4.1.140.300.60.1.1.280</td>
</tr>
</tbody>
</table>

**wleJdbcConPoolSrvId**

**Syntax**  
INTEGER (1..30001)

**Access**  
read-only

**Description**  
Together with the server group name, this value is used to identify a Java server, specified in the SERVERS section of the UBBCONFIG file, for which the connection pool is being described.

**wleJdbcConPoolSrvGrp**

**Syntax**  
DisplayString (SIZE (1..15))

**Access**  
read-only

**Description**  
Name of a server group. This attribute is used to identify a Java server specified in the SERVERS section of the UBBCONFIG file, for which the connection pool is being described.
### wleJdbcConPoolDsName

**Syntax**
DisplayString (SIZE (0..30))

**Access**
read-only

**Description**
The data source name for the connection pool.

### wleJdbcConPoolDriver

**Syntax**
DisplayString (SIZE (0..256))

**Access**
read-write

**Description**
The class name for the Java driver.

### wleJdbcConPoolUrl

**Syntax**
DisplayString (SIZE (2..256))

**Access**
read-write

**Description**
URL for a JDBC driver that is not JDBC 2.0-compliant.

### wleJdbcConPoolState

**Syntax**
INTEGER { valid(1) | invalid(2) } 

**Access**
read-write

**Description**
This object denotes the current state of the tuxDmConnection instance.

**GET requests:**
- valid(1): The object exists.

**SET requests:**
- invalid(1): Delete object.
**wleJdbcConPoolDbName**

**Syntax**  
DisplayString (SIZE (0..30))

**Access**  
read-write

**Description**  
The database name.

**wleJdbcConPoolDbUser**

**Syntax**  
DisplayString (SIZE (0..30))

**Access**  
read-write

**Description**  
The user’s account name.

**wleJdbcConPoolDbPassword**

**Syntax**  
DisplayString (SIZE (0..64))

**Access**  
read-write

**Description**  
The user’s password. The password entered by the user should not exceed 24 bytes.

**wleJdbcConPoolUserRole**

**Syntax**  
DisplayString (SIZE (0..30))

**Access**  
read-write

**Description**  
The user’s SQL role.

**wleJdbcConPoolDbHost**

**Syntax**  
DisplayString (SIZE (0..30))

**Access**  
read-write

**Description**  
The database server name.
wleJdbcConPoolDbNetProtocol

Syntax: DisplayString (SIZE (0..30))
Access: read-write
Description: The protocol used to communicate with the database.

wleJdbcConPoolDbPort

Syntax: INTEGER (1..65535)
Access: read-write
Description: The port used for database connections.

wleJdbcConPoolProps

Syntax: DisplayString (SIZE (0..256))
Access: read-write
Description: Vendor-specific information for the JDBC driver.

wleJdbcConPoolEnableXa

Syntax: INTEGER { yes(1) | no(2) }
Access: read-write
Description: If set to yes, indicates that the pool supports XA mode.

wleJdbcConPoolCreateOnStartUp

Syntax: INTEGER { yes(1) | no(2) }
Access: read-write
Description: If set to yes, indicates that the connection pool is created when the first request arrives.
wleJdbcConPoolLoginDelay

Syntax  INTEGER
Access   read-write
Description  The login delay in seconds.

wleJdbcConPoolInitCapacity

Syntax  INTEGER
Access   read-write
Description  The number of connections initially supported in the connection pool. This number
              should not exceed the value of tuxJdbcConPoolMaxCapacity.

wleJdbcConPoolMaxCapacity

Syntax  INTEGER
Access   read-write
Description  The maximum number of connections supported in the connection pool.

wleJdbcConPoolCapacityIncr

Syntax  INTEGER
Access   read-write
Description  The number of connections added to the pool when the current limit is exceeded but
              the maximum capacity is not yet reached.

wleJdbcConPoolAllowShrinking

Syntax  INTEGER { yes(1) | no(2) }
Access   read-write
Description  If set to yes(1), allows connection pool shrinking.
\section*{WLE MIBs}

**wleJdbcConPoolShrinkPeriod**

Syntax: INTEGER  
Access: read-write  
Description: The interval (in minutes) after which shrinking occurs.

**wleJdbcConPoolTestTable**

Syntax: DisplayString (SIZE (0..256))  
Access: read-write  
Description: The name of a table in the database that is used to test the viability of connections in the connection pool.

The query select count(*) from TESTTABLE is used to test a connection. The table must exist and be accessible to the database user for the connection.

**wleJdbcConPoolRefresh**

Syntax: INTEGER  
Access: read-write  
Description: The refresh interval (in minutes).

\texttt{wleJdbcConPoolRefresh} \texttt{0<num}

**wleJdbcConPoolTestOnReserve**

Syntax: INTEGER \{ yes(1) | no(2) \}  
Access: read-write  
Description: If set to \texttt{yes(1)}, the Java server tests a connection after removing it from the pool and before giving it to the client. The test adds a small delay in serving the client’s request for a connection from the pool, but ensures that the client receives a working connection. A value for \texttt{wleJdbcConPoolTestTable} must be set for this feature to work.
**wleJdbcConPoolTestOnRelease**

Syntax: `INTEGER { yes(1) | no(2) }`

Access: read-write

Description: If set to `yes(1)`, the Java server tests a connection before returning it to the connection pool. If all the connections in the pool are already in use and a client is waiting for a connection, the client’s wait is slightly longer due to the test of the connection. A value for `tuxJdbcConPoolTestTable` must be set for this feature to work.

**wleJdbcConPoolWaitForConn**

Syntax: `INTEGER { yes(1) | no(2) }`

Access: read-write

Description:

If set to `yes(1)`, this feature enables an application to wait for a connection indefinitely if none is currently available. If set to `no(2)`, a request for a connection returns to the caller immediately if there is no connection available.

The default is `yes(1)` unless `wleJdbcConPoolWaitTimeOut` is specified, in which case it becomes `no(2)`.

**wleJdbcConPoolWaitTimeOut**

Syntax: `INTEGER`

Access: read-only

Description: Time (in seconds) that an application waits for a connection to become available.
wleJdbcConPoolExtnTable

An extension of the wleJdbcConPoolTable table.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>wleJdbcConPoolExtnSrvId</td>
<td>.1.3.6.1.4.1.140.300.60.2.1.10</td>
</tr>
<tr>
<td>wleJdbcConPoolExtnSrvGrp</td>
<td>.1.3.6.1.4.1.140.300.60.2.1.20</td>
</tr>
<tr>
<td>wleJdbcConPoolExtnDsName</td>
<td>.1.3.6.1.4.1.140.300.60.2.1.30</td>
</tr>
<tr>
<td>wleJdbcConPoolExtnConnUsed</td>
<td>.1.3.6.1.4.1.140.300.60.2.1.40</td>
</tr>
<tr>
<td>wleJdbcConPoolExtnConnAvailable</td>
<td>.1.3.6.1.4.1.140.300.60.2.1.50</td>
</tr>
<tr>
<td>wleJdbcConPoolExtnHwmConnUsed</td>
<td>.1.3.6.1.4.1.140.300.60.2.1.60</td>
</tr>
<tr>
<td>wleJdbcConPoolExtnHwmConnCreated</td>
<td>.1.3.6.1.4.1.140.300.60.2.1.70</td>
</tr>
<tr>
<td>wleJdbcConPoolExtnAwaitingConn</td>
<td>.1.3.6.1.4.1.140.300.60.2.1.80</td>
</tr>
<tr>
<td>wleJdbcConPoolExtnHwmForWait</td>
<td>.1.3.6.1.4.1.140.300.60.2.1.90</td>
</tr>
</tbody>
</table>

wleJdbcConPoolExtnSrvId

Syntax

INTEGER (1..30001)

Access

read-only

Description

Together with the server group name, this value is used to identify a Java server, specified in the SERVERS section of the UBBCONFIG file, for which the connection pool is being described.

wleJdbcConPoolExtnSrvGrp

Syntax

DisplayString (SIZE (1..15))

Access

read-only

Description

Name of a server group. This attribute is used to identify a Java server specified in the SERVERS section of the UBBCONFIG file, for which the connection pool is being described.
**wleJdbcConPoolExtnDsName**

**Syntax**  
DisplayString (SIZE (0..30))

**Access**  
read-only

**Description**  
The data source name for the connection pool.

**wleJdbcConPoolExtnConnUsed**

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
The number of connections currently in use.

**wleJdbcConPoolExtnConnAvailable**

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
The number of connections available in the pool.

**wleJdbcConPoolExtnHwmConnUsed**

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
The number of connections used above the high-water mark (HWM).

**wleJdbcConPoolExtnHwmConnCreated**

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
The number of connections created above the high-water mark (HWM).
wleJ jdbcConPoolExtnAwaitingConn

Syntax: INTEGER
Access: read-only
Description: T

wleJ jdbcConPoolExtnHwmForWait

Syntax: INTEGER
Access: read-only
Description: T
**wleModuleTable**

This group represents the modules installed on a Java server. The class attributes identify and characterize the module.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>wleModuleSrvId</td>
<td>.1.3.6.1.4.1.140.300.65.1.1.10</td>
</tr>
<tr>
<td>wleModuleSrvGrp</td>
<td>.1.3.6.1.4.1.140.300.65.1.1.20</td>
</tr>
<tr>
<td>wleModuleModule</td>
<td>.1.3.6.1.4.1.140.300.65.1.1.30</td>
</tr>
<tr>
<td>wleModuleClassPath</td>
<td>.1.3.6.1.4.1.140.300.65.1.1.40</td>
</tr>
<tr>
<td>wleModuleModuleArgs</td>
<td>.1.3.6.1.4.1.140.300.65.1.1.50</td>
</tr>
<tr>
<td>wleModuleState</td>
<td>.1.3.6.1.4.1.140.300.65.1.1.60</td>
</tr>
<tr>
<td>wleModuleModuleType</td>
<td>.1.3.6.1.4.1.140.300.65.1.1.70</td>
</tr>
</tbody>
</table>

**wleModuleSrvId**

Syntax: INTEGER (1..30001)  
Access: read-only  
Description: With the server group name, this value is used to identify a Java server, specified in the SERVERS section of the UBBCONFIG file for which the connection pool is being described.

**wleModuleSrvGrp**

Syntax: DisplayString (SIZE (0..20))  
Access: read-only  
Description: The name of a server group. This is used to identify a Java server, specified in the SERVERS section of the UBBCONFIG file, for which the connection pool is being described.
wleModuleModule

Syntax: DisplayString (SIZE (0..256))
Access: read-only
Description: The Java Archive (JAR) file that contains the module.

wleModuleModuleArgs

Syntax: DisplayString (SIZE (0..256))
Access: read-write
Description: The startup arguments for the module.

wleModuleState

Syntax: INTEGER { valid(1) | invalid(2) }
Access: read-write
Description: This object denotes the current state of the wle module.
   GET: valid(1)
       The object exists.
   SET: invalid(2)
       Delete object.

wleModuleModuleType

Syntax: INTEGER
Access: read-only
Description: The type of the module. This is either CORBA or EJB.
Access Control List MIB

An access control list (ACL) specifies who and what is authorized to access Tuxedo or WLE system objects. The ACL MIB enables a system manager to administer Tuxedo or WLE security by authenticating users, setting permissions, and controlling access. The ACL MIB defines the objects controlled by the ACL facility. These MIB objects are grouped into three major categories. The following table lists groups that make up the ACL MIB.

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTAcIcGrpTable</td>
<td>ACL group</td>
</tr>
<tr>
<td>tuxTAcIcPermTable</td>
<td>ACL permissions</td>
</tr>
<tr>
<td>tuxTAcIcPrinTbl</td>
<td>ACL principal (users or domains)</td>
</tr>
</tbody>
</table>

For Tuxedo or WLE security, define application security options in the Domain group. This group lets you specify a user identity and security type used by your Tuxedo application. The users and remote domains in an application that need authentication and authorization are collectively known as principals. The managed objects for getting or setting the values of principals are defined in the tuxTAcIcPrinTbl group. The managed objects for getting or setting the values of ACL groups are defined in the tuxTAcIcGrpTable. The ACL MIB, as a whole, specifies the principals and access control lists for Tuxedo applications services, application queues, and events. You can define these ACL permissions for service, event, and application queue names. The managed objects that enable you to do this are defined in the tuxTAcIcPermTable group. All these ACL MIB groups and their objects are described in the following sections.
The tuxTAclGrpTable group represents groups of Tuxedo application users and domains. The following table lists the managed objects that are part of the tuxTAclGrpTable group. To create a new row in the table, it is necessary to issue a SET request for a non-existing row.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTAclGrpName</td>
<td>.1.3.6.1.4.1.140.300.11.1.1.1.1</td>
</tr>
<tr>
<td>tuxTAclGrpId</td>
<td>.1.3.6.1.4.1.140.300.11.1.1.1.2</td>
</tr>
<tr>
<td>tuxTAclGrpState</td>
<td>.1.3.6.1.4.1.140.300.11.1.1.1.3</td>
</tr>
</tbody>
</table>

**tuxTAclGrpName**

Syntax

DisplayString(SIZE(1..30))

Access read-write

Description Logical name of the group. A group name is a string of printable characters and cannot contain a pound sign, comma, colon, or newline.

**Note:** This object can be set only during row creation.

**tuxTAclGrpId**

Syntax

INTEGER (0..16384)

Access read-write

Description Group identifier associated with this user. A value of 0 indicates the default group other. If the group identifier is not specified at creation time, it defaults to the next available (unique) identifier greater than 0.
tuxTAc1GrpTable

**tuxTAc1GrpState**

**Syntax**

INTEGER { valid(1), invalid(2) }

**Access**

read-write

**Description**

The values for GET and SET operations are as follows:

**GET**: valid(1)

A GET operation retrieves configuration information for the selected tuxTAc1GrpTable instance(s). The following state indicates the meaning of a tuxTAc1GrpState returned in response to a GET request. States not listed are not returned.

- **valid(1)**
  
tuxTAc1GrpTable instance is defined and inactive. Note that this is the only valid state for this class. ACL groups are never active.

**SET**: invalid(2)

A SET operation updates configuration information for the selected tuxTAc1GrpTable instance. The following state indicates the meaning of a tuxTAc1GrpState set in a SET request. States not listed might not be set.

- **invalid(2)**
  
Delete tuxTAc1GrpTable instance for application. Successful return removes the instance from the table.
The \texttt{tuxTAclPermTable} group indicates what groups are allowed to access Tuxedo system entities. These entities are named by a string. The names currently represent service names, event names, and application queue names. To create a new row in this table, it is necessary to issue a \texttt{SET} request for a non-existing row that specifies at least the values for \texttt{tuxTAclPermName} and \texttt{tuxTAclPermType}.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{tuxTAclPermName}</td>
<td>.1.3.6.1.4.1.140.300.11.2.1.1.1</td>
</tr>
<tr>
<td>\texttt{tuxTAclPermType}</td>
<td>.1.3.6.1.4.1.140.300.11.2.1.1.2</td>
</tr>
<tr>
<td>\texttt{tuxTAclPermGrpIds}</td>
<td>.1.3.6.1.4.1.140.300.11.2.1.1.3</td>
</tr>
<tr>
<td>\texttt{tuxTAclPermState}</td>
<td>.1.3.6.1.4.1.140.300.11.2.1.1.4</td>
</tr>
</tbody>
</table>

\texttt{tuxTAclPermName}

\textbf{Syntax} \hspace{1cm} \texttt{DisplayString(SIZE(1..30))}

\textbf{Access} \hspace{1cm} read-write

\textbf{Description} \hspace{1cm} The name of the entity for which permissions are being granted. The name can represent a service name, an event name, and/or a queue name. An ACL name is a string of printable characters and cannot contain a colon, pound sign, or newline.

\textbf{Note:} \hspace{1cm} This object can be set only during row creation.

\texttt{tuxTAclPermType}

\textbf{Syntax} \hspace{1cm} \texttt{INTEGER \{ enq(1), deq(2), service(3), postevent(4) \}}

\textbf{Access} \hspace{1cm} read-write

\textbf{Description} \hspace{1cm} The type of the entity for which permissions are being granted.

\textbf{Note:} \hspace{1cm} This object can be set only during row creation.
**tuxTaclPermGrpIds**

**Syntax**  
`DisplayString(SIZE(0..800))`

**Access**  
read-write

**Description**  
A comma-separated list of group identifiers (numbers) that are permitted access to the associated entity.

**tuxTaclPermState**

**Syntax**  
`INTEGER { valid(1), invalid(2) }`

**Access**  
read-write

**Description**  
The values for GET and SET operations are as follows:

**GET:**  
valid(1)  
A GET operation retrieves configuration information for all selected entities. The following state indicates the meaning of a `tuxTaclPermState` returned in response to a GET request. States not listed are not returned.

valid(1)  
a `tuxTaclPermState` instance is defined. Note that this is the only valid state for this class. ACL permissions are never active.

**SET:**  
invalid(2)  
A SET operation updates configuration information for the selected `tuxTaclPermState` instance. The following state indicates the meaning of a `tuxTaclPermState` set in a SET request. States not listed might not be set.

invalid(2)  
Delete `tuxTaclPermState` instance for application. State change allowed only when in the valid(1) state. Successful return leaves the object in the invalid(2) state.

Note that the `tuxTaclPermTable` instance refers to all groupids related to a particular `tuxTaclPermName` in the table.
The `tuxTAclPrinTbl` group represents users or domains that can access a Tuxedo application and the group with which they are associated. To join the application as a specific user, it is necessary to present a user-specific password. To create a new row in this table, it is necessary to issue a `SET` request for a non-existing row (instance).

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTAclPrinName</code></td>
<td>.1.3.6.1.4.1.140.300.11.3.1.1.1</td>
</tr>
<tr>
<td><code>tuxTAclCltName</code></td>
<td>.1.3.6.1.4.1.140.300.11.3.1.1.2</td>
</tr>
<tr>
<td><code>tuxTAclPrinId</code></td>
<td>.1.3.6.1.4.1.140.300.11.3.1.1.3</td>
</tr>
<tr>
<td><code>tuxTAclPrinGrp</code></td>
<td>.1.3.6.1.4.1.140.300.11.3.1.1.4</td>
</tr>
<tr>
<td><code>tuxTAclPrinPasswd</code></td>
<td>.1.3.6.1.4.1.140.300.11.3.1.1.5</td>
</tr>
<tr>
<td><code>tuxTAclPrinState</code></td>
<td>.1.3.6.1.4.1.140.300.11.3.1.1.6</td>
</tr>
</tbody>
</table>

### `tuxTAclPrinName`  

**Syntax:** `DisplayString(SIZE(1..30))`  
**Access:** read-write  
**Description:** Logical name of the user or domain (a principal). A principal name is a string of printable characters and cannot contain a pound sign, colon, or newline.  

**Note:** This object can be set only during row creation.
tuxTAcIName

Syntax: DisplayString(SIZE(1..30))
Access: read-write
Description: The client name associated with the user. It generally describes the role of the associated user and provides a further qualifier on the user entry. If the client name is not specified at creation time, the default is the wildcard asterisk (*). A client name is a string of printable characters and cannot contain a colon or newline.

tuxTAcPrinId

Syntax: INTEGER(1..131072)
Access: read-write
Description: Unique user identification number. If not specified at creation time, it defaults to the next available (unique) identifier greater than 0.
Note: This object can be set only during row creation.

tuxTAcPrinGrp

Syntax: INTEGER(0..16384)
Access: read-write
Description: Group identifier associated with this user. A value of 0 indicates the default group other. If the group identifier is not specified at creation time, the default value 0 is assigned.

tuxTAcPrinPasswd

Syntax: DisplayString
Access: read-write
Description: The clear-text authentication password for the associated user. Note that the system automatically encrypts this information on behalf of the administrator.
**tuxTAcIPrinState**

**Syntax**
INTEGER { valid(1), invalid(2) }

**Access**
read-write

**Description**
The values for **GET** and **SET** operations are as follows:

**GET:** valid(1)

A **GET** operation retrieves configuration information for the selected tuxTAcIPrinTbl instance(s). The following state indicates the meaning of tuxTAcIPrinState:

valid(1)

tuxTAcIPrinTbl instance is defined and inactive. Note that this is the only valid state for this class. ACL principals are never active.

**SET:** invalid(2)

A **SET** operation updates configuration information for the selected tuxTAcIPrinTbl instance. The following state indicates the meaning of a tuxTAcIPrinState set in a **SET** request. States not listed might not be set.

invalid(2)

Delete tuxTAcIPrinTbl instance for application. State change is allowed only when in the valid(1) state. Successful return leaves the object in the invalid(2) state.
Tuxedo and WLE systems can require clients to run on a workstation for purposes of security, performance, and convenience. A network administrator can define the environment required to control workstation clients using the Workstation MIB. This MIB is an extension of the Tuxedo Core MIB and specifies the information required to control access to a Tuxedo or WLE application from multiple workstations.

The Tuxedo Workstation subsystem consists of a workstation clients (WSC) library, the workstation listener (WSL) executable, and the workstation handler (WSH) executable. The Workstation MIB specifies information about workstation listeners and workstation handlers. The following table lists the two WSL and WSH groups through which you can manage a workstation listener and its associated workstation handler processes.

The Workstation MIB consists of the following groups.

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTwshTbl</td>
<td>Workstation Handler</td>
</tr>
<tr>
<td>tuxTwslTbl</td>
<td>Workstation Listener</td>
</tr>
</tbody>
</table>

You can define new workstation listeners in the tuxTwslTbl group, and you can obtain information about active workstation handlers from the tuxTwshTbl group.
The `tuxTwshTbl` table represents runtime attributes of WSH client processes. These objects characterize workstation statistics specific to a particular WSH client process. Objects in this table are only accessible through a BEA SNMP agent installed on the local machine. Objects are only accessible when the corresponding WSH is active.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTwshTaClientId</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.1</td>
</tr>
<tr>
<td>tuxTwshTaWshClientId</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.2</td>
</tr>
<tr>
<td>tuxTwshTaSrvGrp</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.3</td>
</tr>
<tr>
<td>tuxTwshTaSrvId</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.4</td>
</tr>
<tr>
<td>tuxTwshTaGrpNo</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.5</td>
</tr>
<tr>
<td>tuxTwshTaState</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.6</td>
</tr>
<tr>
<td>tuxTwshTaLmid</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.7</td>
</tr>
<tr>
<td>tuxTwshTaPid</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.8</td>
</tr>
<tr>
<td>tuxTwshTaNaddr</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.9</td>
</tr>
<tr>
<td>tuxTwshTaHwClients</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.10</td>
</tr>
<tr>
<td>tuxTwshTaMultiplex</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.11</td>
</tr>
<tr>
<td>tuxTwshTaCurClients</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.12</td>
</tr>
<tr>
<td>tuxTwshTaTimeleft</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.13</td>
</tr>
<tr>
<td>tuxTwshTaActive</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.14</td>
</tr>
<tr>
<td>tuxTwshTaTotactctime</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.15</td>
</tr>
<tr>
<td>tuxTwshTaTotidltime</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.16</td>
</tr>
<tr>
<td>tuxTwshTaCurwork</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.17</td>
</tr>
</tbody>
</table>
### tuxTwshTbl

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTwshTaFlowcnt</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.18</td>
</tr>
<tr>
<td>tuxTwshTaNumblockQ</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.19</td>
</tr>
<tr>
<td>tuxTwshTaRcvdByt</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.20</td>
</tr>
<tr>
<td>tuxTwshTaRcvdNum</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.21</td>
</tr>
<tr>
<td>tuxTwshTaSentByt</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.22</td>
</tr>
<tr>
<td>tuxTwshTaSentNum</td>
<td>.1.3.6.1.4.1.140.300.1.1.1.23</td>
</tr>
</tbody>
</table>

#### tuxTwshTaClientId

**Syntax**: `DisplayString(SIZE(1..78))`

**Access**: read-only

**Description**: Client identifier for this WSH. The data in this field should not be interpreted directly by the end user except for equality comparison.

#### tuxTwshTaWshClientId

**Syntax**: `DisplayString(SIZE(1..78))`

**Access**: read-only

**Description**: Client identifier for this WSH. The data in this field should not be interpreted directly by the end user except for equality comparison. Value is same as `tuxTwshTaClientId`.

