



BEA Tuxedo

Command Reference

BEA Tuxedo Release 7.1
Document Edition 7.1
May 2000

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

BEA, ObjectBroker, TOP END, and Tuxedo are registered trademarks of BEA Systems, Inc. BEA Builder, BEA Connect, BEA Manager, BEA MessageQ, BEA Jolt, M3, eSolutions, eLink, WebLogic, and WebLogic Enterprise are trademarks of BEA Systems, Inc.

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

BEA Tuxedo Command Reference

Document Edition	Date	Software Version
7.1	May 2000	BEA Tuxedo Release 7.1

Contents

. Section 1 - Commands

Introduction to BEA Tuxedo Commands	8
bldc_dce(1)	10
blds_dce(1)	11
buildclient(1)	12
buildclt(1)	16
buildserver(1)	20
buildtms(1)	26
buildwsh(1)	28
cobcc(1)	30
dmadmin(1)	32
dmloadcf(1)	52
dmunloadcf(1)	56
gencat(1)	58
mkfldhdr, mkfldhdr32(1)	61
mklanginfo(1)	63
qmadmin(1)	66
rex(1)	92
tidl(1)	95
tlisten(1)	102
tmadmin(1)	106
tmboot(1)	121
tmconfig, wtmconfig(1)	128
tmipcrm(1)	139
tmloadcf(1)	141
tmshutdown(1)	145

tmunloadcf(1)	150
topenpasswd(1)	151
tpacladd(1)	152
tpaclcvt(1)	154
tpacldel(1)	155
tpaclmod(1)	156
tpadduser(1)	157
tpdelusr(1)	159
tpgrpadd(1)	160
tpgrpdel(1)	161
tpgrpmod(1)	162
tpmodusr(1)	163
tpusradd(1)	165
tpusrdel(1)	167
tpusrmod(1)	168
tuxadm(1)	170
tuxwsvr(1)	171
txrpt(1)	177
ud, wud(1)	179
uuidgen(1)	184
viewc, viewc32(1)	186
viewdis, viewdis32(1)	189
wlisten(1)	190

Section 1 - Commands

BEA Tuxedo Commands

Name	Description
Introduction to BEA Tuxedo Commands	Provides an introduction to the commands available for setting up and maintaining your BEA Tuxedo application
bldc_dce(1)	Builds a BEA Tuxedo system client that can be called via OSF/DCE
blds_dce(1)	Builds a BEA Tuxedo system server that calls OSF/DCE
buildclient(1)	Constructs a BEA Tuxedo client module
buildclt(1)	Constructs a BEA Tuxedo Workstation client program for MS-DOS, Windows, Windows NT, and OS/2
buildserver(1)	Constructs a BEA Tuxedo server load module
buildtms(1)	Constructs a transaction manager server load module
buildwsh(1)	Builds a customized workstation handler process
cobcc(1)	COBOL compilation interface
dmadmin(1)	Administration command interpreter for BEA Tuxedo Domains
dmloadcf(1)	Parses a DMCONFIG file and loads a binary BDMCONFIG configuration file
dmunloadcf(1)	Unloads a BDMCONFIG file (a binary Domains configuration file)
gencat(1)	Generates a formatted message catalog
mkfldhdr, mkfldhdr32(1)	Creates header files from field tables
mklanginfo(1)	Compiles language-information constants for a locale

Section 1 - Commands

BEA Tuxedo Commands

Name	Description
qmadmin(1)	Administration command interpreter for Queue manager
rex(1)	Off-line regular expression compiler and tester
tidl(1)	Interface Definition Language compiler
tlisten(1)	Generic listener process
tadmin(1)	Command interpreter for BEA Tuxedo bulletin boards
tboot(1)	Brings up a BEA Tuxedo configuration
tmconfig, wtmconfig(1)	Dynamically update and retrieve information about a running BEA Tuxedo application, as either a native client or a workstation client
tmipcrm(1)	Removes IPC resources allocated by a BEA Tuxedo application on a local machine
tmloadcf(1)	Parses a UBBCONFIG file (a text-format configuration file) and loads a TUXCONFIG file (a binary configuration file)
tmshutdown(1)	Shuts down a set of BEA Tuxedo servers
tmunloadcf(1)	Unloads a TUXCONFIG file (a binary configuration file)
topendpasswd(1)	Adds a BEA TOP END password for the TOP END Domain Gateway feature of the BEA Tuxedo system
tpacladd(1)	Adds a new Access Control List entry on the system
tpaclcvt(1)	Converts BEA Tuxedo security data files
tpacldel(1)	Deletes an Access Control List entry
tpaclmod(1)	Modifies an Access Control List entry on the system
tpadduser(1)	Creates a BEA Tuxedo password file
tpdelusr(1)	Deletes a user from a BEA Tuxedo password file
tpgrpadd(1)	Adds a new group to the system
tpgrpdel(1)	Deletes a group from the system

BEA Tuxedo Commands

Name	Description
<code>tpgrpmod(1)</code>	Modifies a group on the system
<code>tpmodusr(1)</code>	Maintains a BEA Tuxedo system password file
<code>tpusradd(1)</code>	Adds a new principal to the system
<code>tpusrdel(1)</code>	Deletes a user from the system
<code>tpusrmod(1)</code>	Modifies user information on the system
<code>tuxadm(1)</code>	CGI gateway for the BEA Tuxedo Web GUI
<code>tuxwsvr(1)</code>	Mini-Web server for use with the BEA Tuxedo Web GUI
<code>txrpt(1)</code>	BEA Tuxedo system server/service report program
<code>ud, wud(1)</code>	BEA Tuxedo driver program
<code>uuidgen(1)</code>	Generates a Universal Unique Identifier (UUID)
<code>viewc, viewc32(1)</code>	Views compiler for BEA Tuxedo views
<code>viewdis, viewdis32(1)</code>	Views disassembler for binary viewfiles
<code>wlisten(1)</code>	BEA Tuxedo Web GUI listener process

Introduction to BEA Tuxedo Commands

Description The *BEA Tuxedo Command Reference* describes, in alphabetic order, shell-level commands delivered with the BEA Tuxedo software.

Reference Page Command Syntax Unless otherwise noted, commands described in the Synopsis section of a reference page accept options and other arguments according to the following syntax and should be interpreted as explained below.

name [*-option* . . .] [*cmdarg* . . .]

where *name* is the name of an executable file and *option* is a string of one of the following two types: *noargletter* . . . or *argletter optarg* [, . . .]

An option is always preceded by a “-”.

noargletter

is a single letter representing an *option* that requires no option-argument. More than one *noargletter* can be grouped after a “-”

optarg

is a character string that satisfies a preceding *argletter*. Multiple *optargs* following a single *argletter* must be separated by commas, or separated by white space and enclosed in quotes.

cmdarg

is a path name (or other command argument) that represents an operand of the command.

-

(dash) by itself means that additional arguments are provided in the standard input.

--

(two dashes) means that what follows are arguments for a subordinate program.

[]

surrounding an *option* or *cmdarg*, mean that the option or argument is not required.

{ } surrounding *cmdargs* that are separated by an `or` sign, mean that one of the choices must be selected if the associated option is used.

. . . . means that multiple occurrences of the *option* or *cmdarg* are permitted.

bldc_dce(1)

- Name** bldc_dce - build a BEA Tuxedo system client that can be called via OSF/DCE
- Synopsis** bldc_dce [-o *output_file*] [-i *idl_options*] [-f *firstfiles*]
 [-l *lastfiles*] [*idl_file* . . .]
- Description** bldc_dce parses any input IDL and related ACF source files and combines them with C source and object files and the OSF/DCE libraries to generate a BEA Tuxedo system client that can be called via DCE RPC (it is a DCE RPC client).

The command-line arguments include the input IDL source file and options to control the actions of the IDL compiler. The options are as follows:

- o *output_file*
 The default filename is a.out.
- i *idl_options*
 Specifies options to be passed to the IDL compiler. Options associated with the C compilation system are automatically provided by this program. This option can be used to provide the `-no_mepv` option such that the application can provide a Manager Entry Point Vector.
- f *firstfiles*
 Specifies compiler options, C source and object files to be included on the compilation before the BEA Tuxedo system and OSF/DCE libraries.
- l *lastfiles*
 Specifies C libraries to be included on the compilation after the BEA Tuxedo system and OSF/DCE libraries.

See Also tidl(1)

blds_dce(1)

Name blds_dce - build a BEA Tuxedo system server that calls OSF/DCE

Synopsis blds_dce [-o *output_file*] [-i *idl_options*] [-f *firstfiles*]
[-l *lastfiles*] [-s *service*] [*idl_file* . . .]

Description blds_dce parses any input IDL and related ACF source files and combines them with C source and object files and the OSF/DCE libraries to generate a BEA Tuxedo system server that can make DCE RPC calls. The primary use of this command is to make a BEA Tuxedo system-to-OSF/DCE gateway process.

The command-line arguments include the input IDL source file and options to control the actions of the IDL compiler. The options are as follows:

-o *output_file*

The default filename is a.out.

-i *idl_options*

Specifies options to be passed to the IDL compiler. Options associated with the C compilation system are automatically provided by this program. This option can be used to provide the `-no_mepv` option such that the application can provide a Manager Entry Point Vector.

-f *firstfiles*

Specifies compiler options, C source and object files to be included on the compilation before the BEA Tuxedo system and OSF/DCE libraries.

-l *lastfiles*

Specifies C libraries to be included on the compilation after the BEA Tuxedo system and OSF/DCE libraries.

-s *service* [, *service* . . .]

Specifies the services to be advertised by the server.

See Also tidl(1)

buildclient(1)

- Name** `buildclient` - construct a BEA Tuxedo client module
- Synopsis** `buildclient [-C] [-v] [{-r rmname | -w }] [-o name] [-f firstfiles] [-l lastfiles]`
- Description** `buildclient` is used to construct a BEA Tuxedo client module. The command combines the files supplied by the `-f` and `-l` options with the standard BEA Tuxedo libraries to form a load module. The load module is built by `buildclient` using the default C language compilation command defined for the operating system in use. The default C language compilation command for the UNIX system is the `cc(1)` command described in UNIX system reference manuals.
- `-v`
specifies that `buildclient` should work in verbose mode. In particular, it writes the compilation command to its standard output.
- `-w`
specifies that the client is to be built using the workstation libraries. The default is to build a native client if both native mode and workstation mode libraries are available. This option cannot be used with the `-r` option.
- `-r rmname`
specifies the resource manager associated with this client. The value *rmname* must appear in the resource manager table located in `$TUXDIR/udataobj/RM`. Each line in this file is of the form:
`rmname:rmstructure_name:library_names`
(See `buildtms(1)` for further details.) Using the *rmname* value, the entry in `$TUXDIR/udataobj/RM` is used to include the associated libraries for the resource manager automatically and to set up the interface between the transaction manager and resource manager properly. Other values can be specified as they are added to the resource manager table. If the `-r` option is not specified, the default is that the client is not associated with a resource manager. Refer to the `UBBCONFIG(5)` reference page.
- `-o`
specifies the filename of the output load module. If not supplied, the load module is named `a.out`.

-f specifies one or more user files to be included in the compilation and link edit phases of `buildclient` first, before the BEA Tuxedo libraries. If more than one file is specified, filenames must be separated by white space and the entire list must be enclosed in quotation marks. This option may be specified multiple times. The `CFLAGS` and `ALTCFLAGS` environment variables, described below, should be used to include any compiler options and their arguments.

-l specifies one or more user files to be included in the compilation and link edit phases of `buildclient` last, after the BEA Tuxedo libraries. If more than one file is specified, filenames must be separated by white space and the entire list must be enclosed in quotation marks. This option may be specified multiple times.

-C specifies COBOL compilation.

Environment Variables

`TUXDIR` `buildclient` uses the environment variable `TUXDIR` to find the BEA Tuxedo system libraries and `include` files to use during compilation of the client process.

`CC` `buildclient` normally uses the default C language compilation command to produce the client executable. The default C language compilation command is defined for each supported operating system platform and is defined as `cc(1)` for UNIX system. In order to allow for the specification of an alternate compiler, `buildclient` checks for the existence of an environment variable named `CC`. If `CC` does not exist in `buildclient`'s environment, or if it is the string "", `buildclient` will use the default C language compiler. If `CC` does exist in the environment, its value is taken to be the name of the compiler to be executed.

`CFLAGS` The environment variable `CFLAGS` is taken to contain a set of arguments to be passed as part of the compiler command line. This is in addition to the command line option `-I${TUXDIR}/include` passed automatically by `buildclient`. If `CFLAGS` does not exist in `buildclient`'s environment, or if it is the string "", no compiler command-line arguments are added by `buildclient`.

ALTCC

When the `-C` option is specified for COBOL compilation, `buildclient` normally uses the BEA Tuxedo shell `cobcc` which in turn calls `cob` to produce the client executable. In order to allow for the specification of an alternate compiler, `buildclient` checks for the existence of an environment variable named `ALTCC`. If `ALTCC` does not exist in `buildclient`'s environment, or if it is the string " ", `buildclient` uses `cobcc`. If `ALTCC` does exist in the environment, its value is taken to be the name of the compiler command to be executed.

Note: On a Windows NT system, the `ALTCC` and `ALTCFLAGS` environment variables are not applicable and setting them will produce unexpected results. You must compile your application first using a COBOL compiler and then pass the resulting object file to the `buildclient(1)` command.

ALTCFLAGS

The environment variable `ALTCFLAGS` is taken to contain a set of additional arguments to be passed as part of the COBOL compiler command line when the `-C` option is specified. This is in addition to the command-line option

`"-I${TUXDIR}/include"`

passed automatically by `buildclient`. When the `-C` option is used, putting compiler options and their arguments in the `buildclient -f` option generates errors; they must be put in `ALTCFLAGS`. If not set, then the value is set to the same value used for `CFLAGS`, as specified above.

Note: See the note under the description of the `ALTCC` environment variable.

COBOPT

The environment variable `COBOPT` is taken to contain a set of additional arguments to be used by the COBOL compiler, when the `-C` option is specified.

COBCPY

The environment variable `COBCPY` indicates which directories contain a set of COBOL copy files to be used by the COBOL compiler, when the `-C` option is specified.

LD_LIBRARY_PATH

The environment variable `LD_LIBRARY_PATH` indicates which directories contain shared objects to be used by the COBOL compiler, in addition to the BEA Tuxedo system shared objects.

Portability The `buildclient` compilation tool is supported on the following platforms:

- Any platform on which the BEA Tuxedo server environment is supported
- Any Workstation platform running a 32-bit Windows operating system

Filename specified in the `buildclient` command line must conform to the syntax and semantics of the resident operating system.

Examples

```
CC=ncc CFLAGS="-I /APPDIR/include"; export CC CFLAGS
buildclient -o empclient -f emp.c -f "userlib1.a userlib2.a"
```

```
COBCPY=$TUXDIR/cobinclude
COBOPT="-C ANS85 -C ALIGN=8 -C NOIBMCOMP -C TRUNC=ANSI -C OSEXT=cbl"
COBDIR=/usr/lib/cobol LD_LIBRARY_PATH=$COBDIR/coblib:$TUXDIR/lib
export COBOPT COBCPY COBDIR LD_LIBRARY_PATH
buildclient -C -o empclient -f name.cbl -f "userlib1.a userlib2.a"
```

See Also `buildclt(1)`, `buildserver(1)`, `buildtms(1)`, `compilation(5)`
`cc(1)`, `ld(1)` in a UNIX system reference manual

buildclt(1)

Name	<code>buildclt</code> - construct a BEA Tuxedo Workstation client program for MS-DOS, Windows 3.x, and OS/2 16-bit.
Synopsis	<code>buildclt [-C] [-v] [-m {m l}] [-c {m b i}] [-o <i>name</i>] [-f <i>firstfiles</i>] [-F <i>Firstlibs</i>] [-l <i>libfiles</i>] [-W] [-O] [-P] [-d <i>deffile</i>]</code>
Description	<p><code>buildclt</code> is used to construct a BEA Tuxedo workstation client program for MS-DOS, Windows 3.x, and OS/2 16-bit. The command combines the files supplied by the <code>-f</code> and <code>-l</code> options with the standard BEA Tuxedo libraries to form an executable program. The load module is built by <code>buildclt</code> using the C and COBOL language compilation commands defined for the operating system in use. The options to <code>buildclt</code> have the following meaning:</p> <ul style="list-style-type: none"><code>-v</code> specifies that <code>buildclt</code> should work in verbose mode. In particular, it writes the compilation command to its standard output.<code>-o <i>name</i></code> specifies the filename of the output program. If not supplied, the program is named <code>client.exe</code>.<code>-f <i>firstfiles</i></code> specifies one or more user files (or options to the compiler or linker) to be included on the command line first, before the BEA Tuxedo libraries. If more than one file is specified, filenames must be separated by white space and the entire list must be enclosed in double quotation marks. This option may be specified multiple times.<code>-F <i>Firstlibs</i></code> specifies one or more standard or import (not dynamic-link) libraries to be included before the BEA Tuxedo libraries. If more than one file is specified, filenames must be separated by white space and the entire list must be enclosed in double quotation marks. This option may be specified multiple times.<code>-l <i>libfiles</i></code> specifies one or more standard or import (not dynamic-link) libraries to be included after the BEA Tuxedo libraries. If more than one file is specified,

filenames must be separated by white space and the entire list must be enclosed in double quotation marks. This option may be specified multiple times.

`-d def file`

specifies a module definition file used for linking an MS Windows 3.x or OS/2 16-bit program.

`-m {m | l}`

specifies the memory model to be used for compilation and linking of a client. The values for this option are `m` and `l` for the medium and large memory models, respectively. The large memory model is the default value for this option. The medium memory model is no longer supported for DOS. The memory model needs only to be specified for Windows 3.x and OS/2 16-bit compilation.

`-c {m | i}`

specifies the compilation system to be used. The supported values for this option are `m` or `i` for the Microsoft C/C++ compiler, or the IBM CSET2 compiler, respectively. The Microsoft C compiler is the default value for this option. The IBM CSET2 compiler can be used only for OS/2 32-bit compilation.

`-W`

compiles and links an MS Windows 3.x client.

`-O`

compiles and links an OS/2 character-mode client.

`-P`

compiles and links an OS/2 Presentation Manager client.

`-C`

specifies COBOL compilation.

The following environment variables must be set for the COBOL environment.

`COBCPY`

The environment variable `COBCPY` indicates which directories contain a set of COBOL copy files to be used by the COBOL compiler.

Microsoft C Compilation

The `buildc1t` command assumes that directories for needed libraries are specified in the environment variables `INCLUDE` and `LIB`. They might look like the following:

```
INCLUDE=C:\TUXEDO\INCLUDE;C:\NET\TOOLKIT\INCLUDE;C:\MSVC\INCLUDE
LIB=C:\NET\TOOLKIT\LIB;C:\WINDEV\LIB;C:\MSVC\LIB;C:\TUXEDO\LIB;
```

Note that in the above example, C:MSVC is the directory where MS Visual C++ resides; earlier versions such as C600 or C700 can be used. Note that in the above example, C:NET is the directory where Novell Lan Workplace resides; earlier versions resided in C:XLN and can be used.

Note that COBOL source files that reference ATMI calls must be compiled with the LITLINK option.

The names of all libraries used for linking the client followed by the files specified in the -l option are put into a temporary *response file* and linking is done using the command line:

```
LINK firstfiles, outname @tmpfile
```

are the filenames specified with the -f option, *outname* is the output filename (default *client.exe*), and *tmpfile* is the temporary response filename. The -f option should be used to include any necessary options to be passed to LINK (for example, /ST:10000 to set the default stack size to 10000 bytes). The -l option should be used to include any necessary network provider libraries (for example, *mlisock.lib*). To create an executable that can be debugged using Codeview (assuming that the object files have been compiled with the -zi option), use -f /CO.

Examples MS-DOS C Compilation:

```
buildclt -cm -ml -o emp.exe -f "/CO/ST:10000/SE:200" -f emp.obj -l  
llibsock.lib
```

WINDOWS C Compilation:

```
buildclt -W -cm -mm -o emp.exe -f "/CO emp.obj" -d emp.def rc -k  
emp.res emp.exe
```

OS2 16-bit:

```
buildclt -O -cm -ml -o emp.exe -f "/NOI/ST:15000/CO emp.obj" -d  
emp.def
```

DOS/WINDOWS/OS2 COBOL Compilation:

```
COBCPY=C:\TUXEDO\COBINC  
COBDIR=C:\COBOL\LBR;C:\COBOL\EXEDLL  
PATH=C:\C700BIN\;C:COBOLEXEDLL;...  
TUXDIR=C:\TUXEDO  
INCLUDE=C:\TUXEDO\INCLUDE;C:\XLN\TOOLKIT\INCLUDE;C:\C700\INCLUDE  
LIB=C:\XLN\TOOLKIT\LIB;C:\C700\LIB;C:\TUXEDO\LIB;C:\COBOL\LIB
```

```
COBOL EMP.CBL OMF"OBJ" LITLINK
```

DOS:

```
BUILDCLT -C -o EMP.EXE -f EMP+MFC7INTF+C7DOSIF+C7DOSLB \  
-f "/NOD/NOE/SE:300/CO/ST:10000" -l LLIBSOCK
```

WINDOWS:

```
BUILDCLT -C -W -o EMP.EXE -f EMP -d EMP.DEF -f "/NOD/NOE/CO/SE:300"
```

OS2:

```
BUILDCLT -C -P -o EMP.EXE -f EMP+MFC6INTF+C6OS2IF+C6OS2LB -d  
EMP.DEF \ -f "/NOD/NOE/SE:300/CO"
```

See Also *Microsoft C/C++ Programming Techniques*, Microsoft Corporation

Micro Focus COBOL/2 Operating Guide, Micro Focus Ltd.

Micro Focus COBOL/2 Workbench for DOS and OS2, Micro Focus Ltd.

buildserver(1)

Name buildserver - construct a BEA Tuxedo server load module

Synopsis buildserver [-C] [-s { @filename | service[,service . . .]
[:func] | :func }] [-v] [-o outfile] [-f firstfiles]
[-l lastfiles] [{-r|-g} rname] [-k] [-t]

Description buildserver is used to construct a BEA Tuxedo server load module. The command combines the files supplied by the -f and -l options with the standard server main routine and the standard BEA Tuxedo libraries to form a load module. The load module is built by the cc(1) command, which buildserver invokes. (See cc(1) in any UNIX system reference manual.) The options to buildserver have the following meaning:

- v specifies that buildserver should work in verbose mode. In particular, it writes the compilation command to its standard output.
- o outfile specifies the name of the file the output load module is to have. If not supplied, the load module is named SERVER.
- f firstfiles specifies one or more user files to be included in the compilation and link edit phases of buildserver first, before the BEA Tuxedo libraries. If more than one file is specified, filenames must be separated by white space and the entire list must be enclosed in quotation marks. This option may be specified multiple times. The CFLAGS and ALTFLAGS environment variables, described below, should be used to include any compiler options and their arguments.
- l lastfiles specifies one or more user files to be included in the compilation and link edit phases of buildserver last, after the BEA Tuxedo libraries. If more than one file is specified, filenames must be separated by white space and the entire list must be enclosed in quotation marks. This option may be specified multiple times.

`-r rmname`

specifies the resource manager associated with this server. The value *rmname* must appear in the resource manager table located in `$TUXDIR/udataobj/RM`. Each line in this file is of the form:

```
rmname:rmstructure_name:library_names
```

(See `builddtms(1)` for further details.) Using the *rmname* value, the entry in `$TUXDIR/udataobj/RM` is used to include the associated libraries for the resource manager automatically and to set up the interface between the transaction manager and resource manager properly. Other values can be specified as they are added to the resource manager table. If the `-r` option is not specified, the default is to use the null resource manager. Refer to the `UBBCONFIG(5)` reference page.

`-s { @filename | service [, service ...] [: func] | : func }]`

specifies the names of services that can be advertised when the server is booted. Service names (and implicit function names) must be less than or equal to 15 characters in length. An explicit function name (that is, a name specified after a colon) can be up to 128 characters in length. Names longer than these limits are truncated with a warning message. When retrieved by `tmadmin(1)` or `TM_MIB(5)`, only the first 15 characters of a name are displayed. (See `servopts(5)`.) All functions that can be associated with a service must be specified with this option. In the most common case, a service is performed by a function that carries the same name; that is, the *x* service is performed by function *x*. For example, the specification

```
-s x,y,z
```

will build the associated server with services *x*, *y*, and *z*, each to be processed by a function of the same name. In other cases, a service (or several services) may be performed by a function of a different name. The specification

```
-s x,y,z:abc
```

builds the associated server with services *x*, *y*, and *z*, each to be processed by the function *abc*. Spaces are not allowed between commas. Function name is preceded by a colon. In another case, the service name may not be known until runtime. Any function that can have a service associated with it must be specified to `buildserver`. To specify a function that can have a service name mapped to it, put a colon in front of the function name. For example, the specification

`-s :pqr`

builds the server with a function `pqr`, which can have a service association. `tpadvertise(3c)` could be used to map a service name to the `pqr` function.

A filename can be specified with the `-s` option by prefacing the filename with the '@' character. Each line of this file is treated as an argument to the `-s` option. You may put comments in this file. All comments must start with the '#' character. This file can be used to specify all the functions in the server that may have services mapped to them.

The `-s` option may appear several times. Note that services beginning with the '.' character are reserved for system use, and `buildserver` will fail if the `-s` option is used to include such a service in the server.

`-C`

specifies COBOL compilation.

`buildserver` normally uses the `cc` command to produce the `a.out`. In order to allow for the specification of an alternate compiler, `buildserver` checks for the existence of a shell variable named `CC`. If `CC` does not exist in `buildservercs` environment, or if it is the string "", `buildserver` will use `cc` as the compiler. If `CC` does exist in the environment, its value is taken to be the name of the compiler to be executed. Likewise, the shell variable `CFLAGS` is taken to contain a set of parameters to be passed to the compiler.

`-k`

keeps the server *main* stub. `buildserver` generates a *main* stub with data structures such as the service table and a `main()` function. This is normally compiled and then removed when the server is built. This option indicates that the source file should be kept (to see what the source filename is, use the `-v` option).

Note: The generated contents of this file may change from release to release; *DO NOT* count on the data structures and interfaces exposed in this file. This option is provided to aid in debugging of build problems.

`-t`

specifies multithreading. If you want your servers to be multithreaded, this option is mandatory. If this option is not specified and you try to boot a server with a configuration file in which the value of `MAXDISPATCHTHREADS` is greater than 1, a warning message is printed in the user log and the server reverts to single-threaded operation.

The purpose of this option is to prevent an administrator from trying to boot, as a multithreaded server, a server that is not programmed in a thread-safe manner.

Environment
Variables

TUXDIR

`buildserver` uses the environment variable `TUXDIR` to find the BEA Tuxedo libraries and include files to use during compilation of the server process.

CC

`buildserver` normally uses the default C language compilation command to produce the server executable. The default C language compilation command is defined for each supported operating system platform and is defined as `cc(1)` for the UNIX System. In order to allow for the specification of an alternate compiler, `buildserver` checks for the existence of an environment variable named `CC`. If `CC` does not exist in `buildserver`'s environment, or if it is the string "", `buildserver` will use the default C language compiler. If `CC` does exist in the environment, its value is taken to be the name of the compiler to be used.

CFLAGS

The environment variable `CFLAGS` is taken to contain a set of arguments to be passed as part of the compiler command line. This is in addition to the command line option `-I${TUXDIR}/include` passed automatically by `buildserver`. If `CFLAGS` does not exist in `buildserver`'s environment, or if it is the string "", no compiler command-line arguments are added by `buildserver`.

ALTCC

When the `-C` option is specified for COBOL compilation, `buildserver` normally uses the BEA Tuxedo shell `cobcc(1)` which in turn calls `cob` to produce the server executable. In order to allow for the specification of an alternate compiler, `buildserver` checks for the existence of an environment variable named `ALTCC`. If `ALTCC` does not exist in `buildserver`'s environment, or if it is the string "", `buildserver` will use `cobcc`. If `ALTCC` does exist in the environment, its value is taken to be the name of the compiler command to be executed.

Note: On a Windows NT system, the `ALTCC` and `ALTCCFLAGS` environment variables are not applicable and setting them will produce unexpected results. You must compile your application first using a COBOL compiler and then pass the resulting object file to the `buildserver(1)` command.

ALTCFLAGS

The environment variable `ALTCFLAGS` is taken to contain a set of additional arguments to be passed as part of the COBOL compiler command line when the `-C` option is specified. This is in addition to the command line option `"-I${TUXDIR}/include"` passed automatically by `buildserver`. When the `-C` option is used, putting compiler options and their arguments in the `buildserver -f` option will generate errors; they must be put in `ALTCFLAGS`. If not set, then the value is set to the same value used for `CFLAGS`, as specified above.

Note: See previous note, under `ALTCC` environment variable.

COBOPT

The environment variable `COBOPT` is taken to contain a set of additional arguments to be used by the COBOL compiler, when the `-C` option is specified.

COBCPY

The environment variable `COBCPY` indicates which directories contain a set of COBOL copy files to be used by the COBOL compiler, when the `-C` option is specified.

LD_LIBRARY_PATH

The environment variable `LD_LIBRARY_PATH` indicates which directories contain shared objects to be used by the COBOL compiler, in addition to the BEA Tuxedo shared objects.

- Compatibility** In earlier releases, the `-g` option was allowed to specify a *genoption* of `sql` or `database`. For upward compatibility, this option is a synonym for the `-r` option.
- Portability** The `buildserver` compilation tool is supported on any platform on which the BEA Tuxedo server environment is supported. RM XA libraries are not supported on the Windows NT platform.
- Notices** Some compilation systems may require some code to be executed within the `main()`. For example, this could be used to initialize constructors in C++ or initialize the library for COBOL. A general mechanism is available for including application code in the server `main()` immediately after any variable declarations and before any executable statements. This will allow for the application to declare variables and execute statements in one block of code. The application exit is defined as follows. `#ifdef TMMAINEXIT #include "mainexit.h" #endif`. To use this feature, the application

should include `"-DTMMAINEXIT"` in the `ALTCFLAGS` (for COBOL) or `CFLAGS` (for C) environment variables and provide a `mainexit.h` in the current directory (or use the `-I` include option to include it from another directory).

For example, Micro Focus Cobol V3.2.x with a PRN number with the last digits greater than 11.03 requires that `cobinit()` be called in `main` before any COBOL routines, if using shared libraries. This can be accomplished by creating a `mainexit.h` file with a call to `cobinit()` (possibly preceded by a function prototype) and following the procedure above.

Examples The following example shows how to specify the resource manager (`-r TUXEDO/SQL`) libraries on the `buildserver` command line:

```
buildserver -r TUXEDO/SQL -s OPEN_ACCT -s CLOSE_ACCT -o ACCT
-f ACCT.o -f appinit.o -f util.o
```

The following example shows how `buildserver` can be supplied `CC` and `CFLAGS` variables and how `-f` can be used to supply a `-lm` option to the `CC` line to link in the math library:

```
CFLAGS=-g CC=/bin/cc buildserver -r TUXEDO/SQL -s DEPOSIT
-s WITHDRAWAL -s INQUIRY -o TLR -f TLR.o -f util.o -f -lm
```

The following example shows use of the `buildserver` command with no resource manager specified:

```
buildserver -s PRINTER -o PRINTER -f PRINTER.o
```

The following example shows COBOL compilation:

```
COBCPY=$TUXDIR/cobinclude COBOPT="-C ANS85 -C ALIGN=8 -C NOIBMCOMP
-C TRUNC=ANSI -C OSEXT=cbl" COBDIR=/usr/lib/cobol
LD_LIBRARY_PATH=$COBDIR/coblib export COBOPT COBCPY COBDIR
LD_LIBRARY_PATH buildserver -C -r TUXEDO/SQL -s OPEN_ACCT
-s CLOSE_ACCT -o ACCT -f ACCT.o -f appinit.o -f util.o
```

See Also `buildtms(1)`, `servopts(5)`, `UBBCONFIG(5)`

C compiler and linker documentation in the reference manual for your operating system

buildtms(1)

Name	<code>buildtms</code> - construct a transaction manager server load module
Synopsis	<code>buildtms [-v] -o name -r rm_name</code>
Description	<code>buildtms</code> is used to construct a transaction manager server load module.

While several TM servers are provided with the BEA Tuxedo system, new resource managers may be provided to work with the BEA Tuxedo system for distributed transaction processing. The resource manager must conform to the X/OPEN XA interface. The following four items must be published by the resource manager vendor: the name of the structure of type `xa_switch_t` that contains the name of the resource manager, flags indicating capabilities of the resource manager, and function pointers for the actual XA functions; the name of the resource manager that is contained in the name element of the `xa_switch_t` structure; the name of the object files that provide the services of the XA interface and supporting software; and the format of the information string supplied to the `OPENINFO` and `CLOSEINFO` parameters in the `UBBCONFIG` configuration file. See `UBBCONFIG(5)`.

When integrating a new resource manager into the BEA Tuxedo system, the file `$TUXDIR/udataobj/RM` must be updated to include the information about the resource manager. The format of this file is

```
rm_name:rm_structure_name:library_names
```

where *rm_name* is the resource manager name, *rm_structure_name* is the name of the `xa_switch_t` structure, and *library_names* is the list of object files for the resource manager. White space (tabs and/or spaces) is allowed before and after each of the values and may be embedded within the *library_names*. The colon (:) character may not be embedded within any of the values. Lines beginning with a pound sign (#) are treated as comments and are ignored.

A transaction manager server for the new resource manager must be built using `buildtms` and installed in `$TUXDIR/bin`. `buildtms` uses the `buildserver(1)` command to build the resulting `a.out`. The options to `buildtms` have the following meaning:

`-v` specifies that `buildtms` should work in verbose mode. In particular, it writes the `buildserver` command to its standard output and specifies the `-v` option to `buildserver`.

`-o name` specifies the name of the file the output load module is to have.

`-r rm_name` specifies the resource manager associated with this server. The value `rm_name` must appear in the resource manager table located in `$TUXDIR/udataobj/RM`. The entry associated with the `rm_name` value is used to include the correct libraries for the resource manager automatically and properly to set up the interface between the transaction manager and resource manager (using the `xa_switch_t` structure).

`buildtms` uses the `buildserver` command to produce the `a.out.buildserver` uses the `CC` and `CFLAGS` environment variables, if set, for the compiler and compiler flags, respectively. See `buildserver(1)` for further details.

Portability `buildtms` is supported as a BEA Tuxedo system-supplied compilation tool on any platform on which the BEA Tuxedo server environment is supported. RM XA libraries are not supported on the Windows NT platform.

Examples `buildtms -o $TUXDIR/bin/TMS_XYZ -r XYZ/SQL # TMS for XYZ resource manager`

See Also `buildserver(1)`, `UBBCONFIG(5)`

buildwsh(1)

Name `buildwsh` - build customized workstation handler process

Synopsis `buildwsh [-v] [-o name] [-f files]`

Description `buildwsh` is used to construct a customized BEA Tuxedo workstation handler module. The files included by the caller should include only the application buffer type switch and any required supporting routines. The command combines the files supplied by the `-f` option with the standard BEA Tuxedo libraries necessary to form a workstation handler load module. The load module is built by the `cc(1)` command described in UNIX system reference manuals, which `buildwsh` invokes. The options to `buildwsh` have the following meaning:

`-v`
specifies that `buildwsh` should work in verbose mode. In particular, it writes the `cc` command to its standard output.

`-o name`
specifies the filename of the output workstation handler load module. The name specified here must also be specified with the `-w WSHname` option of the `WSL(5)` server in the `SERVER` section of the configuration file. If not supplied, the load module is named `WSH`.

`-f firstfiles`
specifies one or more user files to be included in the compilation and/or link edit phases of `buildwsh`. Source files are compiled using either the `cc` command or the compilation command specified through the `CC` environment variable. Object files resulting from compilation of source files and object files specified directly as arguments to the `-f` option are included after all object files necessary to build a base workstation handler process and before the BEA Tuxedo libraries. If more than one file is specified, filenames must be separated by white space and the entire list must be enclosed in quotation marks. This option can be specified multiple times.

`buildwsh` normally uses the `cc` command to produce the `a.out`. In order to allow for the specification of an alternate compiler, `buildwsh` checks for the existence of a shell variable named `CC`. If `CC` does not exist in `buildwsh`'s environment, or if it is the string "", `buildwsh` will use `cc` as the compiler. If `CC` does exist in the environment, its value is taken to be the name of the compiler to be executed. Likewise, the shell variable `CFLAGS` is taken to contain a set of parameters to be passed to the compiler.

If your application uses shared libraries, it is not necessary to go through this compile and link process. See “Managing Typed Buffers” in *Programming a BEA Tuxedo Application Using C*.

Portability The `buildwsh` compilation tool is supported on any platform on which the BEA Tuxedo server environment is supported.

Examples

```
CC=ncc CFLAGS="-I $TUXDIR/include"; export CC CFLAGS buildwsh
-o APPWSH -f apptypsw.o
```

See Also `buildclient(1)`, `WSL(5)`
`cc(1)`, `ld(1)` in a UNIX system reference manual

cobcc(1)

Name `cobcc` - COBOL compilation interface

Synopsis `cobcc [option . . .] filename . . .`

Description `cobcc` is used as an interface shell to the COBOL compiler. It is invoked, by default, when `buildclient(1)` or `buildserver(1)` is executed with the `-C` (COBOL) option. This can be overridden by specifying the `ALTCC` environment variable.

The following list indicates the options recognized by `cobcc`. To use these options, set the environment variable `ALTCFLAGS` to the string of options to be recognized by `cobcc` when running `buildclient` or `buildserver`. Consult your documentation for the COBOL and C compilers to see what effect the various options have.

