Oracle Tuxedo for OpenVMS

Installation Guide

11g Release 1 (11.1.1.2.0)

August 2010

ORACLE

Oracle Tuxedo OpenVMS Users Guide, 11*g* Release 1 (11.1.1.2.0)

Overview

Installing Oracle Tuxedo on OpenVMS

Post-Installation

Sample Applications

CHAPTER 1

Overview

Oracle Tuxedo is now available on the OpenVMS platform. This book is designed for experienced Oracle Tuxedo developers who are porting applications to OpenVMS.

The topics covered in this chapter include:

- Getting Started
- Configuring Oracle Tuxedo on OpenVMS
- Developing Oracle Tuxedo Applications for OpenVMS
- Networking on OpenVMS

Getting Started

The following sections answer general questions about the unique Oracle Tuxedo characteristics on the OpenVMS platform:

- What hardware and software are required?
- Can I run Oracle Tuxedo for OpenVMS with other Oracle Tuxedo releases?
- How are files specified?
- What file naming conventions are used?

Interoperability

Oracle Tuxedo for OpenVMS servers and clients can interoperate with any Oracle Tuxedo release.

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
```

Table 1-1 defines each component of an OpenVMS file specification.

Table 1-1 OpenVMS Components

Component	Definition
node	Optional cluster name (containing no more than 59 characters), followed a dollar sign ($).
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 Oracle 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. Table 1-2 lists valid file extensions and how they are interpreted by OpenVMS and Oracle Tuxedo.

Table 1-2 Valid File Extensions

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, Oracle Tuxedo adds it to the server name at runtime.

Configuring Oracle Tuxedo on OpenVMS

The following sections explain the unique characteristics of configuring Oracle Tuxedo on the OpenVMS platform.

Logical Names

All environment variables needed by Oracle 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 Oracle Tuxedo commands. Most of the processes spawned by Oracle Tuxedo utilities inherit logical names from the current process table.

If you are using the Oracle 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 Oracle Tuxedo commands. Usually you place logical names, such as those of the Oracle Tuxedo shared libraries, in the group or system tables. These logical names are relevant to a specific installation of the Oracle Tuxedo binaries. You should set logical names that are specific to a particular Oracle 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 variable, 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]"
```

Oracle Tuxedo can interpret logical names that include multiple values. Path-related directives in the UBBCONFIG file should be formatted in UPPER CASE.

DCL PATH

The `DCL$PATH` should include the path names for the Oracle 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 Oracle 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 Oracle 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 the UBBCONFIG file. Use the hostname utility to determine your machine node name. An example of the output of the hostname utility is:

```
lcvms1.us.oracle.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 an Oracle Tuxedo application:

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

RM Files

When an Oracle 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 (created when you installed Oracle Tuxedo).

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 Oracle Tuxedo SQL resource manager):

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

User Log File

The Oracle 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. Oracle Tuxedo warning, error, and informational messages can be found in the userlog file as defined by the appropriate `ULOGPFX` variable.

OPENINFO Strings

A Oracle 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:[OracleDEV.APPS.BANKAPP]BANKDL1,BANKDB,readwri
te"
```

DMTYPE Files

An Oracle 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 Oracle Tuxedo Domains):

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

Developing Oracle Tuxedo Applications for OpenVMS

The following sections explain the unique characteristics of developing Oracle 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 Oracle Tuxedo API are to be used.

Table 1-3

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.

Note: Since only 64-bit binary generating on OpenVMS platform is currently supported, `buildclient(1)` and `buildserver(1)` default to add option like `"/po=lo=argv"` to compiler. You should follow this rule to create your own `obj` files as well as to link with Oracle Tuxedo.

When linking Oracle Tuxedo clients and servers, you must include the linker option file, `TUXLIB.OPT` (in the `LIB` directory of the Oracle 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 Oracle 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`.

`buildclient(1)` and `buildserver(1)` sometimes produce warnings about the Oracle Tuxedo libraries. These warnings should be ignored. The Oracle Tuxedo libraries have circular references, and when they are built warnings are produced.