#### tuxTwshTaSrvGrp

**Syntax**: `DisplayString(SIZE(1..30))`

**Access**: read-only

**Description**: Logical name of the server group for the associated WSL.
7 Workstation MIB

**tuxTwshTaSrvId**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER (1..30001)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>Unique (within the server group) server identification number for the associated WSL.</td>
</tr>
</tbody>
</table>

**tuxTwshTaGrpNo**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER (1..30000)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>Group number.</td>
</tr>
</tbody>
</table>

**tuxTwshTaState**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER { active(1), suspended(2), dead(3) }</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>State for the WSH client within the application. Any state defined for the tuxTclientTbl group can be returned or set. State changes to the suspended(2) state are transitive to all clients associated with this WSH as is the resetting of a suspended(2) WSH to active(1). Additionally, suspended(2) WSH clients are not assigned any additional incoming clients by the WSL. Note that the state of a WSH client might not be set to dead(3) when accessing the tuxTclientTbl group. However, the state transition to dead(3) is allowed via the tuxTwshTbl group and results in all connections handled by the targeted WSH being dropped abortively.</td>
</tr>
</tbody>
</table>

**tuxTwshTaLmid**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>DisplayString (SIZE(1..30))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>Current logical machine on which the WSH is running.</td>
</tr>
</tbody>
</table>
tuxTwshTaPid

Syntax: INTEGER
Access: read-only
Description: Native operating system process identifier for the WSH client.

tuxTwshTaNaddr

Syntax: DisplayString(SIZE(1..78))
Access: read-only
Description: Network address of workstation handler. Hexadecimal addresses are converted to an ASCII format with a leading 0x.

tuxTwshTaHwClients

Syntax: INTEGER(1..32767)
Access: read-only
Description: High water number of clients accessing application through this WSH.

tuxTwshTaMultiplex

Syntax: INTEGER(1..32767)
Access: read-only
Description: Maximum number of clients that can access the application through this WSH.

tuxTwshTaCurClients

Syntax: INTEGER(1..32767)
Access: read-only
Description: Current number of clients accessing the application through this WSH.
7 Workstation MIB

**tuxTwshTaTimeleft**

Syntax  INTEGER
Access  read-only
Description  A non-0 value for this attribute indicates that the WSH has been assigned a newly connecting workstation client that has the indicated amount of time, in seconds, to complete the initialization process with the WSH.

**tuxTwshTaActive**

Syntax  INTEGER { yes(1), no(2), unknown(3) }
Access  read-only
Description  A value of yes(1) indicates that the WSH is currently performing work on behalf of one of its associated workstation clients. A value of no(2) indicates that the WSH is currently waiting for work to perform on behalf of one of its associated workstation clients.

**tuxTwshTaTotacttime**

Syntax  INTEGER
Access  read-only
Description  Time, in seconds, that the WSH has been active since it started processing.

**tuxTwshTaTotidltime**

Syntax  INTEGER
Access  read-only
Description  Time, in seconds, that the WSH has been idle since it started processing.
### tuxTwshTaCurwork

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Amount of work processed by this WSH since the last WSH assignment by the WSL. This value is used by the WSL to load balance new incoming connections among a set of WSH processes.

### tuxTwshTaFlowcnt

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Number of times flow control has been encountered by this WSH. This attribute should be considered only in relation to recent past values because it might wrap around during the lifetime of the WSH.

### tuxTwshTaNumblockQ

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Number of times this WSH has been unable to enqueue a message to a local UNIX system message queue due to queue blocking conditions. This attribute should be considered only in relation to recent past values because it might wrap around during the lifetime of the WSH.

### tuxTwshTaRcvdByt

**Syntax**  
INTEGER

**Access**  
read-only

**Description**  
Number of bytes received from the network by this WSH from all its present and past workstation clients. This attribute should be considered only in relation to recent past values because it might wrap around during the lifetime of the WSH.
7 Workstation MIB

**tuxTwshTaRcvdNum**

Syntax: INTEGER

Access: read-only

Description: Number of Tuxedo System/T messages received from the network by this WSH from all its present and past workstation clients. This attribute should be considered only in relation to recent past values because it might wrap around during the lifetime of the WSH.

**tuxTwshTaSentByt**

Syntax: INTEGER

Access: read-only

Description: Number of bytes sent to the network by this WSH to all its present and past workstation clients. This attribute should be considered only in relation to recent past values because it might wrap around during the lifetime of the WSH.

**tuxTwshTaSentNum**

Syntax: INTEGER

Access: read-only

Description: Number of Tuxedo System/T messages sent to the network by this WSH to all its present and past workstation clients. This attribute should be considered only in relation to recent past values because it might wrap around during the lifetime of the WSH.
The `tuxTws1Tbl` table represents configuration and runtime attributes of WSL server processes configured to manage workstation groups. These attribute values identify and characterize workstation-specific configuration attributes for WSL `tuxTsrvrTbl` objects within the application. To create a new row in this table, use a SET request that specifies the values for at least `tuxTws1TaSrvGrp`, `tuxTws1TaSrvId`, and `tuxTws1TaNaddr`.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTws1TaSrvGrp</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.1</td>
</tr>
<tr>
<td><code>tuxTws1TaSrvId</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.2</td>
</tr>
<tr>
<td><code>tuxTws1TaGrpNo</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.3</td>
</tr>
<tr>
<td><code>tuxTws1TaState</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.4</td>
</tr>
<tr>
<td><code>tuxTws1TaLmid</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.5</td>
</tr>
<tr>
<td><code>tuxTws1TaPid</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.6</td>
</tr>
<tr>
<td><code>tuxTws1TaDevice</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.7</td>
</tr>
<tr>
<td><code>tuxTws1TaNaddr</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.8</td>
</tr>
<tr>
<td><code>tuxTws1TaWshName</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.9</td>
</tr>
<tr>
<td><code>tuxTws1TaMinHandlers</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.10</td>
</tr>
<tr>
<td><code>tuxTws1TaMaxHandlers</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.11</td>
</tr>
<tr>
<td><code>tuxTws1TaMultiplex</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.12</td>
</tr>
<tr>
<td><code>tuxTws1TaMaxIdleTime</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.13</td>
</tr>
<tr>
<td><code>tuxTws1TaMaxInitTime</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.14</td>
</tr>
<tr>
<td><code>tuxTws1TaClOpt</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.15</td>
</tr>
<tr>
<td><code>tuxTws1TaEnvFile</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.16</td>
</tr>
<tr>
<td><code>tuxTws1TaGrace</code></td>
<td>.1.3.6.1.4.1.140.300.1.2.1.17</td>
</tr>
</tbody>
</table>
Variable Name          Object ID
\[\text{tuxTws1TaMaxGen} .1.3.6.1.4.1.140.300.1.2.1.18\]
\[\text{tuxTws1TaRcmd} .1.3.6.1.4.1.140.300.1.2.1.19\]
\[\text{tuxTws1TaRestart} .1.3.6.1.4.1.140.300.1.2.1.20\]
\[\text{tuxTws1TaSequence} .1.3.6.1.4.1.140.300.1.2.1.21\]
\[\text{tuxTws1TaCurHandlers} .1.3.6.1.4.1.140.300.1.2.1.22\]
\[\text{tuxTws1TaHwHandlers} .1.3.6.1.4.1.140.300.1.2.1.23\]
\[\text{tuxTws1TaWsProto} .1.3.6.1.4.1.140.300.1.2.1.24\]
\[\text{tuxTws1TaSuspended} .1.3.6.1.4.1.140.300.1.2.1.25\]
\[\text{tuxTws1TaViewRefresh} .1.3.6.1.4.1.140.300.1.2.1.26\]
\[\text{tuxTws1TaKeepAlive} .1.3.6.1.4.1.140.300.1.2.1.28\]
\[\text{tuxTws1TaNetTimeout} .1.3.6.1.4.1.140.300.1.2.1.29\]

\textbf{tuxTws1TaSrvGrp}

\textbf{Syntax} \quad \text{DisplayString( SIZE(1..30) )}
\textbf{Access} \quad \text{read-write}
\textbf{Description} \quad \text{Logical name of the server group. Server group names cannot contain an asterisk (*), comma, or colon.}

\textbf{Note:} This object can be updated only during row creation.

\textbf{tuxTws1TaSrvId}

\textbf{Syntax} \quad \text{INTEGER (1..30001)}
\textbf{Access} \quad \text{read-write}
\textbf{Description} \quad \text{Unique (within the server group) server identification number.}

\textbf{Note:} This object can be updated only during row creation.
tuxTwsITaGrpNo
Syntax: INTEGER (1..30001)
Access: read-only
Description: Group number associated with this servers group.

tuxTwsITaState
Syntax: INTEGER { active(1), inactive(2), migrating(3), cleaning(4),
restarting(5), suspended(6), partitioned(7), dead(8), invalid(9) }
Access: read-write
Description: State for the WSL server within the application. Any state defined for the
tuxTsrvrTbl group can be returned or set as indicated.

tuxTwsITaLmid
Syntax: DisplayString (SIZE (1..30))
Access: read-only
Description: Current logical machine on which the server is running.

tuxTwsITaPid
Syntax: INTEGER
Access: read-only
Description: Native operating system process identifier for the WSL server.

tuxTwsITaDevice
Syntax: DisplayString (SIZE (0..78))
Access: read-write
Description: Device name to be used by the WSL process to access the network. This is a required
value for access to a network through a TLI-based Tuxedo System/T binary. This
attribute is not needed for sockets-based Tuxedo System/T binaries.
7 Workstation MIB

**tuxTwslTaNaddr**

Syntax:  

DisplayString (SIZE (1..78))

Access: read-write

Description: Specifies the complete network address to be used by the WSL process as its listening address. The listening address for a WSL is the means by which it is contacted by workstation client processes participating in the application.

If string has the form 0xhex-digits or \xhex-digits, it must contain an even number of valid hexadecimal digits. These forms are translated internally into a character array containing the hexadecimal representations of the string specified.

**tuxTwslTaWshName**

Syntax:  

DisplayString (SIZE (1..78))

Access: read-write

Description: The name of the executable that provides workstation handler services for this workstation listener. The default value for this is WSH, which corresponds to the system provided workstation handler. Workstation handlers can be customized using the command `buildwsh`.

**tuxTwslTaMinHandlers**

Syntax:  

INTEGER (0..256)

Access: read-write

Description: The minimum number of handlers that should be available in conjunction with this WSL at any given time. Upon being activated, the WSL starts this many WSHs immediately and does not deplete the supply of WSHs below this number until the administrator issues a shutdown to the WSL. Modifications to this attribute for a running WSL might cause additional handlers to be activated.
### tuxTwslTaMaxHandlers

**Syntax**  INTEGER \((0..32767)\)

**Access**  read-write

**Description**  The maximum number of handlers that should be available in conjunction with this WSL at any given time. Handlers are started as necessary to meet the demand of workstation clients attempting to access the system. This attribute must be greater than or equal to the setting for the minimum number of handlers.

### tuxTwslTaMultiplex

**Syntax**  INTEGER \((0..32767)\)

**Access**  read-write

**Description**  Maximum number of clients that are supported by any one handler process concurrently.

### tuxTwslTaMaxIdleTime

**Syntax**  INTEGER

**Access**  read-write

**Description**  Maximum amount of time, in minutes, that a workstation client is permitted to be idle before it is abortively disconnected from the application by the handler. A value of 0 allows clients to be idle as long as is necessary without being timed out.

### tuxTwslTaMaxInitTime

**Syntax**  INTEGER

**Access**  read-write

**Description**  The minimum number of seconds that should be allowed for a workstation client to complete initialization processing through the WSH before being timed out by the WSL.
Workstation MIB

**tuxTwslTaClOpt**

**Syntax**
DisplayString(SIZE(0..128))

**Access**
read-write

**Description**
Command-line options to be passed to the WSL server when it is activated. See the servopts(5) reference page for details.

**Note:** Runtime modifications to this attribute do not affect a running WSL server. Server-specific options (that is, those after a double-dash “--”) cannot be set and are not returned.

**tuxTwslTaEnvFile**

**Syntax**
DisplayString(SIZE(0..78))

**Access**
read-write

**Description**
WSL server-specific environment file. See tuxTmachineEnvFile for a complete discussion of how this file is used to modify the environment.

**Note:** Runtime modifications to this attribute do not affect a running WSL server.

**tuxTwslTaGrace**

**Syntax**
INTEGER

**Access**
read-write

**Description**
The period of time, in seconds, during which the tuxTwslTaMaxGen limit applies. This attribute is meaningful only for restartable WSL servers, that is, if the tuxTwslTaRestart attribute is set to yes(1). When a restarting server would exceed the tuxTwslTaMaxGen limit but the tuxTwslTaGrace period has expired, the system resets the current generation (tuxTsrvrGeneration) to 1 and resets the initial boot time (tuxTsrvrTimeStart) to the current time. A value of 0 for this attribute indicates that the WSL server should always be restarted.
tuxTwslTaMaxGen

Syntax: INTEGER (0..256)
Access: read-write
Description: Number of generations allowed for a restartable WSL server (tuxTwslTaRestart == yes(1)) over the specified grace period (tuxTwslTaGrace). The initial activation of the WSL server counts as one generation and each restart also counts as one. Processing after the maximum number of generations is exceeded is discussed above with respect to tuxTwslTaGrace.

tuxTwslTaRcmd

Syntax: DisplayString (SIZE(0..78))
Access: read-write
Description: Application specified command to be executed in parallel with the system restart of an application server. This command must be an executable file in the native operating system.

tuxTwslTaRestart

Syntax: INTEGER { yes(1), no(2) }
Access: read-write
Description: Restartable (yes(1)) or non-restartable (no(2)) WSL server. If server migration is specified for this server group (tuxTdomainOptions = migrate(2) and tuxTgroupLMID with alternate site), then this attribute must be set to yes(1).

tuxTwslTaSequence

Syntax: INTEGER (1..10000)
Access: read-write
Description: Specifies when this server should be booted (tmboot(1)) or shutdown (tmshutdown(1)) relative to other servers. If two servers are given the same sequence number, it is possible for tmboot(1) to boot them in parallel and for tmshutdown(1) to shut them down in parallel. tuxTwslTbl instances added without a
The `tuxTwslTaSequence` attribute specified or with an invalid value have a value generated for them that is 10,000 or more and is higher than any other automatically selected default value. Servers are booted by `tmboot(1)` in increasing order of sequence number and shutdown by `tmshutdown(1)` in decreasing order. Runtime modifications to this attribute affect only `tmboot(1)` and `tmshutdown(1)` and affect the order in which running servers can be shutdown by a subsequent invocation of `tmshutdown(1)`.

### `tuxTwslTaCurHandlers`

**Syntax**  
`INTEGER`

**Access**  
read-only

**Description**  
Number of currently active handlers associated with this WSL.

### `tuxTwslTaHwHandlers`

**Syntax**  
`INTEGER`

**Access**  
read-only

**Description**  
Maximum number of currently active handlers associated with this WSL at any one time.

### `tuxTwslTaWsProto`

**Syntax**  
`INTEGER`

**Access**  
read-only

**Description**  
The Tuxedo System/T/WS protocol version number for this/WS group. Note that /WS clients connecting to this group might themselves have a different protocol version number associated with them.

### `tuxTwslTaSuspended`

**Syntax**  
`INTEGER { new(1), all(2), none(3) }

**Access**  
read-write
<table>
<thead>
<tr>
<th><strong>tuxTwsITbl</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
</tr>
</tbody>
</table>

**tuxTwsI_VaViewRefresh**

| **Syntax** | INTEGER { yes(1), no-value Returned(2) } |
| **Access** | read-write |
| **Description** | Setting a value of yes(1) causes all active WSHs in the /WS group to refresh their VIEW buffer type cache. A GET request on this object always returns no-value Returned(2) and does not mean anything. This object has meaning only for SET requests. |

**tuxTwsI_KeepAlive**

| **Syntax** | INTEGER { client(1), handler(2), both(3), none(4), not-available(5) } |
| **Access** | read-write |
| **Description** | The network “keep alive” option is configured for the client, the handler, or both the client and the handler, or not on either side of the connection. Changing this value only affects future connections. This object is supported only on Tuxedo 6.4 or later. |

**tuxTwsI_NetTimeOut**

| **Syntax** | INTEGER (0..35204650) |
| **Access** | read-write |
| **Description** | The minimum number of seconds that should be allowed for a workstation client to wait for a response from WSL/WSH. A value of 0 indicates no network time-out. Changing this value affects only future connections. This object is supported only on Tuxedo 6.4. -1 is returned if the object is not available. |
The Tuxedo and WLE systems incorporate the capability to use application queues for time-independent communication. The Tuxedo and WLE Application Queue MIB provides the administrative environment required for managing and controlling access to application queues. The Application Queue MIB defines the structure of the application queues.

In Tuxedo and WLE applications, messages are stored on a queue, and queues are defined within a particular queue space. Queueing and dequeuing is done within a transaction. The Application Queue MIB consists of five different groups for defining queue access, queues, messages, queue spaces, and queue transactions. The following table lists the groups for managing each of the queue components.

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTAppQctrl</td>
<td>Access control to application queues</td>
</tr>
<tr>
<td>tuxTAppQTbl</td>
<td>Application queues within a queue space</td>
</tr>
<tr>
<td>tuxTAppQmsgTbl</td>
<td>Messages within an application queue</td>
</tr>
<tr>
<td>tuxTQspaceTbl</td>
<td>Application queue spaces</td>
</tr>
<tr>
<td>tuxTQtransTbl</td>
<td>Transactions associated with application queues</td>
</tr>
</tbody>
</table>
This is a control MIB to enable controlled access to all Application Queue related MIB groups.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTAppQctrlLmid</td>
<td>.1.3.6.1.4.1.140.300.12.5.1</td>
</tr>
<tr>
<td>tuxTAppQctrlQmConfig</td>
<td>.1.3.6.1.4.1.140.300.12.5.2</td>
</tr>
<tr>
<td>tuxTAppQctrlSpaceName</td>
<td>.1.3.6.1.4.1.140.300.12.5.3</td>
</tr>
<tr>
<td>tuxTAppQctrlQname</td>
<td>.1.3.6.1.4.1.140.300.12.5.4</td>
</tr>
<tr>
<td>tuxTAppQctrlMsgLoPrio</td>
<td>.1.3.6.1.4.1.140.300.12.5.5</td>
</tr>
<tr>
<td>tuxTAppQctrlMsgHiPrio</td>
<td>.1.3.6.1.4.1.140.300.12.5.6</td>
</tr>
<tr>
<td>tuxTAppQctrlMsgEndTime</td>
<td>.1.3.6.1.4.1.140.300.12.5.7</td>
</tr>
<tr>
<td>tuxTAppQctrlMsgStartTime</td>
<td>.1.3.6.1.4.1.140.300.12.5.8</td>
</tr>
<tr>
<td>tuxTAppQctrlMsgExpireEndTime</td>
<td>.1.3.6.1.4.1.140.300.12.5.20</td>
</tr>
<tr>
<td>tuxTAppQctrlMsgExpireStartTime</td>
<td>.1.3.6.1.4.1.140.300.12.5.30</td>
</tr>
</tbody>
</table>

**tuxTAppQctrlLmid**

**Syntax**
INTEGER { local(1), all(2) }

**Access**
read-write

**Description**
This applies to all Application Queue related MIB groups. The value of this object controls the machines for which the values are returned.

If the value is `local(1)`, only the local host where the BEA SNMP Agent is running is considered; alternatively, all LMIDs known to the application are considered if the value is `all(2)`.
The default value of this object is local(1).

**tuxTAppQctrlQmConfig**

**Syntax**  
*DisplayString (SIZE(1..78))*

**Access**  
read-write

**Description**  
This applies to all Application Queue related MIB groups. The value of this object controls the device for which the values are returned.

The default value of this object is “*”, in which case all known devices (which are a part of some group) are considered.

**tuxTAppQctrlSpaceName**

**Syntax**  
*DisplayString (SIZE(1..15))*

**Access**  
read-write

**Description**  
This applies to all Application Queue related MIB groups. The value of this object controls the queue space for which the values are returned.

The default value of this object is “*”, in which case all queue spaces for the devices (qualified by tuxTAppQctrlQmConfig) are considered.

**tuxTAppQctrlQname**

**Syntax**  
*DisplayString (SIZE(1..15))*

**Access**  
read-write

**Description**  
The value of this object controls the queue for which the values are returned. This applies to tuxTAppQTbl and tuxTAppQmsgTbl.

The default value of this object is “*”, in which case all queues for the devices (qualified by tuxTAppQctrlQmConfig) and queue spaces (qualified by tuxTAppQctrlSpaceName) are considered.
tuxTAppQctrlMsgLoPrio

Syntax: INTEGER
Access: read-write
Description: This object applies only to tuxTAppQmsgTbl. The lowest priority within which to search for occurrences of tuxTAppQmsgTbl instances. This is valid only for PRIO-based queues. By default, the minimum value of priority is considered. To revert to the default setting, set this object to 0.


tuxTAppQctrlMsgHiPrio

Syntax: INTEGER
Access: read-write
Description: This object applies only to tuxTAppQmsgTbl. The highest priority within which to search for occurrences of tuxTAppQmsgTbl instances. This is valid only for PRIO-based queues. By default, the maximum value of priority is considered. To revert to the default setting, set this object to 0.


tuxTAppQctrlMsgEndTime

Syntax: DisplayString(SIZE(1..15))
Access: read-write
Description: This object applies only to tuxTAppQmsgTbl. The end time within which to search for occurrences of tuxTAppQmsgTbl instances. The range is inclusive. This is valid only for TIME-based queues. The default value is the maximum number possible on that machine. To use the default setting, set this object to ".*

YY[MM[DD[hh[mm[ss]]]]]

Specifies the year, month, date, hour, minute, and second respectively. Any value which is not specified defaults to its minimum value (e.g., 9506 is taken as 950601000000). The years 00 through 37 are treated as 2000 through 2037, 70 through 99 as 1970 through 1999, and 38 through 69 are invalid.
**tuxTAppQctrlMsgStartTime**

Syntax: \(\text{DisplayString}(\text{SIZE}(1..15))\)

Access: read-write

Description: This object applies only to tuxTAppQmsgTbl. The start time within which to search for occurrences of tuxTAppQmsgTbl instances. The range is inclusive. This is valid only for TIME-based queues. By default, the minimum time value is considered to be 0. To use the default setting, set this object to "*".

YY[MM[DD[hh[mm[ss]]]]]

Specifies the year, month, date, hour, minute, and second respectively. Any value which is not specified defaults to its minimum value (e.g., 9506 is taken as 950601000000). The years 00 through 37 are treated as 2000 through 2037, 70 through 99 as 1970 through 1999, and 38 through 69 are invalid.

**tuxTAppQctrlMsgExpireEndTime**

Syntax: \(\text{DisplayString}(\text{SIZE}(1..12))\)

Access: read-write

Description: This object applies only to tuxTAppQmsgTbl. The expire end time within which to search for occurrences of tuxTAppQmsgTbl instances. The range is inclusive. This is valid only for TIME-based queues. The default value is the maximum number possible on that machine. To use the default setting, set this object to "*".

YY[MM[DD[hh[mm[ss]]]]]

Specifies the year, month, date, hour, minute, and second respectively. Any value which is not specified defaults to its minimum value (e.g., 9506 is taken as 950601000000). The years 00 through 37 are treated as 2000 through 2037, 70 through 99 as 1970 through 1999, and 38 through 69 are invalid.

**tuxTAppQctrlMsgExpireStartTime**

Syntax: \(\text{DisplayString}(\text{SIZE}(1..12))\)

Access: read-write
Description

This object applies only to `tuxTAppQmsgTb1`. The Expire start time within which to search for occurrences of `tuxTAppQmsgTb1` instances. The range is inclusive. This is valid only for TIME-based queues. By default, the minimum time value is considered to be 0. To use the default setting, set this object to “*”.

YY[MM[DD[hh[mm[ss]]]]]

Specifies the year, month, date, hour, minute, and second respectively. Any value which is not specified defaults to its minimum value (e.g., 9506 is taken as 950601000000). The years 00 through 37 are treated as 2000 through 2037, 70 through 99 as 1970 through 1999, and 38 through 69 are invalid.
The `tuxTAppQTbl` group represents application queues. One or more application queues can exist in a single application queue space. Objects in this table are only accessible through a BEA SNMP Agent installed on the local machine.

Creation of a New Queue — For creating a new queue(row), in this group the SET request should have the value of `tuxTAppQname`, `tuxTAppQspaceName`, and `tuxTAppQmConfig`. Also the value of `tuxTAppQgrpNo` (which is a part of the index) should be the corresponding group number for that queue space or "40000" (if no such group exists).