Note: On a Windows NT system, the `ALTCC` and `ALTCFLAGS` environment variables are not applicable and setting them will produce unexpected results. You must compile your application first using a COBOL compiler and then pass the resulting object file to the `buildclient(1)` or `buildserver(1)` command.

Note that for `cobcc`, unlike `cc` and `cob`, all options must come before any filenames.

`-c`

This option specifies that the link phase should be suppressed. That is, compilation will be done but an executable program will not be generated.

`-p -g -r -O`

These options are passed directly to the COBOL compiler.

`-l argument`

This option and its argument are passed directly to the COBOL compiler (with no white space separating them).

`-L argument`

This option and its argument are passed directly to the COBOL compiler (with one space separating them).

`-o output_file`

This option is used to specify the name of the executable file that is output from the link stage.

- `-E -P -S`
These options are passed through the COBOL compiler to the C compiler, and also cause suppression of the link phase.
- `-A -C -H -f -G`
These options are passed through the COBOL compiler to the C compiler.
- `-w`
This option causes warnings to be suppressed from both the COBOL and C compilers.
- `-D argument`
This option and its argument are passed through the COBOL compiler to the C compiler. It is used to define macros in C.
- `{-T -Y -U -I -B -X -F -q} argument`
Each of these options takes an argument. The option and its argument are passed through the COBOL compiler to the C compiler.
- `-V -v`
Each of these options is passed both to the COBOL compiler and the C compiler.
- `-a -s`
Each of these options is passed to the loader.
- `-u argument`
This option and its argument are passed to the loader.
- `-W argument`
The *argument* may consist of up to three comma-separated fields. If the first part of the argument is `-p` or `-0`, it is passed to the C compiler. If it starts with `-a`, it is passed to the assembler. If it starts with `-l`, it is passed to the loader. If it starts with `-c`, it is passed to the COBOL compiler. Otherwise, it is passed through to the C compiler.

The options and their arguments and the filenames are passed to the COBOL compiler with the correct options so that the right information is processed by the COBOL compiler, the C compiler, or the loader. The COBOL compiler name is assumed to be `cob` and already in the `PATH`.

See Also `buildclient(1)`, `buildserver(1)`

`cc(1)` in a UNIX system reference manual

Micro Focus COBOL/2 Operating Guide, Micro Focus Ltd.

dmadmin(1)

Name `dmadmin` - BEA Tuxedo Domains Administration Command Interpreter

Synopsis `dmadmin [-c]`

Description `dmadmin` is an interactive command interpreter used for the administration of domain gateway groups defined for a particular BEA Tuxedo application. This page describes the use of `dmadmin` for both TDomain gateways and the TOP END Domain Gateway (TEDG) feature of the BEA Tuxedo system. `dmadmin` can operate in two modes: administration mode and configuration mode.

`dmadmin` enters *administration* mode when called with no parameters. This is the default. In this mode, `dmadmin` can be run on any active node (excluding workstations) within an active application. Application administrators can use this mode to obtain or change parameters on any active domain gateway group. Application administrators may also use this mode to create, destroy, or reinitialize the `DMTLOG` for a particular local domain. In this case, the domain gateway group associated with that local domain must not be active, and `dmadmin` must be run on the machine assigned to the corresponding gateway group.

`dmadmin` enters *configuration* mode when it is invoked with the `-c` option or when the `config` subcommand is invoked. Application administrators can use this mode to update or add new configuration information to the binary version of the domain configuration file (`BDMCONFIG`).

`dmadmin` requires the use of the Domains administrative server (`DMADM`) for the administration of the `BDMCONFIG` file, and the gateway administrative server (`GWADM`) for the re-configuration of active domain gateway groups. (There is one `GWADM` per gateway group.)

Administration Mode Commands Once `dmadmin` has been invoked, commands may be entered at the prompt (“>”) according to the following syntax:

`command [arguments]`

Several commonly occurring arguments can be given defaults via the `default` command. Commands that accept parameters set via the `default` command check `default` to see if a value has been set. If one has not, an error message is returned.

Once set, a default remains in effect until the session is ended, unless changed by another `default` command. Defaults may be overridden by entering an explicit value on the command line, or unset by entering the value `“*”`. The effect of an override lasts for a single instance of the command.

Output from `dmadmin` commands is paginated according to the pagination command in use (see the `paginate` subcommand below).

Commands may be entered either by their full name or their abbreviation (shown in parentheses) followed by any appropriate arguments. Arguments appearing in square brackets, `[]`, are optional; those in curly braces, `{ }`, indicate a selection from mutually exclusive options. Note that for many commands `local_domain_name` is a required argument, but note also that it can be set with the `default` command.

The following commands are available in `administration` mode:

`advertise (adv) -d local_domain_name [{ | service}]`

Advertise all remote services provided by the named local domain or the specified remote service.

`audit (audit) -d local_domain_name [{off | on}]`

Activate (on) or deactivate (off) the audit trace for the named local domain. If no option is given, then the current setting will be toggled between the values `on` and `off`, and the new setting will be printed. The initial setting is `off`.

`chbktime (chbt) -d local_domain_name -t bktime`

Change the blocking timeout for a particular local domain.

`config (config)`

Enter configuration mode. Commands issued in this mode follow the conventions defined in the section “Configuration Mode Commands” later in this reference page.

`connect (co) -d local_domain_name [-R remote_domain_name]`

Connect the local domain gateway to the remote gateway. If the connection attempt fails and you have configured the local domain gateway to retry a connection, repeated attempts to connect (via automatic connection retry processing) is made. (If `-R` is not specified, then the command applies to all remote domains configured for this local gateway.)

`crdmlog (crdlog)[-d local_domain_name]`

Create the domain transaction log for the named local domain on the current machine (that is, the machine where `dmadmin` is running). The command uses

the parameters specified in the `DMCONFIG` file. This command fails if the named local domain is active on the current machine or if the log already exists.

`default (d) [-d local_domain_name]`

Set the corresponding argument to be the default local domain. Defaults may be unset by specifying "*" as an argument. If the default command is entered with no arguments, the current defaults are printed.

`disconnect (dco) -d local_domain_name [-R remote_domain_name]`

Break the connection between the local domain gateway and the remote gateway and do not initiate connection retry processing. If no connection is active, but automatic connection retry processing is in effect, then stop the automatic retry processing. (If `-R` is not specified, then the command applies to all remote domains configured for this local gateway.)

`dsdmlog (dsdlg) -d local_domain_name [-y]`

Destroy the domain transaction log for the named local domain on the current machine (that is, the machine where `dmadmin` is running). An error is returned if a `DMTLOG` is not defined for this local domain, if the local domain is active, or if outstanding transaction records exist in the log. The term outstanding transactions means that a global transaction has been committed but an end-of-transaction has not yet been written. This command prompts for confirmation before proceeding unless the `-y` option is specified.

`echo (e) [{off | on}]`

Echo input command lines when set to on. If no option is given, then the current setting is toggled, and the new setting is printed. The initial setting is off.

`forgettrans (ft) -d local_domain_name [-t tran_id]`

Forget one or all heuristic log records for the named local domain. If the transaction identifier `tran_id` is specified, then only the heuristic log record for that transaction will be forgotten. The transaction identifier `tran_id` can be obtained from the `printtrans` command or from the `ULOG` file.

`help (h) [command]`

Print help messages. If `command` is specified, the abbreviation, arguments, and description for that command are printed. Omitting all arguments causes the syntax of all commands to be displayed.

`indmlog (indlg) -d local_domain_name [-y]`

Reinitialize the domain transaction log for the named local domain on the current machine (that is, the machine where `dmadmin` is running). An error is returned if a `DMTLOG` is not defined for this local domain, if the local domain is active, or if outstanding transaction records exist in the log. The term outstanding transactions means that a global transaction has been committed but an end-of-transaction has not yet been written. The command prompts for confirmation before proceeding unless the `-y` option is specified.

`paginate (page) [{off | on}]`

Paginate output. If no option is given, then the current setting will be toggled, and the new setting is printed. The initial setting is on, unless either standard input or standard output is a non-tty device. Pagination may only be turned on when both standard input and standard output are tty devices. The shell environment variable `PAGER` may be used to override the default command used for paging output. The default paging command is the indigenous one to the native operating system environment, for example, the command `pg` is the default on UNIX System operating environments.

`passwd (passwd) [-r] local_domain_name remote_domain_name`

Prompts the administrator for new passwords for the specified local and remote domains. The `-r` option specifies that existing passwords and new passwords should be encrypted using a new key generated by the system. The password is limited to at most 30 characters. `passwd` is not supported by the TOP END Domain Gateway.

`printdomain (pd) -d local_domain_name`

Print information about the named local domain. Information printed includes a list of connected remote domains, a list of remote domains being retried (if any), global information shared by the gateway processes, and additional information that is dependent on the domain type instantiation.

`printstats (pstats) -d local_domain_name`

Print statistical and performance information gathered by the named local domain. The information printed is dependent on the domain gateway type.

`printtrans (pt) -d local_domain_name`

Print transaction information for the named local domain.

`quit (q)`

Terminate the session.

`resume (res) -d local_domain_name [{ -all | service}]`
Resume processing of either the specified service or all remote services handled by the named local domain.

`stats (stats) -d local_domain_name [{ off | on | reset }]`
Activate (*on*), deactivate (*off*), or reset (*reset*) statistics gathering for the named local domain. If no option is given, then the current setting will be toggled between the values *on* and *off*, and the new setting will be printed. The initial setting is *off*.

`suspend (susp) -d local_domain_name [{ -all | service}]`
Suspend one or all remote services for the named local domain.

`topendpasswd (tepasswd) [-r] local_domain_name`
Prompts the administrator for a new BEA TOP END password for the specified local domain. The `-r` option specifies that existing passwords and new passwords should be encrypted using a new key generated by the system. The password is limited to at most 12 characters.

The new password is used when sending an RTQ message to a BEA TOP END system on which security is enabled. The user ID that is passed is the DOMAINID of the local domain. `topendpasswd` is supported only for TOP END Domain Gateways.

`unadvertise (unadv) -d local_domain_name [{ -all | service}]`
Unadvertise one or all remote services for the named local domain.

`verbose (v) [{off | on}]`
Produce output in verbose mode. If no option is given, then the current setting will be toggled, and the new setting is printed. The initial setting is *off*.

`! shellcommand`
Escape to shell and execute *shellcommand*.

`!!`
Repeat previous shell command.

`# [text]`
Lines beginning with "#" are comment lines and are ignored.

`<CR>`
Repeat the last command.

Configuration Mode Commands The `dmadmin` command enters configuration mode when executed with the `-c` option or when the `config` subcommand is used. In this mode, `dmadmin` allows run-time updates to the `BDMCONFIG` file. `dmadmin` manages a buffer that contains input field values to be added or retrieved, and displays output field values and status after each operation completes. The user can update the input buffer using any available text editor.

`dmadmin` first prompts for the desired section followed by a prompt for the desired operation.

The prompt for the section is as follows:

Section:

- | | |
|--------------------|-----------------------|
| 1) RESOURCES | 2) LOCAL_DOMAINS |
| 3) REMOTE_DOMAINS | 4) LOCAL_SERVICES |
| 5) REMOTE_SERVICES | 6) ROUTING |
| 7) ACCESS_CONTROL | 8) PASSWORDS |
| 9) TDOMAINS | 10) OSITPS |
| 11) SNADOMS | 12) LOCAL_REMOTE_USER |
| 13) REMOTE_USERS | 14) SNACRMS |
| 15) SNASTACKS | 16) SNALINKS |
| 18) TOPEND | q) QUIT |

Enter Section [1]:

The number of the default section appears in square brackets at the end of the prompt. You can accept the default by pressing `RETURN` or `ENTER`. To select another section enter its number, then press `RETURN` or `ENTER`.

`dmadmin` then prompts for the desired operation.

Operations:

- | | |
|----------------|-----------|
| 1) FIRST | 2) NEXT |
| 3) RETRIEVE | 4) ADD |
| 5) UPDATE | 6) DELETE |
| 7) NEW_SECTION | 8) QUIT |

Enter Operation [1]:

The number of the default operation is printed in square brackets at the end of the prompt. Pressing `RETURN` or `ENTER` selects this option. To select another operation enter its number, then press `RETURN` or `ENTER`.

The currently supported operations are:

1. `FIRST` - Retrieve the first record from the specified section. No key fields are needed (they are ignored if in the input buffer).

2. `NEXT` - Retrieve the next record from the specified section, based on the key fields in the input buffer.
3. `RETRIEVE` - Retrieve the indicated record from the specified section by key field(s) (see fields description below).
4. `ADD` - Add the indicated record in the specified section. Any fields not specified (unless required) take their defaults as specified in `DMCONFIG(5)`. The current value for all fields is returned in the output buffer. This operation can only be done by the BEA Tuxedo administrator.
5. `UPDATE` - Update the record specified in the input buffer in the selected section. Any fields not specified in the input buffer remain unchanged. The current value for all fields is returned in the input buffer. This operation can only be done by the BEA Tuxedo administrator.
6. `DELETE` - Delete the record specified in the input buffer from the selected section. This operation can only be done by the BEA Tuxedo system administrator.
7. `NEW SECTION` - Clear the input buffer (all fields are deleted). After this operation, `dmadmin` immediately prompts for the section again.
8. `QUIT` - Exit the program gracefully (`dmadmin` is terminated). A value of `q` for any prompt also exits the program.

For configuration operations, the effective user identifier must match the BEA Tuxedo administrator user identifier (UID) for the machine on which this program is executed. When a record is updated or added, all defaults and validations used by `dmloadcf(1)` are enforced.

`dmadmin` then prompts you to indicate whether you want to edit the input buffer.

```
Enter editor to add/modify fields [n]?
```

Entering a value of `y` puts the input buffer into a temporary file and executes the text editor. The environment variable `EDITOR` is used to determine which editor is to be used; the default is `ed`. The input format is a set of field name/field value pairs and is described in the “Configuration Input Format” section below. The field names associated with each `DMCONFIG` section are listed in tables in the subsections below. The semantics of the fields and associated ranges, defaults, restrictions, and so on are described in `DMCONFIG(5)` and `DMCONFIG for GWTOPEND(5)`. In most cases, the field name is the same as the `KEYWORD` in the `DMCONFIG` file, prefixed with `"TA_"`. When the user completes editing the input buffer, `dmadmin` reads it. If more than one line is

included for a particular field name, the first line is used and other lines are ignored. If any errors occur, a syntax error is printed and `dmadmin` prompts you to indicate whether you want to edit the file to correct the problem.

```
Enter editor to correct?
```

If the problem is not corrected (response `n`), then the input buffer will contain no fields. Otherwise, the editor is executed again.

Finally, `dmadmin` asks whether the operation should be executed.

```
Perform operation [y]?
```

When the operation completes, `dmadmin` prints the return value as in `Return value TAOK` followed by the output buffer fields. The process then begins again with a prompt for the section. All output buffer fields are available in the input buffer unless the buffer is cleared.

Entering break at any time restarts the interaction at the prompt for the section.

When "QUIT" is selected, `dmadmin` prompts for authorization to create a backup text version of the configuration file:

```
Unload BDMCONFIG file into ASCII backup [y]?
```

If a backup is selected, `dmadmin` prompts for a filename.

```
Backup filename [DMCONFIG]
```

On success, `dmadmin` indicates that a backup was created; otherwise, an error is printed.

Configuration Input Format

Input packets consist of lines formatted as follows:

```
fldname fldval
```

The field name is separated from the field value by one or more tabs (or spaces).

Lengthy field values can be continued on the next line by having the continuation line begin with one or more tabs (which are dropped when read back into `dmadmin`).

Empty lines consisting of a single newline character are ignored.

To enter an unprintable character in the field value or to start a field value with a tab, use a backslash followed by the two-character hexadecimal representation of the desired character (see `ASCII(5)` in a UNIX reference manual). A space, for example,

can be entered in the input data as \20. A backslash can be entered using two backslash characters. `dmadmin` recognizes all input in this format, but its greatest usefulness is for non-printing characters.

Configuration Limitations

The following are general limitations of the dynamic domain reconfiguration capability:

- Values for key fields (as indicated in the following sections) may not be modified. Key fields can be modified, when the system is down, by reloading the configuration file.
- Dynamic deletions cannot be applied when local domains are active (the corresponding gateway group is running).

Domains Terminology Improvements

In this release, some of the domains terminology is changing. The Domains MIB uses improved class and attribute terminology to describe the interaction between local and remote domains. While this improved terminology is more accurate than previous domains terminology, the scope of changes to domains-related documentation and error messages is limited in this release. The improved terminology has been applied to the `DM_MIB` classes, reference page, and error messages, the `DMCONFIG` file syntax, and various `DMCONFIG` error messages.

For backwards compatibility, aliases are provided between the `DMCONFIG` terminology used prior to this release and the improved Domains MIB terminology. In this release, `DMCONFIG` accepts both versions of the terminology. For details, see “Domains Terminology Improvements” in the `DM_MIB(5)` reference page.

Restrictions for Configuration Field Identifiers/ Updates

The following sections describe, for each `DMCONFIG` section, the field identifiers associated with each `DMCONFIG` field, the field type of each identifier, and when each field can be updated. All applicable field values are returned with the retrieval operations. Fields that are allowed and/or required for adding a record are described in `DMCONFIG(5)` and `DMCONFIG for GWTOPEND(5)`. Fields indicated below as *key* are key fields that are used to uniquely identify a record within section. These key fields are required to be in the input buffer when updates are done and are not allowed to be updated dynamically. The `Update` column indicates when a field can be updated. The possible values are:

- `Yes`—Can be updated at any time.
- `NoGW`—Cannot be updated dynamically while the gateway group representing the local domain is running.
- `No`—Cannot be updated dynamically while at least one gateway group is running.

Configuring the
DM_LOCAL_D
OMAINS
Section

The following table lists the fields in the DM_LOCAL_DOMAINS section.

DM_LOCAL_DOMAINS SECTION

Field Identifier	Type	Update	Notes
TA_LDOM	string	NoGW	key
TA_AUDITLOG	string	NoGW	
TA_BLOCKTIME	numeric	NoGW	
TA_CONNECTION_POLICY	string	NoGW	
TA_DOMAINID	string	NoGW	
TA_DMTLOGDEV	string	NoGW	
TA_DMTLOGNAME	string	NoGW	
TA_DMTLOGSIZE	numeric	NoGW	
TA_GWGRP	string	NoGW	
TA_MAXRDOM	numeric	NoGW	
TA_MAXRDTRAN	numeric	NoGW	
TA_MAXRETRY	numeric	NoGW	
TA_MAXTRAN	numeric	NoGW	
TA_RETRY_INTERVAL	numeric	NoGW	
TA_SECURITY	string	NoGW	TDomain format: {NONE APP_PW DM_PW} TEDG format: {NONE CLEAR SAFE PRIVATE}
TA_TYPE	string	NoGW	format:{TDOMAIN OSITP TOPEND}

Section 1 - Commands

Configuring the DM_REMOTE_ DOMAINS Section

The following table lists the fields in the DM_REMOTE_DOMAINS section.

DM_REMOTE_DOMAINS SECTION

Field Identifier	Type	Update	Notes
TA_RDOM	string	No	key
TA_DOMAINID	string	No	
TA_TYPE	string	No	format: {TDOMAIN OSITP TOPEND}

Configuring the DM_TDOMAIN Section

The DM_TDOMAIN section contains the network addressing parameters required by TDOMAIN type domains. The following table lists the fields in this section.

DM_TDOMAIN SECTION

Field Identifier	Type	Update	Notes
TA_LDOM or TA_RDOM	string	No/NoGW	key
TA_NWADDR	string	No/NoGW	text format (no embedded NULL characters)
TA_NWDEVICE	string	No/NoGW	

If the domain identifier (TA_LDOM) is a local domain identifier, then the TA_NWADDR and TA_NWDEVICE fields can be updated if the gateway group representing that local domain is not running.

Configuring the DM_OSITP Section

The DM_OSITP section contains the network addressing parameters required by OSITP type domains. The following table lists the fields in this section.

DM_OSITP SECTION

Field Identifier	Type	Update	Notes
TA_LDOM or TA_RDOM	string	No/NoGW	key
TA_APT	string	No/NoGW	
TA_AEQ	string	No/NoGW	

DM_OSITP SECTION

Field Identifier	Type	Update	Notes
TA_AET	string	No/NoGW	
TA_ACN	string	No/NoGW	
TA_APID	string	No/NoGW	
TA_AEID	string	No/NoGW	
TA_PROFILE	string	No/NoGW	

If the domain identifier (TA_LDOM) is a local domain identifier, then the other fields in this table can be updated if the gateway group representing that local domain is not running.

Configuring the
DM_TOPEND
Section

The DM_TOPEND section contains the network addressing parameters required by TOPEND type domains. The following table lists the fields in this section.

DM_TOPEND SECTION

Field Identifier	Type	Update	Notes
TA_LDOM or TA_RDOM	string	No/NoGW	key
TA_NWADDR	string	No/NoGW	ASCII format (no embedded NULL characters)
TA_NWDEVICE	string	No/NoGW	ASCII format (no embedded NULL characters)
TA_TP_SYSTEM	string	No/NoGW	BEA TOP END system name
TA_TE_PWD	string	NoGW	BEA TOP END password. Applies only to local entries.

If

- The domain identifier is a local domain identifier (TA_LDOM), and
- The gateway group representing that local domain is not running (NoGW)

then the TA_NWADDR, TA_NWDEVICE, TA_TP_SYSTEM, and TA_TE_PWD fields can be updated. If the domain identifier is a remote domain identifier (TA_RDOM), then the TA_NWADDR, TA_NWDEVICE, and TA_TP_SYSTEM fields cannot be updated while any gateway group is running (No). Note that TE_TE_PWD applies only to local domain identifiers (TA_LDOM).

Configuring the
DM_LOCAL_SERVICES Section

The following table lists the fields in the DM_LOCAL_SERVICES section.

DM_LOCAL_SERVICES SECTION

Field Identifier	Type	Update	Notes
TA_SERVICENAME	string	No	key
TA_LDOM	string	Yes	
TA_RNAME	string	Yes	Not applicable to TEDG
TA_ACLNAME	string	Yes	
TA_BUFTYPE	string	Yes	
TA_BUFSTYPE	string	Yes	
TA_OBUFTYPE	string	Yes	
TA_OBUFSTYPE	string	Yes	
TA_TE_PRODUCT	string	NoGW	Applicable to TEDG only
TA_TE_FUNCTION	string	NoGW	Applicable to TEDG only
TA_TE_TARGET	string	NoGW	Applicable to TEDG only
TA_TE_QUALIFIER	numeric	NoGW	Applicable to TEDG only

DM_LOCAL_SERVICES SECTION

Field Identifier	Type	Update	Notes
TA_TE_RTQGROUP	string	NoGW	Applicable to TEDG only
TA_TE_RTQNAME	string	NoGW	Applicable to TEDG only
TA_TYPE	string	NoGW	Format: {SERVICE QSPACE QNAME } Applicable to TEDG only

Configuring the
DM_REMOTE_
SERVICES
Section

The following table lists the fields in the DM_REMOTE_SERVICES section.

DM_REMOTE_SERVICES SECTION

Field Identifier	Type	Update	Notes
TA_SERVICENAME	string	No	key
TA_RDOM	string	No	key
TA_LDOM	string	No	key
TA_RNAME	string	Yes	Not applicable to TEDG
TA_CONV	string	NoGW	format: { Y N }
TA_BUFTYPE	string	Yes	
TA_BUFSTYPE	string	Yes	
TA_OBUFTYPE	string	Yes	
TA_OBUFSTYPE	string	Yes	
TA_ROUTINGNAME	string	Yes	
TA_TRANTIME	numeric	Yes	
TA_TE_PRODUCT	<i>string</i>	Yes	Applicable to TEDG only
TA_TE_FUNCTION	<i>string</i>	Yes	Applicable to TEDG only

DM_REMOTE_SERVICES SECTION

Field Identifier	Type	Update	Notes
TA_TARGET	string	Yes	Applicable to TEDG only
TA_TE_QUALIFIER	numeric	Yes	Applicable to TEDG only
TA_TE_RTQGROUP	string	Yes	Applicable to TEDG only
TA_TE_RTQNAME	string	Yes	Applicable to TEDG only
TA_TYPE	string	Yes	Format: {SERVICE QSPACE QNAME} Applicable to TEDG only

Configuring the
DM_ROUTING
Section

The following table lists the fields in the DM_ROUTING section.

DM_ROUTING SECTION

Field Identifier	Type	Update	Notes
TA_ROUTINGNAME	string	No	key
TA_FIELD	string	Yes	
TA_RANGE	string	Yes	
TA_BUFTYPE	string	Yes	

Configuring the
DM_ACCESS_C
ONTROL
Section

The following table lists the fields in the DM_ACCESS_CONTROL section.

DM_ACCESS_CONTROL SECTION

Field Identifier	Type	Update	Notes
TA_ACLNAME	string	No	key
TA_RDOM	string	Yes	

Configuring the
DM_PASSWORDS
DS Section

The following table lists the fields in the DM_PASSWORDS section. This section does not apply to the TEDG.

***DM_PASSWORDS SECTION**

Field Identifier	Type	Update	Notes
TA_LDOM	string	No	key
TA_RDOM	string	No	key
TA_LPWD	string	Yes	format: { Y N U }
TA_RPWD	string	Yes	format: { Y N U }

The `TA_LPWD` and `TA_RPWD` show the existence of a defined password for the local and/or the remote domain. Passwords are not displayed. If an `UPDATE` operation is selected, the value of the corresponding field must be set to `U`. The program will then prompt with echo turned off for the corresponding passwords.

**Diagnostics in
Configuration
Mode**

`dmadmin` fails if it cannot allocate an FML typed buffer, if it cannot determine the `/etc/passwd` entry for the user, or if it cannot reset the environment variables `FILEDTBLS` or `FLDTBLDIR`.

The return value printed by `dmadmin` after each operation completes indicates the status of the requested operation. There are three classes of return values.

The following return values indicate a problem with permissions or a BEA Tuxedo communications error. They indicate that the operation did not complete successfully.

[TAEPERM]

The calling process specified an `ADD`, `UPDATE`, or `DELETE` operation but it is not running as the BEA Tuxedo administrator. Update operations must be run by the administrator (that is, the user specified in the `UID` attribute of the `RESOURCES` section of the `TUXCONFIG` file).

[TAESYSTEM]

A BEA Tuxedo error has occurred. The exact nature of the error is written to `userlog(3c)`.

[TAEOS]

An operating system error has occurred.

[TAETIME]

A blocking timeout occurred. The input buffer was not updated so no information was returned for retrieval operations. The status of update

operations can be checked by doing a retrieval on the record that was being updated.

The following return values indicate a problem in doing the operation itself and generally are semantic problems with the application data in the input buffer. The string field `TA_STATUS` will be set in the output buffer and will contain short text describing the problem. The string field `TA_BADFLDNAME` will be set to the field name for the field containing the value that caused the problem (assuming the error can be attributed to a single field).

[TAECONFIG]

An error occurred while the `BDMCONFIG` file was being read.

[TAEDUPLICATE]

The operation attempted to add a duplicate record.

[TAEINCONSIS]

A field value or set of field values are inconsistently specified.

[TAENOTFOUND]

The record specified for the operation was not found.

[TAENOSPACE]

The operation attempted to do an update but there was not enough space in the `BDMCONFIG` file.

[TAERANGE]

A field value is out of range or is invalid.

[TAEREQUIRED]

A field value is required but not present.

[TAESIZE]

A field value for a string field is too long.

[TAEUPDATE]

The operation attempted to do an update that is not allowed.

The following return values indicate that the operation was successful.

[TAOK]

The operation succeeded. No updates were made to the `BDMCONFIG` file.

[TAUPDATED]

The operation succeeded. Updates were made to the `BDMCONFIG` file.

When using `dmunloadcf` to print entries in the configuration, optional field values are not printed if they are not set (for strings) or 0 (for integers). These fields will always appear in the output buffer when using `dmadmin`. In this way, it makes it easier for the administrator to retrieve an entry and update a field that previously was not set. The entry will have the field name followed by a tab but no field value.

Configuration Example In the following example, `dmadmin` is used to add a new remote domain. For illustration purposes, `ed(1)` is used for the editor.

```
$ EDITOR=ed dmadmin
> config
Sections:
    1) RESOURCES           2) LOCAL_DOMAINS
    3) REMOTE_DOMAINS     4) LOCAL_SERVICES
    5) REMOTE_SERVICES    6) ROUTING
    7) ACCESS_CONTROL     8) PASSWORDS
    9) TDOMAINS          10) OSITPS
   11) SNADOMS           12) LOCAL_REMOTE_USER
   13) REMOTE_USERS      14) SNACRMS
   15) SNASTACKS         16) SNALINKS
   18) TOPEND            q) QUIT

Enter Section [1]: 2
Operations:
    1) FIRST              2) NEXT
    3) RETRIEVE           4) ADD
    5) UPDATE             6) DELETE
    7) NEW_SECTION       8) QUIT

Enter Operation [1]: 4
Enter editor to add/modify fields [n]? y
a
TA_RDOM           B05
TA_DOMAININID    BA.BANK05
TA_TYPE          TDOMAIN
w
53
q
Perform operation [y]? <return>
Return value TAUPDATED
Buffer contents:
TA_OPERATION      4
TA_SECTION        2
TA_DOMAININID    BA.BANK05
TA_RDOM          B05
TA_TYPE          TDOMAIN
TA_STATUS         Update completed successfully
Operations:
    1) FIRST              2) NEXT
    3) RETRIEVE           4) ADD
```

Section 1 - Commands

```

        5) UPDATE          6) DELETE
        7) NEW_SECTION    8) QUIT
Enter Operation [4]: 7
Section:
    1) RESOURCES          2) LOCAL_DOMAINS
    3) REMOTE_DOMAINS    4) LOCAL_SERVICES
    5) REMOTE_SERVICES   6) ROUTING
    7) ACCESS_CONTROL    8) PASSWORDS
    9) TDOMAINS         10) OSITPS
   11) SNADOMS          12) LOCAL_REMOTE_USER
   13) REMOTE_USERS     14) SNACRMS
   15) SNASTACKS       16) SNALINKS
   18) TOPEND          q) QUIT
Enter Section [1]: 9
Operations:
    1) FIRST            2) NEXT
    3) RETRIEVE        4) ADD
    5) UPDATE          6) DELETE
    7) NEW_SECTION    8) QUIT
Enter Operation [6]: 4
Enter editor to add/modify fields [n]? y
a
TA_RDOM          B05
TA_NWADDR        0x00020401c0066d05
TA_NWDEVICE      /dev/tcp
w
55
q
Perform operation [y]? <return>
Return value TAUPDATED
Buffer contents:
TA_OPERATION     4
TA_SECTION       8
TA_RDOM          B05
TA_NWADDR        0x00020401c0066d05
TA_NWDEVICE      /dev/tcp
TA_STATUS        Update completed successfully
Operations:
    1) FIRST            2) NEXT
    3) RETRIEVE        4) ADD
    5) UPDATE          6) DELETE
    7) NEW_SECTION    8) QUIT
Enter Operation [4]: 8
> quit
```

The `dmadmin` program ends.

-
- Security** If `dmadmin` is run using the UID of the application administrator, it is assumed that the user is a trusted user and security is bypassed. If `dmadmin` is run with another user ID, and the security option is enabled in the `TUXCONFIG` file, then the corresponding application password is required to start the `dmadmin` program. If standard input is a terminal, then `dmadmin` will prompt the user for the password with echo turned off. If standard input is not a terminal, the password is retrieved from the environment variable, `APP_PW`. If this environment variable is not specified and an application password is required, then `dmadmin` will fail to start.
- When running with another user ID (other than the UID of the administrator) only a limited set of commands is available.
- Environment Variables** `dmadmin` resets the `FIELDTBLS` and `FLDTBLDIR` environment variables to pick up the `/${TUXDIR}/udataobj/dmadmin` field table. Hence, the `TUXDIR` environment variable should be set correctly.
- If the application requires security and the standard input to `dmadmin` is not from a terminal, then the `APP_PW` environment variable must be set to the corresponding application password.
- The `TUXCONFIG` environment variable should be set to the path name of the BEA Tuxedo configuration file.
- General Diagnostics** If the `dmadmin` command is entered before the system has been booted, the following message is displayed:
- No bulletin board exists. Only logging commands are available.
- `dmadmin` then prompts for the corresponding commands.
- If an incorrect application password is entered or is not available to a shell script through the environment, then a log message is generated, the following message is displayed, and the command terminates: `Invalid password entered.`
- Interoperability** `dmadmin` must be installed on BEA Tuxedo Release 5.0 or later. Other nodes in the same domain with a Release 5.0 gateway may be BEA Tuxedo Release 4.1 or later.
- Portability** The `dmadmin` administrative tool is supported on any platform on which the BEA Tuxedo server environment is supported.
- See Also** `dmloadcf(1)`, `tmadmin(1)`, `topendpasswd(1)`, `DMADM(5)`, `DMCONFIG(5)`, `DMCONFIG` for `GWTOPEND(5)`
- Using the BEA Tuxedo Domains Component*
- Using the BEA Tuxedo TOP END Domain Gateway*

dmloadcf(1)

Name dmloadcf - parse a DMCONFIG file and load a binary BDMCONFIG configuration file

Synopsis dmloadcf [-c] [-n] [-y] [-b *blocks*] {*DMCONFIG_file* | - }

Description dmloadcf reads a file or standard input that is in DMCONFIG syntax, checks the syntax, and optionally loads a binary BDMCONFIG configuration file. The BDMCONFIG environment variable points to the path name of the BDMCONFIG file where the information should be stored.

dmloadcf prints an error message if it finds any required section of the DMCONFIG file missing. If a syntax error is found while the input file is being parsed, dmloadcf exits without performing any updates to the BDMCONFIG file.

dmloadcf requires the existence of the \$TUXDIR/udataobj/DMTYPE file. This file defines valid domain types. If this file does not exist, dmloadcf exits without performing any updates to the BDMCONFIG file.

The effective user identifier of the person running dmloadcf must match the UID in the RESOURCES section of the TUXCONFIG file.

The -c option to dmloadcf causes the program to print the minimum amount of IPC resources needed for each local domain (gateway group) in this configuration. The BDMCONFIG file is not updated.

The -n option to dmloadcf causes the program to do only syntax checking of the text DMCONFIG file without updating the BDMCONFIG file.

After syntax checking, dmloadcf checks whether the file referenced by the BDMCONFIG environment variable exists, is a valid BEA Tuxedo file, and contains BDMCONFIG tables. If these conditions are not true, dmloadcf gives the user a chance to create and initialize the file by displaying the following prompt:

```
Initialize BDMCONFIG file: path[y, q]?
```

Here *path* is the complete filename of the BDMCONFIG file. Prompting is suppressed if the standard input and output are not directed to a terminal, or if the -y option is specified on the command line. Any response other than “y” or “Y” causes dmloadcf to exit without creating a binary configuration file.

If the BDMCONFIG file is not properly initialized, and the user has entered *y* after the Initialize BDMCONFIG file prompt, then dmloadcf creates the BEA Tuxedo file system and creates the BDMCONFIG tables. If the *-b* option is specified on the command line, its argument defines the number of blocks for the device when the BEA Tuxedo file system is created. If the value of the *-b* option is large enough to hold the new BDMCONFIG tables, dmloadcf uses the specified value to create the new file system; otherwise, dmloadcf prints an error message and exits. If the *-b* option is not specified, dmloadcf creates a new file system large enough to hold the BDMCONFIG tables. The *-b* option is ignored if the file system already exists. The *-b* option is highly recommended if BDMCONFIG is a raw device (that is, a device that has not been initialized). In this case, *-b* should be used to set the number of blocks on the raw device. The *-b* option is not recommended if BDMCONFIG is a regular UNIX file.

If the BDMCONFIG file has been initialized already, dmloadcf ensures that the local domain described by that BDMCONFIG file is not running. If a local domain is running, dmloadcf prints an error message and exits. Otherwise, dmloadcf, to confirm that the file should be overwritten, prompts the user with:

```
"Really overwrite BDMCONFIG file [y, q]?"
```

Prompting is suppressed if the standard input or output are not a terminal or if the *-y* option is specified on the command line. Any response other than "y" or "Y" will cause dmloadcf to exit without overwriting the file.

If the SECURITY parameter is specified in the RESOURCES section of the TUXCONFIG file, then dmloadcf flushes the standard input, turns off terminal echo, and prompts the user for an application password as follows: Enter Application Password? The password is limited to 30 characters. The option to load the text DMCONFIG file via the standard input (rather than a file) cannot be used when this SECURITY parameter is turned on. If the standard input is not a terminal, that is, if the user cannot be prompted for a password (as with a here file, for example), then the environment variable APP_PW is accessed to set the application password. If the environment variable APP_PW is not set with the standard input not a terminal, then dmloadcf will print an error message, generate a log message and fail to load the BDMCONFIG file.

If no errors have occurred and all checks have passed, dmloadcf loads the DMCONFIG file into the BDMCONFIG file. It overwrites all existing information found in the BDMCONFIG tables.

Domains Terminology Improvements

In this release, some of the domains terminology is changing. The Domains MIB uses improved class and attribute terminology to describe the interaction between local and remote domains. While this improved terminology is more accurate than previous domains terminology, the scope of changes to domains-related documentation and

error messages is limited in this release. The improved terminology has been applied to the `DM_MIB` classes, reference page, and error messages, the `DMCONFIG` file syntax, and various `DMCONFIG` error messages. For details, see “Domains Terminology Improvements” in the `DM_MIB(5)` reference page.

