BEA TUXEDO

for OpenVMS User's Guide

BEA TUXEDO Release 6.5
Document Edition 6.5
February 1999

Copyright

Restricted Rights Legend

Trademarks or Service Marks

BEA TUXEDO for OpenVMS User's Guide

Document Edition	Date	Software Version
6.5	February 1999	BEA TUXEDO Release 6.5

Contents

2. Installing the BEA TUXEDO System on OpenVMS

3. Post-Installation

4. Sample Applications

Preface

Purpose of This Document

This document describes the BEA TUXEDO for OpenVMS product and gives instructions for using the tools for building BEA TUXEDO for OpenVMS applications.

Who Should Read This Document

This document is intended for system administrators, network administrators, and developers who are interested in extending secure, scalable, transaction-based processing. It assumes a familiarity with the BEA TUXEDO system and OpenVMS.

How This Document Is Organized

The *BEA TUXEDO for OpenVMS User's Guide* is organized as follows:

- Chapter 1, "Overview," provides an overview of OpenVMS and the BEA TUXEDO system.
- Chapter 2, "Installing the BEA TUXEDO System on OpenVMS," provides installation instructions.
- Chapter 3, "Post-Installation," provides instructions for configuring the BEA TUXEDO system on an OpenVMS platform.

- Chapter 4, “Sample Applications,” contains sample BEA TUXEDO applications for OpenVMS and provides pointers on developing applications.

How to Use This Document

This document is designed primarily as an online, hypertext document. If you are reading this as a paper publication, note that to get full use from this document you should install and access it as an online document via a Web browser.

The following sections explain how to view this document online, and how to print a copy of this document.

Opening the Document in a Web Browser

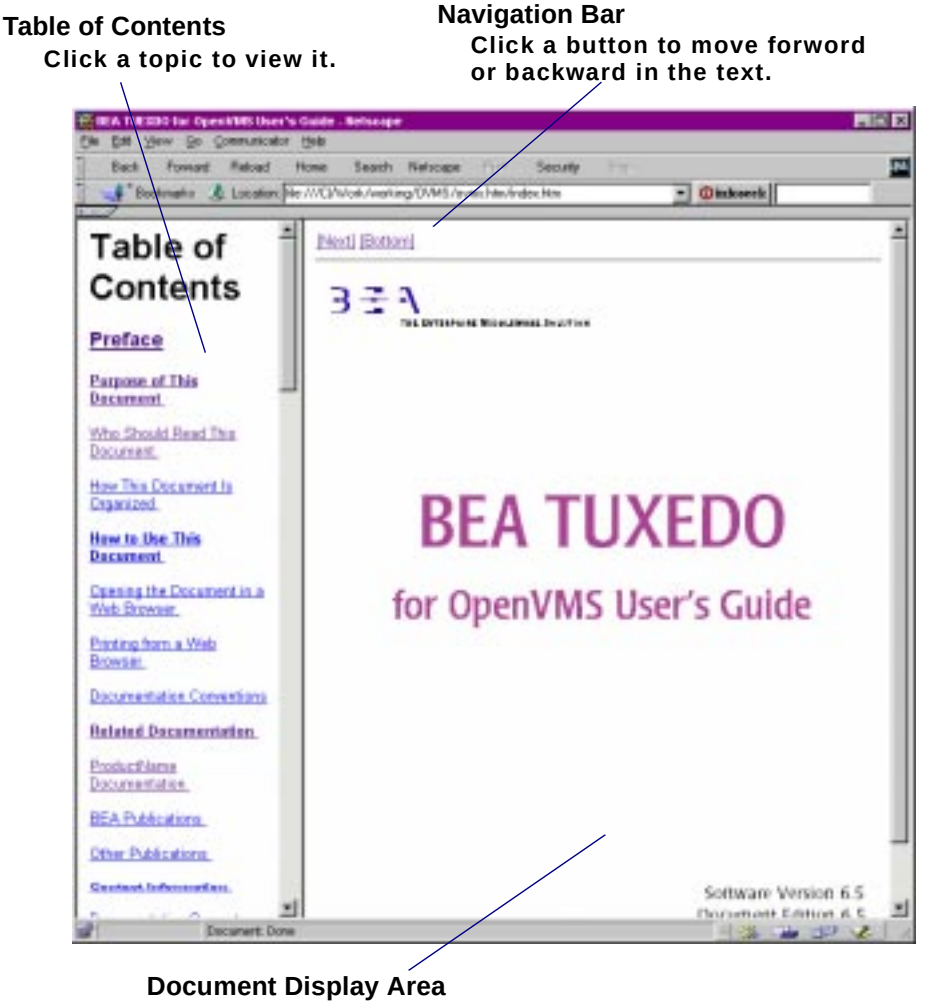
To access the online version of this document, open the following HTML file in a Web browser:

```
/doc/tuxedo/v6_5/ovms/index.htm
```

Note: The online documentation requires a Web browser that supports HTML version 3.0. Netscape Navigator version 2.02 and Microsoft Internet Explorer version 3.0 or later are recommended.

Figure 1 shows the online document with the clickable navigation bar and table of contents.

Figure 1 Online Document Displayed in a Netscape Web Browser



Printing from a Web Browser

You can print a copy of this document, one file at a time, from the Web browser. Before you print, make sure that the chapter or appendix you want is displayed and *selected* in your browser. (To select a chapter or appendix, click anywhere inside the chapter or appendix you want to print. If your browser offers a Print Preview feature, you can use the feature to verify which chapter or appendix you are about to print.)

The BEA TUXEDO Online Documentation CD also includes Adobe Acrobat PDF files of all of the online documents. You can use the Adobe Acrobat Reader to print all or a portion of each document.

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	Item
boldface text	Indicates terms defined in the glossary.
Ctrl+Tab	Indicates that you must press two or more keys sequentially.
italics	Indicates emphasis or book titles.