**Note:** For this and all other Application Queue related MIB groups there is a control MIB which can be used to filter the data returned as a part of all Application Queue related MIB groups. Please refer to `tuxTAppQctrl`.

To create a new row in this table, issue a SET request that specifies at least the values for `tuxTAppQname`, `tuxTAppQspaceName`, and `tuxTAppQmConfig`.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTAppQname</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.1</td>
</tr>
<tr>
<td><code>tuxTAppQspaceName</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.2</td>
</tr>
<tr>
<td><code>tuxTAppQmConfig</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.3</td>
</tr>
<tr>
<td><code>tuxTAppQlmid</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.4</td>
</tr>
<tr>
<td><code>tuxTAppQgrpNo</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.5</td>
</tr>
<tr>
<td><code>tuxTAppQstate</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.6</td>
</tr>
<tr>
<td><code>tuxTAppQorder</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.7</td>
</tr>
<tr>
<td><code>tuxTAppQcmd</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.8</td>
</tr>
<tr>
<td><code>tuxTAppQcmdHw</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.9</td>
</tr>
<tr>
<td><code>tuxTAppQcmdLw</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.10</td>
</tr>
<tr>
<td><code>tuxTAppQmaxRetries</code></td>
<td>.1.3.6.1.4.1.140.300.12.1.1.11</td>
</tr>
</tbody>
</table>
### Application Queue MIB

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTAppQoutOfOrder</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.12</td>
</tr>
<tr>
<td>tuxTAppQretryDelay</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.13</td>
</tr>
<tr>
<td>tuxTAppQcurBlocks</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.14</td>
</tr>
<tr>
<td>tuxTAppQcurMsg</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.15</td>
</tr>
<tr>
<td>tuxTAppQDefExpirationTime</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.30</td>
</tr>
<tr>
<td>tuxTAppQDefDeliveryPolicy</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.40</td>
</tr>
<tr>
<td>tuxTAppQCmdNonPersist</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.50</td>
</tr>
<tr>
<td>tuxTAppQCmdNonPersistHw</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.60</td>
</tr>
<tr>
<td>tuxTAppQCmdNonPersistLw</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.70</td>
</tr>
<tr>
<td>tuxTAppQCurNonPersistBytes</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.80</td>
</tr>
<tr>
<td>tuxTAppQCurNonPersistMsg</td>
<td>.1.3.6.1.4.1.140.300.12.1.1.90</td>
</tr>
</tbody>
</table>

**tuxTAppQname**

**Syntax**  
`DisplayString(SIZE(1..15))`

**Access**  
read-write

**Description**  
Name of the application queue.

**Note:** This object can be updated only during row creation.

**tuxTAppQspaceName**

**Syntax**  
`DisplayString(SIZE(1..15))`

**Access**  
read-write

**Description**  
Name of the application queue space containing the application queue.

**Note:** This object can be updated only during row creation.

---

8-8  
BEA SNMP Agent MIB Reference
**tuxTAppQmConfig**

Syntax: `DisplayString(SIZE(1..78))`

Access: read-write

Description: Absolute pathname of the file or device where the application queue space is located.

**Note:** This object can be updated only during row creation.

**tuxTAppQmid**

Syntax: `DisplayString(SIZE(1..30))`

Access: read-write

Description: Identifier of the logical machine where the application queue space is located.

**Note:** This object can be updated only during row creation.

**tuxTAppQgrpNo**

Syntax: `INTEGER (1..29999)`

Access: read-write

Description: Group number of any server group for which this queue is a resource manager, in other words that group’s openinfo string `tuxTgroupOpenInfo` contains the device name and queue space name for this queue.

**Note:** This object can be updated only during row creation.

**tuxTAppQstate**

Syntax: `INTEGER { valid(1), invalid(2) }`

Access: read-write

Description: The values for GET and SET operations are as follows:

GET: valid(1)
A GET operation retrieves information about the selected application queues. The following list describes the meaning of the \texttt{tuxTAppQstate} attribute returned in response to a GET request. States not listed are not returned.

\texttt{valid(1)}

The specified queue exists.

\texttt{SET: invalid(2)}

A SET operation changes characteristics of the selected application queue or creates a new queue. The following list describes the meaning of the \texttt{tuxTAppQstate} attribute returned by a SET request. States not listed cannot be set.

\texttt{invalid(2)}

Delete the specified queue. If the queue space has processes attached to it, the queue is not deleted. In addition, if the queue has messages in it, it is not deleted. Successful return leaves the object in the \texttt{invalid(2)} state.

\textbf{tuxTAppQorder}

\begin{description}
\item Syntax \texttt{DisplayString(SIZE(1..30))}
\item Access read-write
\item Description The order in which messages in the queue are to be processed. Legal values are \texttt{PRIO} or \texttt{TIME}, followed by a comma, optionally followed by another occurrence of \texttt{PRIO} or \texttt{TIME}, followed by one of the values \texttt{LIFO} or \texttt{FIFO}. If neither \texttt{FIFO} nor \texttt{LIFO} is specified, \texttt{FIFO} is assumed. If nothing is specified when a queue is created, the default is \texttt{FIFO}. For example, these are some legal settings:
\begin{verbatim}
PRIO
PRIO,TIME,LIFO
TIME,PRIO,FIFO
TIME,FIFO
\end{verbatim}
\end{description}

\textbf{tuxTAppQcmd}

\begin{description}
\item Syntax \texttt{DisplayString(SIZE(0..78))}
\item Access read-write
\end{description}
| Description | The command to be automatically executed when the high water mark, tuxTAppQcmdHw, is reached. The command is re-executed when the high water mark is reached again after the low water mark, tuxTAppQcmdLw, has been reached. |

**tuxTAppQcmdHw**

- **Syntax**: DisplayString
- **Access**: read-write
- **Description**: The high water mark. Refer to tuxTAppQcmdLw for further information.

**tuxTAppQcmdLw**

- **Syntax**: DisplayString
- **Access**: read-write
- **Description**: The low water marks that control the automatic execution of the command specified in the tuxTAppQcmd attribute. Each is an integer greater than or equal to zero optionally followed by one of the following keyletters. The keyletters must be consistent for tuxTAppQcmdHw and tuxTAppQcmdLw.

  - **b**: The high and low water marks pertain to the number of bytes used by messages in the queue.
  - **B**: The high and low water marks pertain to the number of blocks used by messages in the queue.
  - **m**: The high and low water marks pertain to the number of messages in the queue.
  - **%**: The high and low water marks are expressed in terms of a percentage of queue capacity.
For example, if tuxtAppQcmdLw is 50m and tuxtAppQcmdHw is 100m, then the command specified in tuxtAppQcmd is executed when 100 messages are on the queue, and it is not executed again until the queue is drained below 50 messages and is filled again to 100 messages.

**tuxtAppQmaxRetries**

**Syntax**

INTEGER

**Access**

read-write

**Description**

The maximum number of retries for a failed queue message. When the number of retries is exhausted, the message is placed on the error queue of the associated application queue space. If there is no error queue, the message is dropped. The default is zero.

**tuxtAppQoutOfOrder**

**Syntax**

INTEGER { none(1), top(2), msgid(3) }

**Access**

read-write

**Description**

The way in which out-of-order message processing is to be handled. The default is none(1).

**tuxtAppQretryDelay**

**Syntax**

INTEGER

**Access**

read-write

**Description**

The delay, in seconds, between retries for a failed queue message. The default is zero.

**tuxtAppQcurBlocks**

**Syntax**

INTEGER

**Access**

read-only

**Description**

The number of disk pages currently consumed by the queue.
tuxTAppQcurMsg

Syntax  INTEGER
Access   read-only
Description  The number of messages currently in the queue.

tuxTAppQDefExpirationTime

Syntax  DisplayString
Access   read-write
Description  This attribute specifies an expiration time for messages enqueued with no explicit expiration time. The expiration time can be either a relative expiration time or none. The relative expiration time is determined by associating a fixed amount of time with a message after the message arrives at the queue manager process. When a message’s expiration time is reached and the message has not been dequeued or administratively deleted, all resources associated with the message are reclaimed by the system and statistics are updated. If a messages expires during a transaction, the expiration does not cause the transaction to fail. Messages that expire while being enqueued or dequeued within a transaction are removed from the queue when the transaction ends. There is no notification that the message has expired. If no default expiration time is specified for a queue, the message without an explicit expiration time does not expire. When the queue’s expiration time is modified, the expiration times of messages that were in the queue before the modification are not changed.

The format is +seconds, where seconds is the number of seconds allowed to lapse between the time that the queue manager successfully completes the operation and the time that the message is to expire. If seconds is set to zero (0), the message expires immediately.

The value of this attribute may also be set to the string “none.” The none string indicates that messages enqueued to the queue with no explicit expiration time do not expire. You can change the expiration time for messages already in a queue with the tuxTAppQmsgExpireTime object of the tuxTAppQmsgTbl group.
tuxTAppQDefDeliveryPolicy

Syntax
INTEGER { persist (1), non-persist (2) }

Access
read-write

Description
This attribute specifies the default delivery policy for the queue when no delivery mode is specified for a message enqueued to the queue. When the value is "persist," messages enqueued to the queue without an explicitly specified delivery mode are delivered using the persistent (disk-based) delivery method. When the value is non-persist, messages enqueued to the queue without an explicitly specified delivery method are delivered using the non-persistent (in memory) delivery method. When a queue’s default delivery policy is modified, the delivery quality of service of messages that are in the queue before the modification are not changed. If the queue being modified is the reply queue named for any messages currently in the queue space, the reply quality of service is not changed for those messages as a result of changing the default delivery policy of the queue.

For non-persistent delivery, if the memory area is exhausted or fragmented so that a message cannot be enqueued, the enqueuing operation fails, even if there is sufficient persistent storage for the message. Similarly, if the persistent storage area is exhausted or fragmented so that a message cannot be enqueued, the enqueuing operation fails, even if there is sufficient non-persistent storage for the message. If the tuxTQspaceMemNonPersist object of the tuxTQspaceTbl group is zero (0) for a queue space, no space is reserved for non-persistent messages. In such a case, any attempt to enqueue a non-persistent message fails. This type of failure results, for example, when no delivery quality of service has been specified for a message and the tuxTAppQDefDeliveryPolicy object for the target queue has been set to "non-persist."

tuxTAppQCmdNonPersist

Syntax
DisplayString {Size(0..78)}

Access
read-write

Description
This attribute specifies the command to be executed automatically when the high-water mark for non-persistent (memory-based delivery) messages, tuxTAppQCmdNonPersistHw, is reached. The command is re-executed when the high-water mark is reached again after the low-water mark for non-persistent (memory-based delivery) messages, tuxTAppQCmdNonPersistLw, has been reached.
### tuxTAppQCmdNonPersistHw

**Syntax**  DisplayString  
**Access**  read-write  
**Description**  These attributes specify the high- and low-water marks that control the automatic execution of the command specified in the `tuxTAppQCmdNonPersist` attribute. Each is an integer greater than or equal to zero, followed by one of the following keyletters. The keyletters must be consistent for `tuxTAppQCmdNonPersistHw` and `tuxTAppQCmdNonPersistLw`.

- **b**  
  The high- and low-water marks are expressed as the number of bytes used by non-persistent (in-memory) messages in the queue.

- **B**  
  The high- and low-water marks are expressed as the number of blocks used by non-persistent (in-memory) messages in the queue.

- **%**  
  The high- and low-water marks are expressed as a percentage of the shared memory capacity reserved for non-persistent messages in the queue space used by the queue.

The messages threshold type specified through the `tuxTAppQCmdHw` and `tuxTAppQCmdLw` attributes (when followed by an m) applies to all messages in a queue, including both persistent and non-persistent messages, and therefore is not available as a threshold type for `tuxTAppQCmdNonPersistHw` and `tuxTAppQCmdNonPersistLw`.

### tuxTAppQCmdNonPersistLw

**Syntax**  DisplayString  
**Access**  read-write  
**Description**  These attributes specify the high- and low-water marks that control the automatic execution of the command specified in the `tuxTAppQCmdNonPersist` attribute. Each is an integer greater than or equal to zero, followed by one of the following keyletters. The keyletters must be consistent for `tuxTAppQCmdNonPersistHw` and `tuxTAppQCmdNonPersistLw`.
The high- and low-water marks are expressed as the number of bytes used by non-persistent (in-memory) messages in the queue.

The high- and low-water marks are expressed as the number of blocks used by non-persistent (in-memory) messages in the queue.

The high- and low-water marks are expressed as a percentage of the shared memory capacity reserved for non-persistent messages in the queue space used by the queue.

The messages threshold type specified through the tuxTAppQCmdHw and tuxTAppQcmdLw attributes (when followed by an m) applies to all messages in a queue, including both persistent and non-persistent messages, and therefore is not available as a threshold type for tuxTAppQCmdNonPersistHw and tuxTAppQCmdNonPersistLw.

**tuxTAppQCurNonPersistBytes**

- **Syntax**: Integer
- **Access**: read-write
- **Description**: This attribute specifies the number of shared memory bytes currently consumed by the non-persistent messages on the queue.

**tuxTAppQCurNonPersistMsg**

- **Syntax**: Integer
- **Access**: read-write
- **Description**: This attribute specifies the number of non-persistent messages currently in the queue. To determine the total number of messages in the queue, add the value of tuxTAppQcurMsg to this value.
The `tuxTAppQmsgTbl` group represents messages stored in application queues. A message is not created by an administrator; instead, it comes into existence as a result of a call to `tpenqueue(3)`. A message can be destroyed either by a call to `tpdequeue(3)` or by an administrator. In addition, certain attributes of a message can be modified by an administrator. For example, an administrator can move a message from one queue to another queue within the same queue space or change its priority.

Objects in this table are only accessible through a BEA SNMP Agent installed on the local machine.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTAppQmsgId</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.1</td>
</tr>
<tr>
<td><code>tuxTAppQmsgSerNo</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.2</td>
</tr>
<tr>
<td><code>tuxTAppQmsgGrpNo</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.3</td>
</tr>
<tr>
<td><code>tuxTAppQmsgQname</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.4</td>
</tr>
<tr>
<td><code>tuxTAppQmsgQmConfig</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.5</td>
</tr>
<tr>
<td><code>tuxTAppQmsgQspaceName</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.6</td>
</tr>
<tr>
<td><code>tuxTAppQmsgLmid</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.7</td>
</tr>
<tr>
<td><code>tuxTAppQmsgState</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.8</td>
</tr>
<tr>
<td><code>tuxTAppQmsgNewQname</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.9</td>
</tr>
<tr>
<td><code>tuxTAppQmsgPrior</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.10</td>
</tr>
<tr>
<td><code>tuxTAppQmsgTime</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.11</td>
</tr>
<tr>
<td><code>tuxTAppQmsgCorId</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.12</td>
</tr>
<tr>
<td><code>tuxTAppQmsgCurRetries</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.13</td>
</tr>
<tr>
<td><code>tuxTAppQmsgSize</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.14</td>
</tr>
<tr>
<td><code>tuxTAppQmsgExpireTime</code></td>
<td>.1.3.6.1.4.1.140.300.12.2.1.20</td>
</tr>
</tbody>
</table>
### Application Queue MIB

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTAppQmsgPersistent</td>
<td>.1.3.6.1.4.1.140.300.12.2.1.30</td>
</tr>
<tr>
<td>tuxTAppQmsgReplyPersistent</td>
<td>.1.3.6.1.4.1.140.300.12.2.1.40</td>
</tr>
</tbody>
</table>

#### tuxTAppQmsgId

- **Syntax**: DisplayString(SIZE(1..32))
- **Access**: read-only
- **Description**: A unique identifier for the queue message, which can be used to select the message for GET or SET operations. No significance should be placed on this value beyond using it for equality comparisons.

#### tuxTAppQmsgSerNo

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: A running number corresponding to tuxTAppQmsgId for the queue message, which is a part of the composite index of this table.

#### tuxTAppQmsgGrpNo

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: Group number of any server group for which this queue is a resource manager, in other words that group’s openinfo string tuxTgroupOpenInfo contains the device name and queue space name for this queue.

#### tuxTAppQmsgQname

- **Syntax**: DisplayString(SIZE(1..15))
- **Access**: read-only
- **Description**: Name of the application queue in which the message is stored.
**tuxTAppQmsgQmConfig**

- **Syntax**: `DisplayString(SIZE(1..78))`
- **Access**: read-only
- **Description**: Absolute pathname of the file or device where the application queue space for the queue containing this message is located.

**tuxTAppQmsgQspaceName**

- **Syntax**: `DisplayString(SIZE(1..15))`
- **Access**: read-only
- **Description**: Name of the application queue space containing the application queue in which this message is located.

**tuxTAppQmsgLmid**

- **Syntax**: `DisplayString(SIZE(1..30))`
- **Access**: read-only
- **Description**: Logical machine id for the machine on which the queue containing this message is located.

**tuxTAppQmsgState**

- **Syntax**: `INTEGER { valid(1), invalid(2) }`
- **Access**: read-write
- **Description**: The values for GET and SET operations are as follows:
  
  **GET**: valid(1)
  
  A GET operation retrieves information about the selected messages. The following list describes the meaning of the `tuxTAppQmsgState` attribute returned in response to a GET request. States not listed are not returned.

  **valid(1)**
  
  The message exists.
A SET operation changes characteristics of the selected message. The following list describes the meaning of the tuxTAppQmsgState attribute returned by a SET request. States not listed cannot be set.

**invalid(2)**

The message is deleted from its queue space. The message must be in state valid(1) before attempting this operation. Successful return leaves the object in the invalid(2) state.

### tuxTAppQmsgNewQname

**Syntax**

DisplayString (SIZE(1..15))

**Access**

read-write

**Description**

Name of the queue into which to move the selected message. This queue must be an existing queue in the same queue space. The message must be in state valid(1) for this operation to succeed. This attribute is not returned by a GET operation.

### tuxTAppQmsgPrior

**Syntax**

INTEGER

**Access**

read-write

**Description**

The priority of the message. This attribute is valid only for PRIO-based queues. The value -1 is returned by a GET operation if the queue is not PRIO-based.

### tuxTAppQmsgTime

**Syntax**

DisplayString (SIZE(1..15))

**Access**

read-write

**Description**

The time when the message is processed. This attribute is valid only for TIME-based queues. The empty string is returned by a GET operation if the queue is not TIME-based. The format is one of the following:

+seconds
Specifies that the message is processed \textit{seconds} in the future. The value zero specifies that the message should be processed immediately.

\texttt{YY[MM[DD[hh[mm[ss]]]]]}

Specifies the year, month, day, hour, minute, and second when the message should be processed. Omitted units default to their minimum possible values. For example, 9506 is equivalent to 950601000000. The years 00 through 37 are treated as 2000 through 2037, 70 through 99 are treated as 1970 through 1999, and 38 through 69 are invalid.

\textbf{tuxTAppQmsgCorId}

\begin{itemize}
\item \textbf{Syntax} \texttt{DisplayString(SIZE(0..32))}
\item \textbf{Access} read-only
\item \textbf{Description} The correlation identifier for this message provided by the application in the \texttt{tpenqueue(3)} request. The empty string indicates that a correlation identifier is not present.
\end{itemize}

\textbf{tuxTAppQmsgCurRetries}

\begin{itemize}
\item \textbf{Syntax} \texttt{INTEGER}
\item \textbf{Access} read-only
\item \textbf{Description} The number of retries that have been attempted so far on this message.
\end{itemize}

\textbf{tuxTAppQmsgSize}

\begin{itemize}
\item \textbf{Syntax} \texttt{INTEGER}
\item \textbf{Access} read-only
\item \textbf{Description} The size of the message, in bytes.
\end{itemize}

\textbf{tuxTAppQmsgExpireTime}

\begin{itemize}
\item \textbf{Syntax} \texttt{DisplayString(SIZE(1..15))}
\item \textbf{Access} read-write
\end{itemize}
This attribute specifies the time at which a message expires (that is, the time at which
the message should be removed from the queue if it has not already been dequeued or
administratively deleted). When a message expires, all resources it uses are reclaimed
by the system and statistics are updated. If a message expires during a transaction, the
expiration does not cause the transaction to fail. Messages that expire while being
enqueued or dequeued within a transaction are removed from the queue when the
transaction ends. There is no notification that the message has expired. Expiration
times cannot be added to messages enqueued by versions of the BEA Tuxedo system
that do not support message expiration, even when the queue manager responsible for
changing this value supports message expiration. Attempts to add an expiration time
fail.

The empty string is returned by a GET operation if the expiration time is not set. The
expiration time format is one of the following:

+seconds

Specifies that the message will be removed after the specified number of
seconds. If the value of seconds is set to zero (0), the message is removed
immediately from the queue. Relative expiration time is calculated on the
basis of the time at which the MIB request arrives and has been processed by
the corresponding queue manager.

YY[MM[DD[hh][mm][ss]]]

Specifies the year, month, day, hour, minute, and second when the message
will be removed if it has not already been dequeued or administratively
deleted. Omitted units default to their minimum possible values. For
example, 9506 is equivalent to 950601000000. The years 00 through 37 are
treated as 2000 through 2037, 70 through 99 are treated as 1970 through
1999, and 38 through 69 are invalid. An absolute expiration time is
determined by the clock on the machine where the queue manager process
resides.

none

Specifies that the message will never expire.

Syntax INTEGER { yes (1), no (2) }

Access read-only
**tuxTAppQmsgTbl**

**tuxTAppQmsgReplyPersistent**

- **Syntax**: INTEGER { yes (1), no (2) }
- **Access**: read-only
- **Description**: This read-only state is set to "no" for non-persistent messages and "yes" for persistent messages. It is the delivery quality that replies to the message.
This group represents application queue spaces. Objects in this table are only accessible through a BEA SNMP Agent installed on the local machine.

**Note:** The values returned by this MIB are controlled by `tuxTAppQctrl`. Refer to the description of the above group for details.

To create a new row in this table, a `SET` request should be issued with an index (`tuxTQspaceGrpNo`) of 40000. This is a reserved value for row creation in the table. The `SET` request also needs to specify values for at least `tuxTQspaceQmConfig`, `tuxTQspaceName`, `tuxTQspaceLmid`, `tuxTQspaceIpckey`, `tuxTQspaceMaxMsg`, `tuxTQspaceMaxPages`, `tuxTQspaceMaxProc`, `tuxTQspaceMaxQueues`, and `tuxTQspaceMaxTrans`. The newly created instance (row) is not visible until it is attached to some server group.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tuxTQspaceName</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.1</td>
</tr>
<tr>
<td><code>tuxTQspaceQmConfig</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.2</td>
</tr>
<tr>
<td><code>tuxTQspaceLmid</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.3</td>
</tr>
<tr>
<td><code>tuxTQspaceGrpNo</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.4</td>
</tr>
<tr>
<td><code>tuxTQspaceState</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.5</td>
</tr>
<tr>
<td><code>tuxTQspaceBlocking</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.6</td>
</tr>
<tr>
<td><code>tuxTQspaceErrQname</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.7</td>
</tr>
<tr>
<td><code>tuxTQspaceForceInit</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.8</td>
</tr>
<tr>
<td><code>tuxTQspaceIpckey</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.9</td>
</tr>
<tr>
<td><code>tuxTQspaceMaxMsg</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.10</td>
</tr>
<tr>
<td><code>tuxTQspaceMaxPages</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.11</td>
</tr>
<tr>
<td><code>tuxTQspaceMaxProc</code></td>
<td>.1.3.6.1.4.1.140.300.12.3.1.12</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Object ID</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>tuxTQspaceMaxQueues</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.13</td>
</tr>
<tr>
<td>tuxTQspaceMaxTrans</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.14</td>
</tr>
<tr>
<td>tuxTQspaceCurExtent</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.15</td>
</tr>
<tr>
<td>tuxTQspaceCurMsg</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.16</td>
</tr>
<tr>
<td>tuxTQspaceCurProc</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.17</td>
</tr>
<tr>
<td>tuxTQspaceCurQueues</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.18</td>
</tr>
<tr>
<td>tuxTQspaceCurTrans</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.19</td>
</tr>
<tr>
<td>tuxTQspaceHwMsg</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.20</td>
</tr>
<tr>
<td>tuxTQspaceHwProc</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.21</td>
</tr>
<tr>
<td>tuxTQspaceHwQueues</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.22</td>
</tr>
<tr>
<td>tuxTQspaceHwTrans</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.23</td>
</tr>
<tr>
<td>tuxTQspacePercentInit</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.24</td>
</tr>
<tr>
<td>tuxTQspaceMaxActions</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.40</td>
</tr>
<tr>
<td>tuxTQspaceMaxHandles</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.50</td>
</tr>
<tr>
<td>tuxTQspaceMaxOwners</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.60</td>
</tr>
<tr>
<td>tuxTQspaceMaxTmpQueues</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.70</td>
</tr>
<tr>
<td>tuxTQspaceMaxCursors</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.80</td>
</tr>
<tr>
<td>tuxTQspaceMemNonPersist</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.90</td>
</tr>
<tr>
<td>tuxTQspaceMemFilters</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.100</td>
</tr>
<tr>
<td>tuxTQspaceMemOverFlow</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.110</td>
</tr>
<tr>
<td>tuxTQspaceMemSystemReserved</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.120</td>
</tr>
<tr>
<td>tuxTQspaceMemTotalAllocated</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.130</td>
</tr>
<tr>
<td>tuxTQspaceCurActions</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.140</td>
</tr>
</tbody>
</table>
### tuxTQspaceName

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTQspaceCurHandles</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.150</td>
</tr>
<tr>
<td>tuxTQspaceCurOwners</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.160</td>
</tr>
<tr>
<td>tuxTQspaceCurTmpQueues</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.170</td>
</tr>
<tr>
<td>tuxTQspaceCurCursors</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.180</td>
</tr>
<tr>
<td>tuxTQspaceCurMemNonPersist</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.190</td>
</tr>
<tr>
<td>tuxTQspaceCurMemFilters</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.200</td>
</tr>
<tr>
<td>tuxTQspaceCurMemOverFlow</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.210</td>
</tr>
<tr>
<td>tuxTQspaceHwActions</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.220</td>
</tr>
<tr>
<td>tuxTQspaceHwHandles</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.230</td>
</tr>
<tr>
<td>tuxTQspaceHwOwners</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.240</td>
</tr>
<tr>
<td>tuxTQspaceHwTmpQueues</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.250</td>
</tr>
<tr>
<td>tuxTQspaceHwCursors</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.260</td>
</tr>
<tr>
<td>tuxTQspaceHwMemNonPersist</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.270</td>
</tr>
<tr>
<td>tuxTQspaceHwMemFilters</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.280</td>
</tr>
<tr>
<td>tuxTQspaceHwMemOverFlow</td>
<td>.1.3.6.1.4.1.140.300.12.3.1.290</td>
</tr>
</tbody>
</table>

**tuxTQspaceName**

- **Syntax**: `DisplayString(SIZE(1..15))`
- **Access**: read-write
- **Description**: Name of the application queue space.