For backwards compatibility, aliases are provided between the `DMCONFIG` terminology used prior to this release and the improved Domains MIB terminology. In this release, `dmloadcf` accepts both versions of the `DMCONFIG` terminology. `dmunloadcf`, however, generates a `DMCONFIG` file that uses the improved domains terminology by default. Use the `-c` option of `dmunloadcf` to generate a `DMCONFIG` file that uses the previous domains terminology.

- Portability** The `dmloadcf` administrative tool is supported on any platform on which the BEA Tuxedo server environment is supported.
- Environment Variables** The environment variable `APP_PW` must be set for applications on which security is required (the `SECURITY` parameter in the `TUXCONFIG` file is set to `APP_PW`) and `dmloadcf` is run with standard input directed to a destination other than a terminal. The `BDMCONFIG` environment variable should point to the `BDMCONFIG` file.
- Examples** The following example shows how a binary configuration file is loaded from the `bank.dmconfig` text file. The `BDMCONFIG` device is created (or reinitialized) with 2000 blocks:
- ```
dmloadcf -b 2000 bank.dmconfig
```
- Diagnostics** If an error is detected in the input, the offending line is printed to standard error, along with a message indicating the problem. If a syntax error is found in the `DMCONFIG` file or the system is currently running, no information is updated in the `BDMCONFIG` file and `dmloadcf` exits with exit code 1.
- If `dmloadcf` is run on an active node, the following error message is displayed:
- ```
*** dmloadcf cannot run on an active node ***
```
- If `dmloadcf` is run by a person whose effective user identifier does not match the `UID` specified in the `TUXCONFIG` file, the following error message is displayed:
- ```
*** UID is not effective user ID ***
```
- Upon successful completion, `dmloadcf` exits with exit code 0. If the `BDMCONFIG` file is updated, a `userlog` message is generated to record this event.

**See Also** `dmunloadcf(1)`, `DMCONFIG(5)`, `DMCONFIG` for `GWTOPEND(5)`, `UBBCONFIG(5)`

*Administering a BEA Tuxedo Application at Run Time*

*Using the BEA Tuxedo Domains Component*

*Using the BEA Tuxedo TOP END Domain Gateway*

# dmunloadcf(1)

|                                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name                             | <code>dmunloadcf</code> - unload binary BDMCONFIG domain configuration file                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Synopsis                         | <code>dmunloadcf [-c]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Description                      | <p><code>dmunloadcf</code> translates the BDMCONFIG configuration file from the binary representation into text. This translation is useful for transporting the file in a compact way between machines with different byte orderings, and for making a backup copy of the file in a compact form for reliability. The text format is the same as that described in DMCONFIG(5).</p> <p><code>dmunloadcf</code> reads values from the BDMCONFIG file referenced by the BDMCONFIG environment variable and writes them to standard output.</p> <p>In this release, <code>dmunloadcf</code>, by default, generates a DMCONFIG file that uses the improved domains terminology. Refer to the following section, “Domains Terminology Improvements,” for details. Use the <code>-c</code> option to generate a DMCONFIG file that uses the previous domains terminology.</p>                                                                                                                                                                                                                                                                                        |
| Domains Terminology Improvements | <p>In this release, some of the domains terminology is changing. The Domains MIB uses improved class and attribute terminology to describe the interaction between local and remote domains. While this improved terminology is more accurate than previous domains terminology, the scope of changes to domains-related documentation and error messages is limited in this release. The improved terminology has been applied to the DM_MIB classes, reference page, and error messages, the DMCONFIG file syntax, and various DMCONFIG error messages. For details, see “Domains Terminology Improvements” in the DM_MIB(5) reference page.</p> <p>For backward compatibility, aliases are provided between the DMCONFIG terminology used prior to this release and the improved Domains MIB terminology. In this release, <code>dmloadcf</code> accepts both versions of the DMCONFIG terminology. <code>dmunloadcf</code>, however, generates a DMCONFIG file that uses the improved domains terminology by default. Use the <code>-c</code> option of <code>dmunloadcf</code> to generate a DMCONFIG file that uses the previous domains terminology.</p> |
| Portability                      | The <code>dmunloadcf</code> command is supported on any platform on which the BEA Tuxedo server environment is supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

**Examples** To unload the configuration in `/usr/tuxedo/BDMCONFIG` into the file `bdmconfig.backup`:

```
BDMCONFIG=/usr/tuxedo/BDMCONFIG dmunloadcf > bdmconfig.backup
```

**Diagnostics** `dmunloadcf` checks that the file referenced by the `BDMCONFIG` environment variable exists, is a valid BEA Tuxedo file system, and contains `BDMCONFIG` tables. If any of these conditions is not met, `dmunloadcf` prints an error message and exits with error code 1. Upon successful completion, `dmunloadcf` exits with exit code 0.

**See Also** `dmloadcf(1)`, `DMCONFIG(5)`, `DMCONFIG` for `GWTOPEND(5)`

*Using the BEA Tuxedo Domains Component*

*Using the BEA Tuxedo TOP END Domain Gateway*

# gencat(1)

Name `gencat` - generate a formatted message catalog

Synopsis `gencat [-m] catfile msgfile . . .`

Description The `gencat` utility merges the message text source file(s) `msgfile` into a formatted message database `catfile`. The database `catfile` is created if it does not already exist. If `catfile` does exist its messages are included in the new `catfile`. If set and message numbers collide, the new message text defined in `msgfile` replaces the old message text currently contained in `catfile`. The message text source file (or set of files) input to `gencat` can contain either set and message numbers or simply message numbers, in which case the set `NL_SETD` (see `nl_types(5)`) is assumed.

The format of a message text source file is defined in the list below. Note that the fields of a message text source line are separated by a single ASCII space or tab character. Any other ASCII spaces or tabs are considered to be part of the subsequent field.

`$set n comment`

Where `n` specifies the set identifier of the following messages until the next `$set`, `$delset`, or end-of-file appears. `n` must be a number in the range  $(1 - \{NL\_SETMAX\})$ . Set identifiers within a single source file need not be contiguous. Any string following the set identifier is treated as a comment. If no `$set` directive is specified in a message text source file, all messages are located in the default message set. `NL_SETD`.

`$delset n comment`

Deletes message set `n` from an existing message catalogue. Any string following the set number is treated as a comment. (Note: if `n` is not a valid set it is ignored.)

`$ comment`

A line beginning with a dollar symbol (\$) followed by an ASCII space or tab character is treated as a comment.

`m message_text`

The `m` denotes the message identifier, which is a number in the range  $(1 - \{NL\_MSGMAX\})$ . (Do not confuse this message text syntax with the `-m` command-line option described under `NOTES`.) The message text is stored in the message catalogue with the set identifier specified by the last `$set` directive, and with message identifier `m`. If the message text is empty, and an

ASCII space or tab field separator is present, an empty string is stored in the message catalog. If a message source line has a message number, but neither a field separator nor message text, the existing message with that number (if any) is deleted from the catalogue. Message identifiers need not be contiguous. The length of message text must be in the range (0- {NL\_TEXTMAX}).

`$quote c`

This line specifies an optional quote character *c*, which can be used to surround message text so that trailing spaces or null (empty) messages are visible in a message source line. By default, or if an empty `$quote` directive is supplied, no quoting of message text is recognized. Empty lines in a message text source file are ignored. Text strings can contain the special characters and escape sequences defined in the following table.

| Description     | Symbol           | Escape Sequence   |
|-----------------|------------------|-------------------|
| newline         | NL(LF)           | <code>\n</code>   |
| horizontal tab  | HT               | <code>\t</code>   |
| vertical tab    | VT               | <code>\v</code>   |
| backspace       | BS               | <code>\b</code>   |
| carriage return | CR               | <code>\r</code>   |
| form feed       | FF               | <code>\f</code>   |
| backslash       | <code>\</code>   | <code>\\</code>   |
| bit pattern     | <code>ddd</code> | <code>\ddd</code> |

The escape sequence `\ddd` consists of a backslash followed by 1, 2, or 3 octal digits, which are taken to specify the value of the desired character. If the character following a backslash is not one of those specified, the backslash is ignored.

A backslash followed by an ASCII newline character is also used to continue a string on the following line. Thus, the following two lines describe a single message string:

```
1 This line continues \
to the next line
```

which is equivalent to:

```
1 This line continues to the next line
```

**Portability** `gencat` is supported on any platform on which the BEA Tuxedo server environment is supported.

**Notices** This version of `gencat` produces a catalog that at run-time is read into `malloc`'ed space. Shared catalogs available with some versions of `gencat` are not available. On some systems, generation of `malloc`'ed catalogs requires that the `-m` option be specified. This option can be specified on the command line, but has no effect. `malloc`'ed catalogs are the default; the `-m` option is supported for compatibility only.

The catalog file generated by this command is limited in size to 64K. Larger catalog files result in an error being reported by this command and no catalog file being generated.

**See Also** `nl_types(5)`

# mkfldhdr, mkfldhdr32(1)

|                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Name</b>        | mkfldhdr, mkfldhdr32 - create header files from field tables                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Synopsis</b>    | mkfldhdr [-d <i>outdir</i> ] [ <i>field_table...</i> ]<br>mkfldhdr32 [-d <i>outdir</i> ] [ <i>field_table...</i> ]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Description</b> | <p>mkfldhdr translates each field table file to a corresponding header file suitable for inclusion in C programs. The resulting header files provide <code>#define</code> macros for converting from field names to field IDs. Header filenames are formed by concatenating a <code>.h</code> to the simple filename for each file to be converted.</p> <p>The field table names may be specified on the command line; each file is converted to a corresponding header file.</p> <p>If the field table names are not given on the command line, then the program uses the <code>FLD_TBL</code> environment variable as the list of field tables to be converted, and <code>FLDTBLDIR</code> environment variable as a list of directories to be searched for the files. <code>FLD_TBL</code> specifies a comma-separated list of field table filenames. If <code>FLD_TBL</code> has no value, <code>fld.tbl</code> is used as the name of the (only) field table file (in this case, the resulting header file will be <code>(fld.tbl.h)</code>). The <code>FLDTBLDIR</code> environment variable is a colon-separated list of directories in which to look for each field table whose name is not an absolute path name; the search for field tables is very similar to the search for executable commands using the UNIX System <code>PATH</code> variable. If <code>FLDTBLDIR</code> is not defined, only the current directory is searched. Thus, if no field table names are specified on the command line and <code>FLD_TBL</code> and <code>FLDTBLDIR</code> are not set, <code>mkfldhdr</code> will convert the field table <code>fld.tbl</code> in the current directory into the header file <code>fld.tbl.h</code>.</p> <p>The <code>-d</code> option is available to specify that the output header files are to be created in a directory other than the present working directory.</p> <p><code>mkfldhdr32</code> is used for 32-bit FML. It uses the <code>FLD_TBL32</code> and <code>FLDTBLDIR32</code> environment variables.</p> |
| <b>Errors</b>      | Error messages are printed if the field table load fails or if an output file cannot be created.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Examples</b>    | <pre>FLDTBLDIR=/project/fldtbls FLD_TBL=maskftbl,DBftbl,miscftbl, export FLDTBLDIR FLD_TBL</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

`mkfldhdr` produces the `#include` files `maskftbl.h`, `DBftbl.h`, and `miscftbl.h` in the current directory by processing the files `maskftbl`, `DBftbl`, and `miscftbl` in directory `/project/fldtbls`.

With environment variables set as in the example above, the command `mkfldhdr -d$FLDTBLDIR` processes the same input field-table files, and produces the same output files, but places them in the directory given by the value of the environment variable `FLDTBLDIR`.

The command `mkfldhdr myfields` processes the input file `myfields` and produces `myfields.h` in the current directory.

**See Also** Introduction to FML Functions, `field_tables(5)`

# mklanginfo(1)

**Name** mklanginfo - compile language information constants for a locale

**Synopsis** mklanginfo [*fname*]

**Description** This program takes the file specified as an argument, and converts the input into a file suitable for placement in `$TUXDIR/locale/xx/LANGINFO` where *xx* is a specific locale. The standard input is used if a file argument is not specified. The language values are used by `setlocale(3c)`, `strftime(3c)`, and `nl_langinfo(3c)`.

mklanginfo reads input lines, ignoring lines that begin with white space or '#'. Value input lines must be of the form

```
<token> = "value"
```

The characters between the *token* and the double-quoted *value* can be anything but a double quote as long as white space appears after the token. If *value* is the null string, the line is ignored. Otherwise, *token* must be either an integer between 1 and 48, inclusive, or a string, as follows.

Integer String Value 1

|         |                                                 |    |
|---------|-------------------------------------------------|----|
| DAY_1   | Day 1 of the week, for example, Sunday          | 2  |
| DAY_2   | Day 2 of the week, for example, Monday          | 3  |
| DAY_3   | Day 3 of the week, for example, Tuesday         | 4  |
| DAY_4   | Day 4 of the week, for example, Wednesday       | 5  |
| DAY_5   | Day 5 of the week, for example, Thursday        | 6  |
| DAY_6   | Day 6 of the week, for example, Friday          | 7  |
| DAY_7   | Day 7 of the week, for example, Saturday        | 8  |
| ABDAY_1 | Abbreviated day 1 of the week, for example, Sun | 9  |
| ABDAY_2 | Abbreviated day 2 of the week, for example, Mon | 10 |
| ABDAY_3 | Abbreviated day 3 of the week, for example, Tue | 11 |
| ABDAY_4 | Abbreviated day 4 of the week, for example, Wed | 12 |
| ABDAY_5 | Abbreviated day 5 of the week, for example, Thu | 13 |
| ABDAY_6 | Abbreviated day 6 of the week, for example, Fri | 14 |
| ABDAY_7 | Abbreviated day 7 of the week, for example, Sat | 15 |
| MON_1   | Month 1 of the year, for example, January       | 16 |
| MON_2   | Month 2 of the year, for example, February      | 17 |
| MON_3   | Month 3 of the year, for example, March         | 18 |
| MON_4   | Month 4 of the year, for example, April         | 19 |
| MON_5   | Month 5 of the year, for example, May           | 20 |
| MON_6   | Month 6 of the year, for example, June          | 21 |
| MON_7   | Month 7 of the year, for example, July          | 22 |

## Section 1 - Commands

---

|           |                                                                    |    |
|-----------|--------------------------------------------------------------------|----|
| MON_8     | Month 8 of the year, for example, August                           | 23 |
| MON_9     | Month 9 of the year, for example, September                        | 24 |
| MON_10    | Month 10 of the year, for example, October                         | 25 |
| MON_11    | Month 11 of the year, for example, November                        | 26 |
| MON_12    | Month 12 of the year, for example, December                        | 27 |
| ABMON_1   | Abbreviated month 1 of the year, for example, Jan                  | 28 |
| ABMON_2   | Abbreviated month 2 of the year, for example, Feb                  | 29 |
| ABMON_3   | Abbreviated month 3 of the year, for example, Mar                  | 30 |
| ABMON_4   | Abbreviated month 4 of the year, for example, Apr                  | 31 |
| ABMON_5   | Abbreviated month 5 of the year, for example, May                  | 32 |
| ABMON_6   | Abbreviated month 6 of the year, for example, Jun                  | 33 |
| ABMON_7   | Abbreviated month 7 of the year, for example, Jul                  | 34 |
| ABMON_8   | Abbreviated month 8 of the year, for example, Aug                  | 35 |
| ABMON_9   | Abbreviated month 9 of the year, for example, Sep                  | 36 |
| ABMON_10  | Abbreviated month 10 of the year, for example, Oct                 | 37 |
| ABMON_11  | Abbreviated month 11 of the year, for example, Nov                 | 38 |
| ABMON_12  | Abbreviated month 12 of the year, for example, Dec                 | 39 |
| RADIXCHAR | Radix character, for example, '.'                                  | 40 |
| THOUSEP   | Separator for thousands                                            | 41 |
| YESSTR    | Affirmative response string, for example, yes                      | 42 |
| NOSTR     | Negative response string, for example, no                          | 43 |
| CRNCYSTR  | Currency symbol                                                    | 44 |
| D_T_FMT   | string for formatting date and time, for example, "%a%b%d%H:%M:OY" | 45 |
| D_FMT     | string for formatting date, for example, "%m/%d/%y"                | 46 |
| T_FMT     | string for formatting time, for example, "H:%M:%S"                 | 47 |
| AM_FMT    | Ante Meridian affix, for example, AM                               | 48 |
| PM_FMT    | Post Meridian affix, for example, PM                               |    |

The input lines may appear in any order. If an input line appears more than once for the same value, the last line for that value is used.

After processing the file, `mklanginfo` prints the string name and string value for each language information constant shown in the previous code listing to the standard error in the order specified in the listing. The null string is used as a value for any language information constant not specified; `nl_langinfo` uses the default value for the C locale (U.S. English values) for these unset constants.

If a filename is specified on the command name, `mklanginfo` writes the *compiled* output to `fname.out`; otherwise, the output is written to the standard output. The format is a list of all of the null-terminated string values (without newlines).

**Diagnostic** If an error occurs in reading the file or in the syntax, an error message is printed to the standard error and the program exits with exit code 1. On success, the program exits with exit code 0.

**Examples** The defaults for the BEA Tuxedo system (locale C) are located in `$TUXDIR/locale/C/lang.text`. To provide French values, an administrator might do the following (on a UNIX system platform):

```
mkdir $TUXDIR/locale/french
cd $TUXDIR/locale/french
cp $TUXDIR/locale/C/lang.text .
ed lang.text
convert to French values
w
q
mklanginfo lang.text > LANGINFO
```

**Files** `$TUXDIR/locale/C/lang.text` - the default values for the C locale  
`$TUXDIR/locale/C/LANGINFO` - the “compiled” file for the C locale  
`$TUXDIR/locale/xx/LANGINFO` - the “compiled” file for the `xx` locale

**Notices** The `mklanginfo` command and the resulting `LANGINFO` file are needed only if the BEA Tuxedo system compatibility functions for `setlocale()`, `strftime()`, or `nl_langinfo()` are used. The functions provided with the UNIX System use a different set and format of files.

**See Also** `nl_langinfo(3c)`, `setlocale(3c)`, `strftime(3c)`, `langinfo(5)`

# qmadmin(1)

Name `qmadmin` - queue manager administration program

Synopsis `[QMCONFIG=<device>] qmadmin [<device>]`

Description With the commands listed in this entry, `qmadmin` supports the creation, inspection, and modification of message queues. The universal device list (UDL) maps the physical storage space on a machine on which the BEA Tuxedo system is run. An entry in the UDL points to the disk space in which the queues and messages of a queue space are stored. The name of the device (file) on which the UDL resides (or will reside) for the queue space may be specified either as a command-line argument or via the environment variable `QMCONFIG`. If both are specified, the command option is used.

As a system-provided command, `qmadmin` does not undergo normal initialization, so it does not pick up the value of `ULOGPFX` from the `UBBCONFIG` file. As a result, any log entries generated by `qmadmin` commands are written to the current working directory. This is corrected by setting and exporting the `ULOGPFX` environment variable to the path name of the directory in which the userlog is located.

`qmadmin` uses the greater than sign (`>`) as a prompt. Arguments are entered separated by white space (tabs and/or spaces). Arguments that contain white space may be enclosed within double quotes; if an argument enclosed within double quotes contains a double quote, the internal double quote must be preceded with a backslash. Commands prompt for required information if it is not given on the command line. A warning message is displayed and the prompt shown again, if a required argument is not entered. Commands do not prompt for information on optional parameters.

A user can exit the program by entering `q` or `<CTRL-d>` when prompted for a command. Output from a command may be terminated by pressing `BREAK`; the program then prompts for the next command. Hitting return when prompted for a command repeats the previously executed command, except after a break.

Note that there is no way to effectively cancel a command once you press `RETURN`; hitting `BREAK` only terminates output from the command, if any. Therefore, be sure that you type a command exactly as you intended before pressing `RETURN`.

Output from `qmadmin` commands is paginated according to the pagination command in use (see the `paginate` subcommand below).

When `qmadmin` is initially entered, no queue space is opened. To create a queue space, run `qspacecreate`; to open it, run `qopen`. The `qaborttrans`, `qclose`, `qchangeprio`, `qchangequeue`, `qchangetime`, `qchangeexptime`, `qcommittrans`, `qchange`, `qcreate`, `qdeletemsg`, `qinfo`, `qlist`, `qprinttrans` and `qset` commands can be executed only when a queue space is open.

The following table lists the `qmadmin` commands grouped by functional type.

| Command Type | Command                     | Purpose                                             |
|--------------|-----------------------------|-----------------------------------------------------|
| General      |                             |                                                     |
|              | <code>echo</code>           | Echo input command lines                            |
|              | <code>help</code>           | Print help messages                                 |
|              | <code>paginate</code>       | Paginate output                                     |
|              | <code>quit</code>           | Terminate the session                               |
|              | <code>verbose</code>        | Produce output in verbose mode                      |
|              | <code>! shellcommand</code> | Escape to shell and execute <i>shellcommand</i>     |
|              | <code>!!</code>             | Repeat previous shell command                       |
|              | <code>#</code>              | Indicate comment lines                              |
|              | <code>&lt;CR&gt;</code>     | Repeat the last command                             |
| Queue Space  |                             |                                                     |
|              | <code>chdl</code>           | Change the name for a universal device list entry   |
|              | <code>crdl</code>           | Create an entry in the universal device list        |
|              | <code>dSDL</code>           | Destroy an entry found in the universal device list |
|              | <code>ipcrm</code>          | Remove IPC data structures used for the queue space |
|              | <code>ipcs</code>           | List IPC data structures used for the queue space   |
|              | <code>lidl</code>           | Print the universal device list                     |
|              | <code>livtoc</code>         | Prints information for all VTOC table entries       |

## Section 1 - Commands

---

| <b>Command Type</b> | <b>Command</b> | <b>Purpose</b>                                                 |
|---------------------|----------------|----------------------------------------------------------------|
|                     | qaddext        | Add an extent to the queue space                               |
|                     | qclose         | Close the currently open queue space                           |
|                     | qopen          | Open and initialize structures for the queue space             |
|                     | qsize          | Compute the size of shared memory needed for a queue space     |
|                     | qspacechange   | Change the parameters for a queue space                        |
|                     | qspacecreate   | Create a queue space for queued messages                       |
|                     | qspacedestroy  | Destroy the named queue space                                  |
|                     | qspacelist     | List the creation parameters for the queue space               |
| Queue               |                |                                                                |
|                     | qchange        | Modify a queue in the currently open queue space               |
|                     | qcreate        | Create a queue in the currently open queue space               |
|                     | qdestroy       | Destroy the named queue                                        |
|                     | qinfo          | List information for associated queue or for all queues        |
| Message             |                |                                                                |
|                     | qchangeexp     | Change the expiration time for messages on a queue             |
|                     | qchangeprio    | Change the priority for messages on a queue                    |
|                     | qchangequeue   | Move messages to a different queue within the same queue space |
|                     | qchangetime    | Change the execution time for messages on a queue              |
|                     | qdeletemsg     | Delete messages from a queue                                   |
|                     | qlist          | Lists messages on a queue                                      |
|                     | qscan          | Sets selection criteria used by other commands                 |
|                     | qset           | Sets the queue name used by other commands                     |

| Command Type | Command      | Purpose                                                          |
|--------------|--------------|------------------------------------------------------------------|
| Transaction  |              |                                                                  |
|              | qaborttrans  | Abort a precommitted transaction                                 |
|              | qcommittrans | Commit a precommitted transaction                                |
|              | qprinttrans  | Print transaction table information for outstanding transactions |

**qmadmin Commands** Commands may be entered either by their full name or their abbreviation (if available, the abbreviation is listed below in parentheses following the full name), followed by any appropriate arguments. Arguments appearing in square brackets ([]) are optional; those in curly braces ({} ) indicate a selection from mutually exclusive options.

`chdl [dlindex [newdevice]]`

Change the name for a universal device list entry. The first argument is the index of the device on the universal device list that is to be changed (device indexes are returned by `lidl`). The program prompts for it if it is not provided on the command line.

The second argument is the new device name. If a device name is not provided on the command line, the program prints the current device name and then prompts for a new one. The name is limited to 64 characters in length. Use this command cautiously; files and data are not accessible via the old name after the device name is changed.

For more information about printing the Universal Device List (UDL) and Volume Table of Contents (VTOC), see *Administering a BEA Tuxedo Application at Run Time*.

`crdl [device [offset [size]]]`

Create an entry in the universal device list. Note: The first entry in the device list must correspond to the device that is referenced by `QMCONFIG` and must have an offset of 0. If arguments are not provided on the command line, the program prompts for them.

The arguments are the device name, the block number at which space may begin to be allocated, and the number of physical pages (disk sectors) to be allocated.

More than one extent can be allocated to a given file. You can, for example, allocate `/app/queues/myspace 0 500`, and then allocate `/app/queues/myspace 1000 500`, for a total of 1000 blocks allocated with blocks 500 through 999 not being used.

Several blocks from the first device entry are used by the device list and table of contents. Up to 25 entries may be created on the device list.

`dSDL [-y] [dindex]`

Destroy an entry found in the universal device list. The *dindex* argument is the index on the universal device list of the device that is to be removed from the device list. If it is not provided on the command line, the program prompts for it. Entry 0 cannot be removed until all VTOC files and other device list entries are destroyed. (Because entry 0 contains the device that holds the device list and table of contents, destroying it also destroys these two tables.) VTOC files can be removed only by removing the associated entities (for example, by destroying a queue space that resides on the device). The program prompts for confirmation unless `-y` is specified.

`echo (e) [{off | on}]`

Echo input command lines when set to `on`. If no option is given, then the current setting is toggled, and the new setting is printed. The initial setting is `off`.

`help (h) [{command | all}]`

Print help messages. If a *command* is specified, the abbreviation, arguments, and description for that command are printed. The `all` argument causes a description of all commands to be displayed.

If no arguments are specified on the command line, the syntax of all commands is displayed.

`ipcrm [-f] [-y] [queue_space_name]`

`ipcrm` removes the IPC data structures used for the specified queue space. If a queue space name is not provided on the command line, the program prompts for one. If the specified queue space is open in `qadmin`, it will be closed. `ipcrm` knows all IPC resources used by the queue space and is the only way that the IPC resources should be removed. `qadmin` ensures that no other processes are attached to the queue space before removing the IPC resources. The `-f` option can be specified to force removal of IPC resources even if other processes are attached. This command prompts for confirmation before execution if the `-f` option is specified, unless the `-y` option is

specified. All non-persistent messages in the specified queue space are permanently lost when this command completes successfully.

`ipcs [queue_space_name]`

`ipcs` lists the IPC data structures used for a queue space, if any (none may be used if the queue space is not opened by any process). If a queue space name is not provided on the command line, the program prompts for one.

`lidl [dlindex]`

Print the universal device list. For each device the following is listed: the index, the name, the starting block, and the number of blocks on the device. In verbose mode, a map is printed that shows free space (starting address and size of free space). If `dlindex` is specified, then only the information for that device list entry is printed.

`livtoc`

Prints information for all VTOC table entries. The information printed for each entry includes the name of the VTOC table, the device on which it is found, the offset of the VTOC table from the beginning of the device and the number of pages allocated for that table. There are a maximum of 100 entries in the VTOC.

`paginate (page) [{off | on}]`

Paginate output. If no option is given, then the current setting is toggled, and the new setting is printed. The initial setting is on, unless either standard input or standard output is a non-terminal device. Pagination may be turned on only when both standard input and standard output are terminal devices.

The default paging command is the pager indigenous to the native operating system environment. The command `pg`, for example, is the default command on the UNIX operating system. The shell environment variable `PAGER` may be used to override the default command used for paging output.

`qaborttrans (qabort) [-y] [tranindex]`

Heuristically abort the precommitted transaction associated with the specified transaction index, `tranindex`. If the transaction index is not specified on the command line, the program prompts for it. If the transaction is known to be decided and the decision was to commit, `qaborttrans` fails. The index is taken from the previous execution of the `qprinttrans` command. Confirmation is requested unless the `-y` option is specified. This command should be used with care.

`qaddext [queue_space_name [pages]]`

Add an extent to the queue space. The queue space must not be active (no processes can be attached to the queue space). If a queue space name and the number of additional physical pages to allocate for the queue space are not specified on the command line, the program prompts for them. If the specified queue space is open in `qadmin`, it will be closed. The number of physical pages is rounded down to the nearest *multiple of four* pages (see `qspacecreate` for clarification and examples). Space is allocated from extents defined in the UDL associated with the `QMCONFIG` device. Each new queue space extent uses an additional entry in the `VTOC` (a maximum of 100 entries are available). The queue manager names the extents such that they can be identified quickly and associated with the queue space. All non-persistent messages in the specified queue space are permanently lost when this command completes successfully.

`qchange [queue_name [out-of-order [retries [delay [high [low [cmd]]]]]]]  
[-d persist|nonpersist] [-n nhigh,nlow,ncmd]  
[-e default_relative_expiration_time]`

Modify a queue in the currently open queue space. The required arguments may be given on the command line or the program will prompt for them. These are the queue name, whether out-of-order enqueueing is allowed (not allowed, top of queue, or before a specified `msgid`); the number of retries and delay time in seconds between retries; and the high and low limits for execution of a threshold command and the threshold command itself for persistent messaging.

The out-of-order values are `none`, `top`, and `msgid`. Both `top` and `msgid` may be specified, separated by a comma.

The threshold values are used to allow for automatic execution of a command when a threshold is reached for persistent messages. The high limit specifies when the command is executed. The low limit must be reached before the command is executed again when the high limit is reached. For example, if the limits are 100 and 50 messages, the command is executed when 100 messages are on the queue, and it is not executed again until the queue is drained down to 50 messages and is filled again to 100 messages. The queue capacity can be specified in bytes or blocks used by the queue (number followed by a `b` or `B` suffix), percentage of the queue space used by the queue (number followed by a `%`), or total number of messages on the queue (number followed by an `m`). The threshold type for the high and low threshold values must be the same. It is optional whether or not the type is specified on the low value, but if specified, it must match the high value type. The message (`m`)

suffix spans both persistent and non-persistent messages. The other threshold suffixes apply only to persistent messages. Use the `-n` option to specify threshold values for non-persistent messages. When specified on the command line, the threshold command should be enclosed in double quotation marks if it contains white space. The retry count indicates how many times a message can be dequeued and the transaction rolled back, causing the message to be put back on the queue. A delay between retries can also be specified. When the retry count is reached, the message is moved to the error queue defined for the queue space. If no error queue has been defined, the message is dropped. The queue ordering values for the queue cannot be changed. Low-priority messages are dequeued after every ten messages, even if the queue still contains high-priority messages.

The `-d` option specifies the default delivery policy for the queue. The valid values for the `-d` option are `persist` and `nonpersist`. When the default delivery policy is `persist`, enqueued messages with no explicitly specified delivery mode are delivered using the persistent (disk-based) delivery method. When the policy is `nonpersist`, enqueued messages with no explicitly specified delivery mode are delivered using the non-persistent (in memory) delivery method. If the `-d` option is not specified, the system does not prompt for information and the default delivery policy is unchanged. When the default delivery policy is modified, the delivery quality of service is not changed for messages already in the queue. 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.

If a non-persistent message cannot be enqueued due to an exhausted or fragmented memory area, the enqueueing operation fails, even if there is sufficient persistent storage for the message. If a persistent message cannot be enqueued due to an exhausted or fragmented disk, the enqueueing operation fails, even if there is sufficient non-persistent storage for the message.

If the amount of memory reserved for non-persistent messages in a queue space is zero (0), no space is reserved for non-persistent messages. (See `qspacecreate` and `qspacechange` for information on specifying the non-persistent message memory area.) In this case, attempts to enqueue a non-persistent message fail. This includes messages with no specified delivery quality of service for which the target queue has a default delivery policy of `nonpersist`.

The `-n` option specifies the threshold values used for automatic execution of a command when a non-persistent storage area threshold is reached. The `nhigh` limit specifies when the command `ncmd` is executed. The `nlow` limit must be reached before the command will be executed again when the `nhigh` limit is reached. If the `-n` option is specified, the `nhigh`, `nlow`, and `ncmd` values must all be supplied, or the command fails. The `ncmd` value may be specified as an empty string. If the `-n` option is not specified, the program does not prompt for information.

The memory capacity (amount of non-persistent data in the queue) can be specified as one of the following threshold types: bytes (`b`), blocks (`B`), or percentage (`number` followed by `%`). The threshold type for the `nhigh` and `nlow` values must be the same. For example, if `nhigh` is set to 100%, then `nlow`, if specified, must also be specified as a percentage. The threshold type of the `nlow` value is optional. If the `-n` option is not specified, the default threshold values for non-persistent messaging are unchanged. If `ncmd` contains white space, it must be enclosed in double quotation marks.

The `m` suffix of the `[... [high[low[cmd]] ...]` thresholds applies to all messages in a queue, including both persistent and non-persistent messages, and therefore is not available with `nhigh` and `nlow`. The `[... [high[low[cmd]] ...]` thresholds specified without the `-m` suffix apply to persistent (disk-based) messages only.

The `-e default_relative_expiration_time` option sets an expiration time for all messages enqueued to the queue that do not have an explicitly specified expiration time. The expiration time may be either a relative expiration time or `none`. When the expiration time is reached and the message has not been dequeued or administratively deleted, the message is removed from the queue, all resources associated with the message are reclaimed by the system, and statistics are updated. If the expiration time is before the message availability time, the message is not available for dequeuing unless either the availability or expiration time is changed so that the availability time is before the expiration time. In addition, these messages are removed from the queue at expiration time even if they were never available for dequeuing. 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 when a message has expired.

If the `-e` option is not specified, the default expiration time of the queue is not changed. When the queue's expiration time is modified using `qchange`, the

expiration times for messages already in the queues are not modified. If the `-e` option is not specified, the program does not prompt for it.

The format of a relative `default_relative_expiration_time` is `+seconds` where `seconds` is the number of seconds from the time that the queue manager successfully completes the operation to the time that the message expires. A value of zero (0) indicates immediate expiration. The value of `default_relative_expiration_time` may also be set to the string `none`. The `none` string indicates that messages that are enqueued with no explicit expiration time will not expire unless an expiration time is explicitly assigned to them.

`qchangeexp (qce) -y [newtime]`

Change the expiration time for messages on a queue. When a message expires, it is removed from the queue, all resources used by the message are reclaimed by the system, and the relevant statistics are updated. If the expiration time is before the message availability time, the message is not available for dequeuing unless either the availability or expiration time is changed so that the availability time is before the expiration time. In addition, these messages are removed from the queue at expiration time even if they were never available for dequeuing. 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 when a message has expired.

The queue for which an expiration time is set is selected using the `qset` command. Selection criteria for limiting the messages to be updated are set with the `qscan` command. If no selection criterion is set, then all messages on the queue are changed. By default, a confirmation is requested before the expiration time is set. The `-y` option specifies no prompt for confirmation. The `newtime` value can be relative to either the current time, an absolute value, or `none`. If the `newtime` value is not provided on the command line, the program prompts for it.

Messages enqueued by versions of the BEA Tuxedo system that do not support message expiration cannot be modified to have an expiration time even when the queue manager responsible for changing the value supports message expiration. If messages affected by `qchangeexp` have been enqueued by one of these versions of the BEA Tuxedo system, an error message indicates that some of the selected messages were not modified due to this limitation.

A relative expiration time is relative to when the request arrives at the queue manager process. The format of a relative *newtime* is *+seconds* where *seconds* is the number of seconds from the time that the queue manager successfully completes the operation to the time that the message expires. If *seconds* is set to zero (0), messages expire immediately. An absolute expiration time is determined by the clock on the machine where the queue manager process resides. The format of an absolute *newtime* is *YY[MM[DD[HH[MM[SS]]]]]* as described in `qscan`. The value of *newtime* may also be set to the string *none*, which indicates that affected messages never expire.

`qchangeprio (qcp) [-y] [newpriority]`

This command can be used to change the priority for messages on a queue. The queue that is affected is set using the `qset` command and the selection criteria for limiting the messages to be updated are set using the `qscan` command.

If no selection criteria are set, then all messages on the queue are changed: confirmation is requested before this is done unless the `-y` option is specified. It is recommended that the `qlist` command be executed to see what messages will be modified (this reduces typographical errors). The *newpriority* value specifies the new priority which will be used when the message(s) are forwarded for processing. It must be in the range 1 to 100, inclusive. If not provided on the command line, the program will prompt for it.

`qchangequeue (qcq) [-y] [newqueue]`

This command can be used to move messages to a different queue within the same queue space. The queue from which messages are moved is set using the `qset` command and the selection criteria for limiting the messages to be moved are set using the `qscan` command. If no selection criteria are set, then all messages on the queue are moved: confirmation is requested before this is done unless the `-y` option is specified. It is recommended that the `qlist` command be executed to see what messages will be moved (this reduces typographical errors). The *newqueue* value specifies the name of the queue to which messages will be moved. If *newqueue* is not specified on the command line, the program prompts for it. The delivery quality of service of a message is not changed to match the default delivery policy of *newqueue*.

When messages with an expiration time are moved, the expiration time is considered an absolute expiration time in the new queue, even if it was previously specified as a relative expiration time.

qchangetime (qct) [-y] [*newtime*]

This command can be used to change the message availability time for messages on a queue. The queue is specified using the `qset` command. The selection criteria for limiting the messages to be updated are set using the `qscan` command.

