



THE ENTERPRISE MIDDLEWARE SOLUTION

BEA eLink for Mainframe TCP

TUXEDO User Guide

BEA eLink for Mainframe TCP Version 3.0
Document Edition 3.0
April 1999

Copyright

Copyright © 1997-1999 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, TUXEDO, and WebLogic are registered trademarks of BEA Systems, Inc. BEA Builder, BEA Connect, BEA® eLink™ for Mainframe, BEA Manager, BEA MessageQ, Jolt and M3 are trademarks of BEA Systems, Inc.

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

BEA eLink for Mainframe TCP for TUXEDO User Guide

Document Edition	Part Number	Date	Software Version
3.0		April 1999	BEA eLink for Mainframe TCP 3.0
2.1	830-001004-003	March 1998	BEA Connect TPS 2.1
1.0	830-001004-002	July 1997	BEA Connect TPS 2.0

Contents

Preface

Purpose of This Document	ix
Who Should Read This Document	x
How This Document Is Organized	x
How to Use This Document	xi
Opening the Document in a Web Browser	xi
Printing from a Web Browser	xii
Document Conventions	xii
Related Documentation	xiii
BEA eLink TCP Documentation	xiv
BEA Publications	xiv
Other Publications	xiv
Contact Information	xv
Documentation Support	xv
Customer Support	xv

1. Introducing BEA eLink TCP for TUXEDO

BEA eLink TCP for TUXEDO and the BEA TUXEDO Architecture	1-2
BEA eLink TCP Functionality	1-4
Domains-based Gateway Connectivity	1-4
Security	1-5
Connection Multiplexing	1-5
Domain Name Server Support	1-5
GWIDOMAIN Gateway Component	1-6
How eLink TCP for TUXEDO Affects BEA TUXEDO Application Programs	1-6
VIEW Definitions	1-7

FML Buffer Support.....	1-7
How eLink TCP for TUXEDO Affects BEA TUXEDO Administration	1-8

2. Understanding How BEA eLink TCP for TUXEDO Works

Initializing eLink TCP for TUXEDO	2-1
Processing Local Service Requests	2-2
Step 1: Receiving a Service Request from BEA TUXEDO Software	2-2
Step 2: Connecting to a Remote System	2-2
Step 3: Converting Input Buffer Types	2-3
Step 4: Translating Input Data.....	2-3
Step 5: Transmitting the Service Request	2-4
Step 6: Receiving a Reply	2-4
Step 7: Translating the Reply	2-4
Step 8: Converting Output Data	2-4
Step 9: Sending the Reply to the Caller.....	2-5
Processing Remote Service Requests	2-5
Processing Shut Down Requests	2-6
Programming Considerations	2-6
Input and Output Issues.....	2-7
Preparing Input and Output Data with eLink TCP for TUXEDO.....	2-7
Service Request Parameters	2-7
Output Data Considerations	2-8
Limitations on the Use of Certain ATMI Functions	2-9
Conversational Communication Functions	2-9
Non-Transactional Communications.....	2-9
The tpsprio() and tpgprio() Functions	2-10
The tpbroadcast() and tpnotify() Functions.....	2-10
Error Handling.....	2-11
Gateway Errors.....	2-11
Remote System Failures.....	2-11
Application Errors	2-11

3. Getting Ready to Configure BEA eLink TCP for TUXEDO

Converting Input and Output Data	3-3
Buffers and Records	3-3

Buffers Received from Local Programs	3-4
Records Received from Remote Programs	3-4
Managing Parameters for Buffer and Record Conversion	3-5
Parameters for Locally Originated Calls	3-6
Guidelines for Mapping Input Buffers to Input Records	3-7
The INBUFTYPE Parameter	3-7
The INRECTYPE Parameter	3-7
Guidelines for Mapping Output Records to Output Buffers	3-8
The OUTBUFTYPE Parameter	3-8
The OUTRECTYPE Parameter	3-8
Parameters for Remotely Originated Calls	3-9
Guidelines for Mapping Input Records to Input Buffers	3-10
The INBUFTYPE Parameter	3-11
The INRECTYPE Parameter	3-11
Guidelines for Mapping Output Buffers to Output Records	3-12
The OUTBUFTYPE Parameter	3-12
The OUTRECTYPE Parameter	3-12
Mapping Buffers to Records	3-13
Setting the INBUFTYPE and INRECTYPE Parameters	3-13
Mapping Records to Buffers	3-15
Setting the OUTRECTYPE and OUTBUFTYPE Parameters	3-16
Creating VIEW Definitions to Facilitate Buffer Conversion	3-18
Preparing VIEW Definitions	3-18
Translating Data	3-19
Data Translation Rules	3-19
NULL Characters in String Length Calculations (C Programs)	3-20
NULL Characters in String Length Calculations (COBOL Programs) ...	3-21
Converting Numeric Data	3-21
Enabling eLink TCP Security	3-22
Security Checking from UNIX to Mainframe	3-22
Security Checking from Mainframe to UNIX	3-23
Setting Up Security	3-24
Sample Security Files	3-24
User Files	3-25
Group File	3-25

ACL File.....	3-26
Considerations for Error Handling	3-26

4. Configuring BEA eLink TCP for TUXEDO

Updating the BEA TUXEDO UBBCONFIG File.....	4-1
Updating the *GROUPS Section to Establish a Server Group	4-2
Syntax.....	4-2
Example.....	4-3
Updating the *SERVERS Section.....	4-3
Syntax.....	4-4
Other Options for Configuring Servers	4-5
Example.....	4-5
Specifying Parameters in the GWICONFIG File	4-6
Defining the *GLOBAL Section of the GWICONFIG File	4-8
Syntax.....	4-8
Required Parameters	4-9
Optional Parameters	4-9
Defining the *NATIVE Section of the GWICONFIG File.....	4-11
Syntax.....	4-11
Required Parameters	4-11
Optional Parameters	4-11
Defining the *FOREIGN Section of the GWICONFIG File	4-12
Syntax.....	4-12
Required Parameters	4-13
Optional Parameters	4-13
Defining the *LOCAL_SERVICES Section of the GWICONFIG File ..	4-15
Syntax.....	4-15
Required Parameters	4-16
Optional Parameters	4-16
Defining the *REMOTE_SERVICES Section of the GWICONFIG File	4-17
Syntax.....	4-17
Required Parameters	4-18
Optional Parameters	4-18
Defining Gateway Configurations in the DMCONFIG File	4-19
*DM_LOCAL_DOMAINS Section.....	4-19