`monospace text`	Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.
	Examples:
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

Convention	Item
`monospace boldface text`	Identifies significant words in code. *Example*: `void` **`commit`** `( )`
`monospace italic text`	Identifies variables in code. *Example*: `String` *`expr`*
UPPERCASE TEXT	Indicates device names, environment variables, and logical operators. *Example*s: LPT1 SIGNON OR
`{ }`	Indicates a set of choices in a syntax line. The braces themselves should never be typed.

`[ ]`	Indicates optional items in a syntax line. The brackets themselves should never be typed.
	Example:
	`buildobjclient [-v] [-o name ] [-f` *`file-list`*`]...`
	`[-l` *`file-list`*`]...`

`|`	Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed.

`...`	Indicates one of the following in a command line:
	♦ That an argument can be repeated several times in a command line
	♦ That the statement omits additional optional arguments
	♦ That you can enter additional parameters, values, or other information
	The ellipsis itself should never be typed.
	Example:
	`buildobjclient [-v] [-o name ] [-f` *`file-list`*`]...`
	`[-l` *`file-list`*`]...`

. . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.

Related Documentation

The following sections list related documentation.

BEA Publications

The following BEA publications are available on the BEA TUXEDO Online Documentation CD:

BEA TUXEDO Reference Manual

BEA TUXEDO Programmer's Guide

Other Publications

For more information about developing BEA TUXEDO applications, refer to the following books:

The TUXEDO System (Andrade, Carges, Dywer, Felts)

TUXEDO: An Open Approach to OLTP (Primatesta)

Building Client/Server Applications Using TUXEDO (Hall)

Contact Information

The following sections provide information about how to obtain support for the documentation and software.

Documentation Support

If you have questions or comments on the documentation, you can contact the BEA Information Engineering Group by e-mail at **docsupport@beasys.com**. (For information about how to contact Customer Support, refer to the following section.)

Customer Support

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

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

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

CHAPTER

1 Overview

The BEA TUXEDO system is now available on the OpenVMS platform. This book is designed for experienced BEA TUXEDO system developers who are porting applications to OpenVMS.

The topics covered in this chapter include:

- Getting Started
- Configuring the BEA TUXEDO System on OpenVMS
- Developing BEA TUXEDO Applications for OpenVMS
- Networking on OpenVMS
- Using a Web Server on OpenVMS

Getting Started

The following sections answer general questions about the unique characteristics of the BEA TUXEDO system on the OpenVMS platform:

- What hardware and software are required?
- Can I run BEA TUXEDO for OpenVMS with other releases of the BEA TUXEDO system?
- How are files specified?
- What file naming conventions are used?

Platform Requirements

The following table lists the system requirements for BEA TUXEDO for OpenVMS.

Component	Requirement
Hardware	Digital Alpha supported by OpenVMS 7.1
	64 MB of RAM
Software	OpenVMS v7.1
	DEC C v5.6 and/or DEC COBOL v5.2 (for development environment)
Network	Digital TCP/IP Service v4.1 for OpenVMS (UCX 4.1) ECO9

Interoperability

BEA TUXEDO for OpenVMS servers and clients interoperate with any 6.x release of the BEA TUXEDO system. BEA TUXEDO 6.5 must be running on the master node in any multi-processor configuration.

File System Specification

A file specification on the OpenVMS operating system consists of up to seven components, several of which assume a default value if they are not specified. A complete OpenVMS file specification takes the following form.

node::*device*:[*root*.][*directory-name*]*filename*.*type*;*version*

The following table defines each component of an OpenVMS file specification.

Component	Definition
node	Optional cluster name (containing no more than 59 characters), followed by two colons (::).
device	Optional device specification (containing no more than 15 characters), followed by a colon (:).
[*root*]	Optional root directory name (containing no more than 39 characters) within square brackets and followed by a period (.)
[*directory-name*]	The path name for the directory in which the file (specified by *filename*) resides. The path name must be specified within square brackets. Directory names within the path name are separated by periods. The path name may be no more than eight levels deep. Each directory name may be no more than 39 characters. Optional component.
filename	File name (containing no more than 39 characters), followed by a period.
type	Type of the file (containing no more than 39 characters), followed by a semi-colon (;).
version	Version number (containing no more than 5 digits) of the file. The largest version number is the most recent copy of a file.

Note: The entire name cannot be longer than 256 characters. All file names are case insensitive. Thus, for example, the names `UBBCONFIG.DAT` and `ubbconfig.dat` refer to the same file.

The following is an example of a file specification.

```
MYCLUSTER::DKA100:[TUXEDO.APPS.SIMPAPP]UBBCONFIG.DAT;2
```

File specifications end with a semi-colon (`;`) and have a version number. Older versions of a file can be removed with the `PURGE` command.

All BEA TUXEDO files that contain file names (such as `TUXCONFIG`) must use the native OpenVMS format. File specifications can be separated by a comma (,). For example, `APPDIR` might be specified as

```
APPDIR="DKA100:[TUXEDO.APPS.SIMPAPP],DKA100:[TUXEDO.APPS.QSAMP]"
```

File Naming Conventions

OpenVMS has several naming conventions. The following table lists valid file extensions and how they are interpreted by OpenVMS and by the BEA TUXEDO system.

Use This Extension	For This Type of File
.C	C source file
.COB	COBOL source file
.COM	Script containing DCL commands that will be run by the command interpreter
.CPY	COBOL copy file
.EXE	Binary executable file or shared library (executable image)
.H	C header file
.OLB	Binary static object library
.OPT	File in which you can specify options (such as functions to be exported from a library) that will be used by the linker
.VV	Binary view file compiled with viewc(1) or viewc32(1)

Note: The .EXE extension is not required for server names listed in the UBBCONFIG file. If the extension is required to make a server executable, the BEA TUXEDO system will add it to the server name at runtime.