If no selection criteria are set, then all messages on the queue are changed: confirmation is requested before this is done unless the `-y` option is specified. It is recommended that the `qlist` command be executed to see what messages will be modified (this reduces typographical errors). The *newtime* value can be either relative to the current time or an absolute value. If not provided on the command line, the program will prompt for it. The format of a relative *onetime* is `+seconds` where *seconds* is the number of seconds from now that the message is to be executed (0 implies immediately). The format of an absolute *newtime* is `YY[MM[DD[HH[MM[SS]]]]]`, as described in `qscan`.

qclose

Close the currently open queue space. All non-persistent messages in the specified queue space are permanently lost when this command completes successfully.

qcommittrans (qcommit) [-y] [*tranindex*]

Heuristically commit the precommitted transaction associated with the specified transaction index *tranindex*. The program will prompt for the transaction index if not specified on the command line. If the transaction is known to be decided and the decision was to abort, `qcommittrans` will fail. The index is taken from the previous execution of the `qprinttrans` command. Confirmation is requested unless the `-y` option is specified. This command should be used with care.

qcreate (qcr) [*queue\_name* [*qorder* [*out-of-order* [*retries* [*delay* [*high* [*low* [*cmd*]]]]]]] [-d *persist|nonpersist*] [-n *nhigh,nlow,ncmd*] [-e *default\_relative\_expiration\_time*]

Create a queue in the currently open queue space. The required arguments may be given on the command line or the program will prompt for them. These are the queue name, the queue ordering (fifo or lifo, by expiration time, by priority, by time); whether out-of-order enqueueing is allowed (not allowed, top of queue, before a specified `msgid`); the number of retries and delay time in seconds between retries; the high and low limits for execution of a threshold command; and the threshold command itself for persistent messages.

The queue ordering values are `fifo`, `lifo`, `priority`, `expiration`, and `time`. When specifying the queue ordering, the most significant sort value must be specified first, followed by the next most significant sort value, and so on; `fifo` or `lifo` can be specified only as the least significant (or only) sort value. If neither `fifo` or `lifo` is specified, then the default is `fifo` within whatever other sort criteria are specified. If `expiration` is specified, messages with no expiration time are dequeued after all messages with an expiration time. Multiple sort values may be specified separated by commas. The out-of-order values are `none`, `top`, or `msgid`. Both `top` and `msgid` may be specified, separated by a comma.

The threshold values are used to allow for automatic execution of a command when a threshold is reached for persistent messages. The high limit specifies when the command is executed. The low limit must be reached before the command will be executed again when the high limit is reached. For example, if the limits are 100 and 50 messages, the command will be executed when 100 messages are on the queue, and will not be executed again until the queue has been drained below 50 messages and has filled again to 100 messages.

The queue capacity can be specified in bytes or blocks used by the queue (number followed by a `b` or `B` suffix), percentage of the queue space used by the queue (number followed by a `%`), or total number of messages on the queue (number followed by an `m`). The threshold type for the high and low threshold values must be the same. The message (`m`) suffix spans both persistent and non-persistent messages. The other threshold suffixes apply only to persistent messages. Use the `-n` option to specify threshold values for non-persistent messages. It is optional whether or not the type is specified on the low value, but if specified, it must match the high value type. When specified on the command line, the threshold command should be enclosed in double quotation marks if it contains white space.

The retry count indicates how many times a message can be dequeued and the transaction rolled back, causing the message to be put back on the queue. A delay between retries can also be specified. When the retry count is reached, the message is moved to the error queue defined for the queue space. If an error queue has not been defined, the message is dropped. Low-priority messages are dequeued after every ten messages, even if the queue still contains high-priority messages.

The `-d` option specifies the default delivery policy for the queue. The valid values for the `-d` option are `persist` and `nonpersist`. When the default delivery policy is `persist`, enqueued messages with no explicitly specified

delivery mode are delivered using the persistent (disk-based) delivery method. When the policy is `nonpersist`, enqueued messages with no explicitly specified delivery mode are delivered using the non-persistent (in memory) delivery method. If the `-d` option is not specified, the system does not prompt for information and the default delivery policy for the queue is `persist`. When the default delivery policy is modified, the delivery quality of service is not changed for messages already in the queue.

If a non-persistent message cannot be enqueued due to an exhausted or fragmented memory area, the enqueueing operation fails, even if there is sufficient persistent storage for the message. If a persistent message cannot be enqueued due to an exhausted or fragmented disk, the enqueueing operation fails, even if there is sufficient non-persistent storage for the message.

If the amount of memory reserved for non-persistent messages in a queue space is zero (0), no space is reserved for non-persistent messages. (See `qspacecreate` and `qspacechange` for information on specifying the non-persistent message memory area.) In this case, attempts to enqueue a non-persistent message fail. This includes messages with no specified delivery quality of service for which the target queue has a default delivery policy of `nonpersist`.

The `-n` option specifies the threshold values used for automatic execution of a command when a non-persistent storage area threshold is reached. The *high* limit specifies when the command `ncmd` is executed. The *low* limit must be reached before the command will be executed again when the *high* limit is reached. If the `-n` option is specified, the *high*, *low*, and `ncmd` values must all be supplied, or the command fails. The `ncmd` value may be specified as an empty string. If the `-n` option is not specified, the program does not prompt for information.

The memory capacity (amount of non-persistent data in the queue) can be specified as one of the following threshold types: bytes (b), blocks (B), or percentage (*number* followed by %). The threshold type for the *high* and *low* values must be the same. For example, if *high* is set to 100%, then *low*, if specified, must also be specified as a percentage. The threshold type of the *low* value is optional. If the `-n` option is not specified, the default threshold values are used (100% for *high* and 0% for *low*) and `ncmd` is set to " ". If `ncmd` contains white space, it must be enclosed in double quotation marks.

The `m` suffix of the `[... [high[low[cmd]] ...]` thresholds applies to all messages in a queue, including both persistent and non-persistent messages, and therefore is not available with `nhigh` and `nlow`. The `[... [high[low[cmd]] ...]` thresholds specified without the `-m` suffix apply to persistent (disk-based) messages only.

The `-e default_relative_expiration_time` option sets an expiration time for all messages enqueued to the queue that do not have an explicitly specified expiration time. The expiration time may be either a relative expiration time or `none`. When the expiration time is reached and the message has not been dequeued or administratively deleted, the message is removed from the queue, all resources associated with the message are reclaimed by the system, and statistics are updated. If the expiration time is before the message availability time, the message is not available for dequeuing unless either the availability or expiration time is changed so that the availability time is before the expiration time. In addition, these messages are removed from the queue at expiration time even if they were never available for dequeuing. 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 when a message has expired.

If the `-e` option is not specified, the default expiration time of the queue is set to `none`. When the queue's expiration time is modified using `qchange`, the expiration times for messages already in the queues are not modified. If the `-e` option is not specified, the program does not prompt for it.

The format of a relative `default_relative_expiration_time` is `+seconds` where `seconds` is the number of seconds from the time that the queue manager successfully completes the operation to the time that the message expires. A value of zero (0) indicates immediate expiration. The value of `default_relative_expiration_time` may also be set to the string `none`. The `none` string indicates that messages that are enqueued with no explicit expiration time will not expire unless an expiration time is explicitly assigned to them.

`qdeletemsg (qdltm) [-y]`

This command can be used to delete messages from a queue. The queue is specified using the `qset` command. The selection criteria for limiting the messages to be deleted are set using the `qscan` command. If no selection criteria are set, then all messages on the queue are deleted; confirmation is requested before this is done. It is recommended that the `qlist` command be

executed to see what messages will be deleted (this reduces typographical errors). This command prompts for confirmation unless the `-y` option is specified.

`qdestroy (qds) [{ -p | -f }] [-y] [queue_name]`

Destroy the named queue. By default, an error is returned if requests exist on the queue or a process is attached to the queue space. The `-p` option can be specified to “purge” any messages from the queue and destroy it, if no processes are attached to the queue space. The `-f` option can be specified to “force” deletion of a queue, even if messages or processes are attached to the queue space; if a message is currently involved in a transaction the command fails and an error is written to the `userlog`. This command prompts for confirmation before proceeding unless the `-y` option is specified.

`qinfo [queue_name]`

List information for associated queue or for all queues. This command lists the following: the number of messages on the specified queue (or all queues if no argument is given); the amount of space used by the messages associated with the queue (both persistent and non-persistent); the number of messages being delivered persistently and non-persistently; the total number of messages in the specified queues, and the amount of space used by the persistent and non-persistent messages. In verbose mode, this command also lists the queue creation parameters for each queue, the default expiration for the queue (if any), the sort criteria, and the default delivery policy for the queue.

`qlist (ql)`

This command lists messages on a queue. The queue is specified using the `qset` command. The selection criteria for limiting the messages to be listed are set using the `qscan` command. If no selection criteria are set, then all messages on the queue will be listed.

For each message selected, the message identifier is printed along with the message priority, the number of retries already attempted, message length, delivery quality of service, the quality of service for any replies, and the expiration time (if any). The message availability time is printed if one is associated with the message, or for messages that have a scheduled retry time (due to rollback of a transaction). The correlation identifier is printed if present and `verbose` mode is on.

`qopen [queue_space_name]`

Open and initialize the internal structures for the specified queue space. If a queue space is not specified on the command line, the program prompts for it. If a queue space is already open in `qadmin`, it is closed.

`qprinttrans (qpt)`

Print transaction table information for currently outstanding transactions. The information includes the transaction identifier, an index used for aborting or committing transactions with `qaborttrans` or `qcommittrans`, and the transaction status.

`qscan [{ [-t time1[-time2]] [-p priority1[-priority2]] [-m msgid] [-i corrid] [-d delivery_mode] [-e time1[-time2]] | none ]}`

This command sets the selection criteria used for the `qchangeprio`, `qchangequeue`, `qchangetime`, `qdeletemsg`, and `qlist` commands. An argument of `none` indicates no selection criteria; all messages on the queue will be affected. Executing this command with no argument prints the current selection criteria values. When command-line options give a value range (for example, `-t`, `-e`, or `-p`) the value range may not contain white space. The `-t` option can be used to indicate a time value or a time range. The format of `time1` and `time2` is: `YY[MM[DD[HH[MM[SS]]]]]` specifying the year, month, day, hour, minute, and second. Units omitted from the date-time value default to their minimum possible values. For example, “7502” is equivalent to “750201000000.” The years 00-37 are treated as 2000-2037, years 70-99 are treated as 1970-1999, and 38-69 are invalid. The `-p` option can be used to indicate a priority value or a priority range. Priority values are in the range 1 to 100, inclusive. The `-m` option can be used to indicate a message identifier value, assigned to a message by the system when it is enqueued. The message identifier is unique within a queue and its value may be up to 32 characters in length. Values that are shorter than 32 characters are padded on the right with nulls (0x0). Backslash and non-printable characters (including white space characters such as space, newline, and tab) must be entered with a backslash followed by a two-character hexadecimal value for the character (for example, space is `\20`, as in “hello\20world”). The `-i` option can be used to indicate an correlation identifier value associated with a message. The identifier value is assigned by the application, stored with the enqueued message, and passed on to be stored with any reply or error message response such that the application can identify responses to particular requests. The value may be up to 32 characters in length. Values that are shorter than 32 characters are padded on the right with nulls (0x0). Backslash and non-printable characters (including white space characters such as space,

newline, and tab) must be entered with a backslash followed by a two-character hexadecimal value for the character (for example, space is \20, as in my\20ID\20value).

The valid values for the `-d delivery_mode` option are `persist` and `nonpersist`. This option specifies the delivery mode of messages selected by `qscan` so that an operator can take action based on the delivery method.

The `-e` option can be used to indicate an expiration time or an expiration time range. The format of `time1` and `time2` is the same as `time1` and `time2` for the `-t` option.

`qset [queue_name]`

This command sets the queue name that is used for the `qchangeprio`, `qchangequeue`, `qchangetime`, `qdeletemsg`, and `qlist` commands. Executing this command with no argument prints the current queue name.

`qsize [pages [queues [transactions [processes [messages]]]]]  
[-A actions] [-H handles] [-C cursors] [-O owners] [-Q tmp_queues]  
[-f filter_memory] [-n nonpersistent_msg_memory[b,B]]  
[-o overflow_memory]`

Compute the size of shared memory needed for a queue space with the specified size in `pages`, `queues`, (concurrent) `transactions`, `processes`, and (queued) `messages`. If the values are not provided on the command line, the program prompts for them. The number of system semaphores needed is also printed. Valid values for the remaining options are described in the `qspacecreate` option.

`qspacechange (qspch) [queue_space_name [ipckey [trans [procs  
[messages [errorq [inityn [blocking]]]]]]] [-A actions] [-H handles]  
[-C cursors] [-O owners] [-Q tmp_queues] [-f filter_memory]  
[-n nonpersistent_msg_memory[b,B]] [-o overflow_memory]`

Change the parameters for a queue space. The queue space must not be active (that is, no processes can be attached to it). If the required information is not provided on the command line, the program prompts for it. Valid values are described in the `qspacecreate` section of this page. If the specified queue space is open in `qmadmin`, it is closed. To add new extents, `qaddext` must be used. The number of queues cannot be modified.

`qspacecreate (qspc) [queue_space_name [ipckey [pages [queues [trans  
[procs [messages [errorq [inityn [blocking]]]]]]]]] [-A actions]`

`[-H handles] [-C cursors] [-O owners] [-Q tmp_queues] [-f filter_memory]  
[-n nonpersistent_msg_memory[b,B]] [-o overflow_memory]`

Create a queue space for queued messages. If not provided on the command line, the program prompts for the following information: the queue space name, the `ipckey` for the shared memory segment and semaphore; number of physical pages to allocate for the queue space; the number of queues; the number of concurrent transactions; the number of processes concurrently attached to the queue space; the number of messages that may be queued at one time; the name of an error queue for the queue space; whether or not to initialize pages on new extents for the queue space; and the blocking factor for doing queue space initialization and warm start disk input/output.

The number of physical pages requested is rounded down to the nearest *multiple of four* pages. For example, a request of 50 pages results in a memory allocation of 48 pages, and a request of 52 pages results in a memory allocation of 52 pages. The error queue is used to hold messages that have reached the maximum number of retries (they are moved from their original queue to the error queue). The administrator is responsible for ensuring that this queue is drained.

The number of physical pages allocated must be large enough to hold the overhead for the queue space (one page plus one page per queue). If the initialization option is specified as 'y' or 'Y,' then the space used to hold the queue space is initialized and this command may run for a while. In verbose mode, a period (.) is printed to the standard output after completing initialization of each 5% of the queue space. If the initialization option is not turned on but the underlying device is not a character special device, then the file will be initialized if it not already the size specified for the extent (that is, the file will be grown to allocate the specified space).

When reading and writing blocks during creation of the queue space and during warm start (restart of the queue space), the size of input and output operations will be calculated as a multiple of the disk page size as specified by the blocking factor.

The `-A actions` option specifies the number of additional actions that the Queuing Services component 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 completes, the action associated with the operation is made available for a subsequent operation. An operation fails if a blocking

operation is requested and cannot be immediately satisfied and there are no actions available. The system reserves actions equivalent to the number of processes that can attach to a queue space so that each queue manager process may have at least one blocking action. Beyond the system-reserved number of blocking actions, the administrator may configure the system to be able to accommodate additional blocking actions beyond the reserve. If the `-A actions` option is not specified, the default is zero. If the `-A` option is not specified, the program does not prompt for it.

The `-H handles` option specifies the number of handles that users of that the Queuing Services component may 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 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 API is used by an application, the administrator must configure the system to accommodate the maximum number of handles that will be opened concurrently. If the `-H handles` option is not specified, the default is zero. If the `-H` option is not specified, the program does not prompt for it. An operation fails if a user attempts to open a queuing services object and there are no handles available. This option is not currently used by the BEA Tuxedo system. Adjusting this value has no effect on BEA Tuxedo applications other than unnecessarily consuming shared memory resources.

The `-C cursors` option specifies the number of cursors that users of that the Queuing Services component may 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 will be allocated concurrently. If the `-C cursors` option is not specified, the default is zero. If the `-C` option is not specified, the program does not prompt for it. An operation fails if a user attempts to create a cursor and there are no cursor resources available. BEA Tuxedo applications need not adjust this value. This option is not currently used by the BEA Tuxedo system. Adjusting this value has no effect on BEA Tuxedo applications other than unnecessarily consuming shared memory resources.

The `-O owners` option specifies the number of additional BEA Engine authenticated users that may concurrently use the 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 for subsequent users. The system reserves owners equivalent to the number of actions so that each action may be initiated by a different owner. Beyond the system-reserved number of owners that may be concurrently using queuing services resources, the administrator may configure the system to accommodate additional owners beyond the reserve. If the `-O owners` option is not specified, the default is zero. If the `-O` option is not specified, the program does not prompt for it. An operation fails if a user attempts to open a handle when they currently do not have any open handles, and there are no owners available. This option is not currently used by the BEA Tuxedo system. Adjusting this value has no effect on BEA Tuxedo applications other than unnecessarily consuming shared memory resources.

The `-Q tmp_queues` option specifies the number of temporary queues that may be opened concurrently in the Queuing Services component of the BEA Engine. Temporary queues reduce the need for administrators to configure each queue used by an application. They are used by dynamic self-configuring applications. 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 will be active concurrently. If the `-Q tmp_queues` option is not specified, the default is zero. If the `-Q` option is not specified, the program does not prompt for it. An open operation fails if a user attempts to open a temporary queue and there are no temporary queue resources available. This option is not currently used by the BEA Tuxedo system. Adjusting this value has no effect on BEA Tuxedo applications other than unnecessarily consuming shared memory resources.

The `-f filter_memory` option 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 may be specified using various grammars but are compiled into an engine normal form and stored in shared memory. Filters are referenced by a handle 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. If the `-f`

*filter\_memory* option is not specified, the default is zero. If the *-f* option is not specified, the program does not prompt for it. 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. This option is not currently used by the BEA Tuxedo system. Adjusting this value has no effect on BEA Tuxedo applications other than unnecessarily consuming shared memory resources.

The *-n nonpersistent\_msg\_memory* option specifies the size of the area to reserve in shared memory for non-persistent messages for all queues in the queue space. The size may be specified in bytes (b) or blocks (B), where the block size is equivalent to the disk block size. The [bB] suffix is optional and, if not specified, the default is blocks. If the *-n* option is not specified, the memory size defaults to zero (0). Also, if the *-n* option is not specified, the program does not prompt for it.

If the value is specified in bytes (b) for *nonpersistent\_msg\_memory*, the system divides the specified value by the number of bytes per page (page size is equivalent to the disk page size), rounds down the result to the nearest integer, and allocates that number of pages of memory. For example, assuming a page size of 1024 bytes (1KB), a requested value of 2000b results in a memory allocation of 1 page (1024 bytes), and a requested value of 2048b results in a memory allocation of 2 pages (2048 bytes). Requesting a value less than the number of bytes per page results in an allocation of 0 pages (0 bytes).

If the value is specified in blocks (B) for *nonpersistent\_msg\_memory* and assuming that one block of memory is equivalent to one page of memory, the system allocates the same value of pages. For example, a requested value of 50B results in a memory allocation of 50 pages.

If the *nonpersistent\_msg\_memory* for a queue space is zero (0), no space is reserved for non-persistent messages. In this case, attempts to enqueue a non-persistent message fail. Persistent and non-persistent storage are not interchangeable. If a non-persistent message cannot be enqueued due to an exhausted or fragmented memory area, the enqueueing operation fails, even if there is sufficient persistent storage for the message. If a persistent message cannot be enqueued due to an exhausted or fragmented disk, the enqueueing operation fails, even if there is sufficient non-persistent storage for the message.

The *-o overflow\_memory* option specifies the size of the memory area to reserve in shared memory to accommodate peek load situations where some

or all of the allocated shared memory resources are exhausted. The memory size is specified in bytes. Additional objects will be 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. If the `-o overflow_memory` option is not specified, the default is zero. If the `-o` option is not specified, the program does not prompt for it. This additional memory space may 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 memory.

`qspacedestroy (qspds) [-f] [-y] [queue_space_name]`

Destroy the named queue space. If not provided on the command line, the program will prompt for it. If the specified queue space is open in `qmadmin`, it will be closed. By default, an error is returned if processes are attached to the queue space or if requests exist on any queues in the queue space. See the `qdestroy` command for destroying queues that contain requests. The `-f` option can be specified to “force” deletion of all queues, even if they may have messages or processes are attached to the queue space. This command prompts for confirmation before proceeding unless the `-y` option is specified. All non-persistent messages in the specified queue space are lost when this command completes successfully.

`(qsp1) [queue_space_name]`

List the creation parameters for the queue space. If not specified on the command line, the program will prompt for it. If a queue space name is not entered, then the parameters for the currently open queue space are printed. (An error occurs if a queue space is not open and a value is not entered.) In addition to printing the values for the queue space (as set when creating the queue space with `qspacecreate` or when they were last changed with `qspacechange`), this command shows the sizes for all queue space extents. It also shows the amount of system-reserved memory as well as the total amount of configured shared memory. The amount of memory allocated for shared memory resources may not match the amount requested *when the amount of memory is requested in bytes* (b); see the `-n nonpersistent_msg_memory` option in `qspacecreate` for clarification and examples.

`quit (q)`

Terminate the session.

```

verbose (v) [{off | on}]
 Produce output in verbose mode. If no option is given, then the current setting
 will be toggled, and the new setting is printed. The initial setting is off.

! shellcommand
 Escape to shell and execute shellcommand.

!!
 Repeat previous shell command.

[text]
 Lines beginning with # are comment lines and are ignored.

<CR>
 Repeat the last command.

```

**Example** The following sequence shows how to set up a queue.

```

$ QMCONFIG=/dev/rawfs qmadmin
qmadmin - Copyright (c) 1987 ATT; 1991 USL. All rights reserved.
QMCONFIG=/dev/rawfs
create the list of devices on which the queue space
can exist; specify two devices, 80000 and 600
blocks, respectively
NOTE: the first one will actually contain the device list
#
create first device on a raw slice
#
> crdl /dev/rawfs 0 80000
Created device /dev/rawfs, offset 0, size 80000 on /dev/rawfs
#
create another device on a UNIX file
#
> crdl /home/queues/FS 0 600
Created device /home/queues/FS, offset 0, size 600 on /dev/rawfs
#
if you want a list of the device list
#
> v Verbose mode is now on

> lidl
universal device index. 0:
 name: /dev/rawfs
 start: 0
 size: 20000
 free space map(1 entry used 47 available):
 size[1]: 79974 addr[1]: 26
universal device index. 1:

```

## Section 1 - Commands

---

```
name: /home/queues/FS
start: 0
size: 600
free space map(1 entry used 47 available):
 size[1]: 600 addr[1]: 0
#
create a queue space
#
> qspacecreate
Queue space name: myqueuespace
IPC Key for queue space: 42000
Size of queue space in disk pages: 50000
Number of queues in queue space: 30
Number of concurrent transactions in queue space: 20
Number of concurrent processes in queue space: 30
Number of messages in queue space: 20000
Error queue name: ERRORQ
Initialize extents (y, n [default=n]): y
Blocking factor [default=16]: 16
.....
#
open queue space
#
> qopen myqueuespace
#
use queue space defaults for queue
> qcreate
Queue name: servicel
queue order (priority, time, fifo, lifo): fifo
out-of-ordering enqueueing (top, msgid, [default=none]): top,msgid
retries [default=0]: 1
retry delay in seconds [default=0]: 30
High limit for queue capacity warning (b for bytes used, B for blocks used,
% for percent used, m for messages [default=100%]): 100m
Reset (low) limit for queue capacity warning [default=0m]: 50
queue capacity command: /usr/app/bin/mailadmin myqueuespace servicel
#
get out of the program
#
> q
```

**Security** The administrator for the queue must be the same as the BEA Tuxedo administrator. The device on which the queue resides must be owned by the administrator and `qmadm` can only be run as the administrator for the queue. All IPC resources allocated for the queue will be owned by the queue administrator and will be created with mode 0600.

**Portability** qmadmin is supported on any platform on which the BEA Tuxedo server environment is supported.

**Windows NT Standard I/O** In order to carry out a command that you have configured within a qmadmin() session, such as the qchange ... Queue capacity command, the NT CreateProcess() function spawns a child process as a DETACHED PROCESS. This type of process does *not* have an associated console for standard input/output. Therefore, for instance, if you use standard DOS syntax to set the qchange ... Queue capacity command to run a built-in DOS command (such as dir or date) and then pipe or redirect the standard output to a file, the file will be empty when the command completes.

As an example of resolving this problem, suppose that for the qchange ... Queue capacity command you want to capture date information in a file using command date /t > x.out. To accomplish this task interactively, you would proceed as follows:

```
qmadmin
> qopen yourQspace
> qchange yourQname
> go through all the setups... the threshold queue capacity warning,
 and so on
> "Queue capacity command: " cmd /c date /t > x.out
```

To accomplish this task from a command file, say *yourFile.cmd*, you would add the command date /t > x.out to *yourFile.cmd* and then proceed as follows:

```
qmadmin
> qopen yourQspace
> qchange yourQname
> go through all the setups... the threshold queue capacity warning,
 and so on
> "Queue capacity command: " yourFile.cmd
```

**See Also** *Administering a BEA Tuxedo Application at Run Time*

# rex(1)

Name `rex` - off-line regular expression compiler and tester

Synopsis Compiling:

```
rex pattern_file C_file
```

Testing:

```
rex pattern [file . . .]
```

Description When invoked without arguments, `rex` reads regular expressions from the standard input and writes initialized character arrays to the standard output. Normally, the output is included in a C program to preclude the need for calling `recomp()`. This saves on both execution time and program size. The command `rex` compiles the regular expressions on the standard input (normally redirected from an input file) and writes the output to the standard output (normally redirected to an output file).

The input file may contain several patterns, each of the following form.

```
name string [string...]
```

Here *name* is the C name to be used for the output array and *string* is the regular expression enclosed with double quotes. Where more than one *string* follows a *name* they are concatenated into one *string*. (Multiple *strings* are strictly a formatting convenience.) If double quotes occur in the pattern they need to be preceded by a backslash.

The output may be included in a C program or compiled and later loaded. In the C program that uses the `rex` output, `rematch(abc, line, 0)` applies the regular expression named `abc` to `line`.

The following is a sample input file.

```
cname "[a-zA-Z_][a-(3c)-Z0-9_]*"
tn "\\((([0-9]{3})$0\\)"
 "([0-9]{3})$1"
 "_"
 "([0-9]{4})$2"
```

Here is the corresponding output.

```

/* pattern: "[a-zA-Z_][a-zA-Z0-9_]*" */
char cname[] = {
040,0,0206,012,0,0210,0141,0172,0210,0101,0132,0137,
... };

/* pattern: "\\((([0-9]{3})$0\\)(([0-9]{3})$1-([0-9]{4})$2" */
char tn[] = {
063,0,050,0202,0225,013,0,03,0206,06,0,0210,060,071,
... };

```

rex can be used to try patterns against test data by invoking it with one or more arguments. The first argument is taken as a pattern (regular expression) to be applied to each line of the files whose names are mentioned in the remaining arguments. If no filename arguments are given the standard input is used. The special filename, -, may be used as an argument to refer to the standard input.

When matching text is found, the line containing the match is printed and the matching portion of the line is underlined. In addition, any text extracted for specified sub-patterns is printed on separate line(s).

For example, the command

```
rex '(^|)([0-9]+)$0(|$)'
```

with input

```

... or 200 programmers in one week.
This sentence has 3 errors.
I need 12 bad men.

```

produces

```

... or 200 programmers in one week.

$0 = `200'

This sentence has 3 errors.

$0 = `3'

I need 12 bad men.

$0 = `12'

```

**Diagnostics** `rex` prints the associated error messages for errors returned from `recomp()` or `rematch()` (see `recomp`, `rematch(3c)`). Other errors include file open errors, argument errors, and so on, and are self-explanatory.

**See Also** `recomp`, `rematch(3c)`

---

# tidl(1)

**Name**     tidl - Interface Definition Language compiler

**Synopsis**   tidl [*option*] . . . *filename* [*option*]. . .

**Description**  tidl parses the input IDL and related ACF source file, and optionally generates a header file, and client and server stubs and auxiliary files. The generated source code can be compiled using compilers for Classic C, ANSI C, or C++.

The command-line arguments include the input IDL source file and options to control the actions of the IDL compiler. The following list describes the options.

`-client type`

This option specifies the client-side files to be generated. The values for *type* are as follows.

`all`

Generates client stub and auxiliary files. This is the default if the `-client` option is not specified.

`stub`

Generates client stub file only.

`aux`

Generates client auxiliary file only. Currently, auxiliary files are not generated so this option has no effect.

`none`

Generates no client files.

`-server type`

This option specifies the server-side files to be generated. The values for *type* are as follows.

`all`

Generates server stub and auxiliary files. This is the default if the `-server` option is not specified.

`stub`

Generates server stub file only.

- `aux`  
Generates server auxiliary file only. Currently, auxiliary files are not generated so this option has no effect.
- `none`  
Generates no server files.
- `-cstub filename`  
Specifies the filename for the client stub file. If the filename does not have a `.c` extension, the IDL compiler will add it. The default client stub name (if `-cstub` is not specified) is the base name of the IDL source file (the simple filename without any directory prefix, or any suffix following a period) with `_cstub.c` appended. The associated client stub object file is the name of the client stub file with `.c` changed to `.o`.
- `-sstub filename`  
Specifies the filename for the server stub file. If the filename does not have a `.c` extension, the IDL compiler will add it. The default server stub name (if `-sstub` is not specified) is the base name of the IDL source file (the simple filename without any directory prefix, or any suffix following a period) with `_sstub.c` appended. The associated server stub object file is the name of the server stub file with `.c` changed to `.o`.
- `-caux filename`  
Specifies the filename for the client auxiliary file. If the filename does not have a `.c` extension, the IDL compiler will add it. The default client auxiliary name (if `-caux` is not specified) is the base name of the IDL source file (the simple filename without any directory prefix, or any suffix following a period) with `_caux.c` appended. The associated client auxiliary object file is the name of the client auxiliary file with `.c` changed to `.o`.
- `-saux filename`  
Specifies the filename for the server auxiliary file. If the filename does not have a `.c` extension, the IDL compiler will add it. The default server auxiliary name (if `-saux` is not specified) is the base name of the IDL source file (the simple filename without any directory prefix, or any suffix following a period) with `_saux.c` appended. The associated server auxiliary object file is the name of the server auxiliary file with `.c` changed to `.o`.
- `-header filename`  
Specifies the filename for the generated header file. The default header filename (if `-header` is not specified) is the base name of the IDL source file

---

(the simple filename without any directory prefix, or any suffix following a period) with `.h` appended.

`-out directory`

Specifies the directory in which output files are created. The default (if `-out` is not specified) is to put the files in the present working directory.

`-keep type`

Specifies which file types to retain. By default, the IDL compiler creates a C source file (for example, a client stub), uses the C compiler to produce an object file, and deletes the C source file. The file types that can be retained are as follows:

`none`

Does not create any files or invoke the C compiler.

`c_source`

Saves only the C source files and does not invoke the C compiler.

`object`

Saves only the object files, deleting the C source files (this is the default).

`all`

Saves both the C source and object files.

`-I directory`

Specifies a directory in which to search for imported IDL files and include files. White space following the `-I` is optional. The `-I` option can be specified multiple times to list multiple directories. The default behavior is to search the present working directory, then the directories specified with the `-I` option in the order specified, and then the system IDL directory (`$TUXDIR/include`). This order is also used to pass include directories to the C preprocessor and C compiler. If a file exists in more than one directory, the first one that is found in the search order is used.

`-no_def_idir`

When used with no `-I` options, specifies that only the present working directory be searched for import and include files. When used with one or more `-I` options, specifies that only the `-I` directories be searched (not the present working directory or the system IDL directory).

`-cpp_cmd "cmd"`

Specifies the C preprocessor command to invoke for expanding source files. By default, the C preprocessor is not invoked on DOS and OS2, and defaults to `/lib/cpp`, `/usr/ccs/lib/cpp`, or `/usr/lib/cpp` (in that order) otherwise.

- `-no_cpp`  
Specifies that the C preprocessor not be invoked to expand source files. This implies that the source files cannot have preprocessor directives (such as macro replacements and `#include`).
- `-cpp_opt "opt"`  
Specifies additional options to be passed to the C preprocessor. The default is the null string. The IDL compiler invokes a command line composed of the values for `-cpp_cmd`, `-cpp_opt`, `-D`, and `-U` arguments (in the order specified), `-I` arguments (in the order specified), and the source filename (the IDL or ACF filename).
- `-D name[=def]`  
Defines a name and optionally a value that is passed to the C preprocessor. More than one symbol can be defined by specifying the `-D` option more than once. White space following the `-D` is optional.
- `-U name`  
Undefines a name for C preprocessor. More than one symbol can be undefined by specifying the `-U` option more than once. White space following the `-U` is optional.
- `-cc_cmd "cmd"`  
Specifies the C compiler command for creating object files. The default (if `-cc_cmd` is not specified) is `"cc -c"`.
- `-cc_opt "opt"`  
Specifies additional C compiler options. The default (if `-cc_opt` is not specified) is the null string. The IDL compiler invokes a command line composed of the values for `-cc_cmd`, `-cc_opt`, `-I` arguments (in the order specified), and the source filename (the stub or auxiliary filename).
- `-syntax_check`  
Specifies that the input source file be checked for syntax errors without generating any output files.
- `-no_warn`  
Specifies that warning messages from the compiler are not to be printed.
- `-confirm`  
Displays IDL compiler options chosen (either explicitly or implicitly) without compilation of the source file. When used with the `-v` option, it indicates what actions would be taken without the `-confirm` option but it does not executing those actions. For example, messages are printed for parsing input files, and for creating and compiling output files.

- 
- `-v` Specifies verbose mode. Messages are printed to the standard error output indicating actions being taken (for example, parsing input files, and creating and compiling output files).
- `-version` Displays the version of the IDL compiler.
- `-stdin` Take the IDL source input from standard input instead of a file. Default filenames are generated as if the input IDL source file is named `a.idl` (for example, the default client stub file is named `a_cstub.c`).
- `-cepv` Generate a Client Entry Point Vector (CEPV). By default, functions in the client stub module are given the same names as operations in the interface definition. However, this naming convention does not allow for multiple versions of the interface, interfaces with the same operation names, or both local and remote versions of the same functions to be linked into the same client program (the operation names will be defined more than once).
- When the `-cepv` option is specified, the function names are declared local to the client stub and a Client Entry Point Vector (array of function pointers) is defined (globally) in the client stub with the name `interface_vmajor_minorc_epv`, where `interface` is the interface name, `major` is the major version number, and `minor` is the minor version number. The interface operations must be called indirectly using the addresses in the CEPV.
- `-no_mepv` Does not generate a Manager (server) Entry Point Vector (MEPV). By default, it is assumed that the application functions in the server are given the same names as operations in the interface definition. However, this naming convention does not allow for multiple versions of the interface, interfaces with the same operation names, or both local and remote versions of the same functions to be linked into the same server program (the operation names are defined more than once). Normally, a Manager Entry Point Vector (array of function pointers) is defined (globally) in the server stub with the name `interface_vmajor_minors_epv`, where `interface` is the interface name, `major` is the major version number, and `minor` is the minor version number, and initialized with the operation names. It is used to call the application service functions. When the `-no_mepv` option is specified, the MEPV is not generated in the server stub and the application is responsible for setting up the structure. In this way, the application can set the entry point

names to those names used by the application instead of names based on the operations.

`-error all`

Specifies additional error checking. By default, the IDL compiler quits after 50 errors are detected.

`-port level`

Specifies the level of portability checking. The following levels are supported:

`case`

Specifies that warnings messages are to be printed if two identifiers differ only in case.

`none`

Specifies no additional portability checking is to be done. This is the default.

`-no_enum_lit`

Specifies that enumeration literals are not to be generated in the stub files. By default, enumeration literals are generated.

`-use_const`

Specifies that ANSI C `const` declarations are to be used for constant values instead of `#define` definitions.

For the IDL source file and any imported IDL files, the compiler searches for a related ACF with a `.acf` suffix added to the basename of the IDL source file. The directories that are searched are the same directories specified for the C preprocessor (see `-I` and `-no_def_idir` above), plus ACF files are searched for in the directory specified in the IDL source filename.

**Notices** The IDL filename `tbase.idl` is reserved for use by the IDL compiler.

**Examples** Here is an example IDL source file, `math1.idl`.

```
[uuid(2048A080-0B0F-14F8-26E0-930269220000)]
interface math1
{
import "math2.idl";

long add_op([in] long first1, [in] long second);
long sub_op([in] long first1, [in] long second);
}
```

Here is a sample ACF source file, math1.acf.

```
[auto_handle]
interface math1
{
 include "stdio";
 [code] add_op([fault_status,comm_status] result);
}
```

The following command line compiles math1.idl, generating client-side only files out/math1\_cs.c and out/math1\_cs.o (no auxiliary files are needed), along with out/math1.h. The IDL compiler looks for math2.idl (which might have the division and multiplication operations) in the current directory, in the app subdirectory, and in \$TUXDIR/include.

```
tidl math1.idl -Iapp -client all -server none -keep all
-cstub math1_cs -out app
```

See Also `uuidgen(1)`

# tlisten(1)

Name `tlisten` - generic listener process

Synopsis `tlisten [-d device] -l nlsaddr [-u {uid-# | uid-name}] [-z bits] [-Z bits]`

Description `tlisten` is a network independent listener process that runs as a *daemon* process on BEA Tuxedo application processors and provides remote service connections for other BEA Tuxedo processes, for example, `tmboot(1)`. The following command-line options are used by `tlisten`.

`-d device`

Full path name of the network device. This parameter is optional. For releases prior to v6.4, it should be used if the underlying network provider requires it.

`-l nlsaddr`

Network address at which the process listens for connections. TCP/IP addresses may be specified in either of the following forms.