By using Oracle Tuxedo shared libraries, Oracle Tuxedo users on OpenVMS can take advantage of the Oracle Tuxedo buffer type switch functionality. See buffer(3c) in the Oracle 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_1030.EXE;1
%LINK-W-SHRWRNERS, compilation warnings
in shareable image file DKA100:[Tuxedo.LIB]LIBBUFT_1030.EXE;1
```

In this example, the linker returns two warning messages about Oracle 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 Oracle Tuxedo `LIBTMIB_1030.EXE` shared library in the LIB directory of your Oracle 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. Oracle Tuxedo 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 Oracle Systems, Inc.
Portions * Copyright 1986-1997 RSA Data Security, Inc.
All Rights Reserved.
Distributed under license by Oracle 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, Oracle 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.

CHAPTER 2

Installing Oracle Tuxedo on OpenVMS

This chapter explains how to install and configure the Oracle Tuxedo system on your OpenVMS system.

The topics covered in this chapter include:

- Preparing to Install Oracle Tuxedo
- Installing Oracle Tuxedo
- Setting Up Your Environment

Preparing to Install Oracle Tuxedo

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

Before installing Oracle Tuxedo, you must:

- Verify that your machine meets the hardware and software requirements.
- Verify that you (as the person installing Oracle Tuxedo) 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 Oracle Tuxedo, you must verify that the machine on which you wish to install Oracle Tuxedo 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 Oracle Tuxedo.
2. Compare your configuration with the hardware and software requirements. For more information, see Oracle Tuxedo 11g Release 1 (11.1.1.2.0) Platform Data Sheets.
3. Verify that the machine on which you wish to install Oracle Tuxedo has at least 300 gigabytes of available disk space.

Verifying the User ID Privileges and Quotas

The person installing Oracle Tuxedo must log on to the machine with the user ID privileges and quotas as listed in Table 2-1

Table 2-1 User ID Privileges and Quotas

User ID Privileges	User ID Quotas
CMKRNL	ASTLM: 400
SETPRV	BIOLM: 500
SYSLCK	DIOLM: 500
SYSNAM	ENGLM: 500
SYSGBL	FILLM: 500
PRMGBL	BYTLM: 5000000

Table 2-1 User ID Privileges and Quotas

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 Oracle Tuxedo

1. Log on to the machine using the user ID that you verified in "Verifying the User ID Privileges and Quotas."
2. Prepare your installation package (PCSI), for example put it under "src_dir" directory.
3. Start the product installation utility (PCSI):

   ```
   product install Tuxedo /source=src_dir/destination=dest_dir
   ```

   where:

   - `src_dir` is the directory in which you put the installation package (PCSI).
   - `dest_dir` is the directory in where Oracle Tuxedo is 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 of the following options:

   1- Full Install

   2- Server Install

3- Full Client Install

4- Jolt Client Install

5- ATMI Client Install

6- Customize ...

Note: full (to install Oracle Tuxedo installation component packages)

server (to install all Oracle Tuxedo server package)

full client (to install only the Oracle Tuxedo all client packages)

jolt client (to install only the Oracle Tuxedo Jolt client)

atmi client (to install only the Oracle Tuxedo ATMI client)

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 Oracle Tuxedo, 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 Oracle Tuxedo 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 Oracle Tuxedo 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 Oracle Tuxedo, 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 Oracle Tuxedo on the same platform. Subsequent versions of the Oracle Tuxedo system can be installed under different group tables.

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

System Table

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

vps_daemon Process

The `vps_daemon` process provides the basic infrastructure for Oracle Tuxedo client and server processes. Oracle 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 Oracle 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 Oracle Tuxedo servers and 200 Oracle Tuxedo clients.

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

Configuring Oracle Tuxedo

This section explains how to configure Oracle Tuxedo once you have finished installing it.

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 Oracle 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 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 Oracle Tuxedo installation directory.
7. The DCL script will prompt you to configure LDAP settings for SSL support.

   Note: if you intend to use SSL encryption in your application, select 'Y' and enter the following LDAP configuration information.

   - LDAP Service Name: The URL of the LDAP server system
   - LDAP PortID: A port number for the URL of the LDAP server system
   - LDAP BaseObject: A base object for search in LDAP server
   - LDAP Filter File Location: The name of LDAP filter file, if the input is null, the default value of `"$TUXDIR.udataobj.security]bea_ldap_filter.dat"` is used

   if you want to use command, epifregedt.exe, to register the LDAP Filter File Location manually, please pay attention its format. The following example shows the correct format and incorrect format:

   correct format:

```
epifreg -u
"filterFileLocation=file:////dka0/tuxedo10gr3/udataobj/security/bea_
ldap_filter.dat"
```

```
epifreg -u
"filterFileLocation=file:///dka0:[tuxedo10gr3.udataobj.security]bea_
ldap_filter.dat"
```

incorrect format:

```
epifreg -u
"filterFileLocation=file://dka0:[tuxedo10gr3.udataobj.security]bea_l
dap_filter.dat"
```

Oracle Tuxedo should now be fully configured and ready to run.

CHAPTER 3

Post-Installation

This chapter contains the following topics:

Running tuxenv.com after Rebooting

Before you can use Oracle Tuxedo, you must configure it using a DCL script. This script must be run each time the machine is rebooted. The Oracle 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 Oracle Tuxedo.
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
   ```

Removing Oracle Tuxedo

Use the following procedure to remove Oracle Tuxedo from a machine.