Configuring the BEA TUXEDO System on OpenVMS

The following sections explain the unique characteristics of configuring the BEA TUXEDO system on the OpenVMS platform.

Environment Variables

All environment variables needed by BEA TUXEDO applications should be defined as logical names. For example, the `TUXCONFIG`, `FLDTBLDIR32`, `FIELDTBLS32`, `VIEWDIR32`, and `VIEWFILES32` variables need to be defined as logical names. These names should be specified in the native OpenVMS format and must be defined in the process table of the process running the BEA TUXEDO commands. Most of the processes spawned by BEA TUXEDO utilities inherit logical names from the current process table.

If you are using the BEA TUXEDO Workstation feature, the `WSNADDR` and `WSTYPE` variables must be defined as logical names. All UNIX system environment variables are logical names in OpenVMS.

For example, you can set the `TUXCONFIG` environment variable with the following DCL command:

```
$ DEFINE TUXCONFIG DKA100:[TUXEDO.APPS.SIMPAPP]TUXCONFIG.
```

We recommend setting these logical names in the process table of the process running the BEA TUXEDO commands. Usually you place logical names, such as those of the BEA TUXEDO shared libraries, in the group or system tables. These logical names are relevant to a specific installation of the BEA TUXEDO binaries. You should set logical names that are specific to a particular BEA TUXEDO application in the process table. If desired, you can specify application-specific logical names in any logical table that a detached process will be able to access.

When assigning a value to an environment varible, you may specify more than one directory. Provide the path name of each directory and use a comma (,) to separate path names. For example, in the following line, two directories (`SIMPAPP` and `BANKAPP`) are assigned to the `APPDIR` variable.

```
$ DEFINE APPDIR DKA100:[TUXEDO.SIMPAPP],DKA100:[TUXEDO:BANKAPP]
```

If the SHOW LOGICAL command is run on APPDIR, the output looks similar to the following:

```
$ SHOW LOGICAL APPDIR
"APPDIR" = "DKA100:[TUXEDO.SIMPAPP]" (LNM$PROCESS_TABLE)
  = "DKA100:[TUXEDO.BANKAPP]"
```

The BEA TUXEDO system can interpret logical names that include multiple values.

DCL PATH

The DCL$PATH should include the path names for the BEA TUXEDO installation directory, as shown in the following example:

```
"DCL$PATH" = "DKA100:[WSCOTT.BIN]" (LNM$PROCESS_TABLE)
     = "DKA100:[TUXEDO.BIN]"
     = "SYS$SYSTEM:"
```

Command-line Programs

All upper-case arguments to BEA TUXEDO commands (such as tmboot and tmshutdown) must be enclosed in double quotes on the command line. If they are not, they will be converted to lower case by OpenVMS and misinterpreted by the system.

```
$ tmboot “-A”
```

Any command that might run a shell script is run through the DCL interpreter, and hence should be a proper DCL script. An example of a BEA TUXEDO service that might run scripts is the qmadmin threshold command.

Hostname Utility

You must enter the name by which your machine is known to the network (your network node name) in the MACHINES section of ubbconfig. Use the hostname utility to determine your machine’s network node name. An example of the output of the hostname utility is:

```
lcvms1.beasys.com
```

You must enclose a network node name in quotes when you enter it in the MACHINES section of the ubbconfig file. (Quotes are not required if a network node name does not contain any periods.)

ID Utility

The id utility displays the UID (user ID) and GID (group ID) of a user account. An example of the output of the id utility is:

```
UID=250456, GID=234
```

Use the id utility to determine the values of UID and GID that you must enter in the ubbconfig file.

Resource Manager Information

This section describes four types of information about resource management that are used by a BEA TUXEDO application:

- Resource manager (RM) files
- User log file
- OPENINFO strings
- DMTYPE files

RM Files

When a BEA TUXEDO application needs to identify or locate resources (such as SQL statements, databases, and libraries), the application refers to a resource manager file, or RM file. At build time, when you use the buildclient(1) or buildserver(1) command with the -R option, the command parses the RM file and puts the library names in a temporary option file. This temporary file is appended to the list of option files read by the linker. When the build is complete, the temporary file is removed.

The RM file for OpenVMS platforms is located in the UDATAOBJ directory (which was created when you installed the BEA TUXEDO system).

Each entry in an RM file consists of a list of resources, such as library names. Fields within an entry are separated by commas; items within a field, by blank spaces, as shown in the following sample RM entry (used for the BEA TUXEDO SQL resource manager):

```
TUXEDO/SQL,tuxsql_switch,TUX_LIBSQL/SHARE TUX_LIBRMS/SHARE TUX_LIBFS/SHARE
```

User Log File

The BEA TUXEDO userlog file is written through the OpenVMS Record Management Services (RMS). Because RMS performs record locking, the userlog file may not be available immediately for viewing. BEA TUXEDO warning, error, and informational messages can be found in the userlog file as defined by the appropriate ULOGPFX variable. No messages are written to an OpenVMS log file.

OPENINFO Strings

A BEA TUXEDO application opens a database for transactions by invoking the tpopen(3c) function. tpopen(), in turn, looks up the setting of the OPENINFO string (in the application code) to find out the name and location of the database to be opened.

Fields within the value of the OPENINFO string are separated by commas, as shown in the following example (from the OPENINFO string in the bankapp sample application):

```
OPENINFO="TUXEDO/SQL,DKA100:[BEADEV.APPS.BANKAPP]BANKDL1,BANKDB,readwrite"
```

DMTYPE Files

A BEA TUXEDO application builds a Domains gateway process by invoking the build_dgw(1) command. This command requires, as an argument, a file called DMTYPE, which contains a list of the libraries to be linked to the new gateway.