`"//hostname:port_number"`

`"//#. #. #. #:port_number"`

In the first format, `tlisten` finds an address for *hostname* using the local name resolution facilities (usually DNS). *hostname* must be the local machine, and the local name resolution facilities must unambiguously resolve *hostname* to the address of the local machine.

In the second example, the string `#. #. #. #` is in dotted decimal format. In dotted decimal format, each `#` should be a number from 0 to 255. This dotted decimal number represents the IP address of the local machine. In both of the above formats, *port\_number* is the TCP port number at which the `tlisten` process listens for incoming requests. The value of *port\_number* can be either a name or a number between 0 and 65535.

**Note:** Some port numbers may be reserved for the underlying transport protocols (such as TCP/IP) used by your system. Check the documentation for your transport protocols to find out which numbers, if any, are reserved on your system.

If *port\_number* is a name, then it must be found in the network services database on your local machine. The address can also be specified in

hexadecimal format when preceded by the characters “0x”. Each character after the initial “0x” is a number between 0 and 9 or a letter between A and F (case insensitive). The hexadecimal format is useful for arbitrary binary network addresses such as IPX/SPX or TCP/IP. The address can also be specified as an arbitrary string. The value should be the same as that specified for the NLSADDR parameter in the NETWORK section of the configuration file.

`-u {uid-# | uid-name}`

tlisten will run as the indicated user. This option supports the startup of tlisten as part of system initialization by root. This option is required if the user running tlisten is root. The tlisten process can therefore be started by root, but will not run as root. Non-root users of the tlisten command do not need to use the -u option. Non-root users can set the -u option, but it can only be set to their own user ID and is effectively a no-op. Each instantiation of a tlisten process on a processor is capable of supporting all BEA Tuxedo applications that use the same application administrator user ID.

`-z [0 | 40 | 56 | 128]`

Specifies the minimum level of encryption required when a network link is being established between a BEA Tuxedo administrative process and tlisten. 0 means no encryption, while 40, 56, and 128 specify the length (in bits) of the encryption key. If this minimum level of encryption cannot be met, link establishment fails. The default value is 0. This option is available only if a license for BEA Tuxedo Security (either International, or US and Canada) is installed.

**Note:** The link-level encryption value of 40 bits is provided for backward compatibility.

`-z [0 | 40 | 56 | 128]`

Specifies the maximum level of encryption allowed when a network link is being established between a BEA Tuxedo administrative process and tlisten. 0 means no encryption, while 40, 56, and 128 specify the length (in bits) of the encryption key. The default value is 128. This option is available only if a license for BEA Tuxedo Security (either International, or US and Canada) is installed.

**Note:** The link-level encryption value of 40 bits is provided for backward compatibility.

The `tlisten` process authenticates most service requests. `tlisten` reads a file with a list of passwords, and any process requesting a service must present at least one of the passwords found in the file. If the `APPDIR` environment variable is set, passwords will be obtained from a file named `APPDIR/.adm/tlisten.pw`. If this file is not found, the system will look for `TUXDIR/udataobj/tlisten.pw`, which is created when the BEA Tuxedo system is installed. A zero-length or missing password file disables password checking. When running in this insecure mode, the `tlisten` and any process connecting to `tlisten` will generate a userlog warning message.

Processes that request services from `tlisten`, such as `tmboot`, find the passwords to be used during authentication in files on their own machines. They use the same methods as the `tlisten` to find their password files.

### Environment Variables

- `TUXDIR` must be set and exported before the `tlisten` command is invoked.
- `LD_LIBRARY_PATH` must be set for SVR4 applications that use shared objects. It must be set to `TUXDIR/lib` prior to starting the `tlisten` process.
- `APPDIR` is set to provide the location of the `tlisten` password file.
- `ULOGPFX` can be used to direct the file in which log messages are placed.

**Note:** During the installation process, an administrative password file is created. When necessary, the BEA Tuxedo system searches for this file in the following directories (in the order shown):

- `APPDIR/.adm/tlisten.pw`
- `TUXDIR/udataobj/tlisten.pw`

To ensure that your administrative password file will be found, make sure you have set the `APPDIR` and/or the `TUXDIR` environment variables.

### Link-level Encryption

If the link-level encryption feature is in operation between `tlisten` and a requesting process such as `tmboot`, then link-level encryption will be negotiated and activated before authentication occurs.

### Termination

The only way to stop a `tlisten` process with normal termination is by sending it a `SIGTERM` signal.

### Recommended Use

We recommend that you start one `tlisten` process for each application upon system startup. Remember to set the `TUXDIR` and `APPDIR` environment variables before invoking `tlisten`.

One alternative method for starting the `tlisten` process is to start it manually. The `-u` option can be omitted if the `tlisten` process is started by the application administrator. Duplicate `tlisten` command invocations using the same network address will terminate automatically and gracefully log an appropriate message.

#### Network Addresses

Suppose the local machine on which the `tlisten` is being run is using TCP/IP addressing and is named `backus.company.com`, with address `155.2.193.18`. Further suppose that the port number at which the `tlisten` should accept requests is `2334`. Assume that port number `2334` has been added to the network services database under the name `bankapp-nlsaddr`. The address specified by the `-l` option can be represented in the following ways.

```
//155.2.193.18:bankapp-nlsaddr
//155.2.193.18:2334
//backus.company.com:bankapp-nlsaddr
//backus.company.com:2334
0x0002091E9B02C112
```

The last of these representations is hexadecimal format. The `0002` is the first part of a TCP/IP address. The `091E` is the port number `2334` translated into a hexadecimal number. After that each element of the IP address `155.2.193.12` is translated into a hexadecimal number. Thus the `155` becomes `9B`, `2` becomes `02`, and so on.

For a STARLAN network, a recommended address of `uname.tlisten` usually yields a unique name.

#### Windows NT Control Panel Applet

Administrative privileges on a remote NT machine are required in order to start a `tlisten` process on that machine through the Control Panel Applet.

#### See Also

UBBCONFIG(5)

# tmadmin(1)

Name `tmadmin` - BEA Tuxedo bulletin board command interpreter

Synopsis `tmadmin [ -r ] [ -c ] [ -v ]`

Description With the commands listed in this entry, `tmadmin` provides for the inspection and modification of bulletin boards and associated entities in a uniprocessor, multiprocessor, or networked environment. The `TUXCONFIG` and `TUXOFFSET` environment variables are used to determine the location and offset at which the BEA Tuxedo configuration file is loaded.

If `tmadmin` is invoked with the `-c` option, it enters configuration mode. The only valid commands are `default`, `echo`, `help`, `quit`, `verbose`, `livtoc`, `crdl`, `lidl`, `dsdl`, `indl`, and `dumptlog`. `tmadmin` may be invoked in this mode on any node, including inactive nodes. A node is considered active if `tmadmin` can join the application as an administrative process or client (via a running BBL).

The `-r` option instructs `tmadmin` to enter the bulletin board as a client instead of as the administrator; in other words, it requests read-only access. This option is useful if you want to leave the administrator slot unoccupied.

**Note:** If you decide to use this option, however, be aware that you will not get all the information you get by running `tmadmin` without the `-r` option. Specifically, `tmadmin -r` does not report load values for servers running at remote sites.

Only one `tmadmin` process can be the administrator at a time. When the `-r` option is specified by a user other than the BEA Tuxedo administrator and security is turned on, the user is prompted for a password.

The `-v` option causes `tmadmin` to display the BEA Tuxedo version number and license number. After printing out the information, `tmadmin` exits. If the `-v` option is entered with either of the other two options, the others are ignored; only the information requested by the `-v` option is displayed.

Normally, `tmadmin` may be run on any active node within an active application. If it is run on an active node that is partitioned, then commands are limited to read-only access to the local bulletin board. These command include `bbls`, `bbparms`, `bbstat`, `default`, `dump`, `dumptlog`, `echo`, `help`, `printclient`, `printnet`, `printqueue`, `printserver`, `printservice`, `printrans`, `printgroup`, `reconnect`, `quit`, `serverparms`, `serviceparms`, and `verbose`, in addition to the configuration

commands. If the partitioned node is the backup node for the MASTER (specified as the second entry on the MASTER parameter in the RESOURCES section of the configuration file), the `master` command is also available to make this node the MASTER for this part of the partitioned application.

If the application is inactive, `tmadmin` can be run only on the MASTER processor. In this mode, all of the configuration mode commands are available plus the TLOG commands (`crlog`, `dslog`, and `inlog`) and `boot`.

Once `tmadmin` has been invoked, commands may be entered at the prompt (`>`) according to the following syntax.

```
command [arguments]
```

Several commonly occurring arguments can be given defaults via the `default` command. Commands that accept parameters set via the `default` command check `default` to see whether a value has been set. If a value has not been set, an error message is returned.

In a networked or multiprocessor environment, a single bulletin board can be accessed by setting a `default machine` (the logical `machine ID` (LMID) as listed in the MACHINES section of the UBBCONFIG file). If the `default machine` is set to `all`, all bulletin boards are accessed. If `machine` is set to `DBBL`, the distinguished bulletin board is addressed. The `default machine` is shown as part of the prompt, as in: `MASTER>`

If `machine` is not set via the `default` command, the `DBBL` is addressed (the local `BBL` is used in a SHM configuration).

The `machine` value for a command can generally be obtained from the `default` setting (`printserver` is an example). A caution is required here, however, because some commands (the TLOG commands, for example) act on devices found through TUXCONFIG; a `default` setting of `DBBL` or `all` results in an error. For some commands, such as `logstart`, you must specify the value of `machine` on the command line; the value does not appear as an argument to the `-m` option.

Once set, a default remains in effect until the session is ended, unless changed by another `default` command. Defaults may be overridden by entering an explicit value on the command line, or unset by entering the value `“*”`. The effect of an override lasts for a single instance of the command.

Output from `tmadmin` commands is paginated according to the pagination command in use (see the description of the `paginate` subcommand later in this entry).

There are some commands that have either verbose or terse output. The `verbose` command can be used to set the default output level. However, each command (except `boot`, `shutdown`, and `config`) takes a `-v` or `-t` option to turn on verbose or terse output for that command only. When output is printed in terse mode, some information (for example, `LMID` or `GROUP` name, service name, or server name) may be truncated. This type of truncation is indicated by a plus sign, `+`, at the end of the value. The entire value may be seen by re-entering the command in verbose mode.

### tmadmin Commands

Commands may be entered either by their full name or their abbreviation (as given in parentheses), followed by any appropriate arguments. Arguments appearing in square brackets, `[]`, are optional; those in curly braces, `{ }`, indicate a selection from mutually exclusive options. Note that command-line options that do not appear in square brackets need not appear on the command line (that is, they are optional) if the corresponding default has been set via the `default` command. Ellipses following a group of options in curly brackets, `{ } . . .`, indicate that more than one of the options listed may appear on the command line (at least one must appear).

`aborttrans (abort) [ -yes ] [ -g groupname ] tranindex`

If *groupname* is specified (on the command line or by default), abort the transaction associated with the specified transaction index *tranindex* at the specified server group. Otherwise, notify the coordinator of the transaction to abort the global transaction. If the transaction is known to be decided and the decision was to commit, `aborttrans` will fail. The index is taken from the previous execution of the `printtrans` command. To completely get rid of a transaction, `printtrans` and `aborttrans` must be executed for all groups that are participants in the transaction. This command should be used with care.

`advertise (adv) { -q qaddress [ -g groupname ]  
[ -i srvid ] | -g groupname -i srvid } service[:func]`

Create an entry in the service table for the indicated service. *service* may be mapped onto a function *func*. If *qaddress* is not specified, then both *groupname* and *srvid* are required to uniquely identify a server. If this *service* is to be added to an MSSQ set, all servers in the set will advertise the service. If all servers in an MSSQ set cannot advertise the service, the advertisement is disallowed. Services beginning with the character `'.` are reserved for use by system servers and will fail to be advertised for application servers.

`bbclean (bbc) machine`

Check the integrity of all accessors of the bulletin board residing on machine *machine*, and the `DBBL` as well. `bbclean` gracefully removes dead servers

and restarts them if they are marked as restartable. It also removes those resources no longer associated with any processes. As its last step, `bbclean` causes the DBBL to check the status of each BBL. If any BBL does not respond within `SCANUNIT` seconds, it is marked as partitioned. To clean only the Distinguished Bulletin Board, `machine` should be specified as DBBL. In SHM mode, `bbclean` restarts the BBL, if it has failed; the `machine` parameter is optional.

`bbparms` (`bbp`)

Print a summary of the bulletin board's parameters, such as the maximum number of servers and services.

`bbsread` (`bbls`) *machine*

List the IPC resources for the bulletin board on machine *machine*. In SHM mode, the *machine* parameter is optional. Information from remote machines is not available.

`bbstats` (`bbs`)

Print a summary of bulletin board statistics. (See also `shmstats`.)

`boot` (`b`) [*options*]

This command is identical to the `tmboot`( ) command. See `tmboot`(1) for an explanation of options and restrictions on use.

`broadcast` (`bcst`) [`-m machine`] [`-u username`] [`-c cltname`] [*text*]

Broadcasts an unsolicited notification message to all selected clients. The message sent is a typed buffer of the type `STRING` with the data being *text*. *text* may be no more than 80 characters in length. If *text* is to contain multiple words, then it must be enclosed in quotation marks ("*text text*"). If any parameter is not set (and does not have a default), then it is taken to be the wildcard value for that identifier.

`changeload` (`chl`) [`-m machine`] { `-q qaddress` [`-g groupname`] [`-i srvid`] `-s service` | `-g groupname` `-i srvid` `-s service` } *newload*

Change the load associated with the specified service to *newload*. If *qaddress* is not specified, then both *groupname* and *srvid* must be specified. If *machine* is set to `all` or is not set, the change is made on all machines; otherwise, a local change is made on the specified *machine*. Local changes are over-ridden by any subsequent global (or local) changes.

`changepriority (chp) [-m machine] {-q qaddress [-g groupname] [-i srvid] -s service | -g groupname -i srvid -s service } newpri`  
Change the dequeuing priority associated with the specified service to *newpri*. If *qaddress* is not specified, then both *groupname* and *srvid* must be specified. If *machine* is set to *all* or is not set, the change is made on all machines; otherwise, a local change is made on the specified *machine*. Local changes are over-riden by any subsequent global (or local) changes.

`changetrace (chtr) [-m machine] [-g groupname] [-i srvid] newspec`  
Change the run-time tracing behavior of currently executing processes to *newspec*. (See `tmtrace(5)` for the syntax of *newspec*.) To change the trace specification of a specific currently running server process, supply the `-g` and `-i` options. To change the configuration of currently-running server processes in a specific group, supply the `-g` option without the `-i` option. To change the configuration of all currently-running client and server processes on a particular machine, specify the `-m` option. If none of the `-g`, `-i`, and `-m` options is supplied, then all non-administrative processes on the default machine are affected. This command does not affect the behavior of clients or servers that are not currently executing, nor Workstation clients.

`changetrtime (chtt) [-m machine] {-q qaddress [-g groupname] - [-i srvid] -s service | -g groupname -i srvid -s service } newtlim`  
Change the transaction timeout value associated with the specified service to *newtlim*. If *qaddress* is not specified, then both *groupname* and *srvid* must be specified. If *machine* is set to *all* or is not set, the change is made on all machines; otherwise, a local change is made on the specified *machine*. Local changes are over-riden by any subsequent global (or local) changes.

`committrans (commit) [ -yes ] -g groupname tranindex`  
Commit the transaction associated with the specified transaction index *tranindex* at the specified server group. `committrans` will fail if the transaction has not been pre-committed at the specified server group or if the transaction is known to be abort-only. The index is taken from the previous execution of the `printtrans` command. This command prompts for confirmation before proceeding unless the `-yes` option is used. This command should be used with care.

`config (conf)`  
This command is identical to the `tmconfig` command. See `tmconfig`, `wtmconfig(1)` for an explanation of its use.

```
crdl -b blocks -z config -o configoffset [-O newdefoffset] [newdevice]
```

Create an entry in the universal device list. *blocks* specifies the number of physical blocks to be allocated on the device. The default *blocks* value is initialized to 1000 blocks. *configoffset* specifies the block number at which space may begin to be allocated. If the *-o* option is not given and a default has not been set, the value of the environment variable `FSOFFSET` is used. If `FSOFFSET` is not set, the default is 0. *config* points to the first device (which contains the device list); it must be an absolute path name (starting with `/`). If the *-z* option is not given and a default has not been set, the path named by the `FSCONFIG` environment variable is used. The *newdevice* argument to the `crdl` command, if specified, points to the device being created; it must be an absolute path name (starting with `/`). If this parameter is not given, the *newdevice* defaults to the config device. *newdefoffset* specifies an offset to the beginning of *newdevice*. If not specified with the *-O* (capital O) option of default, the default is 0 (zero).

```
crlog (crlg) -m machine
```

Create the DTP transaction log for the named or default *machine* (it cannot be “DBBL” or “all”). An error is returned if a `TLOG` is not defined for the machine on the configuration. This command references the `TUXCONFIG` file to determine the BEA Tuxedo file system containing the `TLOG`, the name of the `TLOG` in that file system, the offset, and the size (see `UBBCONFIG(5)`).

```
default (d) [-g groupname] [-i srvid] [-m machine] [-u username] [-c cltname]
[-q qaddress] [-s service] [-b blocks] [-o offset] [-z config] [-a { 0 | 1 | 2 }]
```

Set the corresponding argument to be the default group name, server ID, machine, user name, client name, queue address, service name, device blocks, device offset, or UDL configuration device path (it must be an absolute path name starting with `/`). See `printservice` for information on the *-a* option. All defaults may be unset by specifying `*` as an argument. If *machine* has been set to a machine identifier, and later retrievals are to be done from the Distinguished Bulletin Board, *machine* should be set to `DBBL`. Unsetting the *machine* (*-m \**) is equivalent to setting it to `DBBL`. If the `default` command is entered with no arguments, the current defaults are printed.

```
dsdl [-yes] -z config [-o offset] dlindex
```

Destroy an entry found in the universal device list. The *dlindex* argument is the index on the universal device list of the device that is to be removed from the device list. Entry 0 cannot be removed until all `VTOC` files and other device list entries are destroyed first (because entry 0 contains the device which holds the device list and table of contents, destroying it also destroys these two tables.) *config* points to the device containing the universal device list;

it must be an absolute path name (starting with */*). If the *-z* option is not given and a default has not been set, the path named by the `FSCONFIG` environment variable is used. *offset* specifies an offset into *config*. If the *-o* option is not given and a default has not been set, the value of the environment variable `FSOFFSET` is used. If `FSOFFSET` is not set, the default is 0. This command prompts for confirmation before proceeding unless the *-yes* option is used.

`dslog (dslg) [ -yes ] -m machine`

Destroy the DTP transaction log for the named or default *machine* (it cannot be “DBBL” or “all”). An error is returned if a `TLOG` is not defined for the machine, if the application is not inactive, or if outstanding transaction records exist on the log. The term outstanding transactions means that a global transaction has been committed but an end-of-transaction has not yet been written. This command references the `TUXCONFIG` file to determine the BEA Tuxedo file system containing the `TLOG` and name of the `TLOG` in that file system. This command prompts for confirmation before proceeding unless the *-yes* option is specified.

`dump (du) filename`

Dump the current bulletin board into the file *filename*.

`dumptlog (dl) -z config [ -o offset ] [ -n name ] [ -g groupname ] filename`

Dumps a text version of the `TLOG` into the specified *filename*. The `TLOG` is located on the specified *config* and *offset*, and has the specified *name*. If the *-n* option is not given and a default has not been set, the name “`TLOG`” is used. *config* points to the device containing the universal device list; it must be an absolute path name (starting with */*). If the *-z* option is not given and a default has not been set, the path named by the `FSCONFIG` environment variable is used. The *-o *offset** option can be used to specify an offset into *config*. If the *-o* option is not given and a default has not been set, the value of the environment variable `FSOFFSET` is used. If `FSOFFSET` is not set, the default is 0. If *groupname* is specified, then only log records for transactions where that group is the coordinator are dumped.

`echo (e) [{off | on}]`

Echo input command lines when set to *on*. If no option is given, then the current setting is toggled, and the new setting is printed. The initial setting is *off*.

`help (h) [{command | all}]`

Print help messages. If *command* is specified, the abbreviation, arguments, and description for that command are printed. *all* causes a description of all

commands to be displayed. Omitting all arguments causes the syntax of all commands to be displayed.

`initdl (indl) [ -yes ] -z config [ -o offset ] dlindex`

Reinitializes a device on the device list. The argument *dlindex* is the index of the device on the universal device list of the device that is to be reinitialized. All space on the specified device is freed; this means that any files, etc., stored on the device may be overwritten in the future so this command must be used cautiously. This command prompts for confirmation before proceeding unless the `-yes` option is used. *config* points to the device containing the universal device list; it must be an absolute path name (starting with `/`). If the `-z` option is not given and a default has not been set, the path named by the `FSCONFIG` environment variable is used. The `-o offset` option can be used to specify an offset into *config*. If the `-o` option is not given and a default has not been set, the value of the environment variable `FSOFFSET` is used. If `FSOFFSET` is not set, the default is 0.

`inlog [ -yes ] -m machine`

Reinitialize the DTP transaction log for the named or default *machine* (it cannot be “DBBL” or “all”). An error is returned if a `TLOG` is not defined for the machine or if the application is not inactive. If outstanding transactions exist on the `TLOG`, data may be inconsistent across resource managers acting as participants in these transactions since the resource managers may abort the local transaction instead of correctly committing the transaction. This command references the `TUXCONFIG` file to determine the BEA Tuxedo file system containing the `TLOG` and name of the `TLOG` in that file system. This command prompts for confirmation before proceeding unless the `-yes` option is specified.

`lidl -z config [ -o offset ] [ dlindex ]`

Print the universal device list. For each device the following is listed: the name, the starting block, and the number of blocks on the device. In verbose mode, a map is printed which shows free space (starting address and size of free space). If *dlindex* is specified, then only the information for that device list entry is printed. *config* points to the device containing the universal device list; it must be an absolute path name (starting with `/`). If the `-z` option is not given and a default has not been set, the path named by the `FSCONFIG` environment variable is used. The `-o offset` option can be used to specify an offset into *config*. If the `-o` option is not given and a default has not been set, the value of the environment variable `FSOFFSET` is used. If `FSOFFSET` is not set, the default is 0.

`livtoc -z config [ -o offset ]`

Prints information for all VTOC table entries. The information printed for each entry includes the name of the VTOC table, the device on which it is found, the offset of the VTOC table from the beginning of the device and the number of pages allocated for that table. *config* points to the device containing the universal device list; it must be an absolute path name (starting with /). If the `-z` option is not given and a default has not been set, the path named by the `FSCONFIG` environment variable is used. The `-o offset` option can be used to specify an offset into *config*. If the `-o` option is not specified, the value of the environment variable `FSOFFSET` is used. If `FSOFFSET` is not set, the default is 0.

`loadtlog -m machine filename`

Read the text version of a TLOG from the specified *filename* (produced by `dumpltlog`) into the existing TLOG for the named or default *machine* (it cannot be “DBBL” or “all”).

`logstart machine`

Force a warm start for the TLOG information on the specified *machine*. This should normally be done following a `loadtlog` and after disk relocation during server group migration.

`master (m) [ -yes ]`

If `master` is run on the backup node when partitioned, the backup node takes over as the acting master node and a DBBL is booted to take over administrative processing. If `master` is run on the master node when the backup node is acting as the master, the DBBL is migrated to the master node, and the backup node is no longer the acting master node. This command prompts for confirmation before proceeding unless the `-yes` option is specified.

`migrategroup (migg) [-cancel] group_name`

The `migrategroup` command takes the name of a server group. If the configuration file specifies the `MIGRATE` option and an alternate location for the group, all servers in *group\_name* are migrated to the alternate location. Servers must be shut down for migration with the following command.

```
shutdown -R -g groupname
```

The `-R` option retains server names in the bulletin board so that migration can be done. The migration can be canceled after the `shutdown -R` by the following command.

```
migrategroup -cancel groupname
```

The `-cancel` option deletes the server names from the bulletin board.

`migratemach (migm) [-cancel] machine`

All servers running on the specified *machine* are migrated to their alternate location. Servers must be shut down for migration with the following command.

```
shutdown -R -l machine
```

When the `migratemachine` command is used, all server groups located on *machine* must have the same alternate location (otherwise `migrategroup` must be used). Migration of an LMID (that is, machine) that contains Domains gateway servers implies the migration of these gateway servers to the alternate LMID. Specifying the `-cancel` option causes a migration that is already in progress to be cancelled. In other words, the servers have been shut down—with the `tmshutdown -R` command—but have not yet been migrated.

`paginate (page) [{off | on}]`

Paginate output. If no option is given, then the current setting is toggled, and the new setting is printed. The initial setting is on, unless either standard input or standard output is a non-`tty` device. Pagination may be turned on only when both standard input and standard output are `tty` devices.

The default paging command is indigenous to the native operating system environment. In a UNIX operating system environment, for example, the default paging command is `pg`. The shell environment variable `PAGER` may be used to override the default command used for paging output.

`passwd`

Prompt the administrator for a new application password in an application requiring security.

`pcclean (pcl) machine`

`pcclean` first forces a `bbclean` on the specified *machine* to restart or cleanup any servers that may require it. If *machine* is partitioned, entries for processes and services identified as running on *machine* are removed from all non-partitioned bulletin boards. If *machine* is not partitioned, any processes or services that can not be restarted or cleaned up are removed.

`printclient (pclt) [-m machine] [-u username] [-c cltname]`

Print information for the specified set of client processes. If no arguments or defaults are set, then information on all clients is printed. In a multicontexted client, `printclient` produces a separate entry for each context. The `-m`, `-u`,

and `-c` options or defaults can be used to restrict the information to any combination of machine, user name, or client name.

`printconn (pc) [-m machine]`

Print information about conversational connections. The `-m` option or default can be used to restrict the information to connections to or from the specified machine. A *machine* value of “all” or “DBBL” prints information from all machines.

`printgroup (pg) [-m machine] [-g groupname]`

Print server group table information. The default is to print information for all groups. The `-g` and `-m` options or defaults can be used to restrict the information to a combination of group or machine. The information printed includes the server group name, the server group number, primary and alternate LMIDs, and the current location.

`printnet (pnw) [mach_list]`

Print network connection information. The default is to print information for all machines. The `printnet` command optionally takes a comma-separated list of machines (LMIDs) as arguments. If such a list is provided, information is restricted to network connections involving the specified machines. For each machine, the information indicates whether the machine is partitioned. If a machine is not partitioned, information is printed indicating the other machines to which it is connected and counts of messages in and out.

`printqueue (pq) [qaddress]`

Print queue information for all application and administrative servers. The default is to print information about all queues. The *qaddress* command line or default can be used to restrict information to a specific queue. Output includes the server name and the name of the machine on which the queues reside.

`printserver (psr) [-m machine] [-g groupname] [-i srvid] [-q qaddress]`

Print information for application and administrative servers. The `-q`, `-m`, `-g`, and `-i` options can be used to restrict the information to any combination of queue address, machine, group, and server. In a multicontexted server, `printserver` prints a single entry for all contexts in the server.

`printservice (psc) [-m machine] [-g groupname] [-i srvid] [-a { 0 | 1 | 2 }] [-q qaddress] [-s service]`

Print information for application and administrative services. The `-q`, `-m`, `-g`, `-i` and `-s` options can be used to restrict the information to any combination of queue address, machine, group, server or service. The `-a` option allows you

to select the class of service: `-a0` limits the display to application services, `-a1` selects application services plus system services that can be called by an application, `-a2` selects both of those, plus system services that can be called by the BEA Tuxedo system.

`printtrans (pt) [-g groupname] [-m machine]`

Print global transaction table information for either the specified or the default machine. If *machine* is “all” or “DBBL,” then information is merged together from transaction tables at all non-partitioned machines in the application. The command line or default *groupname* value can be used to restrict the information to transactions in which the group is a participant (including the coordinator).

When printed in terse mode, the following information is provided: the transaction identifier, an index used for aborting or committing transactions with `aborttrans` or `committrans`, the transaction status, and a count of participants.

In verbose mode, transaction timeout information and participant information (for example, server group names and status, including the identity of the coordinator) is also printed.

`quit (q)`

Terminate the session.

`reconnect (rco) non-partitioned_machine1 partitioned_machine2`

Initiate a new connection from the non-partitioned machine to the partitioned machine. `reconnect` forces a new connection from the non-partitioned machine to the partitioned machine. If a connection is already active, it is closed before the reconnect. This may cause in-transit messages to be lost, resulting in transaction timeouts. It is possible for a machine or network connection to be down, but the network interface driver will continue to accept and buffer requests without any error indication to the BRIDGE. In this case, `reconnect` will fail, forcing the BRIDGE to recognize that the remote machine cannot be reached. Note that in most cases, after network problems are resolved, the BRIDGE reconnects automatically, making manual intervention (with `reconnect`) unnecessary.

`resume (res) {-q qaddress | -g groupname | -i srvid | -s service} . . .`

Resume (unsuspend) services. The `-q`, `-g`, `-s`, and `-i` options can be used to restrict the resumed services to any combination of queue, group, service, and server. (At least one of these options must be specified or have a default.)

Thus the following command line provides a shortcut method of unsuspending all services advertised on the queue with the address `servq8`.

```
> resume -q servq8
```

Once a suspended service is resumed, the offering server is selected as a candidate server for that service, as well as for other (unsuspended) services it may offer. If multiple servers are reading from a single queue, the status of a particular service is reflected in all servers reading from that queue.

```
serverparms (srp) -g groupname -i srvid
```

Print the parameters associated with the server specified by *groupname* and *srvid* for a group.

```
serviceparms (scp) -g groupname -i srvid -s service
```

Print the parameters associated with the service specified by *groupname*, *srvid*, and *service*.

```
shmstats (sstats) [ex | app]
```

If `MODEL SHM` is specified in the configuration file, `shmstats` can be used to assure more accurate statistics. When entered with no argument, `shmstats` returns the present setting of the `TMACCSTATS` flag of the `bbparms.options` member of the bulletin board structure. This tells you whether statistics presently being gathered are exact or approximate. If the command is entered with `ex` specified, `shmstats` turns on the `TMACCSTATS` flag, locks the bulletin board, and zeroes out the counters for server table, queue table, and service table entries.

```
shutdown (stop) [options]
```

This command is identical to the `tmshutdown()` command. `tmshutdown options` can be used to select servers to be stopped. See `tmshutdown(1)` for an explanation of options and restrictions on use.

```
suspend (sus) { -q qaddress | -g groupname | -i srvid | -s service } ...
```

Suspend services. The `-q`, `-g`, `-s`, and `-i` options can be used to restrict the suspended services to any combination of queue, group, service, and server (At least one of these options must be specified or have a default.) Thus the following command provides a shortcut method of suspending all services advertised on the queue with the address `servq8`.

```
> suspend -q servq8
```

When a service is suspended, the offering server is no longer selected as a candidate server for that service, although it continues to be selected to

process other services it may offer. Queued requests for the suspended service are processed until the queue is drained. If multiple servers are reading from a single queue, the status of a particular service is reflected in all servers reading from that queue.

`unadvertise (unadv) {-q qaddress [-g groupname] [-i srvid] | -g groupname -i srvid} service`

Remove an entry in the service table for the indicated *service*. If *qaddress* is not specified, then both *groupname* and *srvid* are required to uniquely identify a server. Specifying either a queue or a particular server on that queue achieve the same results. If this *service* is to be removed from a multiple server, single queue (MSSQ) set, then the advertisement for *service* is removed from all servers reading from that queue.

`verbose (v) [{off | on}]`

Produce output in verbose mode. If no option is given, then the current setting is toggled, and the new setting is printed. The initial setting is `off`. The `-v` (verbose) and `-t` (terse) options on individual commands can be used to temporarily override the current setting.

`! shellcommand`

Escape to the shell and execute *shellcommand*.

`!!`

Repeat previous shell command.

`# [text]`

Lines beginning with `"#"` are comment lines and are ignored.

`<CR>`

Repeat the last command.

**Security** When `tmadmin` runs as the administrator, it does not pass through security since it is already checked to be the application administrator's login ID.

The only time that `tmadmin` may run as someone other than the application administrator is if the `-r` option is used to access the application as a client. If such a user invokes `tmadmin` with the `-r` option, and if security is turned on for the application, then the application password is required to access application data. If standard input is a terminal, then `tmadmin` prompts the user for the password with `echo` turned off on the reply. If standard input is not a terminal, the password is retrieved from the `APP_PW` environment variable. If this environment variable is not specified and an application password is required, then `tmadmin` fails.

## Section 1 - Commands

---

|                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Environment Variables | <p><code>tmadmin</code> acts as an application client if the <code>-r</code> option is used or if it cannot register as the application administrator. If this is the case, and if standard input is not from a terminal, then the <code>APP_PW</code> environment variable must be set to the application password in a security application.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Diagnostics           | <p>If the <code>tmadmin</code> command is entered before the system has been booted, the following message is displayed.</p> <pre>No bulletin board exists. Entering boot mode &gt;</pre> <p><code>tmadmin</code> then waits for a <code>boot</code> command to be entered.</p> <p>If the <code>tmadmin</code> command is entered, without the <code>-c</code> option, on an inactive node that is not the <code>MASTER</code>, the following message is displayed and the command terminates.</p> <pre>Cannot enter boot mode on non-master node.</pre> <p>If an incorrect application password is entered or is not available to a shell script through the environment, then a log message is generated and the command terminates, after displaying the following message.</p> <pre>Invalid password entered.</pre> |
| Interoperability      | <p><code>tmadmin</code> may be run on any node within an active interoperating application. However, the commands and command-line arguments available are restricted to those available via <code>tmadmin</code> in the release corresponding to the node on which <code>tmadmin</code> is running. For example, the <code>broadcast</code>, <code>passwd</code>, and <code>printclient</code> commands are not available on Release 4.1 nodes.</p>                                                                                                                                                                                                                                                                                                                                                                    |
| Portability           | <p><code>tmadmin</code> is supported on any platform on which the BEA Tuxedo server environment is supported.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Notices               | <p>The <code>machine</code> option has no effect in a non-networked uniprocessor environment.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| See Also              | <p><code>tmboot(1)</code>, <code>tmloadcf(1)</code>, <code>tmshutdown(1)</code>, <code>compilation(5)</code>, <code>UBBCONFIG(5)</code></p> <p><i>Administering a BEA Tuxedo Application at Run Time</i></p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

# tmboot(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | tmboot - bring up a BEA Tuxedo configuration                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Synopsis    | tmboot [-l <i>lmid</i> ] [-g <i>grpname</i> ] [-i <i>srvid</i> ] [-s <i>aout</i> ] [-o <i>sequence</i> ]<br>[-S] [-A] [-b] [-B <i>lmid</i> ] [-T <i>grpname</i> ] [-e <i>command</i> ] [-w] [-y] [-g]<br>[-n] [-c] [-M] [-dl]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Description | tmboot brings up a BEA Tuxedo application in whole or in part, depending on the options specified. tmboot can be invoked only by the administrator of the bulletin board (as indicated by the <code>UID</code> parameter in the configuration file) or by <code>root</code> . The tmboot command can be invoked only on the machine identified as <code>MASTER</code> in the <code>RESOURCES</code> section of the configuration file, or the backup acting as the <code>MASTER</code> , that is, with the <code>DBBL</code> already running (via the <code>master</code> command in <code>tadmin(1)</code> ). Except, if the <code>-b</code> option is used; in that case, the system can be booted from the backup machine without it having been designated as the <code>MASTER</code> . |

With no options, tmboot executes all administrative processes and all servers listed in the `SERVERS` section of the configuration file named by the `TUXCONFIG` and `TUXOFFSET` environment variables. If the `MODEL` is `MP`, a `DBBL` administrative server is started on the machine indicated by the `MASTER` parameter in the `RESOURCES` section. An administrative server (`BBL`) is started on every machine listed in the `MACHINES` section. For each group in the `GROUPS` section, `TMS` servers are started based on the `TMSNAME` and `TMSCOUNT` parameters for each entry. All administrative servers are started followed by servers in the `SERVERS` sections. Any `TMS` or gateway servers for a group are booted before the first application server in the group is booted. The `TUXCONFIG` file is propagated to remote machines as necessary. tmboot normally waits for a booted process to complete its initialization (that is, `tpsvrinit()`) before booting the next process.

Booting a gateway server implies that the gateway advertises its administrative service, and also advertises the application services representing the foreign services based on the `CLOPT` parameter for the gateway. If the instantiation has the concept of foreign servers, these servers are booted by the gateway at this time.

Booting an `LMID` is equivalent to booting all groups on that `LMID`.

Application servers are booted in the order specified by the `SEQUENCE` parameter, or in the order of server entries in the configuration file (see the description in `UBBCONFIG(5)`). If two or more servers in the `SERVERS` section of the configuration file have the same `SEQUENCE` parameter, then tmboot may boot these servers in

parallel and will not continue until they all complete initialization. Each entry in the `SERVERS` section can have a `MIN` and `MAX` parameter. `tmboot` boots `MIN` application servers (the default is 1 if `MIN` is not specified for the server entry) unless the `-i` option is specified; using the `-i` option causes individual servers to be booted up to `MAX` occurrences.

If a server cannot be started, a diagnostic is written on the central event log (and to the standard output, unless `-q` is specified), and `tmboot` continues—except that if the failing process is a `BBL`, servers that depend on that `BBL` are silently ignored. If the failing process is a `DBBL`, `tmboot` ignores the rest of the configuration file. If a server is configured with an alternate `LMID` and fails to start on its primary machine, `tmboot` automatically attempts to start the server on the alternate machine and, if successful, sends a message to the `DBBL` to update the server group section of `TUXCONFIG`.