**Note**: This object can be updated only during row creation.
**tuxTQspaceTbl**

<table>
<thead>
<tr>
<th>Access</th>
<th>read-write</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Absolute pathname of the file or device where the application queue space is located.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>This object can be updated only during row creation.</td>
</tr>
</tbody>
</table>

**tuxTQspaceLmid**

- **Syntax**: DisplayString(SIZE(1..30))
- **Access**: read-write
- **Description**: Identifier of the logical machine where the application queue space is located.
- **Note**: This object can be updated only during row creation.

**tuxTQspaceGrpNo**

- **Syntax**: INTEGER (1..29999)
- **Access**: read-write
- **Description**: Group number of any server group for which this queue space is a resource manager, in other words that group's openinfo string `tuxTgroupOpenInfo` contains the device name and queue space name for this queue space.
- **Note**: This object can be updated only during row creation.

**tuxTQspaceState**

- **Syntax**: INTEGER { inactive(1), initializing(2), open(3), active(4), cleaning(5), invalid(6) }
- **Access**: read-write
- **Description**: The values for GET and SET operations are as follows:

```
GET: inactive(1)|initializing(2)|open(3)|active(4)
```

A GET operation retrieves information about the selected application queue space. The following list describes the meaning of the `tuxTQspaceState` attribute returned in response to a GET request. States not listed are not returned.
inactive(1)

The queue space exists; i.e., disk space for it has been reserved in a device and the space has been initialized (if requested or if necessary).

initializing(2)

Disk space for the queue space is currently being initialized.

open(3)

Shared memory and other IPC resources for the queue space have been allocated and initialized, but no processes are currently attached to the shared memory.

active(4)

Shared memory and other IPC resources for the queue space have been allocated and initialized, and at least one process is currently attached to the shared memory. These processes can be the queue servers (TMS_QM, TMQUEUE, and perhaps TMQFORWARD) associated with the queue space, or they can be administrative processes such as qmadmin(1), or they can be processes associated with another application.

SET: open(3)|cleaning(5)|invalid(6)

A SET operation changes the selected application queue space or creates a new one. The following list describes the meaning of the tuxTQspaceState attribute returned by a SET request. States not listed cannot be set.

open(3)

Allocate and initialize shared memory and other IPC resources for the queue space. This is allowed only if the queue space is in the inactive(1) state.

cleaning(5)

Remove the shared memory and other IPC resources for the queue space. This is allowed only when the queue space is in the active(4) or open(3) state. Successful return leaves the object in the inactive(1) state.

invalid(6)

Delete the queue space. An error is reported if the state is active(4) or if messages exist on any queues in the queue space. Successful return leaves the object in the invalid(6) state.
**tuxTQspaceBlocking**

**Syntax** INTEGER

**Access** read-write

**Description** The blocking factor used for disk space management of the queue space. The default when a new queue space is created is 16.

**tuxTQspaceErrQname**

**Syntax** DisplayString (SIZE(0..15))

**Access** read-write

**Description** Name of the error queue associated with the queue space. If there is no error queue, an empty string is returned by a GET request.

**tuxTQspaceForceInit**

**Syntax** INTEGER { yes(1), no(2) }

**Access** read-write

**Description** The value of this object determines whether or not to initialize disk pages on new extents for the queue space. The default is not to initialize. Depending on the device type (e.g., regular file or raw slice), initialization can occur even if not requested.

**tuxTQspaceIpckey**

**Syntax** INTEGER { 32769..262143 }

**Access** read-write

**Description** The IPC key used to access queue space shared memory.

**tuxTQspaceMaxMsg**

**Syntax** INTEGER

**Access** read-write

**Description** The maximum number of messages that the queue space can contain.
tuxTQspaceMaxPages

Syntax  INTEGER
Access   read-write
Description  The maximum number of disk pages for all queues in the queue space. Each time the tuxTQspaceMaxPages attribute is increased, a new extent is allocated (see tuxTQspaceCurExtent). It is not possible to decrease the number of pages by setting this attribute to a lower number; an error is reported in this case.

tuxTQspaceMaxProc

Syntax  INTEGER
Access   read-write
Description  The maximum number of processes that can attach to the queue space.

tuxTQspaceMaxQueues

Syntax  INTEGER
Access   read-write
Description  The maximum number of queues that the queue space can contain.

tuxTQspaceMaxTrans

Syntax  INTEGER
Access   read-write
Description  The maximum number of simultaneously active transactions allowed by the queue space.
**tuxTQspaceCurExtent**

*Syntax*  
INTEGER

*Access*  
read-only

*Description*  
The current number of extents used by the queue space. The largest number allowed is 100. Each time the `tuxTQspaceMaxPages` attribute is increased, a new extent is allocated.

**tuxTQspaceCurMsg**

*Syntax*  
INTEGER

*Access*  
read-only

*Description*  
The current number of messages in the queue space. This number can be determined only if the queue space is `open(3)` or `active(4)`, or if the queue space is newly created. If none of the conditions apply, the value -1 is returned.

**tuxTQspaceCurProc**

*Syntax*  
INTEGER

*Access*  
read-only

*Description*  
The current number of processes accessing the queue space.

**tuxTQspaceCurQueues**

*Syntax*  
INTEGER

*Access*  
read-only

*Description*  
The current number of queues existing in the queue space. This number can be determined only if the queue space is `open(3)` or `active(4)`, or if the queue space is newly created. If none of these conditions apply, the value -1 is returned.
tuxTQspaceCurTrans
Syntax: INTEGER
Access: read-only
Description: The current number of outstanding transactions involving the queue space.

tuxTQspaceHwMsg
Syntax: INTEGER
Access: read-only
Description: The highest number of messages in the queue space since the queue space was last opened. The number is reset to 0 when the queue space state is set to cleaning(5).

tuxTQspaceHwProc
Syntax: INTEGER
Access: read-only
Description: The highest number of processes simultaneously attached to the queue space since the queue space was last opened. The number is reset to 0 when the queue space state is set to cleaning(5).

tuxTQspaceHwQueues
Syntax: INTEGER
Access: read-only
Description: The highest number of queues existing in the queue space since the queue space was last opened. The number is reset to 0 when the queue space state is set to cleaning(5).
**tuxTQspaceHwTrans**

**Syntax**  INTEGER

**Access**  read-only

**Description**  The highest number of outstanding transactions involving the queue space since the queue space was last opened. If the queue space is accessed by more than one application, this number reflects all applications — not just the application represented by the TUXCONFIG environment variable. The number is reset to 0 when the queue space state is set to `cleaning(5)`.

**tuxTQspacePercentInit**

**Syntax**  INTEGER (0..100)

**Access**  read-only

**Description**  The percentage (as an integer between 0 and 100 inclusive) of disk space that has been initialized for the queue space.

**tuxTQspaceMaxActions**

**Syntax**  INTEGER

**Access**  read-write

**Description**  This attribute specifies the number of additional actions that the Queuing Services component of the BEA Engine can handle concurrently. When a blocking operation is encountered and additional actions are available, the blocking operation is set aside until it can be satisfied. After setting aside the blocking operation, another operation request can be handled. When the blocking operation is completed, the action associated with the operation is made available for a subsequent operation. The system reserve actions are equivalent to the number of processes that can attach to a queue space, so that each queue manager process can have at least one blocking action. Beyond the system-reserved number of blocking actions, the administrator can configure the system to enable it to accommodate additional blocking actions beyond the reserve. An operation fails if a blocking operation is requested and cannot be immediately satisfied and there are no actions available.
**tuxTQspaceMaxHandles**

**Syntax**
INTEGER

**Access**
read-write

**Description**
This attribute specifies the number of handles that users of the Queueing Services component of the BEA Engine can use concurrently. Objects manipulated by the queuing services API require handles to access the objects. When an object is opened by a call to the Queuing Services API, a new handle is created and returned to the user. When an object handle is closed, the handle is made available for subsequent open object operations. When the Queuing Services API is used by an application, the administrator must configure the system to accommodate the maximum number of handles that are opened concurrently. An operation fails if a user attempts to open a queuing services object and there are no handles available. Adjusting this value has no effect on BEA Tuxedo applications other than unnecessarily consuming shared memory resources.

**tuxTQspaceMaxOwners**

**Syntax**
INTEGER

**Access**
read-write

**Description**
This attribute specifies the number of additional BEA Engine authenticated users that can concurrently use Queuing Services resources. There is one owner record per user, regardless of the number of open handles for the user. When there are no open handles for a user, the owner record is made available to subsequent users. The system reserves a number of owners equivalent to the number of actions, so that each action can be initiated by a different owner. Beyond the system-reserved number of owners that can concurrently use queueing services resources, the administrator can configure the system to accommodate additional owners beyond the reserved number. An operation fails if a user attempts to open a handle when there currently are no open handles, and there are no owners available. Adjusting this value has no effect on BEA Tuxedo applications other than unnecessarily consuming shared memory resources. Adjusting this value has no effect on BEA Tuxedo applications other than unnecessarily consuming shared memory resources.
### \texttt{tuxTQspaceMaxTmpQueues}

**Syntax**  
INTEGER

**Access**  
read-write

**Description**  
This attribute specifies the number of temporary queues that can be opened concurrently in the Queuing Services component of the BEA Engine. Temporary queues are used by dynamic, self-configuring applications and reduce the need for administrators to configure each queue used by an application. Messages enqueued to temporary queues are not persistent. When all handles to a temporary queue are closed, the temporary queue resources are made available for subsequent temporary queue creation. When the temporary queues are used by an application, the administrator must configure the system to accommodate the maximum number of temporary queues that are active concurrently. An open operation fails if a user attempts to open a temporary queue and there are no temporary queue resources available. This attribute specifies the number of additional BEA Engine authenticated users that can concurrently use Queuing Services.

### \texttt{tuxTQspaceMaxCursors}

**Syntax**  
INTEGER

**Access**  
read-write

**Description**  
This attribute specifies the number of cursors that user of the Queuing Services component of the BEA Engine can use concurrently. Cursors are used to navigate a queue. When a cursor is destroyed, the cursor resources are made available for subsequent cursor creation operations. When the cursors are used by an application, the administrator must configure the system to accommodate the maximum number of cursors that are allocated concurrently. An operation fails if a user attempts to create a cursor and there are no cursor resources available. This attribute specifies the number of additional BEA Engine authenticated users that can concurrently use Queuing Services.

### \texttt{tuxTQspaceMemNonPersist}

**Syntax**  
DisplayString

**Access**  
read-write
Description

This attribute specifies the size of the area reserved in shared memory to hold non-persistent messages for all queues in the queue space. The memory size can be specified in bytes (b) or blocks (B). (The size of a block in this context is equivalent to the size of a disk block.)

The [bB] suffix is optional and, if not specified, the default is blocks. Note that the number of bytes requested can be rounded up to the next internal data size. When read, the value is always the actual amount of memory allocated in bytes (b).

All non-persistent messages in the specified queue space are permanently lost when this variable is successfully changed.

If the variable for a queue space is zero (0), no queue space is reserved for non-persistent messages. In this case, any attempt to enqueue a non-persistent message fails. This type of failure results, for example, when no delivery quality of service has been specified for a message and the tuxTAppQDefDeliverPolicy object of the tuxTAppTbl group for the target queue has been set to NONPERSIST. For non-persistent delivery, if the memory area is exhausted or fragmented so that a message cannot be enqueued, the enqueuing operation fails, even if there is sufficient persistent storage for the message. Similarly, if the persistent storage area is exhausted or fragmented so that a message cannot be enqueued, the enqueuing operation fails, even if there is sufficient non-persistent storage for the message.

tuxTQspaceMemFilters

Syntax

INTEGER

Access

read-write

Description

This attribute specifies the size of the memory area to reserve in shared memory to hold the compiled representation of user-defined filters. The memory size is specified in bytes. Filters are used by the Queuing Services component of the BEA Engine for message selection in dequeuing and cursor operations. Filters can be specified using various grammars, but are compiled into an engine normal form and stored in shared memory. Filters are referenced by a handle that is returned when they are compiled. When a filter is destroyed, the memory used by the filter is made available for subsequent compiled filters. When the filters are defined by an application, the administrator must configure the system to accommodate the maximum number of filters that will be concurrently compiled. An operation fails if a user attempts to create a new filter and there is not enough memory allocated for the compiled version of the filter. Adjusting this value has no effect on BEA Tuxedo applications other than unnecessarily consuming shared memory resources.
### txuTQspaceMemOverFlow

**Syntax**
INTEGER

**Access**
read-write

**Description**
This attribute specifies the size of the memory area to reserve in shared memory to accommodate peak load situations where some or all of the allocated shared memory resources are exhausted. The memory size is specified in bytes. Additional objects are allocated from this additional memory on a first-come, first-served basis. When an object created in the additional memory is closed or destroyed, the memory is released for subsequent overflow situations. This additional memory space can yield more objects than the configured number, but there is no guarantee that additional memory is available for any particular object at any given point in time. Currently, only actions, handles, cursors, owners, temporary queues, timers, and filters use the overflow.

### txuTQspaceMemSystemReserved

**Syntax**
INTEGER

**Access**
read-only

**Description**
This attribute specifies the total amount of memory (in bytes) reserved from shared memory for queuing services system use.

### txuTQspaceMemTotalAllocated

**Syntax**
INTEGER

**Access**
read-only

**Description**
This attribute specifies the total amount of memory (in bytes) allocated from shared memory for all queuing services objects.

### txuTQspaceCurActions

**Syntax**
INTEGER

**Access**
read-only
Description  This attribute specifies the current number of actions in use in the queue space. This number can be determined if the queue space is OPEN or ACTIVE, or if the queue space is newly created. If none of the conditions apply, the value −1 is returned.

tuxTQspaceCurHandles

Syntax  INTEGER  
Access  read-only  
Description  This attribute specifies the current number of cursors in use in the queue space. This number can be determined if the queue space is OPEN or ACTIVE, or if the queue space is newly created. If none of the conditions apply, the value −1 is returned.

tuxTQspaceCurOwners

Syntax  INTEGER  
Access  read-only  
Description  This attribute specifies the current number of owners in use in the queue space. This number can be determined if the queue space is OPEN or ACTIVE, or if the queue space is newly created. If none of the conditions apply, the value −1 is returned.

tuxTQspaceCurTmpQueues

Syntax  INTEGER  
Access  read-only  
Description  This attribute specifies the current number of temporary queues in use in the queue space. This number can be determined if the queue space is OPEN or ACTIVE, or if the queue space is newly created. If none of the conditions apply, the value −1 is returned.

tuxTQspaceCurCursors

Syntax  INTEGER  
Access  read-only
### tuxTQspaceCurMemNonPersist

**Syntax**  INTEGER

**Access**  read-only

**Description**  This attribute specifies the current amount of memory, in bytes, consumed by non-persistent messages in the queue space. This number can be determined if the queue space is OPEn or ACTive, or if the queue space is newly created. If none of the conditions apply, the value -1 is returned.

### tuxTQspaceCurMemFilters

**Syntax**  INTEGER

**Access**  read-only

**Description**  This attribute specifies the current number of bytes in use for filters in the queue space. This number can be determined if the queue space is OPEn or ACTive, or if the queue space is newly created. If none of the conditions apply, the value -1 is returned.

### tuxTQspaceCurMemOverFlow

**Syntax**  INTEGER

**Access**  read-only

**Description**  This attribute specifies the current number of bytes of overflow memory in use in the queue space. This number can be determined if the queue space is OPEn or ACTive, or if the queue space is newly created. If none of the conditions apply, the value -1 is returned.

### tuxTQspaceHwActions

**Syntax**  INTEGER (0..100)

**Access**  read-only
8 Application Queue MIB

Description
This attribute specifies the highest number of concurrent actions reached in the queue space since the queue space was last opened. The number is reset to 0 when the queue space is set to CLEaning.

tuxTQspaceHwHandles

Syntax INTEGER (0..100)
Access read-only
Description This attribute specifies the highest number of concurrent handles opened in the queue space since the queue space was last opened. The number is reset to 0 when the queue space is set to CLEaning.

tuxTQspaceHwOwners

Syntax INTEGER
Access read-only
Description This attribute specifies the highest number of concurrent owners reached in the queue space since the queue space was last opened. The number is reset to 0 when the queue space is set to CLEaning.

tuxTQspaceHwTmpQueues

Syntax INTEGER
Access read-only
Description This attribute specifies the highest number of concurrent temporary queues opened in the queue space since the queue space was last opened. The number is reset to 0 when the queue space is set to CLEaning.

8-40 BEA SNMP Agent MIB Reference
### tuxTQspaceTbl

<table>
<thead>
<tr>
<th>Description</th>
<th>This attribute specifies the highest number of concurrent cursors created in the queue space since the queue space was last opened. The number is reset to 0 when the queue space is set to CLEaning.</th>
</tr>
</thead>
</table>

#### tuxTQspaceHwMemNonPersist

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>This attribute specifies the largest amount of memory in bytes consumed by non-persistent messages since the queue space was last opened. The number is reset to 0 when the queue space is set to CLEaning.</td>
</tr>
</tbody>
</table>

#### tuxTQspaceHwMemFilters

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>This attribute specifies the highest number of bytes used for filters in the queue space since the queue space was last opened. The number is reset to 0 when the queue space is set to CLEaning.</td>
</tr>
</tbody>
</table>

#### tuxTQspaceHwMemOverflow

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>This attribute specifies the highest number of bytes used in the overflow memory in the queue space since the queue space was last opened. The number is reset to 0 when the queue space is set to CLEaning.</td>
</tr>
</tbody>
</table>
This group represents runtime attributes of transactions associated with application queue spaces. Objects in this table are only accessible through a BEA SNMP Agent installed on the local machine.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTQtransXid</td>
<td>.1.3.6.1.4.1.140.300.124.1.1</td>
</tr>
<tr>
<td>tuxTQtransIndx1</td>
<td>.1.3.6.1.4.1.140.300.124.1.2</td>
</tr>
<tr>
<td>tuxTQtransIndx2</td>
<td>.1.3.6.1.4.1.140.300.124.1.3</td>
</tr>
<tr>
<td>tuxTQtransIndx3</td>
<td>.1.3.6.1.4.1.140.300.124.1.4</td>
</tr>
<tr>
<td>tuxTQtransIndx4</td>
<td>.1.3.6.1.4.1.140.300.124.1.5</td>
</tr>
<tr>
<td>tuxTQtransIndx5</td>
<td>.1.3.6.1.4.1.140.300.124.1.6</td>
</tr>
<tr>
<td>tuxTQtransGrpNo</td>
<td>.1.3.6.1.4.1.140.300.124.1.7</td>
</tr>
<tr>
<td>tuxTQtranSpaceName</td>
<td>.1.3.6.1.4.1.140.300.124.1.8</td>
</tr>
<tr>
<td>tuxTQtransQmConfig</td>
<td>.1.3.6.1.4.1.140.300.124.1.9</td>
</tr>
<tr>
<td>tuxTQtransLmid</td>
<td>.1.3.6.1.4.1.140.300.124.1.10</td>
</tr>
<tr>
<td>tuxTQtransState</td>
<td>.1.3.6.1.4.1.140.300.124.1.11</td>
</tr>
</tbody>
</table>

**tuxTQtransXid**

**Syntax**: `DisplayString(SIZE(1..78))`

**Access**: read-only

**Description**: Transaction identifier as returned by `tx_info(3)` and mapped to a string representation. The data in this field should not be interpreted directly by the user except for equality comparison.
tuxTQtransTbl

tuxTQtransIndx1
  Syntax    INTEGER
  Access    read-only
  Description  An integer index for tuxTQtransTbl. This should not be interpreted by the user. It is used only for uniquely identifying a particular row in this table by the combination of Indx1 through Indx5.

tuxTQtransIndx2
  Syntax    INTEGER
  Access    read-only
  Description  An integer index for tuxTQtransTbl. This should not be interpreted by the user. It is used only for uniquely identifying a particular row in this table by the combination of Indx1 through Indx5.

tuxTQtransIndx3
  Syntax    INTEGER
  Access    read-only
  Description  An integer index for tuxTQtransTbl. This should not be interpreted by the user. It is used only for uniquely identifying a particular row in this table by the combination of Indx1 through Indx5.

tuxTQtransIndx4
  Syntax    INTEGER
  Access    read-only
  Description  An integer index for tuxTQtransTbl. This should not be interpreted by the user. It is used only for uniquely identifying a particular row in this table by the combination of Indx1 through Indx5.
tuxTQtransIndx5

Syntax  INTEGER
Access   read-only
Description  An integer index for tuxTQtransTbl. This should not be interpreted by the user. It is used only for uniquely identifying a particular row in this table by the combination of Indx1 through Indx5.

tuxTQtransGrpNo

Syntax  INTEGER
Access   read-only
Description  Group number of any server group for which the queue space concerning this transaction is a resource manager, in other words that group’s openinfo string tuxTgroupOpenInfo contains the device name and queue space name for the queue space concerning this transaction.

tuxTQtranSpaceName

Syntax  DisplayString(SIZE(1..15))
Access   read-only
Description  Name of the application queue space associated with the transaction.

tuxTQtransQmConfig

Syntax  DisplayString(SIZE(1..78))
Access   read-only
Description  Absolute pathname of the file or device where the application queue space is located.
tuxTQtransLmid

**Syntax**  
`DisplayString SIZE(1..30)`

**Access**  
read-only

**Description**  
Identifier of the logical machine where the application queue space is located.

---

tuxTQtransState

**Syntax**  
`INTEGER { active(1), abort-only(2), aborted(3), com-called(4), ready(5), decided(6), suspended(7), habort(8), hcommit(9) }`

**Access**  
read-write

**Description**  
The values for GET and SET operations are as follows:

GET: `{active(1)|abort-only(2)|aborted(3)|com-called(4)|ready(5)|decided(6)|suspended(7)}`

A GET operation retrieves runtime information about the selected transactions. The following list describes the meaning of the `tuxTQtransState` attribute returned in response to a GET request. States not listed are not returned.