Each entry in the DMTYPE file contains the name of one or more libraries. Library names are separated by commas. For example, the following line is an entry in the DMTYPE file for GWTDOMAIN (a standard server for BEA TUXEDO Domains):

```
TDOMAIN,TUX_LIBGW/SHARE TUX_LIBNWS/SHARE,,
```

Developing BEA TUXEDO Applications for OpenVMS

The following sections explain the unique characteristics of developing BEA TUXEDO applications on the OpenVMS platform.

Building Clients and Servers

The `buildclient`(1) and `buildserver`(1) utilities are fully supported on the OpenVMS platform. We recommend using these utilities to ensure that the proper options and libraries are used.

When building clients and servers, you must specify which prototypes of the BEA TUXEDO API are to be used.

Table 1-1

If you are using this compiler . . .	Then specify this qualifier . . .

CC	`/prefix=all`
	`/names=as_is`
	`/float=ieee`

COBOL	`/ansi_format`

The `buildclient`(1) and `buildserver`(1) utilities automatically use these qualifiers for any files they compile.

When linking BEA TUXEDO clients and servers, you must include the linker option file, `TUXLIB.OPT` (in the `LIB` directory of the BEA TUXEDO installation) in the link line. Both the `buildclient`(1) and `buildserver`(1) utilities automatically append the `TUXLIB.OPT` file to the link line. Remember that option files must be qualified for the linker with the `/OPT` switch. For an example of how to use an option file in a BEA TUXEDO link line, see the compile and link line in Example 2.

As on the UNIX platform, the CFLAGS logical name allows you to add options to the compile phase of the build. The LINK phase of the OpenVMS build may need different options. You can supply options to the LINK phase of the build with the logical name TMLKFLAGS.

One characteristic of buildclient(1) and buildserver(1) is that they always produce warnings about the BEA TUXEDO libraries. These warnings should be ignored. The BEA TUXEDO libraries have circular references, and when they are built warnings are produced.

By using BEA TUXEDO shared libraries, BEA TUXEDO users on OpenVMS can take advantage of the BEA TUXEDO buffer type switch functionality. See buffer(3c) in the *BEA TUXEDO Reference Manual.*

Example 1

The following example shows how you can run buildclient(1) to create a client called SIMPCL.EXE.

```
$ BUILDCLIENT -f SIMPCL.C -o SIMPCL.EXE
%LINK-W-SHRWRNERS, compilation warnings
   in shareable image file DKA100:[TUXEDO.LIB]LIBTUX_0000.EXE;1
%LINK-W-SHRWRNERS, compilation warnings
   in shareable image file DKA100:[TUXEDO.LIB]LIBBUFT_0000.EXE;1
```

In this example, the linker returns two warning messages about BEA TUXEDO libraries. These messages are not significant; you may ignore them.

Example 2

Option files provide a useful way to specify a large number of files on the link line. In the following example, an option file is used to specify a set of object files—ECHO.OBJ, PROCESS.OBJ, and SECD.OBJ— that have been compiled and will be linked into our server.

```
$ TYPE SECD.OPT
!
! OPTION FILES USE ! in order to denote comment lines
! Any line which starts with ! is ignored by the linker
!
ECHO.OBJ
PROCESS.OBJ
SECD.OBJ
```

```
TUX_LIBTMIB/SHARE
$ BUILDSERVER -f SECD.OPT/OPT -o SECD.EXE -s "SECD:ECHO" -s "PROFILE"
```

Notice the following components of the file:

- TUX_LIBTMIB is a logical name that should point to the BEA TUXEDO LIBTMIB_6400.EXE shared library in the LIB directory of your BEA TUXEDO installation.
- If your server or client needs to access the MIB through tpadmcall(3c), then the TUX_LIBTMIB library should be linked into your client or server.
- The /SHARE qualifier tells the linker that this file is a shared library. (Another option available for the type qualifier is /LIB, which tells the linker to expect a non-shared library file.)
- Double quotes are used to preserve the upper-case spelling of the parameters specified with the -s option ("ECHO" and "PROFILE").

Tips for Calling External Code and Input into your Application

This section provides tips for writing the code in your application for:

- Running executables
- Running DCL scripts
- Redirecting standard I/O and standard error streams

Running Executables

In order to have your application invoke an executable, you must do one of the following:

- Include in your code the definition of a symbolic name for the executable
- Specify the location of the executable through the DCL$PATH variable

To define a symbolic name, enter the name and specify the path for the target executable. The executable must be a DCL script.

For example, to define a symbol for the simpapp client simpcl, enter the following line in your application:

```
$ SIMPCL :== $ DKA100:[TUXEDO.SIMPAPP]SIMPCL.EXE
```

This line defines simpcl as a symbolic name. The SIMPCL.EXE executable can now be run with arguments:

```
$ SIMPCL "Here is a string"
HERE IS A STRING
```

If the DCL$PATH variable includes the directory in which the SIMCL.EXE executable is located, you do not need to define a symbolic name for the executable.

Running DCL Scripts

To run a DCL script from the command line, enter the @ symbol before the name of the script.

Redirecting the Standard I/O and Error Streams

If your code invokes programs that take input from standard input (such as qmadmin or tmadmin), then you will probably want to redirect standard input. The BEA TUXEDO system allows you to do so.

To redirect standard input, standard output, and standard error on an OpenVMS platform, redefine the logical names SYS$STDIN, SYS$STDOUT, and SYS$STDERR.