Syntax.....	4-19
Required Parameters	4-20
Optional Parameters	4-20
*DM_REMOTE_DOMAINS Section	4-22
Syntax.....	4-22
Required Parameters	4-22
Optional Parameters	4-23
*DM_ACCESS_CONTROL Section	4-23
Syntax.....	4-23
Required Parameters	4-24
Optional Parameters	4-24
*DM_LOCAL_SERVICES Section	4-24
Syntax.....	4-24
Required Parameters	4-25
Optional Parameters	4-25
*DM_REMOTE_SERVICES Section	4-26
Syntax.....	4-26
Required Parameters	4-26
Optional Parameters	4-26
*DM_ROUTING Section	4-28
Syntax.....	4-28
Required Parameters	4-28
Optional Parameters	4-30
Sample DMCONFIG File	4-30

5. Starting BEA eLink TCP for TUXEDO

Setting Environment Variables.....	5-1
Invoking eLink TCP for TUXEDO.....	5-2
Administering the Gateways	5-2

A. Operational Considerations

B. Data Area Security

Enabling Data Area Security	B-1
Format	B-2

C. Error and Informational Messages

D. COBOL Data Encoding

Using the COBOL Data Encoding Library	D-2
Encoding for All Services	D-2
Encoding Messages To and From a Specific Host	D-2

Index

Preface

BEA eLink for Mainframe TCP for TUXEDO (hereafter referenced as eLink TCP for TUXEDO) is a gateway connectivity feature that enables application programs on BEA TUXEDO systems to perform various non-transactional tasks with application programs that reside on different kinds of computers.

Purpose of This Document

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

This guide explains how to configure and administer eLink TCP for TUXEDO and how eLink TCP for TUXEDO fits into the BEA TUXEDO environment. In addition, this guide:

- ◆ Explains how eLink TCP for TUXEDO processes service requests, those that originate locally and those that originate on remote systems
- ◆ Explains how eLink TCP for TUXEDO affects BEA TUXEDO application programs
- ◆ Provides conceptual and procedural information that will help you configure and administer eLink TCP for TUXEDO

Who Should Read This Document

This document is intended for system administrators who will configure and administer eLink TCP for TUXEDO. In addition, programmers will find useful pointers for developing client programs and service routines that send data through eLink TCP for TUXEDO.

How This Document Is Organized

The *BEA eLink TCP for TUXEDO User Guide* is organized as follows:

- ◆ *Introducing BEA eLink TCP for TUXEDO* introduces eLink TCP for TUXEDO, explains how eLink TCP for TUXEDO fits into the BEA TUXEDO environment, and lists hardware and software requirements.
- ◆ *Understanding How BEA eLink TCP for TUXEDO Works* explains how to start eLink TCP for TUXEDO, how it processes services requests, and how it is terminated. It also provides information that helps programmers develop clients' programs and service routines that send data through eLink TCP for TUXEDO.
- ◆ *Getting Ready to Configure BEA eLink TCP for TUXEDO* provides information that will prepare you to configure eLink TCP for TUXEDO. It also explains how to create VIEW definitions (descriptions of data structures eLink TCP for TUXEDO uses to convert input and output data into formats that are acceptable to target systems).
- ◆ *Configuring BEA eLink TCP for TUXEDO* provides detailed information about the eLink TCP for TUXEDO configuration file (GWICONFIG) and associated configuration parameters. In addition, this document explains how to update the BEA TUXEDO configuration file (UBBCONFIG) so that eLink TCP for TUXEDO can interact with BEA TUXEDO software.
- ◆ *Starting BEA eLink TCP for TUXEDO* explains how to set environmental variables, test connectivity with remote regions, and invoke eLink TCP for TUXEDO.
- ◆ *Operational Considerations* lists temporary and permanent operating limitations, as well as compatibility issues with other products.
- ◆ *Data Area Security* explains data area security.

-
- ◆ *Error and Informational Messages* describes messages and resolutions for the system.
 - ◆ *COBOL Data Encoding* describes data translation from ASCII-to-EBCDIC and EBCDIC-to-ASCII for various data types using the encoding libraries ConvMVS and ConvMVSC.

How to Use This Document

The *BEA eLink TCP for TUXEDO User Guide* 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 if it is installed on a Web server, open the following HTML file:

`http://(directory path to eLink TCP HTML files)/begin.htm`

Note: The online documentation requires a Web browser that supports HTML 3.0. Netscape Navigator 4.0 or higher or Microsoft Internet Explorer 4.0 or higher are recommended.

Printing from a Web Browser

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

Document Conventions

The following documentation conventions are used throughout this manual:

Item	Examples
Variable names	Variable names represent information you must supply or output information that can change; they are intended to be replaced by actual names. Variable names are displayed in italics and can include hyphens but not underscores. The following are examples of variable names in text: <i>error-file-name</i> The <i>when-return</i> value...
Function names in text	C function names are displayed in lower case type and can include parentheses and possibly underscores, as follows: routine_name() COBOL function or subprogram names are displayed in uppercase type without underscores or hyphens, as follows: ROUTINENAME()
Symbolic constants for languages (keywords, error codes, and flags)	C symbolic constants are displayed in uppercase type and can include underscores, as follows: CONSTANT_NAME COBOL symbolic constants are displayed in uppercase type and can include hyphens, as follows: CONSTANT-NAME

Item	Examples
User input and screen output	<p>For screen displays and other examples of input and output, user input appears as in the first of the following lines; system output appears as in the second through fourth lines:</p> <pre>dir c:\accounting\data</pre> <p>Volume in drive C is WIN_NT_1 Volume Serial Number is 1234-5678 Directory of C:\ACCOUNTING\DATA</p>
Syntax	<p>Code samples can include the following elements:</p> <ul style="list-style-type: none"> ◆ Variable names can include hyphens but not underscores (e.g., <i>error-file-name</i>) ◆ Optional items are enclosed in square brackets: []. If you include an optional item, do not code the square brackets. ◆ A required element for which alternatives exist is enclosed in braces { }. The alternatives are separated by the pipe (vertical bar) character: . You must include only one of the alternatives for that element. Do not code the braces or pipe character. ◆ An ellipsis (...) indicates that the preceding element can be repeated as necessary. <p>C synopsis:</p> <pre>int tpacall(char *svc, char *data, long len, long flags)</pre> <p>COBOL statement:</p> <pre>CALL "TPACALL" USING TPSVCDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.</pre>
Omitted code	<p>An ellipsis (...) is used in examples to indicate that code that is not pertinent to the discussion is omitted. The ellipsis can be horizontal or vertical.</p>

Related Documentation

The following sections list the documentation provided with the eLink TCP software, and other publications related to online transaction processing technology.