For servers in the `SERVERS` section, only `CLOPT`, `SEQUENCE`, `SRVGRP`, and `SRVID` are used by `tmboot`. Collectively, these are known as the server's boot parameters. Once the server has been booted, it reads the configuration file to find its run-time parameters. (See `UBBCONFIG(5)` for a description of all parameters.)

All administrative and application servers are booted with `APPDIR` as their current working directory. The value of `APPDIR` is specified in the configuration file in the `MACHINES` section for the machine on which the server is being booted.

The search path for the server executables is `APPDIR`, followed by `TUXDIR/bin`, followed by `/bin` and `/usr/bin`, followed by any `PATH` specified in the `ENVFILE` for the `MACHINE`. The search path is used only if an absolute path name is not specified for the server. Values placed in the server's `ENVFILE` are not used for the search path.

When a server is booted, the variables `TUXDIR`, `TUXCONFIG`, `TUXOFFSET`, and `APPDIR`, with values specified in the configuration file for that machine, are placed in the environment. The environment variable `LD_LIBRARY_PATH` is also placed in the environment of all servers. Its value defaults to

```
$APPDIR:$TUXDIR/lib:/lib:/usr/lib:lib> where lib> is the value of the first
LD_LIBRARY_PATH= line appearing in the machine ENVFILE. See UBBCONFIG(5) for
a description of the syntax and use of the ENVFILE.
```

The `ULOGPFX` for the server is also set up at boot time based on the parameter for the machine in the configuration file. If not specified, it defaults to `$APPDIR/ULOG`.

All of these operations are performed before the application initialization function, `tpsvrinit()`, is called.

Many of the command-line options of `tmbboot` serve to limit the way in which the system is booted and can be used to boot a partial system. The following options are supported.

`-l lmid`

For each group whose associated LMID parameter is *lmid*, all TMS and gateway servers associated with the group are booted and all servers in the `SERVERS` section associated with those groups are executed.

`-g grpname`

All TMS and gateway servers for the group whose `SRVGRP` parameter is *grpname* are started, followed by all servers in the `SERVERS` section associated with that group. TMS servers are started based on the `TMSNAME` and `TMSCOUNT` parameters for the group entry.

`-i srvid`

All servers in the `SERVERS` section whose `SRVID` parameter is *srvid* are executed.

`-s aout`

All servers in the `SERVERS` section with name *aout* are executed. This option can also be used to boot TMS and gateway servers; normally this option is used in this way in conjunction with the `-g` option.

`-o sequence`

All servers in the `SERVERS` section with `SEQUENCE` parameter *sequence* are executed.

`-S`

All servers in the `SERVERS` section are executed.

`-A`

All administrative servers for machines in the `MACHINES` section are executed. Use this option to guarantee that the `DBBL` and all `BBL` and `BRIDGE` processes are brought up in the correct order. (See also the description of the `-M` option.)

`-b`

Boot the system from the `BACKUP` machine (without making this machine the `MASTER`).

`-B lmid`

A `BBL` is started on a processor with logical name *lmid*.

- M  
This option starts administrative servers on the master machine. If the `MODEL` is `MP`, a `DBBL` administrative server is started on the machine indicated by the `MASTER` parameter in the `RESOURCES` section. A `BBL` is started on the `MASTER` machine, and a `BRIDGE` is started if the `LAN` option and a `NETWORK` entry are specified in the configuration file.
- dl  
Causes command-line options to be printed on the standard output. Useful when preparing to use `sdb` to debug application services.
- T *grpname*  
All `TMS` servers for the group whose `SRVGRP` parameter is *grpname* are started (based on the `TMSNAME` and `TMSCOUNT` parameters associated with the group entry). This option is the same as booting based on the `TMS` server name (`-s` option) and the group name (`-g`).
- e *command*  
Causes *command* to be executed if any process fails to boot successfully. *command* can be any program, script, or sequence of commands understood by the command interpreter specified in the `SHELL` environment variable. This allows an opportunity to bail out of the boot procedure. If *command* contains white space, the entire string must be enclosed in quotes. This command is executed on the machine on which `tmboot` is being run, not on the machine on which the server is being booted.
- w  
Informs `tmboot` to boot another server without waiting for servers to complete initialization. This option should be used with caution. `BBLs` depend on the presence of a valid `DBBL`; ordinary servers require a running `BBL` on the processor on which they are placed. These conditions cannot be guaranteed if servers are not started in a synchronized manner. This option overrides the waiting that is normally done when servers have sequence numbers.
- y  
Assumes a `yes` answer to a prompt that asks if all administrative and server processes should be booted. (The prompt appears only when the command is entered with none of the limiting options.)
- q  
Suppresses the printing of the execution sequence on the standard output. It implies `-y`.

-n The execution sequence is printed, but not performed.

-c Minimum IPC resources needed for this configuration are printed.

When the `-l`, `-g`, `-i`, `-o`, and `-s` options are used in combination, only servers that satisfy all qualifications specified are booted. The `-l`, `-g`, `-s`, and `-T` options cause TMS servers to be booted; the `-l`, `-g`, and `-s` options cause gateway servers to be booted; the `-l`, `-g`, `-i`, `-o`, `-s`, and `-S` options apply to application servers. Options that boot application servers fail if a BBL is not available on the machine. The `-A`, `-M`, and `-B` options apply only to administrative processes.

The standard input, standard output, and standard error file descriptors are closed for all booted servers.

**Interoperability** `tmboot` must run on the master node, which in an interoperating application must be the highest release available. `tmboot` detects and reports configuration file conditions that would lead to the booting of administrative servers such as Workstation listeners on sites that cannot support them.

**Portability** `tmboot` is supported on any platform on which the BEA Tuxedo server environment is supported.

**Environment Variables** During the installation process, an administrative password file is created. When necessary, the BEA Tuxedo system searches for this file in the following directories (in the order shown): `APPDIR/.adm/tlisten.pw` and `TUXDIR/udataobj/tlisten.pw`. To ensure that your password file will be found, make sure you have set the `APPDIR` and/or `TUXDIR` environment variables.

**Link-Level Encryption** If the link-level encryption feature is in operation between `tmboot` and `tlisten`, link-level encryption will be negotiated and activated first to protect the process through which messages are authenticated.

**Diagnostics** If `TUXCONFIG` is set to a non-existent file, two fatal error messages are displayed:

- error processing configuration file
- configuration file not found

If `tmboot` fails to boot a server, it exits with exit code 1 and the user log should be examined for further details. Otherwise `tmboot` exits with exit code 0.

If `tmboot` is run on an inactive non-master node, a fatal error message is displayed.

`tmboot cannot run on a non-master node.`

## Section 1 - Commands

---

If `tmboot` is run on an active node that is not the acting master node, the following fatal error message is displayed.

```
tmboot cannot run on a non acting-master node in an active
application.
```

If the same `IPCKEY` is used in more than one `TUXCONFIG` file, `tmboot` fails with the following message.

```
Configuration file parameter has been changed since last tmboot
```

If there are multiple node names in the `MACHINES` section in a non-LAN configuration, the following fatal error message is displayed.

```
Multiple nodes not allowed in MACHINES for non-LAN application.
```

If `tlisten` is not running on the `MASTER` machine in a LAN application, a warning message is printed. In this case, `tmadmin(1)` cannot run in administrator mode on remote machines; it is limited to read-only operations. This also means that the backup site cannot reboot the master site after failure.

**Examples** To start only those servers located on the machines logically named `CS0` and `CS1`, enter the following command.

```
tmboot -l CS0 -l CS1
```

To start only those servers named `CREDEB` that belong to the group called `DBG1`, enter the following command.

```
tmboot -g DBG1 -s CREDEB1
```

To boot a `BEL` on the machine logically named `PE8`, as well as all those servers with a location specified as `PE8`, enter the following command.

```
tmboot -B PE8 -l PE8
```

To view minimum IPC resources needed for the configuration, enter the following command.

```
tmboot -c
```

The following is an example of the output produced by the `-c` option:

```
IPC sizing (minimum BEA Tuxedo values only) ...
 Fixed Minimums Per Processor
SHMMIN: 1
SHMALL: 1
SEMMAP: SEMMNI
```

| Node  | Variable Minimums Per Processor |        |        |        |        |        |             |
|-------|---------------------------------|--------|--------|--------|--------|--------|-------------|
|       | SEMUME,<br>SEMMNU,<br>SEMMNS    | SEMMSL | A<br>* | SEMMNI | MSGMNI | MSGMAP | SHMMAX<br>* |
| sfpup | 60                              | 1      | 60     | A + 1  | 10     | 20     | 76K         |
| sfsup | 63                              | 5      | 63     | A + 1  | 11     | 22     | 76K         |

where 1 = A = 8.

The number of expected application clients per processor should be added to each MSGMNI value. MSGMAP should be twice MSGMNI. SHMMIN should always be set to 1.

The minimum IPC requirements can be compared to the parameters set for your machine. See the system administration documentation for your machine for information about how to change these parameters. If the -y option is used, the display will differ slightly from the previous example.

**Notices** The tmboot command ignores the hangup signal (SIGHUP). If a signal is detected during boot, the process continues.

Minimum IPC resources displayed with the -c option apply only to the configuration described in the configuration file specified; IPC resources required for a resource manager or for other BEA Tuxedo configurations are not considered in the calculation.

**See Also** tadmin(1), tmloadcf(1), tmshutdown(1), UBBCONFIG(5)

*Administering a BEA Tuxedo Application at Run Time*

# tmconfig, wtmconfig(1)

**Name** `tmconfig, wtmconfig` - dynamically update and retrieve information about a running BEA Tuxedo application, as either a native client or a workstation client

**Synopsis** `tmconfig`  
`wtmconfig`

**Description** `tmconfig` is an interactive program that can be used to update some configuration file parameters, or MIB attributes, and to add records to parts of the `TUXCONFIG` file while the BEA Tuxedo application is running. `tmconfig` manages a buffer that contains input field values to be added, updated, or retrieved. When an operation is completed, `tmconfig` displays output field values and status. The user can update the input buffer using any available text editor.

`tmconfig` is a BEA Tuxedo native client, and `wtmconfig` is a workstation client, as you can see in the output of the `tmadmin/printclient` command sequence. If the application is using the `SECURITY` feature, `tmconfig` prompts for the application password.

**Note:** Because the same functionality is provided by both `tmconfig` and `wtmconfig`—the only difference being that `tmconfig` is a BEA Tuxedo native client and `wtmconfig` is a BEA Tuxedo workstation client—we refer only to `tmconfig` throughout most of this reference page. You can assume that the functionality described here is provided by `wtmconfig`, as well.

`tmconfig` first prompts for the desired section, and then for the desired operation. The prompt for the section appears as follows.

```
Section:1) RESOURCES, 2) MACHINES, 3) GROUPS 4) SERVERS 5)SERVICES
6) NETWORK 7) ROUTING q) QUIT 9) WSL 10) NETGROUPS 11) NETMAP [1]:
```

The default section appears in square brackets at the end of the prompt.

`tmconfig` then prompts for the desired operation.

```
Operation: 1) FIRST 2) NEXT 3) RETRIEVE 4) ADD 5) UPDATE
6) CLEAR BUFFER 7) QUIT [1]:
```

The default operation is shown in square brackets at the end of the prompt. To select the default, simply enter RETURN. To select another option, enter the appropriate number and RETURN.

The following operations are currently supported.

1. `FIRST` - Retrieve the first record from the specified section. No key fields are needed. (Any key fields that are in the input buffer are ignored.)
2. `NEXT` - Retrieve the next record from the specified section, based on the key fields in the input buffer.
3. `RETRIEVE` - Retrieve the indicated record from the specified section by key field(s).
4. `ADD` - Add the indicated record in the specified section. Any fields not specified (unless required) take their default values as specified in `UBBCONFIG(5)`. The current value for all fields is returned in the output buffer. This operation can be done only by the BEA Tuxedo system administrator.
5. `UPDATE` - Update the record specified in the input buffer in the selected section. Any fields not specified in the input buffer remain unchanged. The current value for all fields is returned in the input buffer. This operation can be done only by the BEA Tuxedo administrator.
6. `CLEAR BUFFER` - Clear the input buffer (all fields are deleted). After this operation, `tmconfig` immediately prompts for the section again.
7. `QUIT` - Exit the program gracefully (the client is terminated). A value of `q` for any prompt also exits the program.

For administrator operations, the effective user identifier must match the BEA Tuxedo administrator user identifier (UID) for the machine on which this program is executed. When a record is updated or added, all default values and validations used by `tmloadcf(1)` are enforced.

`tmconfig` then prompts you to indicate whether or not you want to edit the input buffer.

```
Enter editor to add/modify fields [n]?
```

Entering a value of `y` puts the input buffer into a temporary file and executes the text editor. The environment variable `EDITOR` is used to determine which editor is to be used. The default is `ed`. The input format is in `fieldname/field value` pairs and is described in the “Input Format” section below. The field names associated with each

UBBCONFIG section are listed in tables in the sub-sections below. The semantics of the fields and associated ranges, default values, restrictions, and so on, are described in UBBCONFIG(5). Note that permissions values are specified in decimal, not octal. In most cases, the field name is the same as the KEYWORD in the UBBCONFIG file, prefixed with TA\_.

When the user completes editing the input buffer, `tmconfig` reads it. If more than one line occurs for a particular field name, the first occurrence is used and other occurrences are ignored. If any errors occur, a syntax error is printed and `tmconfig` prompts you to indicate whether you want to correct the problem.

```
Enter editor to correct?
```

If the problem is not corrected (response `n`), then the input buffer contains no fields. Otherwise, the editor is executed again.

Finally, `tmconfig` asks whether the operation should be performed.

```
Perform operation [y]?
```

When the operation completes, `tmconfig` prints the return value (as in Return value TAOK), followed by the output buffer fields. The process then begins again with a prompt for the section. All output buffer fields are available in the input buffer unless the buffer is cleared.

Entering a break at any time restarts the interaction at the prompt for the section.

When QUIT is selected, `tmconfig` prompts you to create a backup text version of the configuration.

```
Unload TUXCONFIG file into ASCII backup [y]?
```

If you select a backup, `tmconfig` prompts for a filename.

```
Backup filename [UBBCONFIG]?
```

On success, `tmconfig` indicates that a backup was created; otherwise an error is printed.

**Input Format** Input packets consist of lines formatted as follows.

```
fldname fldval
```

The field name is separated from the field value by one or more tabs.

Lengthy field values can be continued on the next line by having the continuation line begin with one or more tabs (which are dropped when read back into `tmconfig`).

Empty lines consisting of a single newline character are ignored.

To enter an unprintable character in the field value or to start a field value with a tab, use a backslash followed by the two-character hexadecimal representation of the desired character (see ASCII(5) in a UNIX reference manual). A space, for example, can be entered in the input data as `\20`. A backslash can be entered using two backslash characters. `tmconfig` recognizes all input in this format, but the hexadecimal format is most useful for non-printing characters.

**Limitations** The following are general limitations of the dynamic re-configuration capability.

1. Values for key fields (as indicated in the following sections) may not be modified. If key fields are modified in the editor buffer and the operation is done, then a different record will be modified, based on the new values of the key fields. Key fields can be modified, when the system is down, by reloading the configuration file.
2. Fields at the `LMID` level cannot be modified while the `LMID` is booted; similarly fields at the `GROUP` level cannot be modified while the `GROUP` is booted.
3. Many of the `RESOURCES` parameters cannot be updated on a running system.
4. Dynamic deletions are not supported. Deletions must be done off-line.
5. If you attempt to update a parameter in the wrong section (for example, if you try to update the `ENVFILE` parameter in the `MACHINES` section while working in the `RESOURCES` section), the operation appears to succeed (that is, `tmconfig` returns `TAOK`) but the change does not appear in your unloaded `UBBCONFIG` file.

**Relationship between `tmconfig`, `UBBCONFIG`, and the MIBs** In early releases of the BEA Tuxedo system all application configuration was accomplished by editing the `UBBCONFIG` file, a text file that contained all the configuration parameters for an application. Users of later releases of the system compiled the `UBBCONFIG` file into a binary file known as `TUXCONFIG` by running the `tmloadcf(1)` command. A subsequent release introduced `tmconfig`, a command that supports dynamic updating (that is updating of an active system) of various `TUXCONFIG` parameters.

A more recent development was the introduction of the BEA Tuxedo Management Information Bases (MIBs), which redefined BEA Tuxedo resources into classes and attributes. Along with the MIBs, the BEA Tuxedo system also provided an administration API that enabled an administrator (or a user) to access and change the attributes of an application programmatically.

With one exception, this entry of the *BEA Tuxedo Command Reference* (that is, `tmconfig`, `wtmconfig(1)`) provides brief descriptions only of the various classes of the MIBs. The exception is the Network class, for which a detailed description is provided here on `tmconfig()`. For details about all other sections, see `TM_MIB(5)`.

### When Attributes (Fields) Can Be Updated and Who Can Do It

One feature of the former `tmconfig` tables was a column that contained a value indicating whether a field could be updated. That information is provided in the MIB reference pages, but in a form that requires a little more digging on your part. See the description of Permissions in `MIB(5)`. The Permissions columns in MIB tables resemble the read, write, and execute permissions that are used to restrict access to files, but they convey more information and designate more control than simple file permissions. For example, by values in the Permissions columns in MIB tables can indicate whether or not a field can be changed when the system is active.

Study the description in `MIB(5)` before you attempt to use `tmconfig`.

### RESOURCES Section

For attributes in this section, see the description of the `T_DOMAIN` class in `TM_MIB(5)`.

#### Notes

The `ADD` operation is not valid for this section. Because there is only one record in this section, the `RETRIEVE` operation is the same as the `FIRST` operation (no key field is required). The `NEXT` operation always returns a record not found.

Changes to `TA_LDBAL`, `TA_CMTRET`, and `TA_SYSTEM_ACCESS` affect only new clients and servers that are subsequently booted. `TA_SYSTEM_ACCESS` cannot be changed if `NO_OVERRIDE` is specified and any server entries exist that do not match the specified access type (`PROTECTED` or `FASTPATH`). Changes to `TA_NOTIFY` and `TA_AUTHSVC` affect only new clients that are subsequently started.

Updates to parameters other than those listed above do not appear in your unloaded text backup file.

### MACHINES Section

For attributes in this section, see the description of the `T_MACHINE` class in `TM_MIB(5)`.

#### Notes

A machine cannot be added unless `LAN` appears in the `OPTIONS` in the `RESOURCES` section.

Updates to parameters other than those listed above do not appear in your unloaded text backup file.

**GROUPS Section** For attributes in this section, see the description of the `T_GROUP` class in `TM_MIB(5)`.

**SERVERS Section** For attributes in this section, see the description of the `T_SERVER` class in `TM_MIB(5)`.

**Notes** Parameter changes in the `SERVERS` section take effect the next time that an associated server is booted (and not restarted). If multiple servers are defined in an `MSSQ` set (using `TA_RQADDR`), they must have the same services booted (for example, changes to `TA_CLOPT` or `ENVFILE` must not affect the services that are booted such that they do not match currently booted servers). If `TA_MAX` is changed, automatic spawning of conversational servers for the new server identifiers is not performed until one or more servers in the server set are booted.

**Services Section** For attributes in this section, see the descriptions of the `T_SERVICE` and `T_SVCGRP` classes in `TM_MIB(5)`.

**Notes** Parameter changes in the `SERVICES` section take effect the next time a server offering the service is booted (and not restarted). Updates to `TA_ROUTINGNAME` are allowed only if there is no value in the `TA_SRVGRP` field or if the value of that field is `NULL`. In this case, the `TA_ROUTINGNAME` attribute is simultaneously updated in all matching `SERVICES` entries. The `TA_ROUTINGNAME` corresponds to the `ROUTING` field in the `SERVICES` section.

Updates to parameters other than those listed above do not appear in your unloaded text backup file.

**NETWORK Section** The following table lists the fields in the `NETWORK` section.

---

### NETWORK SECTION

---

| Field Identifier       | Field Type | Update | Notes                                      |
|------------------------|------------|--------|--------------------------------------------|
| <code>TA_LMID</code>   | string     | No     | key                                        |
| <code>TA_NADDR</code>  | string     | Sys    | ASCII format (no embedded NULL characters) |
| <code>TA_BRIDGE</code> | string     | Sys    |                                            |

---

---

**NETWORK SECTION**

| Field Identifier | Field Type | Update | Notes                                      |
|------------------|------------|--------|--------------------------------------------|
| TA_NLSADDR       | string     | Sys    | ASCII format (no embedded NULL characters) |

**Notes** A record cannot be added while the associated LMID is booted.

No operations can be done on the NETWORKS section unless LAN appears in the OPTIONS in the RESOURCES section.

Updates to parameters other than those listed above do not appear in your unloaded text backup file.

**ROUTING Section** For attributes in this section, see the description of the T\_ROUTING class in TM\_MIB (5).

**Notes** The ROUTING section cannot be updated while the system is running. New ROUTING section entries may be added if three parameters in the RESOURCES section that control the size of the bulletin board—MAXDRT, MAXRFT, and MAXRTDATA—are set to allow for growth.

**WSL Section** For attributes in this section, see the description of the T\_WSL class in TM\_MIB (5).

**Notes** The T\_WSL class should be used to update the CLOPT for WSL servers, even though this is available via the SERVER section.

**NETGROUPS Section** For attributes in this section, see the description of the T\_WSL class in TM\_MIB (5).

**NETMAP Section** For attributes in this section, see the description of the T\_NETMAP class in TM\_MIB (5).

**Security** If tmconfig is run in a secure application, it requires an application password to access the application. If the standard input is a terminal, tmconfig prompts the user for the password with echo turned off on the reply. If the standard input is not a terminal, the password is retrieved from the APP\_PW environment variable. If this environment variable is not specified and an application password is required, then tmconfig fails.

**Workstation Client** As a Workstation client, the command is named wtmconf on DOS and wtmconfig otherwise. The UPDATE and ADD commands are not supported; TAEPERM is returned.

**Environment Variables** tmconfig resets the FIELDTBLS and FLDTBLDIR environment variables to pick up the `/${TUXDIR}/udataobj/tpadmin` field table. TUXDIR must be set correctly.

APP\_PW must be set to the application password in a secure application if standard input is not from a terminal.

Before your client can join a BEA Tuxedo application, you must set several environment variables:

- For tmconfig you must set the TUXCONFIG environment variable.
- For wtmconfig you must set the WSNADDR environment variable. You may also have to set WSDEVICE, depending on the network protocol used by the BEA Tuxedo system. Which network protocol is used depends, in turn, on the platform on which your application is running. To find out which network protocols are used on your platform, see Appendix A, “Platform Data Sheets,” in the *BEA Tuxedo Installation Guide*.

**Diagnostics** tmconfig fails if it cannot allocate a typed buffer, if it cannot determine the `/etc/passwd` entry for the user, if it cannot become a client process, if it cannot create a temporary file in `/tmp` for editing the input buffer, or if it cannot reset the FIELDTBLS or FLDTBLDIR environment variable.

The return value printed by tmconfig after the completion of each operation indicates the status of the requested operation. There are three classes of return values.

The following return values indicate a problem with permissions or a BEA Tuxedo communications error. They indicate that the operation did not complete successfully.

[TAEPERM]

The calling process specified a TA\_UPDATE or TA\_ADD opcode but is not running as the BEA Tuxedo administrator.

[TAESYSTEM]

A BEA Tuxedo error has occurred. The exact nature of the error is written to `userlog(3c)`.

[TAEOS]

An operating system error has occurred.

[TAETIME]

A blocking timeout occurred. The input buffer is not updated so no information is returned for retrieval operations. The status of update

operations can be checked by doing a retrieval on the record that was being updated.

The following return values indicate a problem in doing the operation itself and generally are semantic problems with the application data in the input buffer. The string field `TA_STATUS` is set in the output buffer indicating the problem. The string field `TA_BADFLDNAME` is set to the field name for the field containing the value that caused the problem (assuming the error can be attributed to a single field).

[TAERANGE]

A field value is out of range or is invalid.

[TAEINCONSIS]

A field value or set of field values is inconsistently specified (for example, an existing `RQADDR` value is specified for a different `SRVGRP` and `SERVERNAME`).

[TAECONFIG]

An error occurred while the `TUXCONFIG` file was being read.

[TAEDUPLICATE]

The operation attempted to add a duplicate record.

[TAENOTFOUND]

The record specified for the operation was not found.

[TAEREQUIRED]

A field value is required but is not present.

[TAE SIZE]

A field value for a string field is too long.

[TAEUPDATE]

The operation attempted to do an update that is not allowed.

[TAENOSPACE]

The operation attempted to do an update but there was not enough space in the `TUXCONFIG` file and/or the Bulletin Board.

The following return values indicate that the operation was successful, at least at the MASTER site.

[TAOK]

The operation succeeded. No updates were done to the `TUXCONFIG` file or the Bulletin Board.

[TAUPDATED]

The operation succeeded. Updates were made to the TUXCONFIG file and/or the Bulletin Board.

[TAPARTIAL]

The operation succeeded at the MASTER site but failed at one or more non-MASTER sites. The non-MASTER sites will be marked as invalid or partitioned. See the administrator's guide for further information.

**Interoperability** The UPDATE and ADD operations are not allowed if a BEA Tuxedo System Release 4.0 or 4.1 node is booted. These nodes must be shut down before these operations are performed. When re-booted, they will pick up the changes.

**tmunloadcf** When `tmunloadcf(1)` is run to print entries in the configuration, certain field values  
**Compatibility** are not printed if they are not set (for strings) or 0 (for integers), or if they match the default value for the field. These fields always appear in the output buffer when `tmconfig` is used. In this way, it makes it easier for the administrator to retrieve an entry and update a field that previously was not set. The entry will have the field name followed by a tab but no field value.

**Example** In the following example, `tmconfig` is used to correct the network address specified on a Workstation Listener server. It happens to be the first entry in the Servers section. The editor used in the example is `ed(1)`.

```
$ EDITOR=ed tmconfig
```

```
Section:1) RESOURCES, 2) MACHINES, 3) GROUPS 4) SERVERS 5)SERVICES
6) NETWORK 7) ROUTING q) QUIT 9) WSL 10) NETGROUPS 11) NETMAP [1]: 4
```

```
Operation: 1) FIRST 2) NEXT 3) RETRIEVE 4) ADD 5) UPDATE
```

```
6) CLEAR BUFFER 7) QUIT [1]: 1
```

```
Enter editor to add/modify fields [n]? <return>
```

```
Perform operation [y]? <return>
```

```
Return value TAOK
```

```
Buffer contents:
```

```
TA_OPERATION 4
TA_SECTION 3
TA_SRVID 2
TA_MIN 1
TA_MAX 1
TA_RQPERM 432
TA_RPPERM 432
TA_MAXGEN 1
TA_GRACE 86400
TA_STATUS Operation completed successfully
TA_SRVGRP WDBG
```

## Section 1 - Commands

---

```
TA_SERVERNAME WSL
TA_CLOPT -A -- -d/dev/tcp -M4 -m2 -x5 -n0x0002fe19c00b6d6b
TA_CONV N
TA_REPLYQ N
TA_RESTART N
Section:1) RESOURCES, 2) MACHINES, 3) GROUPS 4) SERVERS 5)SERVICES
6) NETWORK 7) ROUTING q) QUIT 9) WSL [4]10) NETGROUPS 11) NETMAP [4]: <return>
Operation: 1) FIRST 2) NEXT 3) RETRIEVE 4) ADD 5) UPDATE
6) CLEAR BUFFER 7) QUIT [1]: 5
Enter editor to add/modify fields [n]? y
240
/CLOPT/s/6d6b/690E/p
TA_CLOPT -A -- -d/dev/tcp -M4 -m2 -x5 -n0x0002fe19c00b690E
w
240
q
Perform operation [y]? <return>
Return value TAUPDATED
Buffer contents:
TA_OPERATION 1
TA_SECTION 3
TA_SRVID 2
TA_MIN 1
TA_MAX 1
TA_RQPERM 432
TA_RPPERM 432
TA_MAXGEN 1
TA_GRACE 86400
TA_STATUS Update completed successfully
TA_SRVGRP WDBG
TA_SERVERNAME WSL
TA_CLOPT -A -- -d/dev/tcp -M4 -m2 -x5 -n0x0002fe19c00b690E
TA_CONV N
TA_REPLYQ N
TA_RESTART N
Section:1) RESOURCES, 2) MACHINES, 3) GROUPS 4) SERVERS 5)SERVICES
6) NETWORK 7) ROUTING q) QUIT 9) WSL [1] 10) NETGROUPS 11) NETMAP {1}: q
Unload TUXCONFIG file into ASCII backup [y]? <return>
Backup filename [UBBCONFIG]? <return>
Configuration backed up in UBBCONFIG
$ # boot the changed server
$ tmbot -s WSL -i 2
```

See Also `tmbot(1)`, `tmloadcf(1)`, `userlog(3c)`, `TM_MIB(5)`, `UBBCONFIG(5)`

---

# tmipcrm(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | tmipcrm - remove IPC resources allocated by a BEA Tuxedo application on a local machine                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Synopsis    | tmipcrm [-y] [-n] [ <i>TUXCONFIG_file</i> ]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Description | <p>tmipcrm cleans up the IPC resources allocated by a BEA Tuxedo application such as shared memory, message queues, and semaphores. The command is normally run after an unusual error situation such as a failed shutdown. Under normal conditions, the BEA Tuxedo system cleans up all allocated IPC resources when it is shut down. The IPC resources that are removed include those used by the core BEA Tuxedo system and the Workstation component.</p> <p>tmipcrm works only on the local server machine; it does not clean up IPC resources on the remote machines in a BEA Tuxedo configuration. The name of the <code>TUXCONFIG</code> file must be specified, either as the value of the <code>TUXCONFIG</code> environment variable or on the command line. The <code>TUXCONFIG</code> file must exist and it must be readable.</p> <p>Only an administrator or someone with the proper permissions can run this command successfully. The command assumes that it can attach to the shared memory segments that store the bulletin board (BB), and attempts to remove the IPC resources stored in the bulletin board and referenced from it. Such removal attempts may fail due to other conditions on the system. If detected, such failures are reported.</p> <p>The following options are supported.</p> <ul style="list-style-type: none"><li>-y<br/>Answers yes to all prompts.</li><li>-n<br/>Does not remove IPC resources. Instead, displays a list of IPC resources on stdout and exits.</li></ul> <p><i>TUXCONFIG_file</i><br/>Complete path name of the <code>TUXCONFIG</code> file. If not specified, the default is the value of the <code>TUXCONFIG</code> environment variable.</p> |
| Diagnostics | <p>If the <code>TUXCONFIG</code> file cannot be found, a fatal error occurs and the following message is displayed.</p> <pre>Cannot open tuxconfig file</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

If the TUXCONFIG file cannot be read, a fatal error occurs and the following message is displayed.

```
Execute permission denied, not application administrator
```

If an attempt to attach to the bulletin board shared memory fails, a fatal error occurs and the following message is displayed.

```
Cannot attach to BB!
```

**Examples** The command generally runs in an interactive mode, and prompts the user for responses to questions when necessary. If the `-y` option is specified, `tmipcrm` does not prompt the user, but assumes the answer to every question is *yes*. If the `-n` option is specified, `tmipcrm` simply displays a list of IPC resources (on `stdout`) and exits; no IPC resources are removed.

The following example demonstrates how this command is used typically.

```
$ tmipcrm /home/user/apps/tuxconfig
Looking for IPC resources in TUXCONFIG file
/home/user/apps/tuxconfig
The following IPC resources were found:

Message Queues:
0x2345
0x3456
...

Semaphores:
0x34567
0x45678
...

Shared Memory:
0x45678
0x56789
...

Remove these IPC resources (y/n)? : y
Removing IPC resources ... done!
```

The following example code prints a list of the IPC resources for a BEA Tuxedo application on a local machine in a file called `ipclist`.

```
tmipcrm -n /home/user/apps/tuxconfig >ipclist
```

# tmloadcf(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | tmloadcf - parse a UBBCONFIG file and load binary TUXCONFIG configuration file                                                                                                                                                                                                                                                                                                                                                                                       |
| Synopsis    | tmloadcf [-n] [-y] [-c] [-b <i>blocks</i> ] { <i>UBBCONFIG_file</i>   -}                                                                                                                                                                                                                                                                                                                                                                                             |
| Description | tmloadcf reads a file or the standard input that is in UBBCONFIG syntax, checks the syntax, and optionally loads a binary TUXCONFIG configuration file. The TUXCONFIG and (optionally) TUXOFFSET environment variables point to the TUXCONFIG file and (optional) offset where the information should be stored. tmloadcf can be run only on the MASTER machine, as defined in the RESOURCES section of the UBBCONFIG file, unless the -c or -n option is specified. |

**Note:** The path name specified for the TUXCONFIG environment variable must match exactly (including case) the path name specified for TUXCONFIG parameter within the MACHINES section of the UBBCONFIG file. Otherwise, tmloadcf(1) cannot be run successfully.

tmloadcf prints a warning message if it finds any section of the UBBCONFIG file missing, other than a missing NETWORK section in a configuration for which the LAN OPTION is not specified (see UBBCONFIG(5)) or a missing ROUTING section. If a syntax error is found while parsing the input file, tmloadcf exits without performing any updates to the TUXCONFIG file.

The effective user identifier of the person running tmloadcf must match the UID, if specified, in the RESOURCES section of the UBBCONFIG file.

The -c option to tmloadcf causes the program to print a list of the minimum IPC resources needed for this configuration. Resource requirements that vary on a per-processor basis are printed for each processor in the configuration. The TUXCONFIG file is not updated.

The -n option to tmloadcf causes the program to do only syntax checking of UBBCONFIG (the text version of the configuration file) without actually updating the TUXCONFIG file.

After checking the syntax, tmloadcf checks whether: (a) the file referenced by TUXCONFIG exists; (b) it is a valid BEA Tuxedo system file system; and (c) it contains TUXCONFIG tables. If these conditions are not true, tmloadcf prompts you to indicate whether you want the command to create and initialize TUXCONFIG.

Initialize TUXCONFIG file: path [y, q]?

Prompting is suppressed if the standard input or output is not a terminal, or if the `-y` option is specified on the command line. Any response other than `y` or `Y` causes `tmloadcf` to exit without creating the configuration file.

If the TUXCONFIG file is not properly initialized, and you have indicated approval, `tmloadcf` creates the BEA Tuxedo system file system and then creates the TUXCONFIG tables. If the `-b` option is specified on the command line, its argument is used as the number of blocks for the device when creating the BEA Tuxedo system file system. If the value of the `-b` option is large enough to hold the new TUXCONFIG tables, `tmloadcf` uses the specified value to create the new file system; otherwise, `tmloadcf` prints an error message and exits. If the `-b` option is not specified, `tmloadcf` creates a new file system large enough to hold the TUXCONFIG tables. The `-b` option is ignored if the file system already exists.

The `-b` option is highly recommended if TUXCONFIG is a raw device (that is, if it is a device that has not been initialized) and should be set to the number of blocks on the raw device. The `-b` option is not recommended if TUXCONFIG is a regular UNIX file.

If it is determined that the TUXCONFIG file has already been initialized, `tmloadcf` ensures that the system described by that TUXCONFIG file is not running. If the system is running, `tmloadcf` prints an error message and exits.

If the system is not running and a TUXCONFIG file already exists, `tmloadcf` prompts you to confirm that the file should be overwritten.

Really overwrite TUXCONFIG file [y, q]?

Prompting is suppressed if the standard input or output is not a terminal or if the `-y` option is specified on the command line. Any response other than `y` or `Y` causes `tmloadcf` to exit without overwriting the file.

If the SECURITY parameter is specified in the RESOURCES section of the configuration file, then `tmloadcf` flushes the standard input, turns off terminal echo, and prompts the user for an application password, as follows.

```
Enter Application Password?
Reenter Application Password?
```

The password is limited to 30 characters. The option to load UBBCONFIG (the text version of the configuration file) via standard input (rather than a file) cannot be used when the SECURITY parameter is turned on. If standard input is not a terminal, that is, if the user cannot be prompted for a password (as with a `here` file, for example), then the `APP_PW` environment variable is accessed to set the application password. If the

APP\_PW environment variable is not set and standard input is not a terminal, then tmloadcf prints an error message, generates a log message, and fails to load the TUXCONFIG file.

Assuming no errors, and if all checks have passed, tmloadcf loads the UBBCONFIG file into the TUXCONFIG file. It overwrites all existing information found in the TUXCONFIG tables.