The following example shows how a qmadmin script generates queue spaces and redirects its output to two files: qmadmin.stdout and qmadmin.stderr.

```
$ TYPE QMADMIN.STDIN
echo
crdl DKA100:[TUXEDO.QSAMPLE]QUE 0 400
qspacecreate
QSPACE
62839
100
6
4
9
3
errque
y
16
```

```
q
$ define sys$input "DKA100:[TUXEDO.QSAMPLE]QMADMIN.STDIN"
$ define sys$output "DKA100:[TUXEDO.QSAMPLE]QMADMIN.STDOUT"
$ define sys$error "DKA100:[TUXEDO.QSAMPLE]QMADMIN.STDERR"
$ qmadmin
$ deassign sys$input
$ deassign sys$output
$ deassign sys$error
$ TYPE CRQUE.STDERR
qmadmin - Copyright (c) 1996 BEA Systems, Inc.
Portions * Copyright 1986-1997 RSA Data Security, Inc.
All Rights Reserved.
Distributed under license by BEA Systems, Inc.
TUXEDO is a registered trademark.
$ TYPE CRQUE.STDOUT
%DCL-I-SUPERSEDE, previous value of SYS$ERROR has been superseded
QMCONFIG=DKA100:[TUXEDO.QSAMPLE]QUE
> Echo is now on
> crdl DKA100:[TUXEDO.QSAMPLE]QUE 0 400
Created device DKA100:[TUXEDO.QSAMPLE]QUE, offset 0, size 400 on
DKA100:[TUXEDO.QSAMPLE]QUE
> qspacecreate

Queue space name: IPC Key for queue space: Size of queue space in disk pages:
Number of queues in queue space: Number of concurrent transactions in queue space:
Number of concurrent processes in queue space: Number of messages in queue space:
Error queue name: Initialize extents (y, n [default=n]): Blocking factor
[default=16]:

> q
```

Networking on OpenVMS

When run on other platforms, BEA TUXEDO utilities require a bridge or a device to be specified on the command line. When you run the same utilities on the OpenVMS platform, however, you should not specify a bridge or a device. If you do so, a warning message will be printed in the userlog file and the value will be ignored.

Using a Web Server on OpenVMS

The BEA TUXEDO Web server, `tuxwsvr`, is not supported in the current release of BEA TUXEDO for OpenVMS. Instead, you should use a commercial product, such as Netscape Commerce Server for OpenVMS. Refer to "Using Netscape Commerce Server" in Chapter 3 for more information.

2 Installing the BEA TUXEDO System on OpenVMS

This chapter explains how to install and configure the BEA TUXEDO system on your OpenVMS system.

The topics covered in this chapter include:

- Preparing to install the BEA TUXEDO system
- Installing the BEA TUXEDO system
- Configuring the BEA TUXEDO system
- Installing the BEA TUXEDO Link-Level Encryption Packages

Preparing to Install the BEA TUXEDO System

Note: Before you begin the installation process, you may wish to review the Polycenter product installation utility (PCSI). This utility is used to install the BEA TUXEDO system and you should have a working knowledge of its operation.

Before installing the BEA TUXEDO system, you must:

- Verify that your machine meets the hardware and software requirements.
- Verify that you (as the person installing the BEA TUXEDO system) have the necessary user ID privileges and quotas.

The following sections provide procedures for verifying this information.

Verifying the Hardware and Software Configuration

Before you install the BEA TUXEDO system, you must verify that the machine on which you wish to install the BEA TUXEDO system meets the minimum hardware and software requirements.

Use the following procedure to verify your machine meets the requirements.

1. Review the hardware and software configuration of the machine on which you wish to install the BEA TUXEDO system.
2. Compare your configuration with the hardware and software requirements described in "Platform Requirements."
3. Verify that the machine on which you wish to install the BEA TUXEDO system has at least 30 megabytes of available disk space.

Verifying the User ID Privileges and Quotas

The person installing the BEA TUXEDO system must log on to the machine with the following user ID privileges and quotas.

User ID Privileges	User ID Quotas
CMKRNL	ASTLM: 400
SETPRV	BIOLM: 500
SYSLCK	DIOLM: 500
SYSNAM	ENGLM: 500

User ID Privileges	User ID Quotas
SYSGBL	FILLM: 500
PRMGBL	BYTLM: 5000000
DETACH	DGFLQUOTA: 30000
	PRCLM: 5
	TQELM: 500
	WSDEFAULT: 512
	WSEXTENT: 8192
	WSQUOTA: 1024

Use the following procedure to verify the user ID privileges.

1. Open the user account file for editing.
2. Enter the following line in the user account file.

   ```
   set proc/priv=all
   ```

Installing the BEA TUXEDO System

1. Log on to the machine using the user ID that you verified in "Verifying the User ID Privileges and Quotas."
2. Mount the BEA TUXEDO CD-ROM using the following command.

```
mount/media_format=cdrom/undefined_fat=(fixed:none:8192) -
/override=identification device
```

   where *device* is equal to the device specification of your CD-ROM.

3. Start the product installation utility (PCSI):

```
product install TUXEDO /source=device:[000000...]/destination=dest_dir
```

where

- *device* is the name of the CD-ROM drive.
- *dest_dir* is the directory in which the BEA TUXEDO system will be installed. *dest_dir* must be a valid directory name.

4. The PCSI utility will ask you about the type of installation you want to perform. Select one or more of the following options:
   - BEA TUXEDO Core package
   - BEA TUXEDO Workstation package
   - BEA TUXEDO Documentation package
5. The PCSI utility will install the required files. You can monitor the progress of the installation on the screen.

Setting Up Your Environment

Before you can use the BEA TUXEDO system, you must set up your environment using a DCL script named `postinstall`. The DCL script performs the following tasks:

- Installs any shared images that will be referenced by BEA TUXEDO system processes at run time
- Sets all the logical names in either the group table or system table, depending on the options selected when the DCL script is run
- Sets the environment for the `vps_daemon` process
- Starts the `vps_daemon` if you have requested this process

Note: This script must be run each time the machine is rebooted. The BEA TUXEDO system administrator may put the DCL script into the start-up environment so that when the machine is rebooted the script will be executed.