BEA eLink TCP Documentation

The eLink TCP documentation consists of the following items:

- ◆ *BEA eLink TCP Installation Guide*
- ◆ *BEA eLink TCP for TUXEDO User Guide*
- ◆ *BEA eLink TCP for CICS User Guide*
- ◆ *BEA eLink TCP for IMS User Guide*
- ◆ *BEA eLink TCP Release Notes*

BEA Publications

The following BEA publications are also available:

- ◆ *TUXEDO System 6 Reference Manual*
- ◆ *TUXEDO System 6 Programmer's Guide, Volumes 1 and 2*

Other Publications

For more information about online transaction processing technology, refer to the following books:

- ◆ *The TUXEDO System* (Andrade, Carges, Dwyer, 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 eLink TCP, or if you have problems installing and running BEA eLink TCP, 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



1 Introducing BEA eLink TCP for TUXEDO

The BEA eLink for Mainframe TCP for TUXEDO (hereafter referenced as eLink TCP for TUXEDO) product is a domains-based gateway connectivity feature that makes it possible for application programs on BEA TUXEDO systems to perform non-transactional tasks with application programs in other OLTP systems that support eLink TCP for TUXEDO gateways. These include:

- ◆ CICS on IBM MVS systems
- ◆ IMS/TM on IBM MVS systems
- ◆ IMS/DC on IBM MVS systems
- ◆ IMS/OTMA on IBM MVS systems

The BEA eLink TCP for TUXEDO is designed to provide transparent access to services that reside outside a BEA TUXEDO region. The eLink TCP for TUXEDO gateway validates and acts on behalf of a remote client making a request, without performing user authentication. In addition, eLink TCP for TUXEDO can provide remote application programs with access to local services if this is supported by the release of eLink TCP for TUXEDO software installed on the remote system.

This document provides information about the following topics:

- ◆ BEA eLink TCP for TUXEDO and the BEA TUXEDO Architecture
- ◆ BEA eLink TCP Functionality
- ◆ GWIDOMAIN Gateway Component
- ◆ How eLink TCP for TUXEDO Affects BEA TUXEDO Application Programs

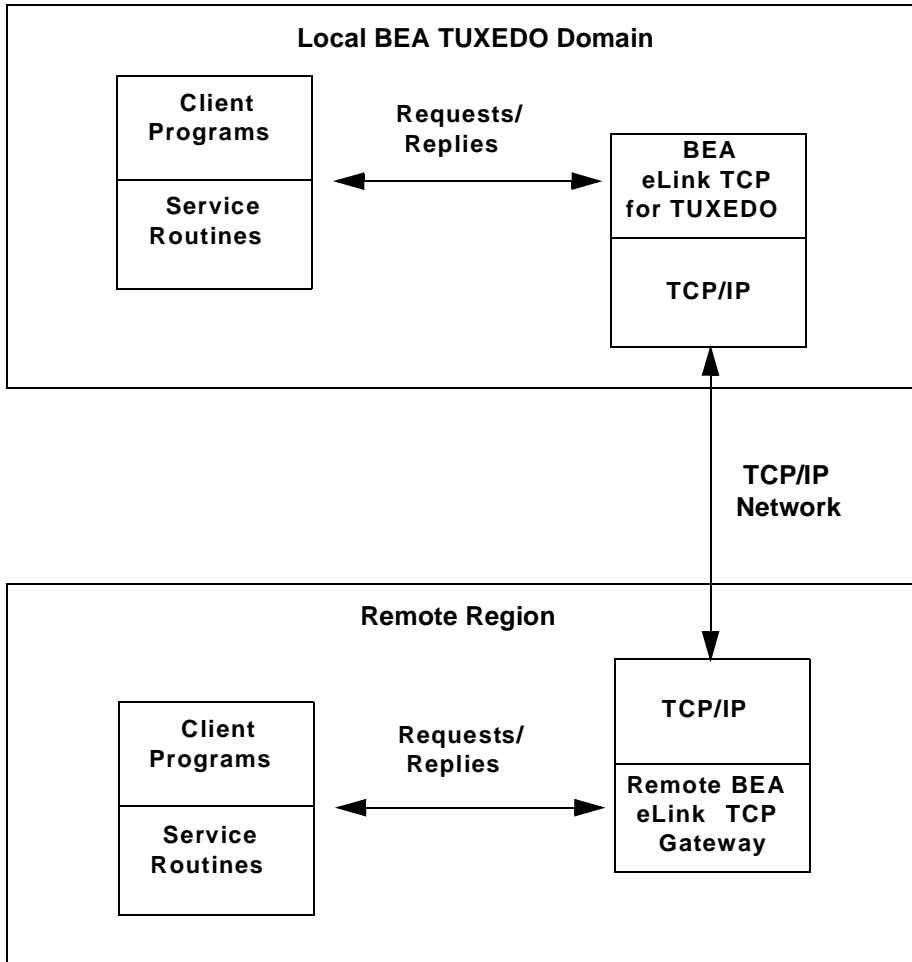
- ◆ How eLink TCP for TUXEDO Affects BEA TUXEDO Administration

BEA eLink TCP for TUXEDO and the BEA TUXEDO Architecture

A BEA TUXEDO region consists of client and server programs that operate across a network of BEA TUXEDO systems or compatible systems. Any client program can request services that are offered by any server program running on any computer in the region. Further, by way of a directory that maps services to servers, the location of server programs is kept transparent.

As Figure 1-1 shows, eLink TCP for TUXEDO extends this transparent access by sending requests to and receiving requests from remote regions and systems through TCP/IP network software.

Figure 1-1 Routing Service Calls through BEA eLink TCP for TUXEDO



As Figure 1-1 suggests, inside a single region, eLink TCP for TUXEDO fits between the BEA TUXEDO software and TCP/IP.