- **active(1)**
  - The transaction is active.

- **abort-only(2)**
  - The transaction has been identified for rollback.

- **aborted(3)**
  - The transaction has been identified for rollback and rollback has been initiated.

- **com-called(4)**
  - The initiator of the transaction has called `tpcommit(3)` and the first phase of two-phase commit has begun.

- **ready(5)**
  - All of the participating groups on the retrieval site have successfully completed the first phase of the two-phase commit and are ready to be committed.
decided(6)
   The second phase of the two-phase commit has begun.

suspended(7)
   The initiator of the transaction has suspended processing on the transaction.

SET: {habort(8) | hcommit(9)}
   A SET operation updates the state of the selected transactions. The following
   list describes the meaning of the tuxTQtransState attribute returned by a
   SET request. States not listed cannot be set.

habort(8)
   Heuristically abort the transaction. Successful return leaves the object in the
   habort(8) state.

hcommit(9)
   Heuristically commit the transaction. Successful return leaves the object in
   the hcommit(9) state.
There are two types of Tuxedo and WLE events: application events and system events. Application events are usually controlled or trapped by the application code. System events are generated by the Tuxedo or WLE run-time system when important changes in that system are detected. Application programs (clients or services) can subscribe to these system events.

The Event Broker MIB defines the characteristics of an event subscription. You can use the Event Broker MIB to obtain the characteristics of current event subscriptions, define new subscriptions, or invalidate subscriptions. To enable both system event and application event notification, you need to define the system event broker and the application event broker in the Tuxedo Core MIB.

Event subscriptions can be temporary or persistent. Persistent subscriptions survive across application activations and can be removed through the Event Broker MIB. The Tuxedo Event Broker MIB contains five groups of event subscriptions through which the Event Broker can be managed. The following table lists the event broker subscription groups.

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxEventClientTbl</td>
<td>Subscriptions that trigger unsolicited notification</td>
</tr>
<tr>
<td>tuxEventCmdTbl</td>
<td>Subscriptions that trigger system commands</td>
</tr>
<tr>
<td>tuxEventQueTbl</td>
<td>Subscriptions for queue-based notification</td>
</tr>
<tr>
<td>tuxEventSvcTbl</td>
<td>Subscriptions for server-based notification</td>
</tr>
<tr>
<td>tuxEventUlogTbl</td>
<td>Subscriptions for writing userlog messages</td>
</tr>
</tbody>
</table>
Each object in these groups represents a single subscription request. Client Notifications (tuxEventClientTbl group) indicate which events trigger an unsolicited message to a client. Service Notifications (tuxEventSvcTbl group) indicate which events trigger a request to an application service. Application Queue Notifications (tuxEventQueTbl group) indicate which events send a message to an application queue. System Command Notifications (tuxEventCmdTbl group) indicate which events trigger an operating system command. Log File Notifications (tuxEventUlogTbl group) indicate which events generate a record in the central event log (ulog). The Event Broker automatically removes temporary subscriptions when it detects that the corresponding target is no longer active.

Event subscriptions and the ability to change the Tuxedo MIB enables system administrators and application designers to write event-adaptive applications. When a failure is detected through a system event notification, a management framework program can perform the corrective measures. For example, a management framework task can be triggered to activate servers on a backup machine when it receives an event notification about a failure on a primary machine.
This represents a set of subscriptions registered with the Event Broker for client-based notification.

When an event is detected, it is compared to each tuxEventClientTbl instance. If the event name matches the value in the event expression and the optional filter rule is true, then the event buffer is sent to the specified client’s unsolicited message handling routine. To create a new row in this table, it is necessary to issue a SET request that at least specifies the values for tuxEventClientExpr and tuxEventClientId.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxEventClientIx</td>
<td>.1.3.6.1.4.1.140.300.2.1.1.1.1</td>
</tr>
<tr>
<td>tuxEventClientExpr</td>
<td>.1.3.6.1.4.1.140.300.2.1.1.1.2</td>
</tr>
<tr>
<td>tuxEventClientFilter</td>
<td>.1.3.6.1.4.1.140.300.2.1.1.1.3</td>
</tr>
<tr>
<td>tuxEventClientState</td>
<td>.1.3.6.1.4.1.140.300.2.1.1.1.4</td>
</tr>
<tr>
<td>tuxEventClientId</td>
<td>.1.3.6.1.4.1.140.300.2.1.1.1.5</td>
</tr>
</tbody>
</table>

**tuxEventClientIx**

Syntax: INTEGER  
Access: read-only  
Description: A running number as the unique identifier for a row in the table.

**tuxEventClientExpr**

Syntax: DisplayString (SIZE(1..255))  
Access: read-only  
Description: Event pattern expression. This expression, in recomp(3) format, controls which event names match this subscription.

Note: This object can be updated only during row creation.
tuxEventClientFilter

Syntax: \textit{DisplayString(SIZE(1..255))}

Access: read-only

Description: Event filter expression. This expression, if present, is evaluated with respect to the posted buffer’s contents. It must evaluate to TRUE or this subscription is not matched. If the value of this is “-”, it means that the filter expression is in binary format.

Note: This object can be updated only during row creation.

tuxEventClientState

Syntax: INTEGER \{ active(1), invalid(2) \}

Access: read-write

Description: The values for \textit{GET} and \textit{SET} operations are as follows:

\textit{GET}: active(1)
A \textit{GET} operation retrieves configuration information for the matching \textit{tuxEventClientTbl} row(s).

\textit{SET}: invalid(2)
A \textit{SET} operation updates configuration information for the row in \textit{tuxEventClientTbl}. The following state indicates the meaning of a \textit{tuxEventClientState} set in a \textit{SET} request. States not listed cannot be set.

invalid(2)
Delete row. Successful return leaves the row in the \textit{invalid(2)} state.

tuxEventClientId

Syntax: \textit{DisplayString(SIZE(1..78))}

Access: read-only

Description: Send an unsolicited notification message to this client when a matching event is detected.

Note: This object can be updated only during row creation.
tuxEventCmdTbl

This represents a set of subscriptions registered with the Event Broker that trigger execution of system commands.

When an event is detected, it is compared to each row in this table. If the event name matches the value in the event expression and the optional filter rule is true, then the event buffer is formatted and passed to the system’s command interpreter.

Create a new Row: To create a new instance of tuxEventCmdTbl the user must specify at least tuxEventCmdExpr and tuxEventCmd. All objects except tuxEventCmdState can be updated only during creation of a new instance.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxEventCmdIndx</td>
<td>.1.3.6.1.4.1.140.300.2.2.1.1.1</td>
</tr>
<tr>
<td>tuxEventCmdExpr</td>
<td>.1.3.6.1.4.1.140.300.2.2.1.1.2</td>
</tr>
<tr>
<td>tuxEventCmdFilter</td>
<td>.1.3.6.1.4.1.140.300.2.2.1.1.3</td>
</tr>
<tr>
<td>tuxEventCmdState</td>
<td>.1.3.6.1.4.1.140.300.2.2.1.1.4</td>
</tr>
<tr>
<td>tuxEventCmd</td>
<td>.1.3.6.1.4.1.140.300.2.2.1.1.5</td>
</tr>
</tbody>
</table>

**tuxEventCmdIndx**

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: A running number as the unique identifier for a row in the table.
Event Broker MIB

**tuxEventCmdExpr**

Syntax  
DisplayString(SIZE(1..255))

Access  
read-write

Description  
Event pattern expression. This expression, in \texttt{recomp}(3) format, controls which event names match this subscription.

**Note:** This object can be updated only during row creation.

**tuxEventCmdFilter**

Syntax  
DisplayString(SIZE(1..255))

Access  
read-write

Description  
Event filter expression. This expression, if present, is evaluated with respect to the posted buffer’s contents. It must evaluate to TRUE or this subscription is not matched. If the value of the filter is 

**Note:** This object can be updated only during row creation.

**tuxEventCmdState**

Syntax  
INTEGER \{ active(1), invalid(2) \}

Access  
read-write

Description  
The values for \texttt{GET} and \texttt{SET} operations are as follows:

**GET:** active(1)

A \texttt{GET} operation retrieves configuration information for the \texttt{tuxEventCmdTbl} instance(s).

**SET:** invalid(2)

A \texttt{SET} operation updates configuration information for the \texttt{tuxEventCmdTbl} instance. The following state indicates the meaning of a \texttt{tuxEventCmdState} set in a \texttt{SET} request. States not listed cannot be set.

invalid(2)
Delete tuxEventCmdTbl instance. Successful return leaves the object in the invalid(2) state.

**tuxEventCmd**

- **Syntax**: `DisplayString(SIZE(1..255))`
- **Access**: read-write
- **Description**: Execute this system command when an event matching this object is detected. For UNIX system platforms, the command is executed in the background using `system(3)`.

**Note**: This object can be updated only during row creation.
This represents a set of subscriptions registered with the Event Broker for queue-based notification.

When an event is detected, it is compared to each `tuxEventQueTbl` instance. If the event name matches the value in the event expression and the optional filter rule is true, then the event buffer is stored in the specified reliable queue. To create a new row in this table, it is necessary to issue a SET request that at least specifies `tuxEventQueExpr`, `tuxEventQspace`, and `tuxEventQname`.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxEventQueIndx</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.1</td>
</tr>
<tr>
<td>tuxEventQueExpr</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.2</td>
</tr>
<tr>
<td>tuxEventQueFilter</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.3</td>
</tr>
<tr>
<td>tuxEventQueState</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.4</td>
</tr>
<tr>
<td>tuxEventQspace</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.5</td>
</tr>
<tr>
<td>tuxEventQname</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.6</td>
</tr>
<tr>
<td>tuxEventQctlQtop</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.7</td>
</tr>
<tr>
<td>tuxEventQctlBeforeMsgid</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.8</td>
</tr>
<tr>
<td>tuxEventQctlQtimeAbs</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.9</td>
</tr>
<tr>
<td>tuxEventQctlQtimeRel</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.10</td>
</tr>
<tr>
<td>tuxEventQctlDeqTime</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.11</td>
</tr>
<tr>
<td>tuxEventQctlPrior</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.12</td>
</tr>
<tr>
<td>tuxEventQctlMsgId</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.13</td>
</tr>
<tr>
<td>tuxEventQctlCorrId</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.14</td>
</tr>
<tr>
<td>tuxEventQctlReplyQ</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.15</td>
</tr>
</tbody>
</table>
### tuxEventQueIdx

**Syntax**: INTEGER  
**Access**: read-only  
**Description**: Running number which is the unique identifier for an event in this table.

### tuxEventQueExpr

**Syntax**: DisplayString(SIZE(1..255))  
**Access**: read-write  
**Description**: Event pattern expression. This expression, in `recomp(3)` format, controls which event names match this subscription.  

**Note**: This object can be updated only during row creation.

### tuxEventQueFilter

**Syntax**: DisplayString(SIZE(1..255))  
**Access**: read-write  
**Description**: Event filter expression. This expression, if present, is evaluated with respect to the posted buffer’s contents. It must evaluate to TRUE or this subscription is not matched. If the value of this object is “-”, it means the filter is in binary format.  

**Note**: This object can be updated only during row creation.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxEventQctlFailQ</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.16</td>
</tr>
<tr>
<td>tuxEventPersist</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.17</td>
</tr>
<tr>
<td>tuxEventTran</td>
<td>.1.3.6.1.4.1.140.300.2.3.1.1.18</td>
</tr>
</tbody>
</table>
Event Broker MIB

**tuxEventQueState**

**Syntax**

`INTEGER { active(1), invalid(2) }`

**Access**

read-write

**Description**

The values for GET and SET operations are as follows:

**GET:** active(1)

A GET operation retrieves configuration information for the matching `tuxEventQueTbl` row(s).

**SET:** invalid(2)

A SET operation updates configuration information for the `tuxEventQueTbl` instance. The following state indicates the meaning of a `tuxEventQueState` set in a SET request. States not listed cannot be set.

invalid(2)

Delete `tuxEventQueTbl` row. Successful return leaves the object in the invalid(2) state.

**tuxEventQspace**

**Syntax**

`DisplayString(SIZE(1..15))`

**Access**

read-write

**Description**

Enqueue a notification message to a reliable queue in this queue space when a matching event is detected.

**Note:** This object can be updated only during row creation.

**tuxEventQname**

**Syntax**

`DisplayString(SIZE(1..15))`

**Access**

read-write

**Description**

Enqueue a notification message to this reliable queue when a matching event is detected.

**Note:** This object can be updated only during row creation.
**tuxEventQctlQtop**

Syntax: INTEGER  
Access: read-write  
Description: This value, if present, is passed in to `tpenqueue(3)`'s TPQCTL control structure to request notification via the /Q subsystem with the message to be placed at the top of the queue.  

**Note:** This object can be updated only during row creation.

**tuxEventQctlBeforeMsgid**

Syntax: INTEGER  
Access: read-write  
Description: This value, if present, is passed in to `tpenqueue(3)`'s TPQCTL control structure to request notification via the /Q subsystem with the message to be placed on the queue ahead of the specified message.  

**Note:** This object can be updated only during row creation.

**tuxEventQctlQtimeAbs**

Syntax: INTEGER  
Access: read-write  
Description: This value, if present, is passed in to `tpenqueue(3)`'s TPQCTL control structure to request notification via the /Q subsystem with the message to be processed at the specified time.  

**Note:** This object can be updated only during row creation.
9 Event Broker MIB

**tuxEventQctlQtimeRel**

**Syntax** INTEGER  
**Access** read-write  
**Description** This value, if present, is passed in to `tpenqueue(3)`’s TPQCTL control structure to request notification via the /Q subsystem with the message to be processed relative to the dequeue time.

**Note:** This object can be updated only during row creation.

**tuxEventQctlDeqTime**

**Syntax** INTEGER  
**Access** read-write  
**Description** This value, if present, is passed in to `tpenqueue(3)`’s TPQCTL control structure.

**Note:** This object can be updated only during row creation.

**tuxEventQctlPrior**

**Syntax** INTEGER  
**Access** read-write  
**Description** This value, if present, is passed in to `tpenqueue(3)`’s TPQCTL control structure.

**Note:** This object can be updated only during row creation.

**tuxEventQctlMsgId**

**Syntax** `DisplayString(SIZE(1..31))`  
**Access** read-write  
**Description** This value, if present, is passed in to `tpenqueue(3)`’s TPQCTL control structure.

**Note:** This object can be updated only during row creation.
tuxEventQctlCorrId

Syntax: DisplayString (SIZE(1..31))
Access: read-write
Description: This value, if present, is passed in to tpenqueue(3)’s TPQCTL control structure.

Note: This object can be updated only during row creation.

---

tuxEventQctlReplyQ

Syntax: DisplayString (SIZE(1..15))
Access: read-write
Description: This value, if present, is passed in to tpenqueue(3)’s TPQCTL control structure.

Note: This object can be updated only during row creation.

---

tuxEventQctlFailQ

Syntax: DisplayString (SIZE(1..15))
Access: read-write
Description: This value, if present, is passed in to tpenqueue(3)’s TPQCTL control structure.

Note: This object can be updated only during row creation.

---

tuxEventPersist

Syntax: INTEGER
Access: read-write
Description: If non-zero, do not cancel this subscription if the designated queue is no longer available.

Note: This object can be updated only during row creation.
tuxEventTran

Syntax  INTEGER
Access  read-write
Description  If non-zero and the client's tpPost(3) call is transactional, include the tpenqueue(3) call in the client's transaction.

Note:  This object can be updated only during row creation.
tuxEventSvcTbl

This represents a set of subscriptions registered with the Event Broker for service-based notification.

When an event is detected, it is compared to each tuxEventSvcTbl instance. If the event name matches the value in the event expression and the optional filter rule is true, then the event buffer is sent to the specified Tuxedo service routine.

To create a new row in this table, a SET request must be issued that specifies values for at least tuxEventSvcExpr and tuxEventSvcName.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxEventSvcIndx</td>
<td>.1.3.6.1.4.1.140.300.2.4.1.1.1</td>
</tr>
<tr>
<td>tuxEventSvcExpr</td>
<td>.1.3.6.1.4.1.140.300.2.4.1.1.2</td>
</tr>
<tr>
<td>tuxEventSvcFilter</td>
<td>.1.3.6.1.4.1.140.300.2.4.1.1.3</td>
</tr>
<tr>
<td>tuxEventSvcState</td>
<td>.1.3.6.1.4.1.140.300.2.4.1.1.4</td>
</tr>
<tr>
<td>tuxEventSvcName</td>
<td>.1.3.6.1.4.1.140.300.2.4.1.1.5</td>
</tr>
<tr>
<td>tuxEventSvcPersist</td>
<td>.1.3.6.1.4.1.140.300.2.4.1.1.6</td>
</tr>
<tr>
<td>tuxEventSvcTran</td>
<td>.1.3.6.1.4.1.140.300.2.4.1.1.7</td>
</tr>
</tbody>
</table>

tuxEventSvcIndx

Syntax: INTEGER

Access: read-only

Description: A running number which is a unique key for a row in this table.
**tuxEventSvcExpr**

- **Syntax**: `DisplayString(SIZE(1..255))`
- **Access**: read-only
- **Description**: Event pattern expression. This expression, in `recomp(3)` format, controls which event names match this subscription.

  **Note**: This object can be updated only during row creation.

**tuxEventSvcFilter**

- **Syntax**: `DisplayString(SIZE(1..255))`
- **Access**: read-only
- **Description**: Event filter expression. This expression, if present, is evaluated with respect to the posted buffer’s contents. It must evaluate to TRUE or this subscription is not matched. If this is “.“, it means the filter is in binary format.

  **Note**: This object can be updated only during row creation.

**tuxEventSvcState**

- **Syntax**: `INTEGER { active(1), invalid(2) }`
- **Access**: read-write
- **Description**: The values for GET and SET operations are as follows:

  **GET**: active(1)

  A GET operation retrieves configuration information for the matching `tuxEventSvcTbl` instance(s).

  **SET**: invalid(2)

  A SET operation updates configuration information for the `tuxEventSvcTbl` instance. The following state indicates the meaning of a `tuxEventSvcState` set in a SET request. States not listed cannot be set.

  invalid(2)
Delete `tuxEventSvcTbl` row. Successful return leaves the object in the `invalid(2)` state.

**tuxEventSvcName**

<table>
<thead>
<tr>
<th>Syntax</th>
<th><code>DisplayString(SIZE(1..15))</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-only</td>
</tr>
<tr>
<td>Description</td>
<td>Call this Tuxedo service when a matching event is detected.</td>
</tr>
</tbody>
</table>

**Note:** This object can be updated only during row creation.

**tuxEventSvcPersist**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>If non-zero, do not cancel this subscription if the <code>tuxEventSvcName</code> service is no longer available.</td>
</tr>
</tbody>
</table>

**Note:** This object can be updated only during row creation.

**tuxEventSvcTran**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>read-write</td>
</tr>
<tr>
<td>Description</td>
<td>If non-zero and the client’s <code>tpost(3)</code> call is transactional, include the <code>tuxEventSvcName</code> service call in the client’s transaction.</td>
</tr>
</tbody>
</table>

**Note:** This object can be updated only during row creation.
tuxEventUlogTbl

This represents a set of subscriptions registered with the Event Broker for writing system userlog(3) messages.

When an event is detected, it is compared to each tuxEventUlogTbl instance. If the event name matches the value in the event expression and the optional filter rule is true, then the event buffer is formatted and passed to the Tuxedo userlog(3) function.

Create a new Row: To create a new instance of tuxEventUlogTbl the user must at least specify values for tuxEventUlogExpr and tuxEventUserlog. All objects except tuxEventUlogState can be updated only during creation of a new instance.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Object ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxEventUlogIndx</td>
<td>.1.3.6.1.4.1.140.300.2.5.1.1.1</td>
</tr>
<tr>
<td>tuxEventUlogExpr</td>
<td>.1.3.6.1.4.1.140.300.2.5.1.1.2</td>
</tr>
<tr>
<td>tuxEventUlogFilter</td>
<td>.1.3.6.1.4.1.140.300.2.5.1.1.3</td>
</tr>
<tr>
<td>tuxEventUlogState</td>
<td>.1.3.6.1.4.1.140.300.2.5.1.1.4</td>
</tr>
<tr>
<td>tuxEventUserlog</td>
<td>.1.3.6.1.4.1.140.300.2.5.1.1.5</td>
</tr>
</tbody>
</table>

`tuxEventUlogIndx`

- **Syntax**: INTEGER
- **Access**: read-only
- **Description**: A running number which is a unique key in this table.
tuxEventUlogExpr

Syntax: DisplayString(SIZE(1..255))

Access: read-write

Description: Event pattern expression. This expression, in recomp(3) format, controls which event names match this subscription.

Note: This object can be updated only during row creation.

---

tuxEventUlogFilter

Syntax: DisplayString(SIZE(1..255))

Access: read-write

Description: Event filter expression. This expression, if present, is evaluated with respect to the posted buffer’s contents. It must evaluate to TRUE or this subscription is not matched. If this is “-”, it means the filter is in binary form.

Note: This object can be updated only during row creation.

---

tuxEventUlogState

Syntax: INTEGER { active(1), invalid(2) }

Access: read-write

Description: The values for GET and SET operations are as follows:

GET: active(1)

A GET operation retrieves configuration information for the matching tuxEventUlogTbl instance(s).

SET: invalid(2)

A SET operation updates configuration information for the tuxEventUlogTbl instance. The following state indicates the meaning of a tuxEventUlogState set in a SET request. States not listed cannot be set.

invalid(2)
Delete tuxEventUlogTbl row. Successful return leaves the object in the invalid(2) state.

**tuxEventUserlog**

- **Syntax**: `DisplayString (SIZE(1..255))`
- **Access**: read-write
- **Description**: Write a userlog(3) message when a matching event is detected.
  
  **Note**: This object can be updated only during row creation.
The event monitor feature of the Tuxedo and WLE systems detects and reports certain predefined events—primarily failures of which a system operator should be aware. The BEA SNMP Agent on the master node subscribes to all system events and generates a corresponding SNMP trap notification whenever any of these events occur. The enterprise ID used for these traps is `.1.3.6.1.4.1.140.tuxedo`, where `tuxedo` is 300. For the BEA SNMP Agent to receive Tuxedo or WLE system events, the Tuxedo system Event Broker (TMSYSEVT) must be running because that is the entity that generates the system events.

The Event Traps MIB defines all the traps that are generated and the objects that are passed in the variable bindings for these traps. The cause and recommended action for each event is described in the following sections.

### Specific Trap Number

Each enterprise-specific trap notification generated by the BEA SNMP Agent has a value in the specific trap ID field of the SNMP trap packet that identifies the Tuxedo or WLE event. For each trap listed in this chapter, “Trap ID” is the specific trap number that is sent in the trap packet.
Variable Bindings

SNMP trap notifications generated by the BEA SNMP Agent contain 12 variables (attribute/value pairs) in the variable bindings of the trap packet:

beaEventsDomainId

This is the ID of the domain that generated the Tuxedo or WLE event notification.

beaEventsIpcKey

This is the IPC key of the Tuxedo or WLE domain.

beaLogicalAgentName

The logical agent name of the BEA SNMP Agent generating the trap. The executable name is the default logical agent name.

The tuxEventTrapVars group contains all objects that are sent as a part of the variable bindings of the traps generated in relation to Tuxedo or WLE system events, as defined in EVENTS.

**tuxEventsName**

Syntax  
DisplayString

Access  
not-accessible

Description  
A string that uniquely identifies this event. All system-generated events begin with .Sys.

**tuxEventsSeverity**

Syntax  
INTEGER { Error(1), Warn(2), Infor(3) }

Access  
not-accessible

Description  
Indicates the severity of the system event.
**tuxEventsLmid**

**Syntax**  
`DisplayString(SIZE(1..30))`

**Access**  
not-accessible

**Description**  
A string that identifies the machine where the event was detected.

**tuxEventsTime**

**Syntax**  
`INTEGER`

**Access**  
not-accessible

**Description**  
A long integer containing the event detection time, in seconds, according to the clock on the machine where detection took place.

**tuxEventsUsec**

**Syntax**  
`INTEGER`

**Access**  
not-accessible

**Description**  
A long integer containing the event detection time, in microseconds, according to the clock on the machine where detection took place. While the units of this value are always microseconds, the actual resolution depends on the underlying operating system and hardware.