Logical Names

Before you can use the BEA TUXEDO system, you must set several logical names. The OpenVMS operating system allows you to set logical names in either the group table or the system table.

Group Table

Setting logical names in the group table gives you the ability to run multiple versions of the BEA TUXEDO system on the same platform. Subsequent versions of the BEA TUXEDO system can be installed under different group tables.

To use this method requires that all users who want to invoke BEA TUXEDO commands must be listed in the group table in which all the required logical names for the BEA TUXEDO system reside.

System Table

Setting logical names in the system table allows you to give all users of a system access to the BEA TUXEDO system. However, subsequent versions of the BEA TUXEDO system cannot be installed simultaneously on the same machine.

vps_daemon Process

The `vps_daemon` process provides the basic infrastructure for BEA TUXEDO client and server processes. BEA TUXEDO client and server processes use the `vps_daemon` process to perform data exchange, synchronization, monitoring, and network communication functions. The `vps_daemon` process also allocates the resources needed for BEA TUXEDO client and server processes to perform these functions.

The `vps_daemon` process requires that the logical name `VPS_INITPATH` point to a valid file containing the configuration parameters for the process. This file, `vps_init.txt`, is located in the `udataobj` directory and should not be modified. The parameters in this file are comparable to the IPC (Inter Process Communication) tunable parameters on the UNIX platform and to the IPC parameters on the Windows NT platform. The default parameters in `vps_init.txt` are capable of supporting approximately 50 BEA TUXEDO servers and 200 BEA TUXEDO clients.

If you want to change the default parameters (or perform other advanced configuration tasks), please consult BEA Support.

BEA TUXEDO License File

The BEA TUXEDO license file, `LIC.TXT`, is installed in the `udataobj` directory of the BEA TUXEDO installation directory. This file should have read permissions for `system`, `user`, `group`, and `world`.

The BEA TUXEDO license file contains the signature that indicates the number of users the installation can support. The BEA TUXEDO system cannot function without this file, which is delivered separately from the BEA TUXEDO CD-ROM.

Configuring the BEA TUXEDO System

This section explains how to configure the BEA TUXEDO system once you have finished installing it.

Preparing for Configuration

Before you begin the configuration procedure, verify that you have the following:

- BEA TUXEDO system license file (`LIC.TXT`)
- `tlisten` password

Configuration Instructions

Use the following procedure to complete the configuration process.

1. Change the current directory to the *`dest_dir`*`.bin` directory, where *`dest_dir`* is the directory you specified during installation.
2. On the command line, enter the following command.

   `set default` *`device`*`:[`*`dest_dir`*`.BIN]`

   where *`device`* is the device specification of your file system and *`dest_dir`* is the directory you specified during installation.

3. On the command line, enter the following command.

   ```
   @postinstall
   ```

4. The DCL script will prompt you for information on configuring the BEA TUXEDO system at the group level or at the system level.

5. If this is not a workstation-only installation, the DCL script will prompt you for the `tlisten` password.

   Note: The DCL script will not echo the password but it will verify the password.

6. The DCL script will prompt you for the location of the license file. The license file, `LIC.TXT`, is installed in the `udataobj` directory of the BEA TUXEDO installation directory. For example, if the BEA TUXEDO system is installed in `DKA100:[TUXDIR]` then the license file is in `DKA100:[TUXDIR.UDATAOBJ]`.

7. The DCL script will prompt you to indicate if the script should start the `vps_daemon` process.

   Note: If you choose not to start the `vps_daemon` process, then you will have to manually start the process. The `vps_daemon.exe` file is located in the `BIN` directory of the BEA TUXEDO installation directory.

Now the BEA TUXEDO system is fully configured and ready to run.

Installing the BEA TUXEDO Link-Level Encryption Packages

The following sections explain how to install link-level encryption packages.

Preparing to Install a Link-Level Encryption Package

Before you install a link-level encryption package, you must do the following.

1. Verify that the BEA TUXEDO system is installed.

2. Verify the full path name of the directory where the BEA TUXEDO system is installed.
3. Stop all application servers and clients.

Installing the 56-bit Link-Level Encryption Package

1. Log on to the machine with the user ID that you used to install the BEA TUXEDO system.
2. Mount the BEA TUXEDO CD-ROM using the following command.

```
mount/media_format=cdrom/undefined_fat=(fixed:none:8192) -
/override=identification device
```

where *device* is equal to the device specification of your CD-ROM.

3. Start the product installation utility (PCSI):

```
product install TUXEDO_ENC40 /source=device:[000000...]/destination=dest_dir
```

where

- *device* is the name of the CD-ROM drive.
- *dest_dir* is the directory in which the BEA TUXEDO system will be installed. *dest_dir* must be a valid directory name.

4. Go to the BIN directory under the BEA TUXEDO installation directory and enter the following command.

```
@inst_enc40
```

Installing the 128-bit Link-Level Encryption Package

1. Log on to the machine with the user ID that you used to install the BEA TUXEDO system.
2. Mount the BEA TUXEDO CD-ROM using the following command.

```
mount/media_format=cdrom/undefined_fat=(fixed:none:8192) -
/override=identification device
```

where *device* is equal to the device specification of your CD-ROM.

3. Start the product installation utility (PCSI):

```
product install TUXEDO_ENC128 /source=device:[000000...]/destination=dest_dir
```

where

- *device* is the name of the CD-ROM drive.
- *dest_dir* is the directory in which the BEA TUXEDO system will be installed. *dest_dir* must be a valid directory name.

4. Go to the `BIN` directory under the BEA TUXEDO installation directory and enter the following command.

```
@inst_enc128
```

CHAPTER

3 Post-Installation

Running tuxenv.com after Rebooting

Before you can use the BEA TUXEDO system, you must configure it using a DCL script. This script must be run each time the machine is rebooted. The BEA TUXEDO system administrator may put the DCL script into the start-up environment so that when the machine is rebooted the script will be executed.