- ◆ When local client programs send requests to remote systems, eLink TCP for TUXEDO transforms those requests into messages formatted appropriately for transmission to the remote system. Also, when remote systems respond, eLink TCP for TUXEDO transforms these responses into replies that local client programs can process.

- ◆ When remote client programs send request messages, eLink TCP for TUXEDO transforms those messages into requests that local service routines can process. Also, when local service routines send replies, eLink TCP for TUXEDO transforms those replies into messages that remote services can process.

BEA eLink TCP for TUXEDO is implemented as a TUXEDO domain gateway. It accepts standard BEA TUXEDO service requests and returns standard replies.

One eLink TCP for TUXEDO gateway connects to multiple communications targets, also referred to as gateways. Each communications target, or gateway, is a unique network endpoint.

The remote gateways must also support eLink TCP for TUXEDO. BEA eLink TCP for TUXEDO servers associated with the local gateway communicate with remote eLink TCP for TUXEDO gateways to non-BEA TUXEDO TP monitors, such as eLink TCP for CICS and eLink TCP for IMS.

Although remote systems are identified in the eLink TCP for TUXEDO configuration, they remain unknown to BEA TUXEDO software. For example, remote systems that are accessible through eLink TCP for TUXEDO are not identified in the *MACHINES section of the UBBCONFIG file.

BEA eLink TCP for TUXEDO maintains its own control information in shared memory, in much the same way that BEA TUXEDO software itself maintains the Bulletin Board. Although eLink TCP for TUXEDO accesses the BEA TUXEDO Bulletin Board, BEA TUXEDO does not access eLink TCP for TUXEDO control information.

BEA eLink TCP Functionality

The following functionality is available in this version of eLink TCP.

Domains-based Gateway Connectivity

The eLink TCP product has a domains-based architecture supporting bidirectional communications, request/response support, and concurrent support for MVS BMP and Open Transaction Manager Access (OTMA) interfaces.

Security

The eLink TCP for TUXEDO product grants access to BEA TUXEDO services based on a user name that the remote gateway supplies.

The eLink TCP for CICS product can initiate transactions or link to programs. BEA TUXEDO security provides the USERID value to the eLink TCP product to test for appropriate security prior to initiating the transactions.

The eLink TCP for IMS product has an OTMA interface that supports enhanced security. This interface allows a BEA TUXEDO requester to pass a USERID through the OTMA server interface for authorization through a third-party security package, such as RACF.

Connection Multiplexing

The eLink TCP for TUXEDO allows multiple requests to process simultaneously over a single connection. This is known as connection multiplexing. Two connecting gateways determine a multiplex count that is acceptable to both sides at connection time. After establishing the connection, clients can send multiple requests (up to the number in the multiplex count) to the server gateway. Connection multiplexing allows for more efficient use of sockets and other system resources by the eLink TCP gateways.

Note: Each connection is one-directional, which means clients on opposing platforms cannot use the same connection to communicate with remote servers.

Domain Name Server Support

The eLink TCP product supports domain name server (DNS) resolution of IP addresses. This support allows you to change the IP address at the Domain Name Server to implement address changes without reconfiguring the eLink TCP gateway.

GWIDOMAIN Gateway Component

The eLink TCP product consists of a single component, the GWIDOMAIN gateway. This gateway is responsible for the processing of all incoming and outgoing requests. It also maintains connections with all remote gateways.

How eLink TCP for TUXEDO Affects BEA TUXEDO Application Programs

BEA eLink TCP for TUXEDO preserves the high degree of location transparency that BEA TUXEDO software provides. In fact, in virtually all cases, programmers do not need to know that particular services are provided by remote systems.

BEA eLink TCP for TUXEDO supports the main BEA TUXEDO communication paradigm: request/reply communications (either synchronous or asynchronous).

All BEA TUXEDO buffer types can be employed for data exchange. These include:

- ◆ X/Open standard XATMI buffer types
 - ◆ X_OCTET
 - ◆ X_C_TYPE
 - ◆ X_COMMON
- ◆ BEA TUXEDO ATMI buffers
 - ◆ CARRAY
 - ◆ STRING
 - ◆ FML
 - ◆ VIEW

Each of the three X/Open buffer types is equivalent to a BEA TUXEDO ATMI buffer type. The following information provides these equivalencies.

- ◆ X_OCTET is equivalent to CARRAY
- ◆ X_C_TYPE is equivalent to VIEW
- ◆ X_COMMON is equivalent to VIEW, but represents only the subset of field types that are common to both the C and COBOL languages

VIEW Definitions

In some circumstances, you must convert typed buffers to formats that are acceptable to target systems. The standard BEA TUXEDO system VIEW definition mechanism is employed for this purpose.

VIEW definitions make it possible to map input data and output data between different programming environments (such as C and COBOL). They also enable eLink TCP for TUXEDO to convert data representations automatically between different systems

VIEW definitions can be created by programmers or system administrators. See “Getting Ready to Configure BEA eLink TCP for TUXEDO,” for details. For more detailed information about programming considerations, see “Understanding How BEA eLink TCP for TUXEDO Works.”

FML Buffer Support

When communicating with systems or regions that do not support FML buffers directly, the eLink TCP for TUXEDO gateway can convert FML buffers to or from user-defined record layouts in a manner transparent to the FML application. Thus, once a VIEW definition that describes the remote application’s record layout is created, it can be used to convert the record to or from an FML buffer. The GWICONFIG and DMCONFIG files (eLink TCP for TUXEDO configuration file) contains VIEW specifications as part of the service description.

Through this conversion between ATMI buffers and record structures, eLink TCP for TUXEDO supports sending fielded buffers containing FML data between regions. The eLink TCP for TUXEDO software converts the data from FML buffers to user-defined records using the VIEW definitions and field descriptions at the originating region.

How eLink TCP for TUXEDO Affects BEA TUXEDO Administration

BEA eLink TCP for TUXEDO administration tools and features are thoroughly integrated with BEA TUXEDO administration tools and features. Here are some specific examples:

- ◆ System administrators define eLink TCP for TUXEDO in the BEA TUXEDO configuration as a regular TUXEDO domain gateway.
- ◆ The eLink TCP for TUXEDO configuration file (GWICONFIG) specifies how local BEA TUXEDO service names are mapped to remote service names. Also, the GWICONFIG file identifies VIEW definitions that eLink TCP for TUXEDO uses to convert and translate input and output data.
- ◆ At runtime, system administrators use BEA TUXEDO `tmadmin` subcommands to manage eLink TCP for TUXEDO and related processes.