Note that some values are rounded during the load and may not match when they are unloaded. These include but are not limited to MAXRFT and MAXRTDATA.

|                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Interoperability      | tmloadcf must run on the master node. In an interoperating application, the master node must be running the highest release available.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Portability           | tmloadcf is supported on any platform on which the BEA Tuxedo server environment is supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Environment Variables | The environment variable APP_PW must be set for applications for which the SECURITY parameter is specified and run tmloadcf with something other than a terminal as standard input.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Examples              | <p>To load a configuration file from UBBCONFIG file BB.shm, initialized the device with 2000 blocks.</p> <pre>tmloadcf -b2000 -y BB.shm</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Diagnostics           | <p>If an error is detected in the input, the offending line is printed to standard error along with a message indicating the problem. If a syntax error is found in the UBBCONFIG file or the system is currently running, no information is updated in the TUXCONFIG file and tmloadcf exits with exit code 1.</p> <p>If tmloadcf is run by a person whose effective user identifier does not match the UID specified in the UBBCONFIG file, the following error message is displayed.</p> <pre>*** UID is not effective user ID ***</pre> <p>If tmloadcf is run on a non-master node, the following error message is displayed.</p> <pre>tmloadcf cannot run on a non-master node.</pre> <p>If tmloadcf is run on an active node, the following error message is displayed.</p> <pre>tmloadcf cannot run on an active node.</pre> <p>Upon successful completion, tmloadcf exits with exit code 0. If the TUXCONFIG file is updated, a userlog message is generated to record this event.</p> |

See Also `tmunloadcf(1)`, `UBBCONFIG(5)`

*Administering a BEA Tuxedo Application at Run Time*

# tmshutdown(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | tmshutdown - shut down a set of BEA Tuxedo servers                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Synopsis    | tmshutdown [ <i>options</i> ]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Description | tmshutdown stops the execution of a set of servers or removes the advertisements of a set of services listed in a configuration file. Only the administrator of the bulletin board (as indicated by the <code>UID</code> parameter in the configuration file) or <code>root</code> can invoke the <code>tmshutdown</code> command. <code>tmshutdown</code> can be invoked only on the machine identified as <code>MASTER</code> in the <code>RESOURCES</code> section of the configuration file, or the backup acting as the <code>MASTER</code> , that is, with the <code>DBBL</code> already running (via the <code>master</code> command in <code>tmadmin(1)</code> ). An exception to this is the <code>-P</code> option which is used on partitioned processors (see below). |

With no options, `tmshutdown` stops all administrative, TMS, and gateway servers, and servers listed in the `SERVERS` section of the configuration file named by the `TUXCONFIG` environment variable, and removes the IPC resources associated with them. For each group, all servers in the `SERVERS` section, if any, are shut down, followed by any associated gateway servers (for foreign groups) and TMS servers. Administrative servers are shut down last.

Application servers without `SEQUENCE` parameters are shut down first in reverse order of the server entries in the configuration file, followed by servers with `SEQUENCE` parameters that are shut down from high to low sequence number. If two or more servers in the `SERVERS` section of the configuration file have the same `SEQUENCE` parameter, then `tmshutdown` may shut down these servers in parallel. Each entry in the `SERVERS` section may have an optional `MIN` and `MAX` parameter. `tmshutdown` shuts down all occurrences of a server (up to `MAX` occurrences) for each server entry, unless the `-i` option is specified; using the `-i` option causes individual occurrences to be shut down.

If it is not possible to shut down a server, or remove a service advertisement, a diagnostic is written on the central event log (see `userlog(3c)`). The following is a description of all options:

`-l lmid`

For each group whose associated `LMID` parameter is *lmid*, all servers in the `SERVERS` section associated with the group are shut down, followed by any TMS and gateway servers associated with the group.

`-g grpname`

All servers in the `SERVERS` section associated with the specified group (that is, for which the `SRVGRP` parameter is set to *grpname*) are shut down, followed by all TMS and gateway servers for the group. TMS servers are shut down based on the `TMSNAME` and `TMSCOUNT` parameters for the group entry. For a foreign group, the gateway servers for the associated entry in the `HOST` section are shut down based on `GATENAME` and `GATECOUNT`. Shutting down a gateway implies not only that the process itself is stopped; it also implies that the administrative service for the gateway and all advertised foreign services are unadvertised.

`-i srvid`

All servers in the `SERVERS` section for which the `SRVID` parameter is set to *srvid* are shut down. Do not enter a value for `SRVID` greater than 30,000; this indicates system processes (that is, TMSs or gateway servers) that should only be shut down via the `-l` or `-g` option.

`-s aout`

All servers listed in the `SERVERS` section with the name *aout* are shut down. This option can also be used to shut down TMS and gateway servers.

`-o sequence`

All servers in the `SERVERS` section for which the `SEQUENCE` parameter is set to *sequence* are shut down.

`-S`

All servers listed in the `SERVERS` section are shut down.

`-A`

All administrative servers are shut down.

`-M`

This option shuts down administrative servers on the master machine. The `BBL` is shut down on the `MASTER` machine, and the `BRIDGE` is shut down if the `LAN` option and a `NETWORK` entry are specified in the configuration file. If the `MODEL` is `MP`, the `DBBL` administrative server is shut down.

`-B lmid`

The `BBL` on the processor with the logical name *lmid* is shut down.

`-T grpname`

All TMS servers for the server group for which the `SRVGRP` parameter is set to *grpname* are shut down (based on the `TMSNAME` and `TMSCOUNT` parameters associated with the server group entry).

`-w delay`

Tells `tmshutdown` to suspend all selected servers immediately and waits for shutdown confirmation for only `delay` seconds before forcing the server to shut down by sending a `SIGTERM` and then a `SIGKILL` signal to the server.

Because the `SIGKILL` signal cannot be trapped, any process that receives it is terminated immediately, regardless of the code being executed by the process at that time. Such behavior may cause structural damage to the Bulletin Board if the process being stopped was updating the Bulletin Board when it was terminated.

Note: Servers to which the `-w` option may be applied should not catch the UNIX signal `SIGTERM`.

`-k {TERM | KILL}`

`tmshutdown` suspends all selected servers immediately and forces them to shut down in an orderly fashion (`TERM`) or preemptively (`KILL`).

Because the `SIGKILL` signal cannot be trapped, any process that receives it is terminated immediately, regardless of the code being executed by the process at that time. Such behavior may cause structural damage to the Bulletin Board if the process being stopped was updating the Bulletin Board when it was terminated.

Note: This option maps to the UNIX signals `SIGTERM` and `SIGKILL` on platforms that support them. By default, a `SIGTERM` initiates an orderly shutdown in a BEA Tuxedo server. If `SIGTERM` is reset by an application, the BEA Tuxedo system may be unable to shut down the server.

`-y`

Assumes a `yes` answer to a prompt that asks whether all administrative and server processes should be shut down. (The prompt appears only when the command is entered with none of the limiting options.)

When the `-y` option is specified, all services are unadvertised immediately from the Bulletin Board and any subsequent service calls fail.

`-q`

Suppresses the printing of the execution sequence on standard output. It implies `-y`.

`-n`

The execution sequence is printed, but not performed.

**-R** For migration operations only, shuts down a server on the original processor without deleting its bulletin board entry in preparation for migration to another processor. The **-R** option must be used with either the **-l** or **-g** option (for example, `tmshutdown -l lmid -R`). The **MIGRATE** option must be specified in the **RESOURCES** section of the configuration file.

**-c** Shuts down BBLs even if clients are still attached.

**-H *lmid*** On a uniprocessor, all administrative and applications servers on the node associated with the specified *lmid* are shut down. On a multiprocessor (for example, 3B4000), all PEs are shut down, even if only one PE is specified.

**-P *lmid*** With this option, `tmshutdown` attaches to the bulletin board on the specified *lmid*, ensures that this *lmid* is partitioned from the rest of the application (that is, that it does not have access to the **DBBL**), and shuts down all administrative and application servers. It must be run on the processor associated with the *lmid* in the **MACHINES** section of the configuration file.

The **-l**, **-g**, **-s**, and **-T** options cause **TMS** servers to be shut down; the **-l**, **-g**, and **-s** options cause gateway servers to be shut down; the **-l**, **-g**, **-i**, **-s**, **-o**, and **-S** options apply to application servers; the **-A**, **-M**, and **-B** options apply only to administrative processes. When the **-l**, **-g**, **-i**, **-o**, and **-s** options are used in combination, only servers that satisfy all the qualifications specified are shut down.

If the distributed transaction processing feature is being used such that global transactions are in progress when servers are shut down, transactions that have not yet reached the point at which a commit is logged after a pre-commit are aborted; transactions that have reached the commit point are completed when the servers (for example, **TMS**) are booted again.

**Interoperability** `tmshutdown` must run on the master node. In an interoperating application the master node must be running the highest release available. `tmshutdown` detects and reports configuration file conditions that would lead to the shutting down of Release 4.2 administrative servers on Release 4.1 sites.

**Portability** `tmshutdown` is supported on any platform on which the BEA Tuxedo server environment is supported.

**Diagnostics** If `tmshutdown` fails to shut down a server or if a fatal error occurs, `tmshutdown` exits with exit code 1 and the user log should be examined for further details; otherwise it exits with exit code 0.

If `tmshutdown` is run on an active node that is not the acting master node, a fatal error message is displayed.

```
tmshutdown cannot run on a non acting-master node in an active application.
```

If shutting down a process would partition active processes from the DBBL, a fatal error message is displayed.

```
cannot shutdown, causes partitioning.
```

If a server has died, the following somewhat ambiguous message is produced.

```
CMDTUX_CAT:947 Cannot shutdown server GRPID
```

**Examples** To shut down the entire system and remove all BEA Tuxedo IPC resources (force it if confirmation is not received in 30 seconds), run the following command.

```
tmshutdown -w 30
```

To shut down only those servers located on the machine for which the value of `lmid` is `CS1`, enter the following command.

```
tmshutdown -l CS1
```

Because the `-l` option restricts the action to servers listed in the `SERVERS` section, the BBL on `CS1` is not shut down.

**Notices** The `tmshutdown` command ignores the hangup signal (`SIGHUP`). If a signal is detected during shutdown, the process continues.

**See Also** `tmadmin(1)`, `tmboot(1)`, `UBBCONFIG(5)`

*Administering a BEA Tuxedo Application at Run Time*

# tmunloadcf(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | tmunloadcf - unload binary TUXCONFIG configuration file                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Synopsis    | tmunloadcf                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Description | <p>tmunloadcf translates the TUXCONFIG configuration file from the binary representation into text format. This translation is useful for transporting the file in a compact way between machines with different byte orderings and backing up a copy of the file in a compact form for reliability. The text format is described in UBBCONFIG(5).</p> <p>tmunloadcf reads values from the TUXCONFIG file referenced by the TUXCONFIG and TUXOFFSET environment variables and writes them to its standard output.</p> <p>Note that some values are rounded during configuration and may not match values set during tmloadcf or via the TMIB interface. These include but are not limited to MAXRFT and MAXRTDATA.</p> |
| Portability | tmunloadcf is supported only on any platform on which the BEA Tuxedo server environment is supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Examples    | <p>Enter the following command to unload the configuration in /usr/tuxedo/tuxconfig into the file tconfig.backup.</p> <pre>TUXCONFIG=/usr/tuxedo/tuxconfig tmunloadcf &gt; tconfig.backup</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Diagnostics | tmunloadcf checks that: (a) the file referenced by the TUXCONFIG and TUXOFFSET environment variables exists; (b) it is a valid BEA Tuxedo system file system; and (c) it contains TUXCONFIG tables. If any of these conditions is not met, tmunloadcf prints an error message and exits with error code 1. Upon successful completion, tmunloadcf exits with exit code 0.                                                                                                                                                                                                                                                                                                                                              |
| See Also    | tmloadcf(1), UBBCONFIG(5)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

*Administering a BEA Tuxedo Application at Run Time*

---

# topendpasswd(1)

- Name** topendpasswd - add a BEA TOP END password for the TOP END Domain Gateway feature of the BEA Tuxedo system
- Synopsis** dmadmin  
> topendpasswd [-r] local\_domain\_name
- Description** topendpasswd can be executed only as a subcommand of dmadmin(1).
- This subcommand prompts the administrator for a new BEA TOP END password for the specified local domain. The -r option specifies that existing passwords and new passwords should be encrypted using a new key generated by the system. The password cannot exceed 12 characters. This password is used when sending an RTQ message to a BEA TOP END system on which security is enabled. The user ID that is passed is the DOMAINID of the local domain.
- Before this subcommand can be run, the application must be configured using either the BEA Administration Console, or the tmloadcf(1) and dmloadcf(1) commands. dmadmin may be run on any active node.
- Diagnostics** The dmadmin topendpasswd subcommand exits with a return code of 0 upon successful completion.
- Example** dmadmin  
>topendpasswd ldom  
Enter Local Domain Password:  
Re-enter Local Domain Password:
- See Also** dmadmin(1)

# tpacladd(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | <code>tpacladd</code> - add a new Access Control List on the system                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Synopsis    | <code>tpacladd [-g <i>GID</i>[,<i>GID</i> . . . ]] [-t <i>type</i>] <i>name</i></code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Description | <p>Invoking <code>tpacladd</code> adds a new Access Control List entry to the BEA Tuxedo security data files. This information is used for BEA Tuxedo access control to services, events, and application queues. A BEA Tuxedo configuration with <code>SECURITY</code> set to <code>USER_AUTH</code>, <code>ACL</code>, or <code>MANDATORY_ACL</code> must be created before you can run this command successfully.</p> <p>The following options are available.</p> <p><code>-g <i>GID</i>, . . .</code><br/>A list of one or more existing group's integer identifiers or character-string names. This option indicates which groups have access to the named object. If this option is not specified, an entry is added with no groups.</p> <p><code>-t <i>type</i></code><br/>The type of the object. It can be one of the following: <code>ENQ</code>, <code>DEQ</code>, <code>SERVICE</code>, or <code>POSTEVENT</code>. The default is <code>SERVICE</code>.</p> <p><i>name</i><br/>A unique string of printable characters that specifies the name of a service, event, or application queue for which access is to be granted. It may not contain a colon (:), pound sign (#), or a newline (\n).</p> <p>Before running this command you must: (a) configure the application, using either the graphical user interface or <code>tmloadcf(1)</code>; and (b) set the <code>TUXCONFIG</code> environment variable to point to your <code>TUXCONFIG</code> file. <code>tpacladd</code> must be run on the configuration <code>MASTER</code> if the application is not active. If the application is active, this command can run on any active node.</p> |
| Portability | This command is available on any platform on which the BEA Tuxedo server environment is supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Diagnostics | The <code>tpacladd</code> command exits with a return code of 0 upon successful completion.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

**See Also** `tpacldel(1)`, `tpaclmod(1)`, `tpgrpadd(1)`, `tpgrpdel(1)`, `tpgrpmod(1)`,  
`AUTHSVR(5)`

*Administering a BEA Tuxedo Application at Run Time*

# tpaclcvt(1)

Name      `tpaclcvt` - convert BEA Tuxedo security data files

Synopsis    `tpaclcvt [-u userfile] [-g groupfile]`

Description    `tpaclcvt` checks and converts the existing user file used by one version of AUTHSVR (the version available with BEA Tuxedo Release 5) into the format used by BEA Tuxedo Release 6. It also generates a group file based on `/etc/group` or a similar file.

The following options are available.

`-u userfile`

The name of the BEA Tuxedo user file. If not specified, the user file is not converted.

`-g groupfile`

The name of the group file, normally `/etc/group`. If not specified, the group file is not converted.

Before running this command you must: (a) configure the application, using either the graphical user interface or `tmloadcf(1)`; and (b) set the `TUXCONFIG` environment variable to point to your `TUXCONFIG` file. `tpaclcvt` must be run on the configuration MASTER when the application is not active.

Portability    This command is available on any platform on which the BEA Tuxedo server environment is supported.

See Also      `tpgrpadd(1)`, `tpusradd(1)`, `AUTHSVR(5)`

*Administering a BEA Tuxedo Application at Run Time*

# tpacldel(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | tpacldel - delete an Access Control List                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Synopsis    | tpacldel [-t <i>type</i> ] <i>name</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Description | <p>Invoking <code>tpacldel</code> deletes an existing Access Control List entry from the BEA Tuxedo security data files. A BEA Tuxedo configuration with <code>SECURITY</code> set to <code>USER_AUTH</code>, <code>ACL</code>, or <code>MANDATORY_ACL</code> must be created before you can run this command successfully.</p> <p>The following options are available.</p> <p><code>-t <i>type</i></code><br/> The type of the object. It can be one of the following: <code>ENQ</code>, <code>DEQ</code>, <code>SERVICE</code>, or <code>POSTEVENT</code>. If not specified, the default type is <code>SERVICE</code>.</p> <p><i>name</i><br/> Identifier for the existing ACL entry to be deleted.</p> <p>Before running this command you must: (a) configure the application, using either the graphical user interface or <code>tmloadcf(1)</code>; and (b) set the <code>TUXCONFIG</code> environment variable to point to your <code>TUXCONFIG</code> file. <code>tpacldel</code> must be run on the configuration <code>MASTER</code> if the application is not active. If the application is active, this command can run on any active node.</p> |
| Portability | This command is available on any platform on which the BEA Tuxedo server environment is supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Diagnostics | The <code>tpacldel</code> command exits with a return code of 0 upon successful completion.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| See Also    | <p><code>tpacladd(1)</code>, <code>tpaclmod(1)</code>, <code>AUTHSVR(5)</code></p> <p><i>Administering a BEA Tuxedo Application at Run Time</i></p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

# tpaclmod(1)

Name `tpaclmod` - modify an Access Control List on the system

Synopsis `tpaclmod [-g GID[,GID...]] [-t type] name`

Description Invoking `tpaclmod` modifies an Access Control List entry in the BEA Tuxedo security data files, replacing the group identifier list. This information is used for BEA Tuxedo access control to services, events, and application queues. A BEA Tuxedo configuration with `SECURITY` set to `USER_AUTH`, `ACL`, or `MANDATORY_ACL` must be created before you can run this command successfully.

The following options are available.

`-g GID, ...`

A list of one or more existing group's integer identifiers or character-string names. This option indicates which groups have access to the named object. If this option is not specified, the entry is modified to have no groups.

`-t type`

The type of the object. It can be one of the following: `ENQ`, `DEQ`, `SERVICE`, or `POSTEVENT`. The default is `SERVICE`.

*name*

An existing ACL name.

Before running this command you must: (a) configure the application, using either the graphical user interface or `tmloadcf(1)`; and (b) set the `TUXCONFIG` environment variable to point to your `TUXCONFIG` file. `tpaclmod` must be run on the configuration `MASTER` if the application is not active. If the application is active, this command can run on any active node.

Portability This command is available on any platform on which the BEA Tuxedo server environment is supported.

Diagnostics The `tpaclmod` command exits with a return code of 0 upon successful completion.

See Also `tpacladd(1)`, `tpacldel(1)`, `AUTHSVR(5)`

*Administering a BEA Tuxedo Application at Run Time*

# tpadduser(1)

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name          | tpaddusr - create a BEA Tuxedo password file                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Synopsis      | tpaddusr <i>usrname file</i> [ <i>cltname</i> [ <i>UID</i> ]]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Description   | <p>This command allows an application administrator to create a UNIX System style password file suitable for use with the BEA Tuxedo AUTHSVR(5) server. <code>tpaddusr</code> adds the user <i>usrname</i> to the password file <i>file</i> (the file cannot be <code>/etc/passwd</code>). The administrator is prompted for an initial password to be associated with the user. If necessary, <i>file</i> is created with permissions 0600. <i>cltname</i>, if specified, indicates a further qualifier on the password entry. <i>usrname</i> and/or <i>cltname</i> may be specified as the asterisk (*) which is considered a wildcard by AUTHSVR(5). If specified, <i>UID</i> indicates the numeric user identifier to be returned with a successful authentication of the user. If not specified, <i>cltname</i> and <i>UID</i> default to * and -1, respectively.</p> |
| Notices       | <p>The <i>cltname</i> values <code>tpsystadm</code> and <code>tpsystop</code> are treated specially by AUTHSVR(5) when authentication requests are processed. These <i>cltname</i> values are not matched against wildcard <i>cltname</i> specifications in the password file.</p> <p>Additionally, regardless of the order of addition to the password file, wildcard entries are considered after explicitly specified values. An authentication request is authenticated against only the first matching password file entry.</p>                                                                                                                                                                                                                                                                                                                                       |
| Portability   | This command is available on any platform on which the BEA Tuxedo server environment is supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Compatibility | <p>This command is used to configure users for <code>SECURITY_USER_AUTH</code>. For compatibility with <code>SECURITY_ACL</code> or <code>MANDATORY_ACL</code> (including the ability to migrate to these security levels), the following restrictions should be applied.</p> <ul style="list-style-type: none"> <li>■ User names should be unique and should not include the wild-card character.</li> <li>■ User identifiers should be unique. They should be greater than 0 and less than 128K.</li> <li>■ The filename should be <code>\$APPDIR/tpusr</code>.</li> </ul> <p>These restrictions are enforced by the <code>tpusradd(1)</code> command.</p>                                                                                                                                                                                                               |
| Examples      | The following sequence of command invocations shows how to construct a simple password file.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

## Section 1 - Commands

---

```
$ # 1. Add username foo with wildcard cltname and no UID
$ tpaddusr foo /home/tuxapp/pwfile
$ # 2. Add username foo with cltname bar and UID 100
$ tpaddusr foo /home/tuxapp/pwfile bar 100
$ # 3. Add username foo with tpsysadm cltname and no UID
$ tpaddusr foo /home/tuxapp/pwfile tpsysadm
$ # 4. Add wildcard username with tpsysop cltname and no UID
$ tpaddusr '*' /home/tuxapp/pwfile tpsysop
$ # 5. Add wildcard username with wildcard cltname and no UID
$ tpaddusr '*' /home/tuxapp/pwfile '*'
```

The following table shows the password file entry (indicated by the numbers shown above) used to authenticate various requests for access to the application. N/A indicates that the request is disallowed because the password file does not include an entry against which a match can be attempted.

| Username | Cltname    | Password | Entry |
|----------|------------|----------|-------|
| "foo"    | "bar"      |          | 2     |
| "foo"    | " "        |          | 1     |
| "foo"    | "tpsysadm" |          | 3     |
| "foo"    | "tpsysop"  |          | 4     |
| "guest"  | "tpsysop"  |          | 4     |
| "guest"  | "bar"      |          | 5     |
| "guest"  | "tpsysadm" |          | N/A   |

The following is an example `SERVERS` section entry for an instance of `AUTHSVR` that works with the password file generated above.

```
AUTHSVR SRVGRP=G SRVID=1 RESTART=Y GRACE=0 MAXGEN=2 CLOPT="-A -- -f
/home/tuxapp/pwfile"
```

**See Also** `tpdelusr(1)`, `tpmodusr(1)`, `tpusradd(1)`, `tpusrdel(1)`, `tpusrmod(1)`, `AUTHSVR(5)`

# tpdelusr(1)

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name          | tpdelusr - delete a user from a BEA Tuxedo password file                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Synopsis      | tpdelusr <i>username file [cltname]</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Description   | <p>This command allows an application administrator to maintain a UNIX System style password file suitable for use with the BEA Tuxedo AUTHSVR(5) server. tpdelusr is used to delete the password file entry for the indicated <i>username/cltname</i> combination (the <i>file</i> cannot be <i>/etc/passwd</i>). <i>cltname</i> defaults to '*' if not specified. Wildcards specified for <i>username</i> and/or <i>cltname</i> match only the corresponding wildcard entry in the password file; they are not expanded to all matching entries.</p>                                    |
| Notices       | <p>The <i>cltname</i> values <i>tpsadm</i> and <i>tpsop</i> are treated specially by AUTHSVR(5) when authentication requests are being processed. These <i>cltname</i> values are not matched against wildcard <i>cltname</i> specifications in the password file.</p> <p>Additionally, regardless of the order in which entries are added to the password file, wildcard entries are considered <i>after</i> explicitly specified values. An authentication request is authenticated against only the first matching password file entry.</p>                                            |
| Portability   | This command is available on any platform on which the BEA Tuxedo server environment is supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Compatibility | <p>This command is used to configure users for SECURITY_USER_AUTH. For compatibility with SECURITY_ACL or MANDATORY_ACL (including the ability to migrate to these security levels), the following restrictions should be applied.</p> <ul style="list-style-type: none"> <li>■ User names should be unique and should not include the wild-card character.</li> <li>■ User identifiers should be unique and should be greater than 0 and less than 128K.</li> <li>■ The filename should be \$APPDIR/tpusr.</li> </ul> <p>These restrictions are enforced by the tpusrdel(1) command.</p> |
| See Also      | tpadduser(1), tpmodusr(1), tpusradd(1), tpusrdel(1), tpusrmod(1), AUTHSVR(5)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

*Administering a BEA Tuxedo Application at Run Time*

# tpgrpadd(1)

**name**      `tpgrpadd` - add a new group on the system

**Synopsis**    `tpgrpadd [-g GID] grpname`

**Description**    The `tpgrpadd` command creates a new group definition on the system by adding the appropriate entry to the BEA Tuxedo security data files. This information is used for BEA Tuxedo system authentication with the `AUTHSVR(5)` server and for access control. A BEA Tuxedo configuration with `SECURITY` set to `USER_AUTH`, `ACL`, or `MANDATORY_ACL` must be created before you can run this command successfully.

The following options are available.

`-g GID`

The group identifier for the new group. This group identifier must be a non-negative decimal integer below 16K. *GID* defaults to the next available (unique) identifier greater than 0. Group identifier 0 is reserved for the “other” group.

`grpname`

A string of printable characters that specifies the name of the new group. It may not include a pound sign (#), comma (,), colon (:), or a newline (\n).

Before running this command you must: (a) configure the application, using either the graphical user interface or `tmloadcf(1)`; and (b) set the `TUXCONFIG` environment variable to point to your `TUXCONFIG` file. `tpgrpadd` must be run on the configuration `MASTER` if the application is not active. If the application is active, this command can run on any active node.

**Portability**    This command is available on any platform on which the BEA Tuxedo server environment is supported.

**Diagnostics**    The `tpgrpadd` command exits with a return code of 0 upon successful completion.

**See Also**        `tpgrpdel(1)`, `tpgrpmod(1)`, `tpusradd(1)`, `tpusrdel(1)`, `tpusrmod(1)`, `AUTHSVR(5)`

*Administering a BEA Tuxedo Application at Run Time*

---

# tpgrpdel(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | tpgrpdel - delete a group from the system                                                                                                                                                                                                                                                                                                                                                                                             |
| Synopsis    | tpgrpdel <i>grpname</i>                                                                                                                                                                                                                                                                                                                                                                                                               |
| Description | The <code>tpgrpdel</code> command removes a group definition from the system by deleting the entry for the relevant group from the BEA Tuxedo security data files. It does not, however, remove the group ID from the user file. A BEA Tuxedo configuration with <code>SECURITY</code> set to <code>USER_AUTH</code> , <code>ACL</code> , or <code>MANDATORY_ACL</code> must be created before you can run this command successfully. |

The following options are available.

*grpname*

The name of an existing group to be deleted.

Before running this command you must: (a) configure the application, using either the graphical user interface or `tmloadcf(1)`; and (b) set the `TUXCONFIG` environment variable to point to your `TUXCONFIG` file. `tpgrpdel` must be run on the configuration `MASTER` if the application is not active. If the application is active, this command can run on any active node.

|             |                                                                                                                                                                |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Portability | This command is available on any platform on which the BEA Tuxedo server environment is supported.                                                             |
| Diagnostics | The <code>tpgrpdel</code> command exits with a return code of 0 upon successful completion.                                                                    |
| See Also    | <code>tpgrpadd(1)</code> , <code>tpgrpmod(1)</code> , <code>tpusradd(1)</code> , <code>tpusrdel(1)</code> , <code>tpusrmod(1)</code> , <code>AUTHSVR(5)</code> |

*Administering a BEA Tuxedo Application at Run Time*

# tpgrpmod(1)

Name `tpgrpmod` - modify a group on the system

Synopsis `tpgrpmod [-g GID] [-n name] grpname`

Description The `tpgrpmod` modifies the definition of the specified group by modifying the appropriate entry to the BEA Tuxedo security data files. A BEA Tuxedo configuration with `SECURITY` set to `USER_AUTH`, `ACL`, or `MANDATORY_ACL` must be created before you can run this command successfully.

The following options are available.

`-g GID`

The new group identifier for the group. This group identifier must be a non-negative decimal integer below 16K. Group identifier 0 is reserved for the “other” group.

`-n name`

A string of printable characters that specifies the new name of the group. It may not include a comma (,), colon (:), or a newline (\n).

`grpname`

The current name of the group to be modified.

Before running this command you must: (a) configure the application, using either the graphical user interface or `tmloadcf(1)`; and (b) set the `TUXCONFIG` environment variable to point to your `TUXCONFIG` file. `tpgrpmod` must be run on the configuration MASTER if the application is not active. If the application is active, this command can run on any active node.

Portability This command is available on any platform on which the BEA Tuxedo server environment is supported.

Diagnostics The `tpgrpmod` command exits with a return code of 0 upon successful completion.

See Also `tpgrpadd(1)`, `tpgrpdel(1)`, `tpusradd(1)`, `tpusrdel(1)`, `tpusrmod(1)`, `AUTHSVR(5)`

*Administering a BEA Tuxedo Application at Run Time*

# tpmodusr(1)

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name          | tpmodusr - maintain a BEA Tuxedo system password file                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Synopsis      | tpmodusr <i>usrname file [cltname]</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Description   | <p>This command allows an application administrator to maintain a UNIX System style password file suitable for use with the BEA Tuxedo system AUTHSVR(5) server. A BEA Tuxedo configuration with SECURITY set to USER_AUTH, ACL, or MANDATORY_ACL must be created before you can run this command successfully.</p> <p>tpmodusr is used to modify the password for the indicated user in the password file <i>file</i> (the file cannot be <i>/etc/passwd</i>). The administrator is prompted for a new password to be associated with the user. <i>cltname</i> defaults to '*' if not specified. Wildcards specified for <i>usrname</i> and/or <i>cltname</i> match only the corresponding wildcard entry in the password file; they are not expanded to all matching entries.</p> |
| Notices       | <p>The <i>cltname</i> values <i>tpsysadm</i> and <i>tpsysop</i> are treated specially by AUTHSVR(5) when authentication requests are being processed. These <i>cltname</i> values are not matched against wildcard <i>cltname</i> specifications in the password file.</p> <p>Additionally, regardless of the order in which entries are added to the password file, wildcard entries are considered <i>after</i> explicitly specified values. An authentication request is authenticated against only the first matching password file entry.</p>                                                                                                                                                                                                                                  |
| Portability   | This command is available on any platform on which the BEA Tuxedo server environment is supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Compatibility | <p>This command is used to configure users for SECURITY USER_AUTH. For compatibility with SECURITY ACL or MANDATORY_ACL (including the ability to migrate to these security levels), the following restrictions should be applied.</p> <ul style="list-style-type: none"> <li>■ User names should be unique and should not include the wild-card.</li> <li>■ User identifiers should be unique. They should be greater than 0 and less than 128K.</li> <li>■ The filename should be \$APPDIR/tpusr.</li> </ul> <p>These restrictions are enforced by the <code>tpusrmod(1)</code> command.</p>                                                                                                                                                                                      |

**See Also**    `tpadduser(1)`, `tpdelusr(1)`, `tpusradd(1)`, `tpusrdel(1)`, `tpusrmod(1)`,  
              `AUTHSVR(5)`

*Administering a BEA Tuxedo Application at Run Time*

# tpusradd(1)

|             |                                                                                                                                                                                                                 |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | tpusradd - add a new principal on the system                                                                                                                                                                    |
| Synopsis    | tpusradd [-u <i>UID</i> ] [-g <i>GID</i> ] [-c <i>client_name</i> ] <i>username</i>                                                                                                                             |
| Description | Invoking <code>tpusradd</code> adds a new principal (user or domain) entry to the BEA Tuxedo security data files. This information is used for per-user authentication with the <code>AUTHSVR(5)</code> server. |

Before you can run this command successfully, you must:

- Configure the application, using either the graphical user interface or `tmloadcf(1)`.
- Set the `TUXCONFIG` environment variable to point to your `TUXCONFIG` file.
- Set `SECURITY` to `USER_AUTH`, `ACL`, or `MANDATORY_ACL`.

`tpusradd` must be run on the configuration `MASTER` if the application is not active; if active, this command can run on any active node.

The system file entries created with this command have a limit of 512 characters per line. Specifying long arguments to several options may exceed this limit.

The following options are available.

-u *UID*

The user identification number. *UID* must be a positive decimal integer below 128K. *UID* must be unique within the list of existing identifiers for the application. *UID* defaults to the next available (unique) identifier greater than 0.

-g *GID*

An existing group's integer identifier or character-string name. This option defines the new user's group membership. It defaults to the "other" group (identifier 0).

-c *client\_name*

A string of printable characters that specifies the client name associated with the user. If specified, it generally describes the role of the associated user, and provides a further qualifier on the user entry. It may not contain a colon (:)

or a newline (\n). If not specified, the default is the wildcard '\*' which will authenticate successfully for any client name specified.

*usrname*

A string of printable characters that specifies the new login name of the user. It may not contain a colon (:), pound sign (#), or a newline (\n). The user name must be unique within the list of existing users for the application.

The administrator is prompted for an initial password to be associated with the user.

See AUTHSVR (5) for further information about per-user authentication and configuring administrator permissions.

**Portability** This command is available on any platform on which the BEA Tuxedo server environment is supported.

**Diagnostics** The `tpusradd` command exits with a return code of 0 upon successful completion.

**Examples** The following sequence of command invocations shows the construction of a simple user file.

```
$ # 1. Add username foo with cltname bar and UID 100
$ tpusradd -u 100 -c bar foo
$ # 2. Add username foo with tpsysadm cltname and no UID
$ tpusradd -c tpsysadm foo
```

The following table shows the user entry (indicated by the numbers shown above) used to authenticate various requests for access to the application and the associated UID and GID. N/A indicates that the request is disallowed because there is no entry in the user file against which a match can be attempted.

| Username | Cltname    | Password | Entry | Uid | Gid  |
|----------|------------|----------|-------|-----|------|
| "foo"    | "bar"      | 2        |       | 100 | 0    |
| "foo"    | " "        | 1        |       | 1   | 0    |
| "foo"    | "tpsysadm" | 3        |       | 0   | 8192 |
| "guest"  | "tpsysadm" | N/A      |       | N/A | N/A  |

The following is an example "SERVERS" section entry for an instance of AUTHSVR that works with the user file generated above.

```
AUTHSVR SRVGRP=G SRVID=1 RESTART=Y GRACE=0 MAXGEN=2 CLOPT="-A"
```

**See Also** `tpgrpadd(1)`, `tpgrpdel(1)`, `tpgrpmod(1)`, `tpusrdel(1)`, `tpusrmod(1)`, `AUTHSVR(5)`

*Administering a BEA Tuxedo Application at Run Time*

---

# tpusrdel(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | tpusrdel - delete a user from the system                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Synopsis    | tpusrdel <i>usrname</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Description | <p>The <code>tpusrdel</code> command deletes a principal (user or domain name) definition from the system. It removes the definition of the specified user. <i>usrname</i> specifies the existing user name to be deleted.</p> <p>Before you can run this command successfully, you must:</p> <ul style="list-style-type: none"><li>■ Configure the application, using either the graphical user interface or <code>tmloadcf(1)</code>.</li><li>■ Set the <code>TUXCONFIG</code> environment variable to point to your <code>TUXCONFIG</code> file.</li><li>■ Set <code>SECURITY</code> to <code>USER_AUTH</code>, <code>ACL</code>, or <code>MANDATORY_ACL</code>.</li></ul> <p><code>tpusradd</code> must be run on the configuration <code>MASTER</code> if the application is not active. If the application is active, this command can run on any active node.</p> |
| Portability | This command is available on any platform on which the BEA Tuxedo server environment is supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Diagnostics | The <code>tpusrdel</code> command exits with a return code of 0 upon successful completion.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| See Also    | <code>tpgrpadd(1)</code> , <code>tpgrpdel(1)</code> , <code>tpgrpmod(1)</code> , <code>tpusradd(1)</code> , <code>tpusrmod(1)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

# tpusrmod(1)

Name `tpusrmod` - modify user information on the system

Synopsis `tpusrmod [-u UID ] [-g GID] [-c client_name] [-l new_login] [-p] usrname`

Description Invoking `tpusrmod` modifies a principal (user or domain) entry to the BEA Tuxedo security data files. This information is used for BEA Tuxedo system authentication with the `AUTHSVR(5)` server.

Before you can run this command successfully, you must:

- Configure the application, using either the graphical user interface or `tmloadcf(1)`.
- Set the `TUXCONFIG` environment variable to point to your `TUXCONFIG` file.
- Set `SECURITY` to `USER_AUTH`, `ACL`, or `MANDATORY_ACL`.

`tpusradd` must be run on the configuration `MASTER` if the application is not active. If the application is active, this command can run on any active node.

The system file entries created with this command have a limit of 512 characters per line. Specifying long arguments to several options may exceed this limit.

The following options are available.

`-u UID`

The new user identification number. *UID* must be a positive decimal integer below 128K. *UID* must be unique within the list of existing identifiers for the application.

`-g GID`

An existing group's integer identifier or character-string name. It redefines the user's group membership.

`-c client_name`

A string of printable characters that specifies the new client name for the user. It may not contain a colon (`:`) or a newline (`\n`).

*-l new\_login*

A string of printable characters that specifies the new login name of the user. It may not contain a colon (:), pound sign (#), or a newline (\n). The user name must be unique within the list of existing users for the application. This option also implies the *-p* option to re-set the password.

*-p*

*tpusrmod* modifies the password for the indicated user. The administrator is prompted for a new password to be associated with the user.

*usrname*

A string of printable characters that specifies the name of an existing user to be modified.

See *AUTHSVR(5)* for further information about per-user authentication and configuring administrator permissions.