1. Log on to the machine with the user ID that you used to install the BEA TUXEDO system.
2. Change the current directory to the *dest_dir*`.bin` directory, where *dest_dir* is the directory you specified during installation.
3. On the command line, enter the following command.

   ```
   @tuxenv
   ```

Modifying the VPS_INIT.TXT file

Please contact BEA Support for help in modifying the `vps_init.txt` file.

Removing the BEA TUXEDO System

Use the following procedure to remove the BEA TUXEDO system from a machine.

Note: This procedure removes all the files and directories under the BEA TUXEDO installation directory. It does not remove temporary files, which are not part of the Core package and may have been created after installation. Temporary files may have to be removed manually using the `delete` command.

1. Log on to the machine with the user ID that you used to install the BEA TUXEDO system.
2. Change the current directory to the *`dest_dir`*`.bin` directory, where *`dest_dir`* is the directory you specified during installation.
3. On the command line, enter the following command.

   ```
   @tuxenvdel
   ```

4. On the command line, enter the following command.

   ```
   product remove TUXEDO
   ```

Removing the BEA TUXEDO Link-Level Encryption Packages

Use the following procedures to remove a BEA TUXEDO link-level encryption package from a machine.

Removing the 56-bit Link-Level Encryption Package

1. Log on to the machine with the user ID that you used to install the BEA TUXEDO system.

2. Change the current directory to the *dest_dir*.bin directory, where *dest_dir* is the directory you specified during installation.
3. On the command line, enter the following command.

   ```
   @remove_enc40
   ```

4. On the command line, enter the following command.

   ```
   product remove TUXEDO_ENC40
   ```

Removing the 128-bit Link-Level Encryption Package

1. Log on to the machine with the user ID that you used to install the BEA TUXEDO system.
2. Change the current directory to the *dest_dir*.bin directory, where *dest_dir* is the directory you specified during installation.
3. On the command line, enter the following command.

   ```
   @remove_enc128
   ```

4. On the command line, enter the following command.

   ```
   product remove TUXEDO_ENC128
   ```

Copying the License File

Use the following procedure to copy the license file.

1. Log on to the machine with the user ID that you used to install the BEA TUXEDO system.
2. On the command line, enter the following command.

   copy *source_dir*LIC.TXT DKA100:[*dest_dir*.UDATAOBJ]LIC.TXT.

   where

   - *source_dir* is the root directory of the CD-ROM.

- *dest_dir* is the directory under which the BEA TUXEDO system is installed.

Using Netscape Commerce Server

Before you can use Netscape Commerce Server for VMS with the BEA TUXEDO system, you must configure certain directories and their URL mappings. We assume that Netscape Commerce Server is already installed on your system.

Because it will run the `TUXADM.EXE` cgi-bin script, the Web server must have access to the relevant environment variables. If the BEA TUXEDO system is installed as `SYSTEM`, then no further configuration is necessary. However, if the BEA TUXEDO system is installed in a specific group, then Netscape Commerce Server must run under the same group. Otherwise, Netscape Commerce Server will not have the proper variables set up in its environment.

1. Collect the following information about your installation of Netscape Commerce Server.
   - administrative port
   - administrative user name
   - administrative password
2. Connect to the administrative port for the server from your browser. For example, if the administrative port is `8080`, and the server resides on `myvmsbox.mycompany.com`, then the URL for connecting to the administrative port is `http://myvmsbox.mycompany.com:8080/`.
3. Enter the administrative user name and administrative password.
4. Your browser will display a list of installed commerce servers. Click on the link to the commerce server that you would like to configure.
5. Your browser will display a page titled Netscape Server Manager. Scroll down to the link labeled “Map a URL to a local directory” and click on the link.
6. In the URL Prefix field, enter the name of the directory in which the BEA TUXEDO system is installed.

7. In the Directory To Map To field, enter the name of the directory where the file `webguitop.html` is located. In a standard BEA TUXEDO installation, this is `DKA100:[TUXEDO.UDATAOBJ.WEBGUI]`.
8. From the Template for Configuration pull-down menu, select the NONE option.
9. Click on the Make These Changes button.
10. Click on the Map a URL to a Local Directory link.
11. In the URL Prefix field, enter `cgi-bin`.
12. In the Directory To Map To field, enter the full path name of the `BIN` directory of your BEA Tuxedo system installation. For example, if the BEA TUXEDO system is installed at `DKA100:[TUXEDO]`, enter `DKA100:[TUXEDO.BIN]` in the field.
13. From the Template for Configuration pull-down menu, select the `cgi-bin` option.
14. Click on the Make These Changes button.
15. Click on the Map a URL to a Local Directory link.
16. In the URL Prefix field, enter `java`.
17. In the Directory To Map To field, enter the full path name of the Java directory of your BEA Tuxedo system installation. For example, if the BEA TUXEDO system is installed at `DKA100:[TUXEDO]`, enter `DKA100:[TUXEDO.UDATAOBJ.WEBGUI.JAVA]` in the field.
18. From the Template for Configuration pull-down menu, select the NONE option.
19. Click on the Make These Changes button.
20. Restart your Web server.

4 Sample Applications

This chapter provides instructions for running the simpapp sample applications delivered with BEA TUXEDO for OpenVMS. The other sample applications are documented in the Read Me file located in the sample applications directory.

simpapp

This section provides a procedure for building, configuring, running, and shutting down a sample application based on `simapp` (a sample application delivered with the BEA TUXEDO system).

Before you begin, make sure that:

- `TUXDIR` is installed on `DKA100:[TUXEDO]`.
- The logical name `APPDIR` is pointing to `DKA100:[TUXAPP]`.