Note: This procedure removes all the files and directories under the Oracle 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 Oracle Tuxedo.
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
   ```

   Note: Uninstall Oracle Tuxedo after the vps_daemon has stopped. Uninstall procedure will remove all files installed, including files changed after installation. For example:

   SYSREGIIOP.RDP
   SYSREGTGIOP.RDP
   SYSTEM.RDP
   TLISTEN.PW
   RM
   WEBGUI.INI
   SNMP.INI
   vps_init.txt

Using Apache Non-Commerce Server

Before using Apache Non-commerce Server for OpenVMS with Oracle Tuxedo, you must configure certain directories and their URL mappings.

Suppose Apache Non-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 logical names. If Oracle Tuxedo is installed as SYSTEM, then no further configuration is necessary. However, if Oracle Tuxedo is installed in a specific group, then Apache Non-commerce Server must run under the same group. Otherwise, Apache Non-commerce Server will not have the proper variables set up in its environment.

You must do the following steps:

1. Collect the following information about your installation of Apache Non-commerce Server

- the existing apache instances and their configuration-specific root directory

2. Select the existing apache instance that you would like to configure. Users can also create a new apache instance to deploy the Oracle Tuxedo Web console application.

3. Modify the selected apache instance's configuration file, httpd.conf. Replace `DocumentRoot` value with the name of the directory where the file webguitop.html is located. In a standard Oracle Tuxedo installation, this is `'/tuxroot/udataobj/webgui'`.

   Add alias '/java' to the local directory, `'/tuxroot/udataobj/webgui/java'`, and set attributes to this directory.

   Add script alias 'CGI-BIN' to the local directory, `'/tuxroot/bin'`, and set attributes to this directory.

4. Restart your Web server.

Setting Up the Web Console

The following example will show you how to set up the Web console application.

Suppose Oracle Tuxedo is installed on DKA100:[TUXEDO].

Prerequisites

Before you begin, make sure that:

- `TUXDIR` is set as 'DKA100:[TUXEDO]'. If not set, please run the following command:
  `$ define tuxdir dka100:[tuxedo]`
- The logical name 'tuxroot' is set 'DKA100:[TUXEDO]'. If not set, please run the following command: `$ define/system/trans=(conc,term) tuxroot dka100:[tuxedo]`

Setting Up Apache

Run `APACHE$MENU.COM`. The Apache menu appears as follows:

Apache$Menu

```
1. Configure the Secure Web Server
2. Create an Apache instance
3. Delete an Apache instance
4. Manage suEXEC users
5. Run OpenSSL Certificate tool
```

```
6. Convert directory tree to Stream_LF
7. Start up an Apache instance
8. Shut down an Apache instance
9. Show status of an Apache instance
10. Add a node to CSWS in a cluster environment
11. Exit
```

Select option 9 to show the existing Apache instances:

```
Registered Apache Instances
1. SWS     APACHE$COMMON:[CONF]HTTPD.CONF
2. csws    sys$sysdevice:[APACHE$WWW.CONF]httpd.conf
3. Exit
```

For example, select the first instance to deploy the Tuxedo Web console application as follows:

```
Modify APACHE$COMMON:[CONF]HTTPD.CONF
Replace DocumentRoot value
Original:
DocumentRoot "/apache$root/htdocs"
<Directory "/apache$root/htdocs">
…
</Directory>
                    New:
                           DocumentRoot "/tuxroot/udataobj/webgui"
                           <Directory "/tuxroot/udataobj/webgui">
                           …
                           </Directory>
Add alias '/java'
Alias /java "/tuxroot/udataobj/webgui/java/"
<Directory "/tuxroot/udataobj/webgui/java">
Options Indexes
```