For more detailed information about configuring eLink TCP for TUXEDO, see “Getting Ready to Configure BEA eLink TCP for TUXEDO,” and “Configuring BEA eLink TCP for TUXEDO.” For detailed information about commands for administering eLink TCP for TUXEDO, refer to the *BEA TUXEDO Administrator's Guide* and the *BEA TUXEDO Domain User Guide*.

2 Understanding How BEA eLink TCP for TUXEDO Works

To understand how eLink for Mainframe TCP for TUXEDO (hereafter referenced as eLink TCP for TUXEDO) works, you need to understand how it performs the following functions:

- ◆ Initializing eLink TCP for TUXEDO
- ◆ Processing Local Service Requests
- ◆ Processing Remote Service Requests
- ◆ Processing Shut Down Requests
- ◆ Programming Considerations

Each of these operations is described in the following subsections.

Initializing eLink TCP for TUXEDO

When you boot BEA TUXEDO software using the `tmboot` command, eLink TCP for TUXEDO initializes in the following manner:

1. BEA eLink TCP for TUXEDO parses the GWICONFIG configuration file and initializes all parameters. If syntax errors are encountered during parsing, eLink TCP for TUXEDO writes a message to the ULOG file and initialization fails.
2. After reading the GWICONFIG file, eLink TCP for TUXEDO advertises remote services that are named in the file dynamically. This includes services for all remote gateways.

Processing Local Service Requests

When eLink TCP for TUXEDO receives a BEA TUXEDO service request from a local client program, it processes the request in the following manner.

Step 1: Receiving a Service Request from BEA TUXEDO Software

When a client program sends a request for a remote service that is accessible through eLink TCP for TUXEDO, BEA TUXEDO forwards the request to the gateway pending the requested service.

Step 2: Connecting to a Remote System

If no connection to the target remote system exists or an existing connection has been broken, the eLink TCP for TUXEDO gateway opens a new connection at this time.

If the remote system returns a connection failure indication, the BEA TUXEDO service request fails and an error is returned to the caller. The actual error value returned depends on the timing of the connection failure. Information about failures is logged to the ULOG file.

Step 3: Converting Input Buffer Types

In some circumstances, typed buffers associated with service requests must be converted before service requests can be sent to remote systems. Type conversion involves changing the layout of a buffer to a format that is acceptable to a remote service.

For example, if a local client program places user input in an FML buffer that a remote service cannot process, the buffer must be converted into the structure the remote service expects.

In situations where input type conversion is required, programmers or administrators must perform the following tasks:

- ◆ Determine the format of the input data the remote service expects.
- ◆ If necessary, create a VIEW definition that describes the format of the input data. VIEW definitions are descriptions of data structures that are used for input and output in the BEA TUXEDO environment.
- ◆ Specify this information in the eLink TCP for TUXEDO configuration files (GWICONFIG and DMCONFIG).

Once these tasks have been completed, eLink TCP for TUXEDO performs all necessary type conversions automatically.

For information about creating VIEW definitions to facilitate type conversion, see “Getting Ready to Configure BEA eLink TCP for TUXEDO.” For information about the eLink TCP for TUXEDO configuration file, see “Configuring BEA eLink TCP for TUXEDO.”

Step 4: Translating Input Data

BEA eLink TCP for TUXEDO automatically translates data as required. Translation refers to a change in how intrinsic data types are represented with respect to word length, byte ordering, and character encoding.

To facilitate data translation, administrators must specify certain parameters in the GWICONFIG configuration file. For detailed information about how BEA eLink TCP for TUXEDO translates data, refer to “Getting Ready to Configure BEA eLink TCP for TUXEDO.”

Step 5: Transmitting the Service Request

BEA eLink TCP for TUXEDO constructs a request message. This message includes the following items and is sent to the remote system:

- ◆ The remote service name
- ◆ The input data record that eLink TCP for TUXEDO has converted and translated, as required
- ◆ An indication of whether the remote service should return a reply to the caller

Step 6: Receiving a Reply

After sending a request message, eLink TCP for TUXEDO performs a receive operation. If the eLink TCP for TUXEDO receive timeout expires before a message arrives from the remote system, a TPETIME error is returned to the caller.

Step 7: Translating the Reply

After eLink TCP for TUXEDO receives a reply, data representations are translated as needed, in the reverse of the input translation. For details, see the “Step 4: Translating Input Data” section in this document.

Step 8: Converting Output Data

If the format of the reply is not suitable for the local client program, eLink TCP for TUXEDO converts the reply into an appropriate buffer format.

In situations where output conversion is required, programmers or administrators must do the following:

- ◆ Determine the type and the format of the output buffer the local client program expects
- ◆ If necessary, create a VIEW definition that describes the format of the output buffer
- ◆ Specify this information in the eLink TCP for TUXEDO configuration files (GWICONFIG and DMCONFIG)

Once these tasks have been completed, eLink TCP for TUXEDO performs all necessary conversions automatically.

For information about creating VIEW definitions to facilitate type conversion, see “Getting Ready to Configure BEA eLink TCP for TUXEDO.” For information about the BEA eLink TCP for TUXEDO configuration file, see “Configuring BEA eLink TCP for TUXEDO.”

Step 9: Sending the Reply to the Caller

The BEA TUXEDO buffer resulting from output type conversion and output data translation is returned to the caller with a TPSUCCESS or TPFAIL indication.

Processing Remote Service Requests

BEA eLink TCP for TUXEDO processes remote service requests (those which originate on remote systems) in much the same way that it processes local requests. Here is a brief summary:

1. The eLink TCP for TUXEDO gateway receives the service request and, if necessary, the input record associated with the request is translated and converted into the specified input buffer format.
2. Acting as a client program, the gateway passes the service request to BEA TUXEDO software.

3. When the service routine terminates, the output buffer is converted into the specified output record format and/or translated (when required).
4. Finally, the output is sent to the requester on the remote system.

For more detailed information about record and buffer conversion, and data translation, see “Processing Local Service Requests.”

Processing Shut Down Requests

When you send a shutdown request to eLink TCP for TUXEDO using the `tmsshutdown` command, eLink TCP for TUXEDO performs the following tasks:

1. Completes outstanding requests
2. Drops all open connections
3. Terminates

Programming Considerations

In general, BEA TUXEDO application programs that send requests through eLink TCP for TUXEDO are developed just like other BEA TUXEDO application programs.