**Portability** This command is available on any platform on which the BEA Tuxedo server environment is supported.

**Diagnostics** The *tpusrmod* command exits with a return code of 0 upon successful completion.

**See Also** *tpgrpadd(1)*, *tpgrpdel(1)*, *tpgrpmod(1)*, *tpusradd(1)*, *tpusrdel(1)*, *AUTHSVR(5)*

*Administering a BEA Tuxedo Application at Run Time*

# tuxadm(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | tuxadm - BEA Tuxedo Web GUI CGI gateway                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Synopsis    | <code>http://cgi-bin/tuxadm[TUXDIR=tuxedo_directory   INIFILE=initialization_file][other_parameters]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Description | <p>tuxadm is a common gateway interface (CGI) process used to initialize the Web GUI from a browser. As shown in the “Synopsis” section, this program can be used only as a location, or URL from a Web browser; normally it is not executed from a standard command-line prompt. Like other CGI programs, tuxadm uses the <code>QUERY_STRING</code> environment variable to parse its argument list.</p> <p>tuxadm parses its arguments and finds a Web GUI initialization file. If the <code>TUXDIR</code> parameter is present, the initialization file is taken to be <code>\$TUXDIR/udataobj/webgui/webgui.ini</code> by default. If the <code>INIFILE</code> option is present, then the value of that parameter is taken to be the full path to the initialization file. Other parameters may also be present.</p> <p>Any additional parameters can be used to override values in the initialization file. See the <code>wlisten</code> reference page for a complete list of initialization file parameters. The <code>ENCRYPTBITS</code> parameter may not be overridden by the tuxadm process unless the override is consistent with the values allowed in the actual initialization file.</p> <p>The normal action of tuxadm is to generate, to its standard output, HTML commands that build a Web page that launches the Web GUI applet. The general format of the Web page is controlled by the <code>TEMPLATE</code> parameter of the initialization file, which contains arbitrary HTML commands, with the special string <code>%APPLET%</code> on a line by itself in the place where the Web GUI applet should appear. Through the use of other parameters from the initialization file (such as <code>CODEBASE</code>, <code>WIDTH</code>, <code>HEIGHT</code>, and so on) a correct <code>APPLET</code> tag is generated that contains all the parameters necessary to create an instance of the Web GUI.</p> |
| Errors      | <p>tuxadm generates HTML code that contains an error message if a failure occurs. Because of the way CGI programs operate, there is no reason to return an error code of any kind from tuxadm.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| See Also    | <code>tuxwsvr(1)</code> , <code>wlisten(1)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

# tuxwsvr(1)

**Name** tuxwsvr - Mini Web Server for use with BEA Tuxedo Web GUI

**Synopsis** tuxwsvr -l *nlsaddr* [-d *device*] [-L *logfile*] [-F]  
-i *initialization\_file*

**Description** tuxwsvr is a World Wide Web server process that can be used to support the BEA Tuxedo Web GUI by customers who do not have a commercial Web server or a public-domain Web server on the machine on which the BEA Tuxedo Web GUI processes are running. tuxwsvr places itself in the background when invoked unless otherwise specified, and continues running until the machine shuts down or the tuxwsvr process is killed using an operating system command.

tuxwsvr contains all functionality necessary to support the BEA Tuxedo Web GUI, but does not include many features present in commercial Web servers, such as preforked processes, server-side HTML includes (.shtml files), default directory indexes, and https connections. (Note, however, that the BEA Tuxedo Web GUI can be run in secure mode without an https connection since it implements its own encryption protocol.) For performance reasons, the generic Web server does not perform reverse DNS lookups for received requests.

The following command-line options are used by tuxwsvr.

-l *nlsaddr*

Network address at which the process listens for connections. TCP/IP addresses may be specified in the following forms.

"//*hostname:port\_number*"

"//#. #. #. #:*port\_number*"

In the first format, tuxwsvr finds an address for *hostname* using the local name resolution facilities (usually DNS). *hostname* must be the local machine, and the local name resolution facilities must unambiguously resolve *hostname* to the address of the local machine.

In the second example, the dotted decimal format (#. #. #. #) is used. In dotted decimal format, each # should be a number from 0 to 255. This dotted decimal number represents the IP address of the local machine.

In both of the above formats, *port\_number* is the TCP port number at which the `tlisten` process will listen for incoming requests. *port\_number* can either be a number between 0 and 65535 or a name. If *port\_number* is a name, then it must be found in the network services database on your local machine. The address can also be specified in hexadecimal format when preceded by the characters “0x”. Each character after the initial “0x” is either a number between 0 and 9, or a letter between A and F (case insensitive). The hexadecimal format is useful for arbitrary binary network addresses such as IPX/SPX or TCP/IP. The address can also be specified as an arbitrary string. For example, string addresses are used in STARLAN networks.

`-d device`

Full path name of the network device. For Release 6.4 or higher, this parameter is optional. For prior releases, it should be used if required by the underlying network provider (for example, `tcp`).

`-L logfile`

Prefix of the name of the file used by `tuxwsvr` to log Web requests and error messages. The actual name of the logfile is formed by adding a seven-character string (`.mmddy`—indicating the month, day, and year) to this prefix. If this option is not specified, the Web server log file prefix is `WB` in the current directory. The first log message written on each successive day that the `tuxwsvr` process runs is written to a new file.

`-F`

Specifies that `tuxwsvr` should run in the foreground rather than placing itself in the background. This option is mainly useful for testing and debugging. (The `tuxwsvr` process automatically runs in the background unless otherwise specified; the trailing ampersand (&) on the command line is not required.)

`-i initialization_file`

An initialization file must be specified on every `tuxwsvr` command line. The command-line option that lets you do so is `-i`. The following section describes the format of an initialization file.

Initialization File  
Format

An initialization file contains mappings to directories needed by the Web server and, possibly, some comment lines. (The latter are marked by # signs at the beginning of the line.) Each non-comment line consists of three fields separated by white space.

## Initialization File Format

| Field | Contents                                                                                                                                   |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------|
| 1     | Either HTML or CGI, indicating the type of files (HTML files or executable CGI programs) residing in the directory described in this line. |
| 2     | A path prefix. (If a particular request matches more than one prefix, the first matching prefix mapping in the file is chosen.)            |
| 3     | The directory or file to which the path prefix (in Field 2) is mapped.                                                                     |

The last non-comment line in the initialization file must have a prefix of '/'. If any line prior to the last non-comment line in the initialization file has a prefix of '/', a warning message is generated.

### A Note about Changing the Initialization File

The initialization file is read once at startup time. Thus, if you make any changes to this file, you must stop and restart `tuxwsvr` before your changes will take effect.

### Example of a UNIX System Initialization File

The following is an example of an initialization file for a UNIX system.

```
CGI /cgi-bin /home/tuxedo/udataobj/webgui/cgi-bin
CGI /webgui /home/tuxedo/udataobj/webgui/cgi-bin
HTML /java /home/tuxedo/udataobj/webgui/java
HTML /doc /home/tuxedo/doc
HTML / /home/tuxedo/udataobj/webgui
```

Suppose the Web server is running on port 8080 on the following machine.

```
tuxmach.acme.com
```

Enter a request to either of the following URLs.

```
http://tuxmach.acme.com:8080/cgi-bin/tuxadm?TUXDIR=/home/tuxedo
http://tuxmach.acme.com:8080/webgui/tuxadm?TUXDIR=/home/tuxedo
```

Your request has two effects:

- It invokes the following program.
 

```
/home/tuxedo/udataobj/webgui/tuxadm
```
- It sets the environment variable `QUERY_STRING` to `TUXDIR=/home/tuxedo` in the program, as stated in the World Wide Web CGI specification.

Note that it is not a good idea to specify `$TUXDIR/bin` as a value for an initialization file CGI directory since doing so makes it possible for Web users to invoke any BEA Tuxedo executable. (Such users cannot, however, see the output from executables other than `tuxadm` since these other executables are not written as CGI programs.)

Also, note that in the previous example the first HTML line is redundant since the second HTML line maps subdirectories of `/java` to the same path. Nevertheless, we have included this line since some users might wish to place their java class files in a location other than the one in which they have stored their HTML documents.

### Example of a Windows NT Initialization File

The following is an example of an initialization file for a Windows NT system.

```
HTML /tuxedo/webgui D:\\tuxedo\\htmldocs
CGI /cgi-bin C:\\cgi-bin
HTML /java D:\\tuxedo\\udataobj\\webgui\\java
HTML / D:\\tuxedo\\udataobj\\webgui
```

Suppose the Web server is running on port 80 on machine `ntsvr1`. Enter the following URL.

```
http://ntsvr1/tuxedo/webgui/page1.html
```

The following file is retrieved.

```
D:\\tuxedo\\htmldocs\\page1.html
```

Presumably this file is a customer-created page that invokes the Web GUI.

### Termination

There is only one way to achieve a normal termination of a `tuxwsvr` process: by sending it a `SIGTERM` signal.

### Recommended Use

In the current release of the BEA Tuxedo system the `tuxwsvr` process is provided as a Web server for the BEA Tuxedo administrative GUI for those customers who do not have a commercial Web server. On UNIX systems, we recommend adding a command line of the following format to UNIX initialization scripts so that the Web server will be started automatically.

```
TUXDIR=tuxdir_path_name $TUXDIR/bin/tuxwsvr -l nlsaddr -i initialization_file
```

*tuxdir\_path\_name* represents the full path name of the location of the BEA Tuxedo system software for that application. *nlsaddr* is the network-dependent address to be used by this `tuxwsvr` process.

One alternative method for starting the `tuxwsvr` process is to start it manually using the command line recommended above. A second alternative is to use `cron` jobs to start the `tuxwsvr` process periodically (daily, or perhaps even more often). Duplicate `tuxwsvr` command invocations using the same network address terminate automatically and gracefully log an appropriate message.

#### Network Addresses

The only restriction on the network address specified for the `tuxwsvr` process by the application administrator is that it be a unique address on the specified network. For a STARLAN network, a recommended address of `uname.tuxwsvr` usually yields a unique name. For TCP/IP, the address is formed from a unique port selected by the application administrator paired with the node identifier for the local machine, that is, `0x0002ppppnnnnnnnn`. Unique port values for a particular machine (`pppp`) need to be negotiated among users of that network/machine combination; higher port numbers tend to be better since lower numbers are frequently used for system-related services. The appropriate value for the node field (`nnnnnnnn`) can be found in the `/etc/hosts` file by completing the following steps.

1. Enter `uname -n`

The system returns `node_name`.

2. Enter `grep node_name /etc/hosts`

The system returns `182.11.108.107 node_name`.

3. Convert the dot notation into eight hexadecimal digits.

#### Examples of Network Addresses

Suppose the local machine on which the `tuxwsvr` is being run is using TCP/IP addressing. The machine is named `backus.company.com` and its address is `155.2.193.18`. Further suppose that the port number at which the `tuxwsvr` should accept requests is `2334`. Assume that port number `2334` has been added to the network services database under the name `bankapp-tuxwsvr`. The address specified by the `-l` option can be represented in any of several ways:

- `//155.2.193.18:bankapp-tuxwsvr`
- `//155.2.193.18:2334`
- `//backus.company.com:bankapp-tuxwsvr`
- `//backus.company.com:2334`
- `0x0002091E9B02C112`

The last line shows how to represent the address in hexadecimal format: 0002 is the first part of a TCP/IP address, 091E is a hexadecimal translation of the port number 2334, and 9B02C112 is the hexadecimal translation of the IP address, 155.2.193.18. (In the latter translation, 155 becomes 9B, 2 becomes 02, and so on.)

For a STARLAN network, a recommended address of *uname.tuxsvr* usually yields a unique name.

See Also `tuxadm(1)`, `wlisten(1)`

# txrpt(1)

**Name** txrpt - BEA Tuxedo system server/service report program

**Synopsis** txrpt [-t] [-n *names*] [-d *mm/dd*] [-s *time*] [-e *time*]

**Description** txrpt analyzes the standard error output of a BEA Tuxedo system server to provide a summary of service processing time within the server. The report shows the number of times dispatched and average elapsed time in seconds of each service in the period covered. txrpt takes its input from the standard input or from a standard error file redirected as input. Standard error files are created by servers invoked with the `-r` option from the `servopts(5)` selection; the file can be named by specifying it with the `-e servopts` option. Multiple files can be concatenated into a single input stream for txrpt. Options to txrpt have the following meaning.

-t

Order the output report by total time usage of the services, with those consuming the most total time printed first. If this option is not specified, the report is ordered by total number of invocations of a service.

-n *names*

Restrict the report to those services specified by *names*. *names* is a comma-separated list of service names.

-d *mm/dd*

Limit the report to service requests on the month (*mm*) and day (*dd*) specified. The default is the current day.

-s *time*

Restrict the report to invocations starting after the time given by the *time* argument. The format for *time* is `hr[:min[:sec]]`.

-e *time*

Restrict the report to invocations that finished before the specified *time*. The format for *time* is the same as the format of `-s` flag.

The report produced by txrpt covers only a single day. If the input file contains records from more than one day, the `-d` option controls the day reported on.

## Section 1 - Commands

---

**Notices** Make sure that the `ULOGDEBUG` variable is not set to `y` when a server is collecting statistics for analysis via `txrpt`. Debugging messages in the file will be misinterpreted by `txrpt`.

**Examples** For the following command line

```
txrpt -nSVC1 -d10/15 -s11:01 -e14:18 newr
```

the report produced looks like the following.

```
START AFTER: Thu Oct 15 11:01:00 1992
END BEFORE: Thu Oct 15 14:18:00 1992
 SERVICE SUMMARY REPORT

SVCNAME 11a-12n 13p-14p 14p-15p TOTALS
 Num/Avg Num/Avg Num/Avg Num/Avg

SVC1 2/0.25 3/0.25 1/0.96 6/0.37

TOTALS 2/0.25 3/0.25 1/0.96 6/0.37
```

The above example shows that `SVC1` was requested a total of six times within the specified period and that it took an average of 0.37 seconds to process the request.

**See Also** `servopts(5)`

# ud, wud(1)

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | ud, wud - BEA Tuxedo driver program                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Synopsis    | <pre>ud [-p] [-ddelay] [-eerror_limit] [-r] [-ssleeptime] [-ttimeout]   [-n] [-u {n   u   j}] [-Uusrname] [-Ccltname] [-Sbuffersize] ud32 [options] wud [options] wud32 [options]</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Description | <p>ud reads an input packet from its standard input using <code>Fextread()</code> (see <code>Fextread</code>, <code>Fextread32(3fm1)</code> for details). The packet must contain a field identified as the name of a service. The input packet is transferred to an FML fielded buffer (FBFR) and sent to the service. If the service that receives the FBFR is one that adds records to a database, ud provides a method for entering bulk fielded data into a database known to the BEA Tuxedo system.</p> <p>By using flags (see “Input Format”) to begin the lines of the input packet, you can use ud to test BEA Tuxedo services.</p> <p>By default, after sending the FBFR to the service, ud expects a return FBFR. The sent and reply FBFRs are printed to ud’s standard output; error messages are printed to standard error.</p> <p>ud32 uses FML32 buffers of type FBFR32.</p> <p>wud and wud32 are versions of ud and ud32 built using the Workstation libraries. On sites that support only Workstation, only the wud and wud32 commands are provided.</p> |
| Options     | <p>ud supports the following options.</p> <ul style="list-style-type: none"> <li>-p Suppress printing of the fielded buffers that were sent and returned.</li> <li>-d Expect a delayed reply for every request. <i>delay</i> specifies the maximum delay time, in seconds, before timeout. If timeout occurs, an error message is printed on <code>stderr</code>. If ud receives reply messages for previous requests within the delay time, they are indicated as delayed RTN packets. Hence, it is possible to receive more than one reply packet within a delay time interval. The -d option is not available for wud on DOS operating systems.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                             |

- `-e error_limit`  
ud stops processing requests when errors exceed the limit specified in *error\_limit*. If no limit is specified, the default is 25.
- `-r`  
ud should not expect a reply message from servers.
- `-s sleeptime`  
Sleep between sends of input buffers. *sleeptime* is the time, in seconds, of the sleep.
- `-t timeout`  
ud should send requests in transaction mode. *timeout* is the time, in seconds, before the transaction is timed out. The `-d delay` and `-r` (no reply) options are not allowed in combination with the `-t` option.
- `-u {n|u|j}`  
Specify how the request buffer is modified before reading each new packet. The `n` option indicates that the buffer should be reinitialized (treated as new). The `u` option indicates that the buffer should be updated with the reply buffer using `Fupdate()`. The `j` option indicates that the reply buffer should be joined with the request buffer using `Fojoin()`. (See `Fupdate`, `Fupdate32(3fml)` and `Fojoin`, `Fojoin32(3fml)` for details.)
- `-n`  
Reinitialize the buffer before reading each packet (that is, treat each buffer as a new buffer). This option is equivalent to `-un` and is maintained for compatibility.
- `-U usrname`  
Use *usrname* as the user name when joining the application.
- `-S buffersize`  
If the default buffer size is not large enough, the `-S` option can be used to raise the limit. The value of *buffersize* can be any number up to `MAXLONG`.

The `-d delay` and `-r` options are mutually exclusive.

**Input Format** Input packets consist of lines formatted as follows.

```
[flag]fldname fldval
```

*flag* is optional. If *flag* is not specified, a new occurrence of the field named by *fldname* with value *fldval* is added to the fielded buffer. If *flag* is specified, it should be one of the following.

- + Occurrence 0 of *fldname* in FBFR should be changed to *fldval*.
- Occurrence 0 of *fldname* should be deleted from FBFR. The tab character is required; *fldval* is ignored.
- = The value in *fldname* should be changed. In this case, *fldval* specifies the name of a field whose value should be assigned to the field named by *fldname*.
- # The line is treated as a comment; it is ignored.

If *fldname* is the literal value `SRVCNM`, *fldval* is the name of the service to which FBFR is to be passed.

Lengthy field values can be continued on the next line by putting a tab at the beginning of the continuation line.

A line consisting only of the newline character ends the input and sends the packet to `ud`.

If an input packet begins with a line consisting of the character `n`, followed by the newline character, the FBFR is reinitialized. FBFR reinitialization can be specified for all packets with the `-un` option on the command line.

To enter an unprintable character in the input packet, use the escape convention followed by the hexadecimal representation of the desired character. (For details, see `ASCII(5)` in a UNIX reference manual.) An additional backslash is needed to protect the escape from the shell. A space, for example, can be entered in the input data as `\20`. `ud` recognizes all input in this format, but its greatest usefulness is for non-printing characters.

#### Processing Model

Initially, `ud` reads a fielded buffer from its standard input and sends it to the service whose name is given by the *fldval* of the line in which *fldname* equals `SRVCNM`. Unless the `-r` option is selected, `ud` waits for a reply fielded buffer. After obtaining the reply, `ud` reads another fielded buffer from the standard input. In so doing, `ud` retains the returned buffer as the current buffer. This means that the lines on the standard input that form the second fielded buffer are taken to be additions to the buffer just returned. That is, the default action is for `ud` to maintain a current buffer whose contents are added to by a set of input lines. The set is delimited by a blank line. `ud` may be instructed to discard the current buffer (that is, to reinitialize its FBFR structure), either by specifying the `-un` option on the command line, or by including a line in which the

only character is the letter `n`, specified as the first line of an input set. `ud` may be instructed to merge the contents of the reply buffer into the request buffer by specifying either the `-uu` option (`Fupdate` is used) or the `-uj` option (`Fojoin` is used).

- Security** If `ud` is run in a security application, it requires an application password to access the application. If standard input is a terminal, `ud` prompts the user for the password with echo turned off on the reply. However, since `ud` accepts bulk input on standard input, standard input is typically a file rather than a terminal. In this case, the password is retrieved from the environment variable `APP_PW`. If this environment variable is not specified and an application password is required, then `ud` fails.
- Portability** These commands are supported on any platform on which the BEA Tuxedo server environment is supported.
- Environment Variables** `FLDTBLDIR` and `FIELDTBLS` must be set and exported. `FLDTBLDIR` must include `$TUXDIR/udataobj` in the list of directories. `FIELDTBLS` must include `Usysflds` as one of the field tables.
- `APP_PW` must be set to the application password in a security application if standard input is not from a terminal. `TPIDATA` must be set to the application-specific data necessary to join the application in a security application with an authentication server if standard input is not from a terminal.
- `WSNADDR`, `WSDEVICE`, and optionally `WSTYPE` must be set if access is from a workstation. See `compilation(5)` for more details on setting environment variables for client processes.
- Diagnostics** `ud` fails if it cannot become a client process, if it cannot create the needed `FBFRs`, or if it encounters a UNIX system error. It also fails if it encounters more than 25 errors in processing a stream of input packets. These can be syntax errors, missing service names, errors in starting or committing a transaction, timeouts, and errors in sending the input `FBFR` or in receiving the reply `FBFR`.
- Notices** The final fielded buffer in the input stream should be terminated by a blank line.
- Examples**
- ```
$ud <EOF>
SRVCNM      BUY
CLIENT      J. Jones
ADDR        21 Valley Road
STOCK       AAA
SHARES      100
<CR>
+SRVCNM     SELL
+STOCK      XXX
+SHARES     300
```

```

STOCK      YYY
SHARES    150
<CR>
n
SRVCNM     BUY
CLIENT     T. Smith
ADDR       1 Main Street
STOCK      BBB
SHARES     175
<CR>
+SRVCNM    SELL
+STOCK     ZZZ
+SHARES    100
<CR>
EOF
$

```

In this example, `ud` first sends a fielded buffer to the service `BUY` with the `CLIENT` field set to `J. Jones`, the `ADDR` field set to `21 Valley Road`, the `STOCK` field set to `AAA`, and the `SHARES` field set to `100`.

When the fielded buffer is returned from the `BUY` service, `ud` uses the next set of lines to change `SRVCNM` to `SELL`, `STOCK` to `XXX`, and `SHARES` to `300`. Also, it creates an additional occurrence of the `STOCK` field with value `YYY` and an additional occurrence of the `SHARES` field with a value of `150`. This fielded buffer is then sent to the `SELL` service (the new value of the `SRVCNM` field).

When `SELL` sends back a reply fielded buffer, `ud` discards it by beginning the next set of lines with a line containing only the character `n`. `ud` then begins building an entirely new input packet with `SRVCNM` set to `BUY`, `CLIENT` set to `T. Smith`, and so on.

See Also `Fextread`, `Fextread32(3fml)`, `compilation(5)`

`ascii(5)` in a UNIX system reference manual

Programming a BEA Tuxedo Application Using C

Programming a BEA Tuxedo Application Using FML

Administering a BEA Tuxedo Application at Run Time

uuidgen(1)

Name **uuidgen** - generate a Universal Unique Identifier (UUID)

Synopsis **uuidgen** [-o *filename*] [{-i | -n *number*}] [-v] [-h] [-?]

Description **uuidgen**, by default, generates a Universal Unique Identifier (UUID) on the standard output. The UUID is used to uniquely identifier an IDL interface definition. The format for a UUID string consists of eight hexadecimal digits followed by a dash, followed by three groups of four hexadecimal digits separated by dashes, followed by a dash and twelve hexadecimal digits. (For samples of this format, see “Examples.”)

The following `uuidgen(1)` options are supported.

`-i`

Produces an IDL file template, including a UUID string. (See “Examples” for samples of the file format.) This option cannot be specified with the `-n` option.

`-n number`

Generates the specified number of UUID strings. This option cannot be specified with the `-i` option.

`-o filename`

Redirects the output to the specified file.

`-v`

Displays the version number for `uuidgen(1)` but does not generate a UUID string.

`-h` or `-?`

Displays help information for `uuidgen(1)`.

Network Address The generation of the UUID requires the availability of a 48-bit IEEE 802 address. Because this type of address is not available in all environments and the method of determination is not portable, several methods are available for use with the BEA Tuxedo system version of `uuidgen`.

- If the `NADDR` environment variable is set to a value of the form

num.num.num.num

it is assumed to be an Internet-style address and converted accordingly.

- Otherwise, if the WSNADDR environment variable is set to a value of the form `0xxxxxxxxxxxxxxxxxxx` it is assumed to be a hexadecimal network address, as used in Workstation.
- Otherwise, if not DOS, the `uname` for the machine is used to look up the machine entry in `/etc/hosts` to get the Internet-style address.
- Otherwise, a warning is printed and `00.00.00.00` is used.

Note that in each of these cases, a 32-bit address is formed and the remainder of the address (for 48-bits) is treated as `00.00`.

Diagnostics `uuidgen` exits with a non-zero exit code if an invalid command-line option is specified, or if it cannot open the output file. A warning is printed if an invalid network address value is given and the value `00.00.00.00` is used.

Examples To generate a UUID string, enter the following command.

```
uuidgen
23C67E00-71B6-11C9-9DFC-08002B0ECEf1
```

To generate an IDL template for developing an interface definition, enter the following command.

```
uuidgen -i
[uuid(B5F8DB80-3CCA-14F8-1E78-930269370000)]
interface INTERFACE
{
}
```

To generate two UUID strings, enter the following command.

```
uuidgen -n 2
C0B37080-3CCA-14F8-265F-930269370000
C0B37081-3CCA-14F8-2CDB-930269370000
```

See Also `tidl(1)`

viewc, viewc32(1)

- Name** viewc, viewc32 - view compiler for BEA Tuxedo views
- Synopsis** viewc [-n] [-d *viewdir*] [-C] *viewfile* [*viewfile* . . .]
viewc32 [-n] [-d *viewdir*] [-C] *viewfile* [*viewfile* . . .]
- Description** viewc is a view compiler program. It takes a source viewfile and produces:
- A binary file, which is interpreted at run time to effect the actual mapping of data between FML buffers and C structures
 - One or more header files
 - Optionally COBOL copy files. (When viewc is executed a C compiler must be available.)

viewc32 is used for 32-bit FML. It uses the FIELDTBLS32 and FLDTBLDIR32 environment variables.

The *viewfile* is a file containing source view descriptions. More than one *viewfile* can be specified on the viewc command line as long as the same VIEW name is not used in more than one *viewfile*.

By default, all views in the *viewfile* are compiled and two or more files are created: a view object file (with a .v suffix) and a C header file (with a .h suffix). The name of the object file is *viewfile.v* in the current directory unless an alternate directory is specified through the -d option. C header files are created in the current directory.

If the -c option is specified, then one COBOL copy file is created for each VIEW defined in the *viewfile*. These copy files are created in the current directory.

At viewc compile time, the compiler matches each *fieldid* and field name specified in the *viewfile* with information obtained from the field table file, and stores mapping information in an object file for later use. Therefore, it is essential to set and export the environment variables FIELDTBLS and FLDTBLDIR to point to the related field table file. For more information on FIELDTBLS and FLDTBLDIR, see *Programming a BEA Tuxedo Application Using FML* and *Programming a BEA Tuxedo Application Using C*.

If the `viewc` compiler cannot match a field name with its `fieldid` because either the environment variables are not set properly or the field table file does not contain the field name, a warning message, `Field not found`, is displayed.

With the `-n` option, it is possible to create a view description file for a C structure that is not mapped to an FML buffer. *Programming a BEA Tuxedo Application Using C* discusses how to create and use such an independent view description file.

The following options are interpreted by `viewc`.

`-n` Used when compiling a view description file for a C structure that does not map to an FML buffer. It informs the view compiler not to look for FML information.

`-d viewdir` Used to specify that the view object file is to be created in a directory other than the current directory.

`-C` Used to specify that COBOL copy files are to be created.

Environment Variables

`CC` `viewc` normally uses the default C language compilation command to produce the client executable. The default C language compilation command is defined for each supported operating system platform and is defined as `cc(1)` for UNIX system. In order to allow for the specification of an alternate compiler, `viewc` checks for the existence of an environment variable named `CC`. If `CC` does not exist in `viewc`'s environment, or if it is the string "", `viewc` will use the default C language compiler. If `CC` does exist in the environment, its value is taken to be the name of the compiler to be executed.

CFLAGS

The environment variable `CFLAGS` is taken to contain a set of arguments to be passed as part of the compiler command line. If `CFLAGS` does not exist in `viewc`'s environment, or if it is the string "", no compiler command-line arguments are added by `buildclient`.

Portability

The output view file is a binary file that is machine and compiler dependent. It is not possible to generate a view on one machine with a specific compiler and use that view file on another machine type or with a compiler that generates structure offsets differently (for example, with different padding or packing).

When a view file description file is compiled on DOS or OS/2, the name of the object file has a `.vv` suffix instead of a `.v` suffix because the file names are not case dependent. The following additional options are recognized.

`-m { m | l }`

Specifies the memory model to be used for compilation and linking of a client. The supported values for this option are `m` and `l` for the medium and large memory models, respectively. The large memory model is the default for this option. The `-m` option is supported for DOS only.

`-c { m | b }`

Specifies the C compilation system to be used. The supported value for this option is `m` for the Microsoft C compiler. The Microsoft C compiler is the default for this option. The `-c` option is supported for DOS and WINDOWS only.

`-1 filename`

Specifies that pass 1 should be run, and the resulting batch file should be called `filename.bat` should be created. After this file is created, it should be executed before running pass 2. Using pass 1 and pass 2 increases the size of the views that can be compiled.

`-2 filename`

Specifies that pass 2 should be run to complete processing, using the output from pass 1.

See Also Introduction to FML Functions

Programming a BEA Tuxedo Application Using C

viewdis, viewdis32(1)

Name viewdis, viewdis32 - view disassembler for binary viewfiles

Synopsis viewdis viewobjfile . . . viewdis32 viewobjfile . . .

Description viewdis disassembles a view object file produced by the view compiler and displays view information in viewfile format. In addition, it displays the offsets of structure members in the associated structure.

One or more *viewobjfiles* (with a .v suffix) can be specified on the command line. By default, the *viewobjfile* in the current directory is disassembled. If this is not found, an error message is displayed.

Because the information in the *viewobjfile* was obtained from a match of each `fieldid` and field name in the viewfile with information in the field table file, it is important to set and export the environment variables `FIELDTBLS` and `FLDTBLDIR`.

The output of `viewdis` looks the same as the original view description(s), and is mainly used to verify the accuracy of the compiled object view descriptions.

`viewdis32` is used for 32-bit FML. It uses the `FIELDTBLS32` and `FLDTBLDIR32` environment variables.

See Also `viewc`, `viewc32(1)`

Programming a BEA Tuxedo Application Using FML

wlisten(1)

Name `wlisten` - BEA Tuxedo Web GUI listener process

Synopsis `wlisten [-i initialization_file]`

Description `wlisten` is a listener process that receives incoming connections from Web GUI applets and starts a Web GUI gateway process (`wgated`). All `wlisten` options are taken from an initialization file that is specified by the `-i` option. If the `-i` option is not given, then `$TUXDIR/udataobj/webgui/webgui.ini` is used as the default initialization file. The format and parameters allowed in the initialization file are described below. A default initialization file is generated during system installation.

`wlisten` places itself in the background when invoked (unless the initialization file contains the `FOREGROUND=Y` parameter), and continues running until the machine shuts down or the `wlisten` process is killed through an operating system command.

The following command-line option is used by `wlisten`.

`-i initialization_file`

Specifies that `wlisten` should use the *initialization_file* specified for parameters used during Web GUI sessions. The format of the initialization file is specified below. Most parameters of the initialization file are set to reasonable values when the BEA Tuxedo system is installed. If this option is not specified on the command line, then the default initialization file location is `$TUXDIR/udataobj/webgui/webgui.ini`.

Initialization File The initialization file specified by the `-i` option contains parameters that allow the applet, `wlisten` process, and gateway process to coordinate certain pieces of configuration information necessary for the connection and subsequent operation of the Web GUI.

Most of the parameters in the initialization file are configured when the BEA Tuxedo system is installed. Other parameters may be added automatically when the Web GUI is being run, in response to user input. For example, if you connect to a domain, the GUI adds a listing for that domain to the initialization file. The next time you use the pull-down Domain menu (above the Power Bar in the main GUI window), you will see the new domain listed. Do not be alarmed if you notice that lines have been added or changed in your initialization file without your having explicitly edited the file.

The initialization file consists of commentary lines (blank lines or lines beginning with the # character) and keyword lines. Keyword lines are of the form *keyword=value*. The following list describes valid keywords and values for them.

TUXDIR=*directory*

The directory in which the BEA Tuxedo software is installed. There is no default for this parameter; you must assign a value. Note that if the `-i` option is not given to `wlisten`, then TUXDIR must be set in the environment (and normally should be set to the value specified in the initialization file.)

NADDR=*network_address*

Specifies the network address to be used by `wlisten`. There is no default for this parameter; you must assign a value. The format of the network address is the same as that allowed by `tlisten` and other BEA Tuxedo commands. (See “Network Addresses,” below, for a complete description.)

DEVICE=*device*

Specifies the network device to be used by `wlisten`. This variable is optional. For releases prior to v6.4, the default is the empty string, which means that no network device has been selected. (This is appropriate for some systems, such as Microsoft NT.) Use the same value here that you would use for the `-d` option of `tlisten`. On some UNIX systems the value should be `/dev/tcp`. Whether or not you assign this value depends on the operating system.

BACKGROUND=[Y | N]

Specifies whether `wlisten` should run in the foreground. The default is N, meaning that `wlisten` puts itself in the background automatically. The only reason to use this option is for testing and debugging.

WIDTH=*pixels* and HEIGHT=*pixels*

Specifies the width and height, respectively, for the applet. This area is used for password prompting if security is enabled. The defaults are 400 and 150, respectively.

FRAMEWIDTH=*pixels* and FRAMEHEIGHT=*pixels*

Specifies the width and height, respectively, for the main applet window in which administration tasks are performed. The defaults are 750 and 550, respectively.

ENCRYPTBITS=[0 | 40]

Sets the encryption mode used by the gateway and applet connection. The default is 0, meaning there is no encryption. If the 40 option is chosen, then 40-bit RC4 encryption will take place. In this case, a `tlisten` password file

must exist and authentication must occur in order to exchange encryption keys.

`DOCBASE=`*document_root*

Specifies the document base where the BEA Tuxedo Web GUI help files are found. This parameter is set during installation of the BEA Tuxedo system and, under normal circumstances, it should not be changed afterward. There is no default for this parameter; you must assign a value in the initialization file.

`CODEBASE=`*applet_root*

Specifies the URL for the code base where the BEA Tuxedo Web GUI applet files are found. This parameter is set during installation of the BEA Tuxedo system and, under normal circumstances, it should not be changed afterward. There is no default for this parameter; you must assign a value in the initialization file.

`SNAPDIR=`*snapshot_directory*

Specifies the server directory path in which userlog snapshot files and event log snapshot files are stored. (The value of `SNAPDIR` is a full path name rather than a URL.) It is set during installation of the BEA Tuxedo system and, under normal circumstances, it should not be changed afterward. There is no default for this parameter; you must assign a value in the initialization file.

`SNAPBASE=`*http_root*

Specifies the URL base in which userlog snapshot files and event log snapshot files are stored. (The value of `SNAPBASE` is a URL rather than a full path name.) It is set during installation of the BEA Tuxedo system and, under normal circumstances, it should not be changed afterward. There is no default for this parameter; you must assign a value in the initialization file.

`TEMPLATE=`*template_path*

Specifies the path name of the template file used to deliver the Web GUI applet to the user at startup time. The template file must contain the string `%APPLET%` on a line by itself, which is the place in the file where the Web GUI applet will appear. The rest of the file should be a standard HTML format file that typically contains instructions, a logo, or other information for use by the Web GUI administrator. The default path name is:

```
$TUXDIR/udataobj/webgui/webgui.html
```

`INIFILE=`*init_file*

Specifies the full path for the initialization file to be used by the applet. Under normal circumstances, the initialization file itself is used, but it is technically possible for the applet user to use an initialization file other than the one used

by the gateway process. We do not recommend using an alternative initialization file, however, because if two initialization files are used they must be kept consistent with each other. For example, the `NADDR` and `CODEBASE` parameters, as well as various directory parameters, must be set to identical values, and the value of the `ENCRYPTBITS` parameter must be consistent between the two files. Thus an application in which two files are used is more error prone than an application in which only one is used.

`FLDTBLDIR32=field_table_dir` and `FLDTBLS32=field_tables`

Specifies the field table directories and values, respectively, for use with the Web GUI. These parameters are set to the proper values by the installation program for the BEA Tuxedo system; under normal circumstances they should not be changed later.

Termination	The only way to stop a <code>wlisten</code> process with a normal termination is by sending it a <code>SIGTERM</code> signal.
Recommended Use	<ul style="list-style-type: none"> ■ To ensure automatic starting of the listener—To make sure the Web GUI listener is started automatically, we recommend adding a command line in the following format to your UNIX system initialization scripts. <pre>\$TUXDIR/bin/wlisten -i initialization_file</pre> ■ To start the <code>wlisten</code> process manually, enter the command line shown above after a system prompt. ■ To ensure the administrative password will be found—During the installation process, an administrative password file is created. When necessary, the BEA Tuxedo system searches for this file in the following directories (in the order shown). <pre>APPDIR/.adm/tlisten.pw TUXDIR/udataobj/tlisten.pw</pre> <p>To ensure that your administrative password file will be found, make sure you have set the <code>APPDIR</code> and/or <code>TUXDIR</code> environment variables.</p>
Network Addresses	<p>Suppose the local machine on which <code>wlisten</code> is being run is using TCP/IP addressing. The machine is named <code>backus.company.com</code> and its address is <code>155.2.193.18</code>.</p> <p>Further suppose that the port number at which <code>wlisten</code> should accept requests is <code>2334</code>.</p> <p>Note: Some port numbers may be reserved for the underlying transport protocols (such as TCP/IP) used by your system. Check the documentation for your</p>

transport protocols to find out which numbers, if any, are reserved on your system.

Assume that port number 2334 has been added to the network services database under the name `bankapp-nlsaddr`. The address specified by the `-l` option may be represented in any of several ways:

- `//155.2.193.18:bankapp-nlsaddr`
- `//155.2.193.18:2334`
- `//backus.company.com:bankapp-nlsaddr`
- `//backus.company.com:2334`
- `0x0002091E9B02C112`

The last line shows how to represent the address in hexadecimal format: `0002` is the first part of a TCP/IP address, `091E` is the hexadecimal translation of the port number 2334, and `9B02CU2` is an element-by-element hexadecimal translation of the IP address, `155.2.193.18`. (In the latter translation, 155 becomes `9B`, 2 becomes `02`, and so on.)

For a STARLAN network, a recommended address of `uname.wlisten` usually yields a unique name.

See Also `tuxadm(1)`, `tuxwsvr(1)`