**tuxEventsDescription**

**Syntax**  
`DisplayString`

**Access**  
not-accessible

**Description**  
A one-line string summarizing the event.
## tuxEventsClass

**Syntax**  
*DisplayString*

**Access**  
not-accessible

**Description**  
The class of the object associated with the event. Depending on TA_CLASS, the event notification buffer can contain additional fields specific to an object of this class.

## tuxEventsUlogCat

**Syntax**  
*DisplayString*

**Access**  
not-accessible

**Description**  
Catalog name from which the message was derived, if any.

## tuxEventsUlogMsgNum

**Syntax**  
*INTEGER*

**Access**  
not-accessible

**Description**  
Catalog message number, if the message was derived from a catalog.

## tuxTdomainID

**Syntax**  
*DisplayString*(SIZE(0..30))

**Access**  
not-accessible

**Description**  
Domain identification string. Refer to Chapter 2, “Tuxedo Core MIB.”
tuxTdomainKey

Syntax INTEGER (32769..262143)

Access not-accessible

Description Numeric key for the well-known address in a Tuxedo System/T bulletin board. In a single processor environment, this key “names” the bulletin board. In a multiple processor or LAN environment, this key names the message queue of the DBBL. In addition, this key is used as a basis for deriving the names of resources other than the well-known address, such as the names for bulletin boards throughout the application. Refer to Chapter 2, “Tuxedo Core MIB.”

beaLogicalAgentName

Syntax DisplayString

Access not-accessible

Description The logical name of the agent as provided in the -l option (service name in case of Windows NT) when the agent was started. This is the agent that is monitoring this domain. If there are multiple SNMP agents running on a managed node, this name needs to be appended to the community with an @ sign to get the MIB values from the appropriate agent. For example, if there are two logical agents, simp_snmpd and bank_snmpd, the communities used to query values from these agents would be public@simp_snmpd and public@bank_snmpd respectively. The component after the @ sign is used to identify the agent to which the MIB query is to be sent.

This object is passed in the variable binding of all SNMP traps generated on behalf of Tuxedo system events.

Note: To run multiple SNMP agents on the same managed node, they must be started as sub-agents (without -s option) and run after starting the agent integrator.

Trap Definitions

This section defines all the traps generated by the BEA SNMP Agent when system events occur.
Tuxedo Traps MIB

DOMAIN Traps

The Domain Traps group defines the Tuxedo domain specific event traps.

resourceConfigTrap

<table>
<thead>
<tr>
<th>Enterprise</th>
<th>tuxedo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables</td>
<td>{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName}</td>
</tr>
<tr>
<td>Description</td>
<td>This trap is generated when .SysResourceConfig occurs. It denotes a system configuration change.</td>
</tr>
<tr>
<td>Action</td>
<td>This is an informational message.</td>
</tr>
<tr>
<td>Trap ID</td>
<td>1</td>
</tr>
</tbody>
</table>
MACHINE Traps

The Machine Traps group defines the Tuxedo machine specific event traps.

**machineBroadcastTrap**

- **Enterprise**: tuxedo
- **Variables**: `{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName }
- **Description**: This trap is generated when `.SysMachineBroadcast` occurs. It denotes broadcast delivery failure. This message indicates that `tpbroadcast()` failed for at least one accessor on the Lmid of the application.
- **Action**: Since the broadcast messages are sent in no-blocking mode, it is possible that the process doing the broadcasting encountered a blocking condition and dropped a message. Configure larger message queues or load-balance clients and servers such that excessive load is not put on some machines.
- **Trap ID**: 2

**machineConfigTrap**

- **Enterprise**: tuxedo
- **Variables**: `{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName }
- **Description**: This trap is generated when `.SysMachineConfig` occurs. It denotes a change in a particular machine configuration.
- **Action**: This is an informational message.
- **Trap ID**: 3
Tuxedo Traps MIB

machineFullMaxAccessersTrap

Enterprise tuxedo

Variables { tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName }

Description This trap is generated when .SysMachineFullMaxaccessers occurs. This message indicates that the given LMID reached the capacity limit on the number of accessers.

Action Increase the MAXACCESSERS value for the particular machine. Or, if the hardware/software limits have been reached for the maximum number of users on the machine, move additional users to other machines.

Trap ID 4

machineFullMaxConvTrap

Enterprise tuxedo

Variables { tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName }

Description This trap is generated when .SysMachineFullMaxconv occurs. This message indicates that the given LMID reached the capacity limit on the number of concurrent conversations.

Action Increase the value of MAXCONV for the particular machine to the point that this event is not generated.

Trap ID 5
**Trap Definitions**

**machineFullMaxGttTrap**

**Enterprise**  
tuxedo

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName}

**Description**  
This trap is generated when .SysMachineFullMaxGtt is raised. This message  
indicates that the given machine reached the capacity limit on the number of  
concurrent transactions.

**Action**  
Increase the value of MAXGTT for the particular machine to the point that this event  
is not generated.

**Trap ID**  
6

**machineFullMaxWsClientsTrap**

**Enterprise**  
tuxedo

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName}

**Description**  
This trap is generated when .SysMachineFullMaxWsClients is raised. This message  
indicates that the given machine reached the capacity limit on the number of  
workstation clients.

**Action**  
Increase the value of MAXWSCliENTS for the particular machine to the point that  
this event is not generated.

**Trap ID**  
7
**machineMsgQTrap**

**Enterprise**  
tuxedo

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName}

**Description**  
This trap is generated when `.SysMachineMsgq` occurs. This message indicates that the  
server posting a message encountered a blocking condition while putting a message on  
the message queue.

**Action**  
Configure larger message queues and/or distribute the load equally on all the  
machines.

**Trap ID**  
8

**machinePartitionedTrap**

**Enterprise**  
tuxedo

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName}

**Description**  
This trap is generated when `.SysMachinePartitioned` occurs. This message  
indicates that DBBL partitioned the stated machine either because the BBL on the  
machine is slow or the network link between the master and the machine is broken.

**Action**  
This can occur due to several reasons:

- The entire network might be bogged down due to heavy traffic.
- The BBL or BRIDGE on the non-master is either dead or slow.
- The BRIDGE process on the non-master is extremely busy.

The software is capable of unpartitioning the machine if things stabilize.

**Trap ID**  
9
Trap Definitions

**machineSlowTrap**

**Enterprise**  
tuxedo

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName }

**Description**  
This trap is generated when .SysMachineSlow occurs. This message indicates that BBL on the non-master machine is slow in generating IAMOK messages. These messages are sent periodically from BBLs to the DBBL that helps the DBBL maintain the pulse of the system.

**Action**  
This can occur due to several reasons:

- The entire network might be bogged down due to heavy traffic.
- The BBL on the non-master might be either dead or slow.
- The BRIDGE process on the non-master is extremely busy.

This problem can be intermittent.

**Trap ID**  
10

**machineStateTrap**

**Enterprise**  
tuxedo

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName }

**Description**  
This trap is generated when .SysMachineState occurs. This denotes that a particular machine changed its state.

**Action**  
This is an informational message.

**Trap ID**  
11
BRIDGE Traps

The Bridge Traps group defines the Tuxedo bridge specific traps.

**networkConfigTrap**

**Enterprise**  tuxedo

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName}

**Description**  This trap is generated when .SysNetworkConfig occurs. This message indicates the network link between the two machines specified changed to a new state.

**Action**  This is an informational message.

**Trap ID**  12

**networkDroppedTrap**

**Enterprise**  tuxedo

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName}

**Description**  This trap is generated when .SysNetworkDropped occurs. This message indicates the network link between the two machines specified was dropped abnormally.

**Action**  This can happen either because the BRIDGE on either machine died or one of the machines crashed.

**Trap ID**  13
**networkFailureTrap**

**Enterprise**  tuxedo

**Variables**  
\{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, 
  tuxEventsUsec, tuxEventsDescription, tuxEventsClass, 
  tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, 
  beaLogicalAgentName \}

**Description**  This trap is generated when .SysNetworkFailure occurs. This indicates a network connection failure between BRIDGE processes.

**Action**  This can happen either because the BRIDGE on remote machine died or the remote machine itself crashed.

**Trap ID**  14

**networkFlowTrap**

**Enterprise**  tuxedo

**Variables**  
\{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, 
  tuxEventsUsec, tuxEventsDescription, tuxEventsClass, 
  tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, 
  beaLogicalAgentName \}

**Description**  This trap is generated when .SysNetworkFlow occurs. This message states that the virtual circuit between machines changed to a new state.

**Action**  This is an informational message.

**Trap ID**  15
networkStateTrap

Enterprise  tuxedo

Variables  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName }

Description  This trap is generated when .SysNetworkState occurs. This message indicates that  
the server died abnormally and BBL cleaned up the slot allocated by the server.

Action  Debug the server and fix the problem before the server is restarted.

Trap ID  16

SERVER Event Traps

The Server Traps group defines the Tuxedo server specific traps.

serverCleaningTrap

Enterprise  tuxedo

Variables  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName }

Description  This trap is generated when .SysServerCleaning occurs. This message indicates that  
the server died abnormally and BBL cleaned up the slot allocated by the server.

Action  Debug the server and fix the problem before the server is restarted.

Trap ID  17
**serverConfigTrap**

**Enterprise**  
bea

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName}

**Description**  
This trap is generated when .SysServerConfig occurs. This message indicates that  
the configuration parameters for the server have been updated.

**Action**  
This is an informational message.

**Trap ID**  
18

**serverDiedTrap**

**Enterprise**  
bea

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime,  
tuxEventsUsec, tuxEventsDescription, tuxEventsClass,  
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey,  
beaLogicalAgentName}

**Description**  
This trap is generated when .SysServerDied occurs. This message indicates that the  
server died abnormally and the BBL detected this condition in its periodic scan of the  
BB.

**Action**  
Debug the server and fix the problem before the server is restarted.

**Trap ID**  
19
**serverInitTrap**

**Enterprise**  tuxedo

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName}

**Description**  This trap is generated when .SysServerInit occurs. This message indicates that the server specified above failed in tpsvrint() and therefore could not be booted.

**Action**  Fix the problem and then reboot the server. This problem might be due to a Tuxedo resource limit or an application-specific problem.

**Trap ID**  20

---

**serverMaxgenTrap**

**Enterprise**  tuxedo

**Variables**  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName}

**Description**  This trap is generated when .SysServerMaxgen occurs. This message indicates that the server died abnormally. Since the server has been marked as restartable, it has been restarted MAXGEN-1 times in the specified GRACE period.

**Action**  Tuxedo application servers should not die abnormally. If this happens, it is most likely due to an application-specific problem. Debug the server and resolve the problem before restarting the server.

**Trap ID**  21
serverRestartingTrap

Enterprise  tuxedo

Variables  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, 
tuxEventsUsec, tuxEventsDescription, tuxEventsClass, 
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, 
beaLogicalAgentName }

Description  This trap is generated when .SysServerRestarting occurs. This message indicates 
that the server died abnormally. Since this has been marked as a restartable server, it 
has been restarted.

Action  Tuxedo application servers should not die abnormally. If this happens, it is most likely 
due to an application-specific problem. Debug the server and resolve the problem 
before restarting the server.

Trap ID  22

serverStateTrap

Enterprise  tuxedo

Variables  
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, 
tuxEventsUsec, tuxEventsDescription, tuxEventsClass, 
tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, 
beaLogicalAgentName }

Description  This trap is generated when .SysServerState occurs. This message indicates that the 
server changed state.

Action  This is an informational message.

Trap ID  23
Tuxedo Traps MIB

**serverTpExitTrap**

Enterprises: tuxedo

Variables: { tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName }

Description: This trap is generated when .SysServerTpexit occurs. This message indicates that the server received a request and the service routine code did a `tpreturn(TPEXIT)` while the server was executing application-specific code.

Action: This is an informational message.

Trap ID: 24

**CLIENT Traps**

The Client Traps group defines the Tuxedo client-specific traps.

**clientConfigTrap**

Enterprises: tuxedo

Variables: { tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName }

Description: This trap is generated when .SysClientConfig is raised. This denotes that a particular user on a machine changed its configuration.

Action: This is an informational message.

Trap ID: 25
**clientDiedTrap**

**Enterprise**
tuxedo

**Variables**
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName }

**Description**
This trap is generated when .SysClientDied occurs. This message indicates that the client exited the application without doing a tpterm(). Normally, clients should do a tpterm() before exiting the application.

**Action**
This is an informational message.

**Trap ID** 26

**clientSecurityTrap**

**Enterprise**
tuxedo

**Variables**
{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, tuxEventsUsec, tuxEventsDescription, tuxEventsClass, tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName }

**Description**
This trap is generated when .SysClientSecurity occurs. This message indicates that the client failed security validation when trying to join the application.

**Action**
Check to make sure that this is not an unauthorized user trying to gain access to your application data.

**Trap ID** 27
Tuxedo Traps MIB

clientStateTrap

Enterprise  tuxedo

Variables  { tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, 
             tuxEventsUsec, tuxEventsDescription, tuxEventsClass, 
             tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, 
             beaLogicalAgentName}

Description  This trap is generated when .SysClientState occurs. This message indicates that a 
              particular client on a machine changed state.

Action  This is an informational message.

Trap ID  28

TRANSACTION Traps

The Transaction Traps group defines the Tuxedo transaction-specific traps.

transHeuristicAbortTrap

Enterprise  tuxedo

Variables  { tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, 
             tuxEventsUsec, tuxEventsDescription, tuxEventsClass, 
             tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, 
             beaLogicalAgentName}

Description  This trap is generated when .SysTransactionHeuristicAbort occurs. This 
              message indicates that the database in a particular group performed an heuristic abort 
              on a transaction.

Action  Check to make sure that the coordinator of the transaction is still running.

Trap ID  29
transHeuristicCommitTrap

Enterprise  tuxedo

Variables  ( txEventsName, txEventsSeverity, txEventsLmid, txEventsTime, txEventsUsec, txEventsDescription, txEventsClass, txEventsUlogCat, txEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName)

Description  This trap is generated when .SysTransactionHeuristicCommit occurs. This message indicates that the database in a particular group performed an heuristic commit on a transaction.

Action  Check to make sure that the coordinator of the transaction is still running.

Trap ID  30

EVENT Traps

The Event Traps group defines the Tuxedo event specific traps.

eventDeliveryTrap

Enterprise  tuxedo

Variables  ( txEventsName, txEventsSeverity, txEventsLmid, txEventsTime, txEventsUsec, txEventsDescription, txEventsClass, txEventsUlogCat, txEventsUlogMsgNum, beaDomainId, beaDomainKey, beaLogicalAgentName)

Description  This trap is generated when .SysEventDelivery occurs. This message indicates that the event server failed to perform at least one notification for a posted event.

Action  Check to make sure that the notifications specified in the subscriptions that match the posted event are doable.

Trap ID  31
eventFailureTrap

Enterprise tuxedo

Variables 
\{ tuxEventsName, tuxEventsSeverity, tuxEventsLmid, tuxEventsTime, 
  tuxEventsUsec, tuxEventsDescription, tuxEventsClass, 
  tuxEventsUlogCat, tuxEventsUlogMsgNum, beaDomainId, beaDomainKey, 
  beaLogicalAgentName \}

Description 
This trap is generated when .SysEventFailure occurs. The system event server periodically sends a message to itself to detect blocking conditions on the message queues. This event is generated if the server cannot put a message on the queue in no-block mode. It can also be generated if the received message does not match what was sent out earlier. The second possible case is very unlikely. This denotes a system event monitor subsystem failure on a particular host.

Action 
Configure larger message queues or distribute the load in the application equally among all the machines.

Trap ID 32
Index

A
absolute OIDs 1-4
access control list
  MIB groups for 6-1
  what it is 6-1
access control lists
  retrieving and modifying values of 6-1
access to Tuxedo System entities, of groups 6-4
active applications
  runtime statistics by machine 2-76
active transactions, attributes of
  as MIB objects 2-136
administrative processes on master machine,
  deactivating or activating 2-36
Agent Integrator
  supported MIB objects 1-5
application attributes
  by machine 2-62
application events 9-1
application queues, attributes of
  as MIB objects 2-89
application queues, features of
  as MIB objects 8-1
application security, type of
  as MIB object 2-46
application users and domains, attributes of
  groups of 6-2
applications, attributes of
  for server group 2-55
attributes of domain

as MIB objects 2-31

B
BBL
  See Also bulletin board 2-47
beaDomainId 4-2
beaDomainKey 4-1
beaDomainList 4-1
beaDomainStatus 4-3
beaDomainTuxconfig 4-3
beaDomainTuxdir 4-2
beaEvtAgentName 2-165
beaEvtExpr 2-166
beaEvtFilter 2-166
beaEvtFilterid 2-165
beaEvtFilterState 2-167
beaEvtFilterTable 2-165
beaIntAgtTable
  role of 1-5
beaLogicalAgentName 4-2
beaRoutingType 2-97
bridge, Tuxedo
  MIB table 2-3
bulletin board
  machine table 2-40
  number of queue table entries 2-49
  number of server table entries 2-50
  number of service table entries 2-50
  number of type table entries 2-50
  queue table 2-41
routing criteria range table 2-41
routing table entries 2-48
server group table entries 2-49
server table 2-42
service table 2-42
starting 2-65
string pool table 2-41

C
client
    network address of 2-18
client, death of
    trap 10-19
client, elapsed time of
    calculation of 2-18
clientConfigTrap 10-18
clientDiedTrap 10-19
clients, attributes of
    as MIB objects 2-9
clientStateTrap 10-20
conversations, attributes of
    as MIB objects 2-21
CORBA 5-4
CORBA interfaces
    configuration and runtime attributes as MIB objects 5-5
customer support contact information vii

D
device lists, Tuxedo System /T
    files storing 2-27
dip-in notification method
    activating 2-44
documentation, where to find it vi
domain, global application attributes of
    as MIB objects 2-31
domains, Tuxedo or wle
    list of as MIB table 4-1
domain-specific traps 10-6

DTP transaction log size, by machine
    as MIB object 2-71

E
encryption level 2-158
tuxTBridgeCurEncryptBits 2-8
enterprise OID
    as value of enterprise field in traps 1-9
entry object in a table 1-8
Event Broker subscriptions
    for service-based notifications 9-15
    for system commands 9-5
    for writing system userlog messages 9-18
event filters, Tuxedo
    as MIB objects 2-165
event subscription requests
    as MIB objects 9-2
eventDeliveryTrap 10-21
eventFailureTrap 10-22

F
factory-based routing specifications 2-94

G
groups with access to Tuxedo System
    as MIB objects 6-4

H
handler, workstation
    See WSH 7-6

L
listener processes, attributes of
    as MIB objects 2-135
listener, workstation
    See WSL 7-9
load balancing
   as MIB object 2-43
local machine, statistics of
   as MIB objects 2-76
logical agent name 4-2
logical machine identifier
   where originator is accessing application
      2-23
logical machines
   network group membership 2-156
logical machines, connectivity of
   as MIB objects 2-3

M
machineBroadcastTrap 10-7
machineConfigTrap 10-7
machineFullMaxAccessesTrap 10-8
machineFullMaxConvTrap 10-9
machineFullMaxGttTrap 10-9
machineFullMaxWsClientsTrap 10-9
machineMsgQTrap 10-10
machinePartitionedTrap 10-10
machineSlowTrap 10-11
machineStateTrap 10-11
master machine
   deactivating or activating administrative
      processes on 2-35
message queues, Tuxedo System/T
   as MIB objects 2-85
message, moving
   via SNMP SET request 8-20
messages, in queues
   as MIB objects 8-17
MIB objects
   values returned when nonexistent 1-7
MIB, TUXEDO standard
   See TMIB 1-9
MIBs
   how to use 1-7

N
network address
   of workstation clients 2-18
network address of logical machine 2-72
network groups, attributes of
   as MIB objects 2-154
networkConfigTrap 10-12
networkDroppedTrap 10-12
networkFailureTrap 10-13
networkFlowTrap 10-13
networkStateTrap 10-14

O
object identifier
   of TUXEDO 1-9
   See OID 1-1
OID
   expressing with textual names 1-4
   relative and absolute 1-4
   use with Agent Integrator polling rules
      1-5
   what it is 1-2
OID tree
   what it is 1-2

P
printing product documentation vi

Q
queue space, attributes of
   as MIB objects 8-24
queue space, state of
   as MIB object 8-27
queue-based notifications
   subscriptions for 9-8
queues
   maximum number of in bulletin board
      queue table 2-41
queues, creating
   via SNMP SET request 8-7
queues, number of
   as MIB object 8-31

R
related information vii
relative OIDs 1-4
resourceConfigTrap 10-6
RFC 1212 1-4
RFC 1215 1-8
routing specifications, of applications
   as MIB objects 2-94

S
security, application
   as MIB object 2-46
security, of application
   as MIB object 2-46
security, Tuxedo 6-1
server group
   application attributes 2-56
server table, extension of 2-113
server workload
   as MIB object 2-117
server, death of
   trap 10-15
serverCleaningTrap 10-14
serverConfigTrap 10-15
serverDiedTrap 10-15
serverInitTrap 10-16
serverMaxgenTrap 10-16
serverRestartingTrap 10-17
servers
   maximum number of in bulletin board
      server table 2-42
servers in an application, attributes of
   as MIB objects 2-99
servers, administrative
   starting 2-65
servers, Tuxedo
   traps specific to 10-14
serverStateTrap 10-17
serverTpExitTrap 10-18
services
   maximum number of in bulletin board
      service table 2-42
services in applications, attributes of
   as MIB objects 2-123
services within an application, attributes of
   as MIB objects 2-129
severity, of system event 10-2
slow, response to DBBL
   trap 10-11
specific trap ID 1-9
support
   technical vii