```
AllowOverride None
Order allow,deny
Allow from all
</Directory>
Add scriptalias 'CGI-BIN'
ScriptAlias /CGI-BIN/ "/tuxroot/bin/"
<Directory "/tuxroot/bin">
AllowOverride None
Options None
Order allow,deny
Allow from all
</Directory>
```

Final Steps

Do the following steps:

1. Rerun APACHE$MENU.COM to restart your web server
2. Set up the Oracle Tuxedo Web console application
3. Modify tuxroot:[udataobj.webgui]webguitop.html
4. Replace 'HOST' to web server's real IP address
5. Run application `wlisten`

The Tuxedo web console application is now ready.

Programming and Compiling Considerations

For this release please note the following considerations:

- Building Multi-Threaded Oracle Tuxedo Applications
- Process Quota: BIOLM Setting
- MQ XA Switch

- Buildmqadapter
- MAXSERVICES
- AUTHSVR
- Environment Variable Settings
- Logical Name:VPS_MAX_PAGES_PER_IO
- Logical and Symbol Length Limitation
- Multi-Threaded Stack Overflow
- TMTRACE
- Dynamic Library

Building Multi-Threaded Oracle Tuxedo Applications

The link qualifier "`/THREADS_ENABLE`" is required to build multi-threaded Oracle Tuxedo applications. Applications built without `/THREADS_ENABLE` may be unreliable or have unexpected behavior. Single threaded Oracle Tuxedo applications do not use this qualifier.

Building a Single-Threaded Oracle Tuxedo Application

- Client
  - `buildclient  -o empclient -f emp.c`
- Server
  - `buildserver -s "PRINTER" -o printer.c`

Building a Multi-Threaded Oracle Tuxedo Application

- Client
  - A: `define tmlkflags "threads_enable"`
  - B: `buildclient -o empclient -f emp.c`
- Server
  - `buildserver -t -s "PRINTER" -o printer.c`

Note: "`-t`" specifies multi-threading, and that "`/THREADS_ENABLE`" is included in link options.

Process Quota: BIOLM Setting

In multi-threaded Oracle Tuxedo applications, each thread has its own mailbox to communicate with other processes. In order to accelerate receiving messages, four requests are queued in each mailboxes. This means that every thread requires four BIOs.

Note: When BIOLM quota is exhausted, the process is in RWINS state. BIOLM size should be at least greater than `4 times` the number of threads in process.

MQ XA Switch

Oracle Tuxedo for OpenVMS only supports the static XA switch `MQRMIXASwitch`.

Configure TCP Latency

In certain cases, setting `tcpnodelack` can improve network performance. Do the following steps:

```
1. $tcpip
2. TCPIP> sysconfig -r inet rcpnodelack = 1
```

Buildmqadapter

`Buildmqadapter` does not support the `XCBINDIR` environment variable. the binary is created in the local directory if the '`-o`' option not supplied.

MAXSERVICES

If `MAXSERVICES` value is very large and requires a heap size that exceeds the system-defined heap size, some servers cannot boot. In order to support maximum `MAXSERVICES` value `1048575`, please modify the `sysgen pql_mpgflquo` parameter to `740,000`.

AUTHSVR

The `tpusr.dat` file is required for AUTHSVR.

Environment Variable Settings

All environment variable values must be in upper case.

Logical Name:VPS_MAX_PAGES_PER_IO

By default, the maximum allowable size of any single event is 64K bytes. It can be changed using the `VPS_MAX_PAGES_PER_IO` environment variable setting as follows:

```
Maximum size = (512*VPS_MAX_PAGES_PER_IO).
```

Logical and Symbol Length Limitation

According to OpenVMS documentation, the logical and symbol length range is `[1,255]`.

Multi-Threaded Stack Overflow

Oracle Tuxedo applications register `tpterm()` to be called during normal process termination. In multi-threaded processes, `DECthreads` provides a special thread to execute this routine. Because the default stack size is less than `tpterm()` requires, stack overflow occurs.

To resolve this issue, apply patch `VMS831H1I_PTHREAD-V0200`.

TMTRACE

When the `TMTRACE` environment variable is set, the application crashes due to a c compiler issue. When `/pointer=long=argv` is specified, `argv` is not always `NULL` terminated.

To resolve this issue, use c compiler v.7.3, and rebuild the application. Otherwise do not set `TMTRACE`.

Dynamic Library

If you compile a dynamic library with a warning message, `dlopen` fails and displays a “key not found” error message. To resolve this issue, apply patch `VMS831H1I_LIBRTL-V0100`.

Sample Applications

This chapter provides instructions for running the simpapp sample applications delivered with Oracle 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 Oracle Tuxedo).

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.vms`, 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 `id` 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 Oracle Tuxedo sample application.

```
$ tmshutdown -y
```