BEA eLink TCP for TUXEDO supports all request/reply communications functions that are included in the ATMI and XATMI interfaces. In addition, all supported functions can be used in the standard manner. In particular:

- ◆ All XATMI and ATMI buffer types are supported.
- ◆ All system errors are reported in the standard manner.

Input and Output Issues

This subsection explores several input and output issues that programmers need to consider when developing application programs that use eLink TCP for TUXEDO.

Preparing Input and Output Data with eLink TCP for TUXEDO

As described in “Processing Local Service Requests,” there are many circumstances in which it is necessary to convert input and output parameters into formats that are acceptable to remote systems or regions and the local system.

BEA eLink TCP for TUXEDO provides powerful configuration capabilities that make it possible for you to convert or map parameters easily rather than requiring you to program in a different way.

The eLink TCP for TUXEDO configuration files (GWICONFIG and DMCONFIG) are a centralized mechanism that you can use to define and maintain relationships between the local system and remote systems or regions. In addition to input and output parameter mappings, these relationships include service name mappings (where remote service names are mapped to local service names) and error record mappings.

For more information about the GWICONFIG configuration file, see “Getting Ready to Configure BEA eLink TCP for TUXEDO,” and “Configuring BEA eLink TCP for TUXEDO.”

Service Request Parameters

BEA TUXEDO application programs can request the following two categories of remote services through eLink TCP for TUXEDO:

- ◆ Existing application programs that were originally developed for traditional OLTP environments and have been adapted for use with BEA TUXEDO
- ◆ Services that were developed specifically for the XATMI or ATMI (BEA TUXEDO) environments. Included in this category are services that reside in remote BEA TUXEDO regions.

If a remote service was developed specifically for the BEA TUXEDO environment, the input it requires is shaped by three factors:

- ◆ Application-specific requirements

- ◆ Normal ATMI or XATMI requirements for defining and sending input
- ◆ Machine-specific requirements for how input is formatted (such as those described in “Processing Local Service Requests”)

On the other hand, if a remote service is an existing OLTP application program, there are often additional requirements for input. For example, many systems require input that includes terminal data.

Often, by creative use of the eLink TCP for TUXEDO configuration capabilities previously introduced, you can eliminate the need to include control information, such as terminal data, in the BEA TUXEDO application source code that you develop. For instance, you can include terminal control codes in VIEW definitions that are associated with the eLink TCP for TUXEDO configuration.

For information about the normal input requirements of BEA TUXEDO services, see the *BEA TUXEDO Programmer's Guide*.

Output Data Considerations

To maintain the location transparency of the BEA TUXEDO environment, eLink TCP for TUXEDO does not preserve data from BEA TUXEDO input buffers in the associated output buffers. Hence, the consequences of using the same buffer for input and output must be understood to avoid problems.

In particular, some existing BEA TUXEDO applications may use FML buffers to accumulate results or to maintain application context across service requests. Developers adding eLink TCP for TUXEDO to such an application must do one of the following:

- ◆ Maintain a copy of the necessary data in the client program (or service routine) that makes requests.
- ◆ Ensure that the remote service returns input data with the output record.

This requirement is no different from the requirement that existed before the use of eLink TCP for TUXEDO. That is, application programs that accumulate output data in FML buffers must ensure that services return replies in original FML input buffers (with output data added)—not in new or re-initialized buffers.

Limitations on the Use of Certain ATMI Functions

The ATMI interface includes several features and functions—related primarily to conversations, transactions and client identities—that application programs cannot propagate to other application programs through eLink TCP for TUXEDO.

In this guide, permanent limitations of this sort are referred to as operational considerations. Specific operational considerations are described in the following sections.

Note: In this discussion, a local application program is one that resides within the immediate BEA TUXEDO region. A remote application program is one that resides outside the immediate BEA TUXEDO region.

Conversational Communication Functions

Conversational communication functions are subject to the following operational considerations:

- ◆ Local client and server programs cannot use the `tpconnect()` function to establish conversations with remote services.
- ◆ Remote client and server programs cannot use the `tpconnect()` function to establish conversations with local services.
- ◆ Similarly, the `tpsend()`, `tprecv()` and `tpdiscon()` functions may not be used for communication through eLink TCP for TUXEDO.

Non-Transactional Communications

BEA eLink TCP for TUXEDO supports only non-transactional communications. Therefore, all communications via eLink TCP for TUXEDO are subject to the following operational considerations:

- ◆ Remote services called by local client or server programs are always invoked outside the boundaries of any local transaction.
- ◆ Local services called by remote client or server programs are always invoked outside the boundaries of any local transaction.

- ◆ Local client or server programs calling remote services should invoke the `tpcall()` function, or the `tpacall()` and `tpgetrply()` functions, outside the boundaries of any local transaction (i.e., outside of any `tx_begin()/tx_commit()` or `tpbegin()/tpcommit()` pair).
- ◆ If local client or server programs must call remote services within the boundaries of a local transaction, `TPNOTRAN` must be specified as one of the flags to the `tpcall()` or `tpacall()` function.

The `tpsprio()` and `tpgprio()` Functions

The `tpsprio()` and `tpgprio()` functions are subject to the following operational considerations:

- ◆ Local client and server programs cannot use the `tpsprio()` function to set the priority at which remote services are processed. Instead, a call to the `tpsprio()` function causes the priority of a local eLink TCP for TUXEDO gateway to be set.
- ◆ Remote client and server programs cannot use the `tpsprio()` function to set the priority at which local services are processed.
- ◆ When local client or server programs use the `tpgprio()` function to determine the priority of a remote service, the priority of a local eLink TCP for TUXEDO Requester is returned.

The `tpbroadcast()` and `tpnotify()` Functions

The `tpbroadcast()` and `tpnotify()` functions are subject to the following operational considerations:

- ◆ Local client programs cannot use the `tpbroadcast()` function to send unsolicited messages to remote client programs (and the reverse).
- ◆ Local services cannot use the `tpbroadcast()` or `tpnotify()` functions to send messages to remote client programs (and the reverse).

Error Handling

There are three kinds of errors that local application programs can encounter when they send requests through eLink TCP for TUXEDO gateways:

- ◆ Gateway errors
- ◆ Problems on remote systems
- ◆ Errors from remote application programs

The following sections explain how eLink TCP for TUXEDO handles these different kinds of errors.