T
tables
   Entry object under 1-8
timeout, of service requests
   as MIB object 2-132
TMIB 1-7
   differences from SNMP MIBs 1-9
transactions
   traps specific to 10-20
transactions, active
   MIB table for 2-136
   state of 2-138
transactions, attributes of
   as MIB objects 8-42
transHeuristicAbortTrap 10-20
transHeuristicCommitTrap 10-21
traps
   domain-specific 10-6
   machine-specific 10-7
   Tuxedo bridge-specific 10-12
   Tuxedo client-specific 10-18
Tuxedo event-specific 10-21
Tuxedo server-specific 10-14
Tuxedo transaction-specific 10-20

traps, SNMP
generated by Tuxedo agent 10-1

\text{tuxDmAclName} 3-3
\text{tuxDmAcTTable} 3-3
\text{tuxDmConDmCurEncryptBits} 3-6
\text{tuxDmConDmAccessPoint} 3-5
\text{tuxDmConDmRAccessPoint} 3-5
\text{tuxDmConDmType} 3-5
\text{tuxDmConnectionTable} 3-5
\text{tuxDmConState} 3-6
\text{tuxDmExpDmAclName} 3-8
\text{tuxDmExpDmConv} 3-9
\text{tuxDmExpDmInBufType} 3-9
\text{tuxDmExpDmlAccessPoint} 3-8
\text{tuxDmExpDmOutBufType} 3-10
\text{tuxDmExpDmRemoteName} 3-9
\text{tuxDmExpDmResourceName} 3-7
\text{tuxDmExpDmResourceType} 3-9
\text{tuxDmExpDmTopendFunction} 3-11
\text{tuxDmExpDmTopendProduct} 3-10
\text{tuxDmExpDmTopendQualifier} 3-11
\text{tuxDmExpDmTopendRtqGroup} 3-11
\text{tuxDmExpDmTopendRtqName} 3-12
\text{tuxDmExpDmTopendTarget} 3-11
\text{tuxDmExportTable} 3-7
\text{tuxDmExpState} 3-8
\text{tuxDmImpDmAutoTran} 3-15
\text{tuxDmImpDmConv} 3-15
\text{tuxDmImpDmInBufType} 3-17
\text{tuxDmImpDmlAccessPoint} 3-14
\text{tuxDmImpDmLoad} 3-15
\text{tuxDmImpDmOutBufType} 3-17
\text{tuxDmImpDmPrio} 3-15
\text{tuxDmImpDmrAccessPointList} 3-14
\text{tuxDmImpDmRemoteName} 3-16
\text{tuxDmImpDmResourceName} 3-14
\text{tuxDmImpDmResourceType} 3-16
\text{tuxDmImpDmRoutingName} 3-16
\text{tuxDmImpDmteFunction} 3-17
\text{tuxDmImpDmteProduct} 3-17
\text{tuxDmImpDmteQualifier} 3-18
\text{tuxDmImpDmteRtqGroup} 3-18
\text{tuxDmImpDmteRtqName} 3-19
\text{tuxDmImpDmteTarget} 3-18
\text{tuxDmImpDmTranTime} 3-16
\text{tuxDmImportTable} 3-13
\text{tuxDmImpState} 3-14
\text{tuxDmLclDmAccessPoint} 3-21
\text{tuxDmLclDmAuditLog} 3-22
\text{tuxDmLclDmBlobShmSize} 3-27
\text{tuxDmLclDmBlockTime} 3-22
\text{tuxDmLclDmConnectionPolicy} 3-24
\text{tuxDmLclDmConnPrincipalName} 3-26
\text{tuxDmLclDmMachineType} 3-26
\text{tuxDmLclDmMaxRapTran} 3-22
\text{tuxDmLclDmMaxRetry} 3-25
\text{tuxDmLclDmMaxTran} 3-23
\text{tuxDmLclDmRetryInterval} 3-25
\text{tuxDmLclDmSecurity} 3-23
\text{tuxDmLclDmSrvGroup} 3-21
\text{tuxDmLclDmTlogDev} 3-24
\text{tuxDmLclDmTlogName} 3-24
\text{tuxDmLclDmTlogSize} 3-24
\text{tuxDmLclDmType} 3-21
\text{tuxDmLclState} 3-21
\text{tuxDmLocalTable} 3-20
\text{tuxDmOsiDmAccessPoint} 3-28
\text{tuxDmOsiDmAcn} 3-29
\text{tuxDmOsiDmAeid} 3-30
\text{tuxDmOsiDmAeq} 3-29
\text{tuxDmOsiDmApid} 3-30
\text{tuxDmOsiDmApt} 3-29
\text{tuxDmOsiDmMaxListeningEp} 3-30
\text{tuxDmOsiDmNwDevice} 3-29
\text{tuxDmOsiDmState} 3-28
\text{tuxDmOsiDmUrch} 3-30
\text{tuxDmOsiDmXatmiEncoding} 3-31
\text{tuxDmOsiDmTable} 3-28
<table>
<thead>
<tr>
<th>Variable</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxDmPasswdDmlAccessPoint</td>
<td>3-32</td>
</tr>
<tr>
<td>tuxDmPasswdDmlPWD</td>
<td>3-32</td>
</tr>
<tr>
<td>tuxDmPasswdDmrAccessPoint</td>
<td>3-32</td>
</tr>
<tr>
<td>tuxDmPasswdDmrPWD</td>
<td>3-33</td>
</tr>
<tr>
<td>tuxDmPasswdState</td>
<td>3-33</td>
</tr>
<tr>
<td>tuxDmPasswordTable</td>
<td>3-32</td>
</tr>
<tr>
<td>tuxDmPrincipalMapTable</td>
<td>3-34</td>
</tr>
<tr>
<td>tuxDmPrincipalMapDirection</td>
<td>3-35</td>
</tr>
<tr>
<td>tuxDmPrincipalMapDmlAccessPoint</td>
<td>3-34</td>
</tr>
<tr>
<td>tuxDmPrincipalMapDmlPrinName</td>
<td>3-34</td>
</tr>
<tr>
<td>tuxDmPrincipalMapDmrAccessPoint</td>
<td>3-34</td>
</tr>
<tr>
<td>tuxDmPrincipalMapDmrPrinName</td>
<td>3-35</td>
</tr>
<tr>
<td>tuxDmPrincipalMapState</td>
<td>3-35</td>
</tr>
<tr>
<td>tuxDmRemoteDmlAccessPoint</td>
<td>3-36</td>
</tr>
<tr>
<td>tuxDmRemoteDmlAccessPointID</td>
<td>3-36</td>
</tr>
<tr>
<td>tuxDmRemoteDmCodePage</td>
<td>3-37</td>
</tr>
<tr>
<td>tuxDmRemoteDmMachineType</td>
<td>3-38</td>
</tr>
<tr>
<td>tuxDmRemoteState</td>
<td>3-37</td>
</tr>
<tr>
<td>tuxDmRemoteTable</td>
<td>3-36</td>
</tr>
<tr>
<td>tuxDmRemoteType</td>
<td>3-37</td>
</tr>
<tr>
<td>tuxDmResourcesDmVersion</td>
<td>3-39</td>
</tr>
<tr>
<td>tuxDmResourcesTable</td>
<td>3-39</td>
</tr>
<tr>
<td>tuxDmRoutingDmBufType</td>
<td>3-40</td>
</tr>
<tr>
<td>tuxDmRoutingDmField</td>
<td>3-41</td>
</tr>
<tr>
<td>tuxDmRoutingDmFieldType</td>
<td>3-42</td>
</tr>
<tr>
<td>tuxDmRoutingDmRanges</td>
<td>3-42</td>
</tr>
<tr>
<td>tuxDmRoutingDmRoutingName</td>
<td>3-40</td>
</tr>
<tr>
<td>tuxDmRoutingState</td>
<td>3-43</td>
</tr>
<tr>
<td>tuxDmRoutingTable</td>
<td>3-40</td>
</tr>
<tr>
<td>tuxDmrPrincipalDmrAccessPoint</td>
<td>3-44</td>
</tr>
<tr>
<td>tuxDmrPrincipalDmrPrinName</td>
<td>3-44</td>
</tr>
<tr>
<td>tuxDmrPrincipalDmrPrinPasswd</td>
<td>3-44</td>
</tr>
<tr>
<td>tuxDmrPrincipalState</td>
<td>3-45</td>
</tr>
<tr>
<td>tuxDmrPrincipalTable</td>
<td>3-44</td>
</tr>
<tr>
<td>tuxDMSnaCRMDmAccessPoint</td>
<td>3-46</td>
</tr>
<tr>
<td>tuxDMSnaCRMDmNWAddr</td>
<td>3-47</td>
</tr>
<tr>
<td>tuxDMSnaCRMDmNWDevice</td>
<td>3-47</td>
</tr>
<tr>
<td>tuxDMSnaCRMDmSNACRM</td>
<td>3-46</td>
</tr>
<tr>
<td>tuxDMSnaCRMState</td>
<td>3-46</td>
</tr>
<tr>
<td>tuxDMSnaCRMTable</td>
<td>3-46</td>
</tr>
</tbody>
</table>
tuxDmTransactionDmBranchIndex 3-63
 tuxDmTransactionDmBranchNo 3-64
 tuxDmTransactionDmBranchState 3-64
 tuxDmTransactionDmNetTranID 3-64
 tuxDmTransactionDmTPTranID 3-62
 tuxDmTransactionDmTxAccessPoint 3-63
 tuxDmTransactionDmTxNetTranID 3-63
 tuxDmTransactionState 3-62
 tuxDmTransactionTable 3-61
 TUXEDO
   object identifier of 1-9
 Tuxedo application, access to 6-6
 Tuxedo configuration file, by machine
   as MIB object 2-64
 Tuxedo events
   types of 9-1
 TUXEDO MIB
   See TMIB 1-1
 Tuxedo software pathname, by machine
   as MIB object 2-65
 Tuxedo software, pathname to
   as MIB object 4-2
 TUXEDO standard MIB
   See TMIB 1-10
 Tuxedo System /T
   listener processes 2-135
 Tuxedo system events 9-1
 Tuxedo, states of 1-5
 tuxEventClientExpr 9-3
 tuxEventClientFilter 9-4
 tuxEventClientId 9-4
 tuxEventClientIndx 9-3
 tuxEventClientState 9-4
 tuxEventClientTbl 9-3
 tuxEventCmd 9-7
 tuxEventCmdExpr 9-6
 tuxEventCmdFlter 9-6
 tuxEventCmdIndx 9-5
 tuxEventCmdState 9-6
 tuxEventCmdTbl 9-5
 tuxEventPersist 9-13
 tuxEventQctlBeforeMsgid 9-11
 tuxEventQctlCorrId 9-13
 tuxEventQctlDeqTime 9-12
 tuxEventQctlFailQ 9-13
 tuxEventQctlMsgid 9-12
 tuxEventQctlPrior 9-12
 tuxEventQctlQtimeAbs 9-11
 tuxEventQctlQtimeRel 9-12
 tuxEventQctlQtop 9-11
 tuxEventQctlReplyQ 9-13
 tuxEventQname 9-10
 tuxEventQSpace 9-10
 tuxEventQueExpr 9-9
 tuxEventQueFilter 9-9
 tuxEventQueIndx 9-9
 tuxEventQueState 9-10
 tuxEventQueTbl 9-8
 tuxEventsClass 10-4
 tuxEventsDescription 10-3
 tuxEventsLmid 10-3
 tuxEventsName 10-2
 tuxEventsSeverity 10-2
 tuxEventsTime 10-3
 tuxEventsUlogCat 10-4
 tuxEventsUlogMsgNum 10-4
 tuxEventsUsec 10-3
 tuxEventSvcExpr 9-16
 tuxEventSvcFilter 9-16
 tuxEventSvcIndx 9-15
 tuxEventSvcName 9-17
 tuxEventSvcPersist 9-17
 tuxEventSvcState 9-16
 tuxEventSvcTbl 9-15
 tuxEventSvcTran 9-17
 tuxEventTran 9-14
 tuxEventUlogExpr 9-19
 tuxEventUlogFilter 9-19
 tuxEventUlogIndx 9-18
 tuxEventUlogState 9-19
 tuxEventUlogTbl 9-18
I-8     BEA SNMP Agent MIB Reference
tuxTclientBirthTime 2-12

<table>
<thead>
<tr>
<th>tuxTclientCClnName 2-13</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTclientCmtRet 2-17</td>
</tr>
<tr>
<td>tuxTclientContextID 2-20</td>
</tr>
<tr>
<td>tuxTclientCurConv 2-17</td>
</tr>
<tr>
<td>tuxTclientCurReq 2-18</td>
</tr>
<tr>
<td>tuxTclientCurTime 2-18</td>
</tr>
<tr>
<td>tuxTclientId 2-20</td>
</tr>
<tr>
<td>tuxTclientIdleTime 2-13</td>
</tr>
<tr>
<td>tuxTclientLastGrp 2-18</td>
</tr>
<tr>
<td>tuxTclientMachineId 2-12</td>
</tr>
<tr>
<td>tuxTclientNotify 2-19</td>
</tr>
<tr>
<td>tuxTclientNumConv 2-15</td>
</tr>
<tr>
<td>tuxTclientNumDeque 2-16</td>
</tr>
<tr>
<td>tuxTclientNumEnque 2-16</td>
</tr>
<tr>
<td>tuxTclientNumPost 2-16</td>
</tr>
<tr>
<td>tuxTclientNumReq 2-16</td>
</tr>
<tr>
<td>tuxTclientNumSubscribe 2-16</td>
</tr>
<tr>
<td>tuxTclientNumTran 2-17</td>
</tr>
<tr>
<td>tuxTclientNumTranAbt 2-17</td>
</tr>
<tr>
<td>tuxTclientNumTranCmt 2-17</td>
</tr>
<tr>
<td>tuxTclientNumUnSol 2-19</td>
</tr>
<tr>
<td>tuxTclientPid 2-13</td>
</tr>
<tr>
<td>tuxTclientReg 2-13</td>
</tr>
<tr>
<td>tuxTclientRelease 2-15</td>
</tr>
<tr>
<td>tuxTclientRpid 2-19</td>
</tr>
<tr>
<td>tuxTclientSrvGrp 2-14</td>
</tr>
<tr>
<td>tuxTclientState 2-11</td>
</tr>
<tr>
<td>tuxTclientTbl 2-9</td>
</tr>
<tr>
<td>tuxTclientTimeLeft 2-19</td>
</tr>
<tr>
<td>tuxTclientTimeStart 2-20</td>
</tr>
<tr>
<td>tuxTclientTranLev 2-20</td>
</tr>
<tr>
<td>tuxTclientUsrName 2-14</td>
</tr>
<tr>
<td>tuxTclientWsc 2-14</td>
</tr>
<tr>
<td>tuxTclientWsh 2-14</td>
</tr>
<tr>
<td>tuxTclientWshClientId 2-15</td>
</tr>
<tr>
<td>tuxTclientWsProto 2-15</td>
</tr>
<tr>
<td>tuxTconnClientId 2-23</td>
</tr>
<tr>
<td>tuxTconnOgrpNo 2-23</td>
</tr>
<tr>
<td>tuxTconnOlimid 2-23</td>
</tr>
</tbody>
</table>

<p>| tuxTconnOpid 2-23      |
| tuxTconnOsndcnt 2-23   |
| tuxTconnOsrvld 2-24    |
| tuxTconnSgrpNo 2-24    |
| tuxTconnSmtdcnt 2-24   |
| tuxTconnSpid 2-24      |
| tuxTconnSsrsvld 2-25   |
| tuxTconnState 2-22     |
| tuxTconnSvcName 2-22   |
| tuxTconnTable 2-21     |
| tuxTdevCfgDev 2-28     |
| tuxTdeviceName 2-28    |
| tuxTdeviceTbl 2-27     |
| tuxTDevIndex 2-29      |
| tuxTDevLmid 2-27       |
| tuxTDevOffset 2-28     |
| tuxTDevSize 2-28       |
| tuxTDevState 2-29      |
| tuxTdomain 2-31        |
| tuxTdomainAuthsvc 2-47 |
| tuxTdomainBBLQuery 2-47|
| tuxTdomainBlockTime 2-48|
| tuxTdomainCMTRT 2-43   |
| tuxTdomainCurDRT 2-48  |
| tuxTdomainCurGroups 2-49|
| tuxTdomainCurMachines 2-49|
| tuxTdomainCurQueues 2-49|
| tuxTdomainCurRFT 2-49  |
| tuxTdomainCurServers 2-49|
| tuxTdomainCurServices 2-50|
| tuxTdomainCursType 2-50|
| tuxTdomainDbblWait 2-48|
| tuxTdomainEncryptionRequired 2-54|
| tuxTdomainGID 2-37     |
| tuxTdomainHwDRT 2-50   |
| tuxTdomainHwGroups 2-51|
| tuxTdomainHWmachines 2-51|</p>
<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Reference Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTdomainHwQueues</td>
<td>2-51</td>
</tr>
<tr>
<td>tuxTdomainHwRFT</td>
<td>2-51</td>
</tr>
<tr>
<td>tuxTdomainHwRTdata</td>
<td>2-51</td>
</tr>
<tr>
<td>tuxTdomainHwServers</td>
<td>2-52</td>
</tr>
<tr>
<td>tuxTdomainID</td>
<td>2-37</td>
</tr>
<tr>
<td>tuxTdomainKey</td>
<td>2-34</td>
</tr>
<tr>
<td>tuxTdomainLoadBalance</td>
<td>2-43</td>
</tr>
<tr>
<td>tuxTdomainMask</td>
<td>2-38</td>
</tr>
<tr>
<td>tuxTdomainMaster</td>
<td>2-34</td>
</tr>
<tr>
<td>tuxTdomainMaxAccessers</td>
<td>2-38</td>
</tr>
<tr>
<td>tuxTdomainMaxACLgroups</td>
<td>2-42</td>
</tr>
<tr>
<td>tuxTdomainMaxBufsType</td>
<td>2-39</td>
</tr>
<tr>
<td>tuxTdomainMaxBufType</td>
<td>2-39</td>
</tr>
<tr>
<td>tuxTdomainMaxConv</td>
<td>2-38</td>
</tr>
<tr>
<td>tuxTdomainMaxDRT</td>
<td>2-39</td>
</tr>
<tr>
<td>tuxTdomainMaxGroups</td>
<td>2-40</td>
</tr>
<tr>
<td>tuxTdomainMaxGTT</td>
<td>2-39</td>
</tr>
<tr>
<td>tuxTdomainMaxMachines</td>
<td>2-40</td>
</tr>
<tr>
<td>tuxTdomainMaxNetGroups</td>
<td>2-52</td>
</tr>
<tr>
<td>tuxTdomainMaxQueues</td>
<td>2-41</td>
</tr>
<tr>
<td>tuxTdomainMaxRFT</td>
<td>2-41</td>
</tr>
<tr>
<td>tuxTdomainMaxRTData</td>
<td>2-41</td>
</tr>
<tr>
<td>tuxTdomainMaxServers</td>
<td>2-42</td>
</tr>
<tr>
<td>tuxTdomainMaxServices</td>
<td>2-42</td>
</tr>
<tr>
<td>tuxTdomainMode</td>
<td>2-35</td>
</tr>
<tr>
<td>tuxTdomainNotify</td>
<td>2-43</td>
</tr>
<tr>
<td>tuxTdomainOptions</td>
<td>2-45</td>
</tr>
<tr>
<td>tuxTdomainPerm</td>
<td>2-37</td>
</tr>
<tr>
<td>tuxTdomainSanityScan</td>
<td>2-48</td>
</tr>
<tr>
<td>tuxTdomainScanUnit</td>
<td>2-47</td>
</tr>
<tr>
<td>tuxTdomainSecurity</td>
<td>2-46</td>
</tr>
<tr>
<td>tuxTdomainSignal</td>
<td>2-45</td>
</tr>
<tr>
<td>tuxTdomainSignatureAhead</td>
<td>2-53</td>
</tr>
<tr>
<td>tuxTdomainSignatureBehind</td>
<td>2-54</td>
</tr>
<tr>
<td>tuxTdomainSignatureRequired</td>
<td>2-54</td>
</tr>
<tr>
<td>tuxTdomainState</td>
<td>2-35</td>
</tr>
<tr>
<td>tuxTdomainSystemAccess</td>
<td>2-44</td>
</tr>
<tr>
<td>tuxTdomainUID</td>
<td>2-37</td>
</tr>
<tr>
<td>tuxTgroupCloseInfo</td>
<td>2-59</td>
</tr>
<tr>
<td>tuxTgroupLMID</td>
<td>2-61</td>
</tr>
<tr>
<td>tuxTgroupLMID</td>
<td>2-56</td>
</tr>
<tr>
<td>tuxTgroupName</td>
<td>2-55</td>
</tr>
<tr>
<td>tuxTgroupNo</td>
<td>2-56</td>
</tr>
<tr>
<td>tuxTgroupOpenInfo</td>
<td>2-59</td>
</tr>
<tr>
<td>tuxTgroupSignatureRequired</td>
<td>2-61</td>
</tr>
<tr>
<td>tuxTgroupState</td>
<td>2-57</td>
</tr>
<tr>
<td>tuxTgroupTable</td>
<td>2-55</td>
</tr>
<tr>
<td>tuxTgroupTMScount</td>
<td>2-60</td>
</tr>
<tr>
<td>tuxTgroupTMName</td>
<td>2-60</td>
</tr>
<tr>
<td>tuxTlistenLmid</td>
<td>2-135</td>
</tr>
<tr>
<td>tuxTlistenState</td>
<td>2-135</td>
</tr>
<tr>
<td>tuxTlistenTbl</td>
<td>2-135</td>
</tr>
<tr>
<td>tuxTmachineAclCacheAccess</td>
<td>2-83</td>
</tr>
<tr>
<td>tuxTmachineAclCacheHits</td>
<td>2-83</td>
</tr>
<tr>
<td>tuxTmachineAclFail</td>
<td>2-83</td>
</tr>
<tr>
<td>tuxTmachineActive</td>
<td>2-76</td>
</tr>
<tr>
<td>tuxTmachineAppDir</td>
<td>2-65</td>
</tr>
<tr>
<td>tuxTmachineBridge</td>
<td>2-71</td>
</tr>
<tr>
<td>tuxTmachineCmpLimit</td>
<td>2-72</td>
</tr>
<tr>
<td>tuxTmachineCurAccessers</td>
<td>2-77</td>
</tr>
<tr>
<td>tuxTmachineCurClients</td>
<td>2-78</td>
</tr>
<tr>
<td>tuxTmachineCurConv</td>
<td>2-78</td>
</tr>
<tr>
<td>tuxTmachineCurGTT</td>
<td>2-78</td>
</tr>
<tr>
<td>tuxTmachineCurLoad</td>
<td>2-78</td>
</tr>
<tr>
<td>tuxTmachineCurWsClients</td>
<td>2-78</td>
</tr>
<tr>
<td>tuxTmachineEncryptionRequired</td>
<td>2-75</td>
</tr>
<tr>
<td>tuxTmachineEnvFile</td>
<td>2-67</td>
</tr>
<tr>
<td>tuxTmachineGid</td>
<td>2-67</td>
</tr>
<tr>
<td>tuxTmachineHwAccessers</td>
<td>2-79</td>
</tr>
<tr>
<td>tuxTmachineHwAclCache</td>
<td>2-82</td>
</tr>
<tr>
<td>tuxTmachineHwClients</td>
<td>2-79</td>
</tr>
<tr>
<td>tuxTmachineHwConv</td>
<td>2-79</td>
</tr>
<tr>
<td>tuxTmachineHwGTT</td>
<td>2-79</td>
</tr>
<tr>
<td>tuxTmachineHwWsClients</td>
<td>2-79</td>
</tr>
<tr>
<td>tuxTmachineLicExpires</td>
<td>2-81</td>
</tr>
<tr>
<td>tuxTmachineLicMaxUsers</td>
<td>2-82</td>
</tr>
<tr>
<td>tuxTmachineLicSerial</td>
<td>2-82</td>
</tr>
<tr>
<td>tuxTmachineLmid</td>
<td>2-64</td>
</tr>
<tr>
<td>tuxTmachineLogDevice</td>
<td>2-70</td>
</tr>
<tr>
<td>tuxTmachineMaxAccessers</td>
<td>2-69</td>
</tr>
<tr>
<td>Variable</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>tuxTmachineMaxActCache</td>
<td>2-70</td>
</tr>
<tr>
<td>tuxTmachineMaxConv</td>
<td>2-69</td>
</tr>
<tr>
<td>tuxTmachineMaxGtt</td>
<td>2-69</td>
</tr>
<tr>
<td>tuxTmachineMaxWsClients</td>
<td>2-70</td>
</tr>
<tr>
<td>tuxTmachineMinor</td>
<td>2-74</td>
</tr>
<tr>
<td>tuxTmachineNaddr</td>
<td>2-71</td>
</tr>
<tr>
<td>tuxTmachineNlsaddr</td>
<td>2-72</td>
</tr>
<tr>
<td>tuxTmachineNumConv</td>
<td>2-80</td>
</tr>
<tr>
<td>tuxTmachineNumDequeue</td>
<td>2-80</td>
</tr>
<tr>
<td>tuxTmachineNumEnqueue</td>
<td>2-80</td>
</tr>
<tr>
<td>tuxTmachineNumPost</td>
<td>2-80</td>
</tr>
<tr>
<td>tuxTmachineNumSubscribe</td>
<td>2-81</td>
</tr>
<tr>
<td>tuxTmachineNumTran</td>
<td>2-81</td>
</tr>
<tr>
<td>tuxTmachineNumTranAbt</td>
<td>2-81</td>
</tr>
<tr>
<td>tuxTmachineNumTranCmt</td>
<td>2-81</td>
</tr>
<tr>
<td>tuxTmachinePageSize</td>
<td>2-82</td>
</tr>
<tr>
<td>tuxTmachinePerm</td>
<td>2-68</td>
</tr>
<tr>
<td>tuxTmachinePmid</td>
<td>2-64</td>
</tr>
<tr>
<td>tuxTmachineRelease</td>
<td>2-74</td>
</tr>
<tr>
<td>tuxTmachineRole</td>
<td>2-73</td>
</tr>
<tr>
<td>tuxTmachineSignatureRequired</td>
<td>2-75</td>
</tr>
<tr>
<td>tuxTmachineSpinCount</td>
<td>2-73</td>
</tr>
<tr>
<td>tuxTmachineState</td>
<td>2-65</td>
</tr>
<tr>
<td>tuxTmachineSWrelease</td>
<td>2-82</td>
</tr>
<tr>
<td>tuxTmachineTable</td>
<td>2-62</td>
</tr>
<tr>
<td>tuxTmachineTlogName</td>
<td>2-71</td>
</tr>
<tr>
<td>tuxTmachineTlogSize</td>
<td>2-71</td>
</tr>
<tr>
<td>tuxTmachineTmNetLoad</td>
<td>2-73</td>
</tr>
<tr>
<td>tuxTmachineTuxConfig</td>
<td>2-64</td>
</tr>
<tr>
<td>tuxTmachineTuxDir</td>
<td>2-64</td>
</tr>
<tr>
<td>tuxTmachineType</td>
<td>2-68</td>
</tr>
<tr>
<td>tuxTmachineUid</td>
<td>2-67</td>
</tr>
<tr>
<td>tuxTmachineUlogPfx</td>
<td>2-68</td>
</tr>
<tr>
<td>tuxTmachineWkComplete</td>
<td>2-83</td>
</tr>
<tr>
<td>tuxTmachineWkInitiated</td>
<td>2-84</td>
</tr>
<tr>
<td>tuxTmsgCbytes</td>
<td>2-86</td>
</tr>
<tr>
<td>tuxTmsgCtime</td>
<td>2-87</td>
</tr>
<tr>
<td>tuxTmsgCurTime</td>
<td>2-86</td>
</tr>
<tr>
<td>tuxTmsgId</td>
<td>2-85</td>
</tr>
<tr>
<td>tuxTmsgLrPid</td>
<td>2-87</td>
</tr>
</tbody>
</table>