Now you are ready to begin.

1. Build the client and server executables (`simpcl` and `simpserv`, respectively):
   a. Verify that all the post-installation steps have been performed and that all the logical names and the symbols were created properly. Please see the "Post-Installation" chapter for more information.
   b. Copy all the files from `DKA100:[TUXEDO.APPS.SIMPAPP]` to `DKA100:[TUXAPP]`. The copied files include `SIMPCL.C`, `SIMPSERV.C`, `README`, and `UBBBSIMPLE`.
   c. On the command line, generate the client executable.

      ```
      buildclient -o SIMPCL.EXE -f SIMPCL.C
      ```

d. Define the symbol for SIMPCL.EXE.

```
SIMPCL:==$ DKA100:[TUXAPP]SIMPCL.EXE
```

e. Generate the server program.

```
buildserver -o SIMPSERV.EXE -f SIMPSERV.C -s "TOUPPER"
```

2. Modify UBBSIMPLE.

   a. Assign a unique integer value to IPCKEY.

   b. Assign DKA100:[TUXAPP] to APPDIR.

   c. Assign DKA100:[TUXEDO] to TUXDIR.

   d. Assign DKA100:[TUXAPP]TUXCONFIG to TUXCONFIG.

   e. Run the hostname command on your machine to determine the name of the machine. Then replace the *machine_name* entry with the name of your machine.

   f. Run the getuidgid command on your VMS machine to determine your UID and GID. Then change the UID and GID values in the *MACHINE section.

3. Create a binary version of the UBBSIMPLE configuration file. A file named DKA100:[TUXAPP]TUXCONFIG will be created.

```
$ tmloadcf -y UBBSIMPLE
```

4. Boot the application.

```
$ tmboot -y
```

5. Run the simple client.

```
$ simpcl abc
```

   The simple client calls the TOUPPER service and specifies input: the string abc. TOUPPER converts the string to upper case (ABC) and sends the new string to the client.

6. Shut down the BEA TUXEDO sample application.

```
$ tmshutdown -y
```

Configuration Issues for COBOL simpapp

Keep in mind the following issues when using the `simpapp` sample application on an OpenVMS platform:

- The suffix for the name of a COBOL program is `.cob` instead of `.cbl`. The suffix for the name of a COBOL copy file is `.cpy` instead of `.cbl`.
- To invoke the COBOL compiler, specify the `/ansi_format` parameter, as in

  ```
  cobol /ansi_format csimpcl.cob
  ```

  Use the `buildclient` and `buildserver` commands to build the COBOL client and server executables. Use double quotes around the `-C` option:

```
buildclient "-C" -v -o CSIMPCL.exe  -f CSIMPCL.cob
buildserver "-C" -v -o CSIMPSRV.exe -s "CSIMPSRV" -f CSIMPSRV.cob -f
TPSVRINIT.cob
```

csimpcl.cob

- To make sure that command-line arguments are received, include the following lines.

  ```
  SPECIAL-NAMES.
  SYSERR IS STANDARD-ERR
  ARGUMENT-VALUE IS COMMAND-LINE-ARGUMENT.
  ```

- VMS COBOL uses the following method to call `COPY`.

  ```
  *
    01  TPTYPE-REC.
    COPY "TUX$COBINCLUDE:TPTYPE.CPY".
  *
    01  TPSTATUS-REC.
    COPY "TUX$COBINCLUDE:TPSTATUS.CPY".
  *
    01  TPSVCDEF-REC.
    COPY "TUX$COBINCLUDE:TPSVCDEF.CPY".
  *
    01  TPINFDEF-REC.
    COPY "TUX$COBINCLUDE:TPINFDEF.CPY".
  ```

  `TUX$COBINCLUDE` is defined in `tuxenv.com`.

- The length of `logmsg` needs to be computed.

```
COMPUTE LOGMSG-LEN = FUNCTION LENGTH (LOGMSG)
```

- The format of spaces and tabs (especially leading spaces and tabs) is very important for DEC COBOL programs.

csimpsrv.cob

- OpenVMS uses the following syntax for the `COPY` function.

```
  01  TPSVCRET-REC.
  COPY "tux$cobinclude:TPSVCRET.cpy".
*
  01  TPTYPE-REC.
  COPY "tux$cobinclude:TPTYPE.cpy".
*
  01 TPSTATUS-REC.
  COPY "tux$cobinclude:TPSTATUS.cpy".
*
  01  TPSVCDEF-REC.
  COPY "tux$cobinclude:TPSVCDEF.cpy".
```

  `tux$cobinclude` is defined in `tuxenv.com`.

- The lengths of `LOGMSG` and `RECV-STRING` need to be computed.

```
COMPUTE LOGMSG-LEN = FUNCTION LENGTH (LOGMSG)
COMPUTE LEN = FUNCTION LENGTH (RECV-STRING)
```

- In lines 86 and 103 of `csimpsrv.cob`, use the following method to call `TPRETURN`.

```
COPY "tux$cobinclude:TPRETURN.cpy" REPLACING
DATA-REC BY RECV-STRING.
```

  `tux$cobinclude` is defined in `tuxenv.com`.

- The format of spaces and tabs (especially leading spaces and tabs) is very important for DEC COBOL programs.

tpsvrinit.cob

- Use the following method to call `COPY`.

```
COPY "TUX$COBINCLUDE:TPSTATUS.CPY"
COPY "TUX$COBINCLUDE:TPSTATUS.CPY"
```

  `tux$cobinclude` is defined in `tuxenv.com`.

- The length of `LOGMSG` needs to be computed.

  ```
  COMPUTE LOGMSG-LEN = FUNCTION LENGTH (LOGMSG)
  ```

- The format of spaces and tabs (especially leading spaces and tabs) is very important for DEC COBOL programs.