Gateway Errors

When local or remote gateway errors occur, they are logged in the BEA TUXEDO ULOG file and associated service requests fail. Also, appropriate error codes are returned to callers.

Remote System Failures

When remote systems encounter problems, service requests may fail or time out. The exact outcome depends on whether the remote system provides a means for eLink TCP for TUXEDO to detect failure.

If the remote target system does not make it possible for eLink TCP for TUXEDO to detect particular types of failure, the eLink TCP for TUXEDO blocking timeout parameter can be tuned to provide timely detection of the problem.

For more information about the blocking timeout parameter, see “Configuring BEA eLink TCP for TUXEDO.”

Application Errors

Application errors are similar to remote system failures. That is, remote systems may or may not use error indicators to pass information back to the local eLink TCP for TUXEDO gateway resulting in the generation of error messages. If no such error indicators exist, service routines typically use their own mechanisms to report failures to callers.

When application errors occur, some service routines may not return their usual output records at all. Instead, they may return some other data indicating that there has been an error, such as a string that contains a failure message.

When eLink TCP for TUXEDO receives a service failure message from a remote system, it:

- ◆ Converts the output buffer as required based on instructions it finds in the GWICONFIG file
- ◆ Writes an error record in the ULOG file if the configuration directs it to do so
- ◆ Returns the appropriate type of BEA TUXEDO buffer to the caller

Note: When BEA TUXEDO applications detect service failures, they should not assume that returned buffers are the expected type. In practice this may not be an issue, since BEA TUXEDO application programs may ignore return buffers when failures occur. If you need to check a buffer type, you can use the BEA TUXEDO `tptypes()` function. Once the type is known, the buffer can be handled accordingly (for example, by displaying a window containing an error string).

3 Getting Ready to Configure BEA eLink TCP for TUXEDO

One of the major benefits of using eLink for Mainframe TCP for TUXEDO (hereafter referenced as eLink TCP for TUXEDO) to connect dissimilar systems is the degree to which different programming environments can be isolated. BEA TUXEDO programmers rarely need to know that services are handled by dissimilar systems or by systems in remote regions. Application programs do not need to be developed in any special way.

The key to this high degree of transparency is the eLink TCP for TUXEDO configuration. It is through this mechanism that environmental differences, such as naming conventions and data formats, are concealed from programmers and programs.

This document provides information about the following topics:

- ◆ Converting Input and Output Data
- ◆ Managing Parameters for Buffer and Record Conversion
- ◆ Translating Data
- ◆ Enabling eLink TCP Security
- ◆ Considerations for Error Handling

There are three kinds of environmental differences that are isolated in the eLink TCP for TUXEDO configuration files (GWICONFIG and DMCONFIG).

3 *Getting Ready to Configure BEA eLink TCP for TUXEDO*

Service names	Different systems have different rules for naming services. Service names can differ in length, allowable characters, and even conventions as to how they are constructed or chosen.
Input and output data formats	Different systems have different conventions for formatting input and output data (such as structure, character set, and so forth).
Error handling	Different systems report application errors in different ways.

The technique that is used to hide these differences is called mapping. Generally speaking, when you map things, you associate local values or entities with values or entities that are meaningful to programs on remote systems.

The procedure for mapping service names is self-explanatory; you create a configuration file record in which a local name for a service is paired with a remote name for that service. On the other hand, procedures for mapping input data, output data, and application errors are more complex. Conceptual information and other background information is required.

This document explains how eLink TCP for TUXEDO performs the following tasks:

- ◆ Converts input and output data into formats that are acceptable to target systems
- ◆ Translates data
- ◆ Processes application errors

This document also introduces configuration file parameters that you must set. These configuration file parameters:

- ◆ Map input data and output data that flows between the local region and remote systems and thereby facilitate the eLink TCP for TUXEDO conversion process
- ◆ Specify how data is translated
- ◆ Specify how to set up security
- ◆ Specify how application errors are processed

In addition, this document provides information about creating VIEW definitions. VIEW definitions are descriptions of data structures that are used for input and output in the BEA TUXEDO environment. eLink TCP for TUXEDO uses VIEW definitions to determine how to convert input data and output data into formats that are acceptable to target systems.

For detailed information about updating the eLink TCP for TUXEDO configuration files (GWICONFIG and DMCONFIG), see “Configuring BEA eLink TCP for TUXEDO.”

Note: All eLink TCP for TUXEDO configuration parameters are described in “Configuring BEA eLink TCP for TUXEDO.” This document focuses on complex parameters that require a separate introduction.

The task of configuring data mappings could be considered a programming activity because it requires knowledge of the BEA TUXEDO programming environment. However, because configuration parameters affect many application programs, configuration is usually an administrator's responsibility.

Converting Input and Output Data

This section introduces procedures that eLink TCP for TUXEDO follows to process and convert input and output data.

Buffers and Records

In this guide, the following terms are used to describe input and output data:

Buffer	Input or output data as it exists inside the local BEA TUXEDO region. This includes all the buffer types that BEA TUXEDO software supports—both BEA TUXEDO ATMI buffer types and X/Open XATMI buffer types.
Record	Input or output data as it exists outside the local BEA TUXEDO region—on different kinds of systems.

These terms make it easier to understand how eLink TCP for TUXEDO handles input and output data.

Buffers Received from Local Programs

When eLink TCP for TUXEDO receives a buffer from a local program, it automatically determines the buffer's type.

- ◆ BEA eLink TCP for TUXEDO automatically “types” input buffers that local client programs send to remote services.
- ◆ BEA eLink TCP for TUXEDO automatically “types” output buffers that local services return to remote client programs.

After eLink TCP for TUXEDO determines a buffer's type, it consults the configuration file (GWICONFIG) to determine whether the buffer needs to be converted to a different format.

- ◆ Input buffers sent to remote services may need to be converted to record formats that are meaningful to those services.
- ◆ Output buffers returned to remote client programs may need to be converted to record formats that are meaningful to those programs.

If the configuration indicates that conversion is required, eLink TCP for TUXEDO transforms the buffer into the record format that is specified in the configuration.

Records Received from Remote Programs

When eLink TCP for TUXEDO receives a record from a remote system, it consults the configuration file (GWICONFIG) to determine the record's type.