**BEA SNMP Agent MIB Reference**
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>tuxTQspaceName</td>
<td>8-26</td>
</tr>
<tr>
<td>tuxTQspacePercentInit</td>
<td>8-33</td>
</tr>
<tr>
<td>tuxTQspaceQmConfig</td>
<td>8-26</td>
</tr>
<tr>
<td>tuxTQspaceState</td>
<td>8-27</td>
</tr>
<tr>
<td>tuxTQspaceTbl</td>
<td>8-24</td>
</tr>
<tr>
<td>tuxTQtransGrpNo</td>
<td>8-44</td>
</tr>
<tr>
<td>tuxTQtransIndx1</td>
<td>8-43</td>
</tr>
<tr>
<td>tuxTQtransIndx2</td>
<td>8-43</td>
</tr>
<tr>
<td>tuxTQtransIndx3</td>
<td>8-43</td>
</tr>
<tr>
<td>tuxTQtransIndx4</td>
<td>8-43</td>
</tr>
<tr>
<td>tuxTQtransIndx5</td>
<td>8-44</td>
</tr>
<tr>
<td>tuxTQtransLmid</td>
<td>8-45</td>
</tr>
<tr>
<td>tuxTQtransQmConfig</td>
<td>8-44</td>
</tr>
<tr>
<td>tuxTQtransSpaceName</td>
<td>8-44</td>
</tr>
<tr>
<td>tuxTQtransState</td>
<td>8-45</td>
</tr>
<tr>
<td>tuxTQtransTbl</td>
<td>8-42</td>
</tr>
<tr>
<td>tuxTQtransXid</td>
<td>8-42</td>
</tr>
<tr>
<td>tuxTqueueNqueued</td>
<td>2-92</td>
</tr>
<tr>
<td>tuxTqueueRqAddr</td>
<td>2-89</td>
</tr>
<tr>
<td>tuxTqueueRqId</td>
<td>2-91</td>
</tr>
<tr>
<td>tuxTqueueSource</td>
<td>2-92</td>
</tr>
<tr>
<td>tuxTqueueSrvrCnt</td>
<td>2-91</td>
</tr>
<tr>
<td>tuxTqueueState</td>
<td>2-90</td>
</tr>
<tr>
<td>tuxTqueueTable</td>
<td>2-89</td>
</tr>
<tr>
<td>tuxTqueueTotNqueued</td>
<td>2-91</td>
</tr>
<tr>
<td>tuxTqueueWkQueued</td>
<td>2-92</td>
</tr>
<tr>
<td>tuxTqueueWkQueued</td>
<td>2-93</td>
</tr>
<tr>
<td>tuxTranCoordLmid</td>
<td>2-136</td>
</tr>
<tr>
<td>tuxTranGrpCnt</td>
<td>2-140</td>
</tr>
<tr>
<td>tuxTranGrpIndex</td>
<td>2-140</td>
</tr>
<tr>
<td>tuxTranGrpNo</td>
<td>2-140</td>
</tr>
<tr>
<td>tuxTranGstate</td>
<td>2-140</td>
</tr>
<tr>
<td>tuxTranIndx1</td>
<td>2-137</td>
</tr>
<tr>
<td>tuxTranIndx2</td>
<td>2-137</td>
</tr>
<tr>
<td>tuxTranIndx3</td>
<td>2-137</td>
</tr>
<tr>
<td>tuxTranIndx4</td>
<td>2-138</td>
</tr>
<tr>
<td>tuxTranIndx5</td>
<td>2-138</td>
</tr>
<tr>
<td>tuxTranState</td>
<td>2-138</td>
</tr>
<tr>
<td>tuxTranTbl</td>
<td>2-136</td>
</tr>
<tr>
<td>tuxTranTimeOut</td>
<td>2-140</td>
</tr>
<tr>
<td>tuxTranXid</td>
<td>2-137</td>
</tr>
<tr>
<td>tuxTranzBufferType</td>
<td>2-95</td>
</tr>
<tr>
<td>tuxTranzField</td>
<td>2-95</td>
</tr>
<tr>
<td>tuxTranzName</td>
<td>2-95</td>
</tr>
<tr>
<td>tuxTranzRanges</td>
<td>2-96</td>
</tr>
<tr>
<td>tuxTranzState</td>
<td>2-97</td>
</tr>
<tr>
<td>tuxTranzTable</td>
<td>2-94</td>
</tr>
<tr>
<td>tuxTranzQmConfig</td>
<td>2-92</td>
</tr>
<tr>
<td>tuxTranzSrvrCnt</td>
<td>2-91</td>
</tr>
<tr>
<td>tuxTranzSrvrState</td>
<td>2-92</td>
</tr>
<tr>
<td>tuxTranzSrvrTable</td>
<td>2-91</td>
</tr>
<tr>
<td>tuxTranzSrvrSrvId</td>
<td>2-91</td>
</tr>
<tr>
<td>tuxTranzSrvrClOpt</td>
<td>2-94</td>
</tr>
<tr>
<td>tuxTranzSrvrCltLmid</td>
<td>2-94</td>
</tr>
<tr>
<td>tuxTranzSrvrCltPid</td>
<td>2-94</td>
</tr>
<tr>
<td>tuxTranzSrvrCltReply</td>
<td>2-94</td>
</tr>
<tr>
<td>tuxTranzSrvrCltRet</td>
<td>2-94</td>
</tr>
<tr>
<td>tuxTranzSrvrConv</td>
<td>2-94</td>
</tr>
<tr>
<td>tuxTranzSrvrTimeout</td>
<td>2-94</td>
</tr>
<tr>
<td>tuxTranzSrvrTimeLeft</td>
<td>2-94</td>
</tr>
<tr>
<td>tuxTranzSrvrTranLev</td>
<td>2-94</td>
</tr>
<tr>
<td>tuxTsrvrBaseSrvId</td>
<td>2-104</td>
</tr>
<tr>
<td>tuxTsrvrClOpt</td>
<td>2-104</td>
</tr>
<tr>
<td>tuxTsrvrCltLmid</td>
<td>2-117</td>
</tr>
<tr>
<td>tuxTsrvrCltPid</td>
<td>2-117</td>
</tr>
<tr>
<td>tuxTsrvrCltReply</td>
<td>2-118</td>
</tr>
<tr>
<td>tuxTsrvrCltRet</td>
<td>2-118</td>
</tr>
<tr>
<td>tuxTsrvrConv</td>
<td>2-108</td>
</tr>
<tr>
<td>tuxTsrvrCurConv</td>
<td>2-118</td>
</tr>
<tr>
<td>tuxTsrvrCurDispatchThreads</td>
<td>2-122</td>
</tr>
<tr>
<td>tuxTsrvrCurReq</td>
<td>2-118</td>
</tr>
<tr>
<td>tuxTsrvrCurService</td>
<td>2-119</td>
</tr>
<tr>
<td>tuxTsrvrCurTime</td>
<td>2-119</td>
</tr>
<tr>
<td>tuxTsrvrEnvFile</td>
<td>2-104</td>
</tr>
<tr>
<td>tuxTsrvrGeneration</td>
<td>2-109</td>
</tr>
<tr>
<td>tuxTsrvrGrace</td>
<td>2-104</td>
</tr>
<tr>
<td>tuxTsrvrGrp</td>
<td>2-100</td>
</tr>
<tr>
<td>tuxTsrvrGrpExt</td>
<td>2-120</td>
</tr>
<tr>
<td>tuxTsrvrGrpNo</td>
<td>2-101</td>
</tr>
<tr>
<td>tuxTsrvrGrpNoExt</td>
<td>2-115</td>
</tr>
<tr>
<td>tuxTsrvrHwDispatchThreads</td>
<td>2-122</td>
</tr>
<tr>
<td>tuxTsrvrId</td>
<td>2-101</td>
</tr>
<tr>
<td>MIB Variable</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>tuxTsrvrIdExt</td>
<td>2-114</td>
</tr>
<tr>
<td>tuxTsrvrLastGrp</td>
<td>2-119</td>
</tr>
<tr>
<td>tuxTsrvrMax</td>
<td>2-106</td>
</tr>
<tr>
<td>tuxTsrvrMaxDispatchThreads</td>
<td>2-111</td>
</tr>
<tr>
<td>tuxTsrvrMaxgen</td>
<td>2-105</td>
</tr>
<tr>
<td>tuxTsrvrMin</td>
<td>2-106</td>
</tr>
<tr>
<td>tuxTsrvrMinDispatchThreads</td>
<td>2-111</td>
</tr>
<tr>
<td>tuxTsrvrName</td>
<td>2-101</td>
</tr>
<tr>
<td>tuxTsrvrNumConv</td>
<td>2-115</td>
</tr>
<tr>
<td>tuxTsrvrNumDeque</td>
<td>2-115</td>
</tr>
<tr>
<td>tuxTsrvrNumDispatchThreads</td>
<td>2-122</td>
</tr>
<tr>
<td>tuxTsrvrNumEnque</td>
<td>2-115</td>
</tr>
<tr>
<td>tuxTsrvrNumPost</td>
<td>2-115</td>
</tr>
<tr>
<td>tuxTsrvrNumReq</td>
<td>2-116</td>
</tr>
<tr>
<td>tuxTsrvrNumSubscribe</td>
<td>2-116</td>
</tr>
<tr>
<td>tuxTsrvrNumTran</td>
<td>2-116</td>
</tr>
<tr>
<td>tuxTsrvrPid</td>
<td>2-109</td>
</tr>
<tr>
<td>tuxTsrvrRcmd</td>
<td>2-106</td>
</tr>
<tr>
<td>tuxTsrvrReplyQ</td>
<td>2-108</td>
</tr>
<tr>
<td>tuxTsrvrRestart</td>
<td>2-107</td>
</tr>
<tr>
<td>tuxTsrvrRpPerm</td>
<td>2-110</td>
</tr>
<tr>
<td>tuxTsrvrRqAddr</td>
<td>2-109</td>
</tr>
<tr>
<td>tuxTsrvrRqld</td>
<td>2-110</td>
</tr>
<tr>
<td>tuxTsrvrRqPerm</td>
<td>2-109</td>
</tr>
<tr>
<td>tuxTsrvrSequence</td>
<td>2-107</td>
</tr>
<tr>
<td>tuxTsrvrState</td>
<td>2-101</td>
</tr>
<tr>
<td>tuxTsrvrStateExt</td>
<td>2-120</td>
</tr>
<tr>
<td>tuxTsrvrSvcTimeOut</td>
<td>2-119</td>
</tr>
<tr>
<td>tuxTsrvrSystemAccess</td>
<td>2-108</td>
</tr>
<tr>
<td>tuxTsrvrTbl</td>
<td>2-99</td>
</tr>
<tr>
<td>tuxTsrvrTblExt</td>
<td>2-113</td>
</tr>
<tr>
<td>tuxTsrvrThreadStackSize</td>
<td>2-112</td>
</tr>
<tr>
<td>tuxTsrvrTimeLeft</td>
<td>2-119</td>
</tr>
<tr>
<td>tuxTsrvrTimeRestart</td>
<td>2-110</td>
</tr>
<tr>
<td>tuxTsrvrTimeStart</td>
<td>2-111</td>
</tr>
<tr>
<td>tuxTsrvrTotReqC</td>
<td>2-117</td>
</tr>
<tr>
<td>tuxTsrvrTotWorkL</td>
<td>2-117</td>
</tr>
<tr>
<td>tuxTsrvrTranAbt</td>
<td>2-116</td>
</tr>
<tr>
<td>tuxTsrvrTranCmt</td>
<td>2-116</td>
</tr>
<tr>
<td>tuxTsrvrTranLev</td>
<td>2-120</td>
</tr>
<tr>
<td>tuxTsvcAutoTran</td>
<td>2-125</td>
</tr>
<tr>
<td>tuxTsvcBufType</td>
<td>2-127</td>
</tr>
<tr>
<td>tuxTsvcEncryptionRequired</td>
<td>2-128</td>
</tr>
<tr>
<td>tuxTsvcGrp</td>
<td>2-129</td>
</tr>
<tr>
<td>tuxTsvcGrpAutoTran</td>
<td>2-132</td>
</tr>
<tr>
<td>tuxTsvcGrpLoad</td>
<td>2-132</td>
</tr>
<tr>
<td>tuxTsvcGrpName</td>
<td>2-131</td>
</tr>
<tr>
<td>tuxTsvcGrpNo</td>
<td>2-131</td>
</tr>
<tr>
<td>tuxTsvcGrpPrio</td>
<td>2-132</td>
</tr>
<tr>
<td>tuxTsvcGrpState</td>
<td>2-131</td>
</tr>
<tr>
<td>tuxTsvcGrpSvcName</td>
<td>2-130</td>
</tr>
<tr>
<td>tuxTsvcGrpSvcTimeOut</td>
<td>2-132</td>
</tr>
<tr>
<td>tuxTsvcGrpTranTime</td>
<td>2-133</td>
</tr>
<tr>
<td>tuxTsvcLoad</td>
<td>2-125</td>
</tr>
<tr>
<td>tuxTsvcName</td>
<td>2-124</td>
</tr>
<tr>
<td>tuxTsvcPrio</td>
<td>2-126</td>
</tr>
<tr>
<td>tuxTsvcrName</td>
<td>2-134</td>
</tr>
<tr>
<td>tuxTsvcRoutingName</td>
<td>2-128</td>
</tr>
<tr>
<td>tuxTsvcSignatureRequired</td>
<td>2-128</td>
</tr>
<tr>
<td>tuxTsvcSrvrId</td>
<td>2-133</td>
</tr>
<tr>
<td>tuxTsvcSrvrLmid</td>
<td>2-133</td>
</tr>
<tr>
<td>tuxTsvcSrvrNcompleted</td>
<td>2-134</td>
</tr>
<tr>
<td>tuxTsvcSrvrNqueued</td>
<td>2-134</td>
</tr>
<tr>
<td>tuxTsvcSrvrRqAddr</td>
<td>2-133</td>
</tr>
<tr>
<td>tuxTsvcState</td>
<td>2-124</td>
</tr>
<tr>
<td>tuxTsvcTbl</td>
<td>2-123</td>
</tr>
<tr>
<td>tuxTsvcTimeOut</td>
<td>2-126</td>
</tr>
<tr>
<td>tuxTsvcTranTime</td>
<td>2-126</td>
</tr>
<tr>
<td>tuxTsvcType</td>
<td>2-124</td>
</tr>
<tr>
<td>tuxTulogCat</td>
<td>2-147</td>
</tr>
<tr>
<td>tuxTulogCatCtrl</td>
<td>2-152</td>
</tr>
<tr>
<td>tuxTulogContextID</td>
<td>2-148</td>
</tr>
<tr>
<td>tuxTulogCtrl</td>
<td>2-149</td>
</tr>
<tr>
<td>tuxTulogEndTimeCtrl</td>
<td>2-151</td>
</tr>
<tr>
<td>tuxTulogLine</td>
<td>2-146</td>
</tr>
<tr>
<td>tuxTulogLineCtrl</td>
<td>2-151</td>
</tr>
<tr>
<td>tuxTulogLmid</td>
<td>2-145</td>
</tr>
<tr>
<td>tuxTulogLmidCtrl</td>
<td>2-150</td>
</tr>
<tr>
<td>tuxTulogMmDdDYy</td>
<td>2-145</td>
</tr>
<tr>
<td>tuxTulogMmddyyCtrl</td>
<td>2-150</td>
</tr>
<tr>
<td>tuxTulogMsg</td>
<td>2-146</td>
</tr>
</tbody>
</table>
tuxTulogMsgCtrl 2-151

tuxTulogMsgNum 2-147

tuxTulogMsgNumCtrl 2-153

tuxTulogPid 2-146

tuxTulogPidCtrl 2-152

tuxTulogPmid 2-145

tuxTulogPmidCtrl 2-150

tuxTulogProcNameCtrl 2-153

tuxTulogProcName 2-147

tuxTulogSerNo 2-145

tuxTulogSeverity 2-147

tuxTulogSeverityCtrl 2-152

tuxTulogTable 2-144

tuxTulogThreadID 2-147

tuxTulogTime 2-145

tuxTulogTimeCtrl 2-150

tuxTulogTpTranId 2-146

tuxTulogTptranIdCtrl 2-151

tuxTulogXid 2-146

tuxTulogXidCtrl 2-152

tuxTwshTaActive 7-6

tuxTwshTaClientId 7-3

tuxTwshTaCurClients 7-5

tuxTwshTaCurwork 7-7

tuxTwshTaFlowcnt 7-7

tuxTwshTaGrpNo 7-4

tuxTwshTaHwClients 7-5

tuxTwshTaLmid 7-4

tuxTwshTaMultiplex 7-5

tuxTwshTaNumblockQ 7-7

tuxTwshTaPId 7-5

tuxTwshTaRcvdByt 7-7

tuxTwshTaRcvdNum 7-8

tuxTwshTaSentByt 7-7

tuxTwshTaSentNum 7-8

tuxTwshTaSrvGrp 7-3

tuxTwshTaSrvId 7-4

tuxTwshTaState 7-4

tuxTwshTaTimeleft 7-6

tuxTwshTaTotacttime 7-6

tuxTwshTaTotidtime 7-6

tuxTwshTaWsclentId 7-3

tuxTwslTaClOpt 7-14

tuxTwslTaCurHandlers 7-16

tuxTwslTaDevice 7-11

tuxTwslTaEnvFile 7-14

tuxTwslTaGrace 7-14

tuxTwslTaGrpNo 7-11

tuxTwslTaHwHandlers 7-16

tuxTwslTaKeepAlive 7-17

tuxTwslTaLmid 7-11

tuxTwslTaMaxGen 7-15

tuxTwslTaMaxHandlers 7-13

tuxTwslTaMaxIdleTime 7-13

tuxTwslTaMaxInitTime 7-13

tuxTwslTaMinHandlers 7-12

tuxTwslTaMultiplex 7-13

tuxTwslTaNaddr 7-12

tuxTwslTaNetTimeOut 7-17

tuxTwslTaPId 7-11

tuxTwslTaRcmd 7-15

tuxTwslTaRestart 7-15

tuxTwslTaSequence 7-15

tuxTwslTaSrvGrp 7-10

tuxTwslTaSrvId 7-10

tuxTwslTaState 7-11

tuxTwslTaSuspended 7-16

tuxTwslTaViewRefresh 7-17

tuxTwslTaWshName 7-12

tuxTwslTaWsProto 7-16

tuxTwslTbl 7-9

tuxWshTaNaddr 7-5

**Userlog messages**

Event Broker subscriptions for 9-18

Userlog table messages, control of 2-149

Userlogs, attributes of

as MIB objects 2-144
variable bindings
  contents of 1-9, 10-2
variable bindings, trap
  components of 10-2

wleCurInterfaces 2-53
wleFactoryId 5-3
wleFactoryIfName 5-4
wleFactorySerNo 5-3
wleFactoryState 5-4
wleFactoryTable 5-3
wleHwInterfaces 2-53
wleIfAutoTran 5-9
  effect of m3IfTxPolicy on 5-9
  effect of wleIfTxPolicy on 5-18
wleIfFnRoutingName 5-11
wleIfLmid 5-11
wleIfLoad 5-10
wleIfName 5-7
wleIfNumServers 5-11
wleIfPrio 5-10
wleIfQueueAutoTran 5-18
wleIfQueueFnRoutingName 5-19
wleIfQueueLmid 5-19
wleIfQueueLoad 5-18
wleIfQueueName 5-16
wleIfQueueNumServers 5-20
wleIfQueuePrio 5-18
wleIfQueueRqAddr 5-16
wleIfQueueSerNo 5-16
wleIfQueueSrvGrp 5-16
wleIfQueueState 5-17
wleIfQueueTimeout 5-19
wleIfQueueTpPolicy 5-20
wleIfQueueTranTime 5-19
wleIfQueueTxPolicy 5-20
wleIfSerNo 5-6
wleIfSrvGrp 5-7
wleIfState 5-7
wleIfTimeout 5-10
wleIfTpPolicy 5-12
wleIfTranTime 5-11
wleIfTxPolicy 5-12
  effect on m3IfAutoTran 5-9
  effect on wleIfAutoTran 5-18
wleInterfaceTable 5-5
wleJdbcConPoolAllowShrinking 5-29
wleJdbcConPoolCapacityIncr 5-29
wleJdbcConPoolCreateOnStartUp 5-28
wleJdbcConPoolDbHost 5-27
wleJdbcConPoolDbName 5-27
wleJdbcConPoolDbNetProtocol 5-28
wleJdbcConPoolDbPassword 5-27
wleJdbcConPoolDbPort 5-28
wleJdbcConPoolDbUser 5-27
wleJdbcConPoolDriver 5-26
wleJdbcConPoolDsName 5-26
wleJdbcConPoolEnableXa 5-28
wleJdbcConPoolExtnConnAvailable 5-33
wleJdbcConPoolExtnConnUsed 5-33
wleJdbcConPoolExtnDsName 5-33
wleJdbcConPoolExtnHwmForWait 5-34
wleJdbcConPoolExtnSrvGrp 5-32
wleJdbcConPoolExtnSrvId 5-32
wleJdbcConPoolExtnTable 5-32
wleJdbcConPoolHwmConnCreated 5-33
wleJdbcConPoolHwmConnUsed 5-33
wleJdbcConPoolInitCapacity 5-29
wleJdbcConPoolLoginDelay 5-29
wleJdbcConPoolMaxCapacity 5-29
wleJdbcConPoolProps 5-28
wleJdbcConPoolRefresh 5-30
wleJdbcConPoolShrinkPeriod 5-30
wleJdbcConPoolSrvGrp 5-25
wleJdbcConPoolSrvId 5-25
wleJdbcConPoolState 5-26
wleJdbcConPoolTable 5-24
wleJdbcConPoolTestOnRelease 5-31
wleJdbcConPoolTestOnReserve 5-30
wleJdbcConPoolTestTable  5-30
wleJdbcConPoolUrl  5-26
wleJdbcConPoolUserRole  5-27
wleJdbcConPoolWaitForConn  5-31
wleJdbcConPoolWaitTimeOut  5-31
wleLclfName  5-13
wleLclfN completed  5-14
wleLclfN queued  5-14
wleLclfQueueCurObjs  5-23
wleLclfQueueCurTrans  5-23
wleLclfQueueName  5-21
wleLclfQueueN completed  5-22
wleLclfQueueN queued  5-22
wleLclfQueueRqAddr  5-22
wleLclfQueueSerNo  5-21
wleLclfQueueSrvGrp  5-22
wleLclfQueueTable  5-21
wleLclfSerNo  5-13
wleLclInterfaceTable  5-13
wleLclSrvGrp  5-14
wleMachineCurObjects  2-84
wleMachineHwObjects  2-84
wleMachineMaxObjects  2-74
wleMaxInterfaces  2-53
wleMaxObjects  2-52
wleModuleModule  5-36
wleModuleModuleArgs  5-36
wleModuleModuleType  5-36
wleModuleSrvGrp  5-35
wleModuleSrvld  5-35
wleModuleState  5-36
wleModuleNameTable  5-35
wleRoutingFieldIdType  2-98
wleSrvrClassPath  2-121
wleSrvrCurInterfaceExt  2-120
wleSrvrCurObjsExt  2-120
wleSrvrjavaHeap  2-121
wleSrvrjavaHeapuse  2-121
wleSrvrjavaVendor  2-121
wleSrvrjavaVersion  2-121
wleSsrvrSrvType  2-112

wleJdbcConPoolExtnAwaitingConn  5-34

work station handler
See WSH  7-2
workload, of server
as MIB object  2-117
workstation client
status of  2-11
WSH client processes, attributes of
as MIB objects  7-2
WSL server processes, attributes of
as MIB objects  7-9