After eLink TCP for TUXEDO determines a record's type, it consults the domain configuration (DMCONFIG) to determine whether the record needs to be converted to a different format.

- ◆ Input records from remote client programs may need to be converted to buffer formats that are acceptable to local service routines.

- ◆ Output records returned from remote services may need to be converted to buffer formats that are acceptable to local client programs.

If the configuration indicates that conversion is required, eLink TCP for TUXEDO transforms the record into the buffer format that is specified in the configuration.

Managing Parameters for Buffer and Record Conversion

BEA eLink TCP for TUXEDO provides four configuration parameters you can use to map buffers and records. For more information about buffers and records, see section “Buffers and Records.”

Specify the following buffer configuration parameters in the domain configuration file (DMCONFIG).

- ◆ INBUFTYPE - Identifies the type, and in some cases the format, of a buffer received from a TUXEDO client or server
- ◆ OUTBUFTYPE - Identifies the type, and in some cases the format, of a buffer to be sent to a TUXEDO client or server

Specify the following record configuration parameters in the gateway configuration file (GWICONFIG).

- ◆ INRECTYPE - Identifies the type, and in some cases the format, of a buffer to be sent to a remote gateway
- ◆ OUTRECTYPE - Identifies the type, and in some cases the format, of a buffer received from a remote gateway

Each of these four parameters has two possible meanings or interpretations—one for service requests that originate locally, and one for service requests that originate on remote systems.

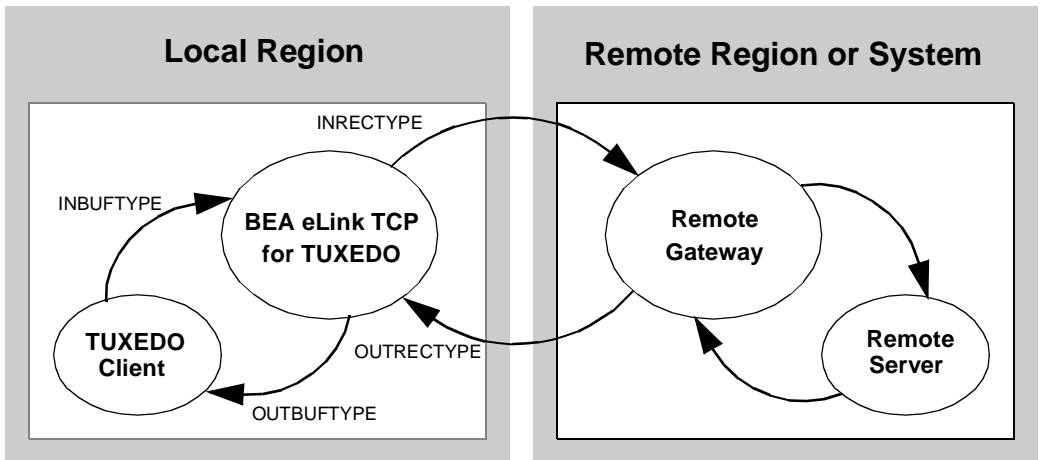
The next two sections (“Parameters for Locally Originated Calls” and “Parameters for Remotely Originated Calls” explore these different meanings in detail.

Parameters for Locally Originated Calls

This section takes a closer look at how eLink TCP for TUXEDO handles service calls that originate locally, within the immediate BEA TUXEDO region. Also, it explains how the INBUFTYPE, INRECTYPE, OUTRECTYPE, and OUTBUFTYPE parameters can be used to manage the conversion of buffers and records that flow between local client programs and remote services.

In Figure 3-1, a local BEA TUXEDO client program issues a service call that a local eLink TCP for TUXEDO gateway routes to a remote server through eLink TCP for TUXEDO.

Figure 3-1 How Parameters Are Mapped During Locally Originated Calls



In this situation, the four configuration parameters that are shown in the figure have the following meanings:

- ◆ The INBUFTYPE parameter describes the BEA TUXEDO input buffer that the local client program provides to the eLink TCP for TUXEDO gateway through BEA TUXEDO software.
- ◆ The INRECTYPE parameter describes the input record that is sent to the service on the remote system.
- ◆ The OUTRECTYPE parameter describes the output record that is received from the service on the remote system.

- ◆ The OUTBUFTYPE parameter describes the BEA TUXEDO output buffer that is returned to the local client program.

Guidelines for Mapping Input Buffers to Input Records

The following sections provide detailed information explaining how to use the INBUFTYPE and INRECTYPE parameters for service calls that originate locally (where local client programs call remote services).

The INBUFTYPE Parameter

The INBUFTYPE parameter is used to specify the buffer type that is provided to a local eLink TCP for TUXEDO gateway when a local client program issues a service request.

Because the gateway determines the type of incoming client buffers automatically at runtime, this parameter is described here for conceptual completeness only.

The INRECTYPE Parameter

The INRECTYPE parameter is used to specify the type, and in some cases the format, of the input record that a particular remote service requires. eLink TCP for TUXEDO uses this information to convert BEA TUXEDO input buffers into records that remote services can process.

You must specify the INRECTYPE parameter when one of the circumstances described in the following table is true.

Circumstance	Explanation
The remote service uses an input record that is structured differently than the client program's input buffer.	In this circumstance, the remote service uses a record that is structured differently than the client program's VIEW, X_C_TYPE, or X_COMMON buffer. For example, the remote service may expect structure members to be sequenced differently.

Circumstance	Explanation
The remote service uses an input record that differs from the client program's input buffer in both type and structure.	In this case, the client program uses a BEA TUXEDO FML buffer and the remote service expects a corresponding record with an appropriate structure.

The INRECTYPE parameter may be omitted if the input buffer is identical, in type and structure, to the record the remote service expects.

Guidelines for Mapping Output Records to Output Buffers

The following sections provide detailed information explaining how to use the OUTRECTYPE and OUTBUFTYPE parameters for service calls that originate locally (where local client programs call remote services and receive output from those services).

The OUTBUFTYPE Parameter

The OUTBUFTYPE parameter is used to specify the type, and in some cases the structure, of the output buffer that a local client program expects. eLink TCP for TUXEDO uses this information to map output records from remote services to the appropriate kinds of output buffers.

The OUTRECTYPE Parameter

The OUTRECTYPE parameter is used to specify the type, and in some cases the format, of the output record that a particular remote service returns to the local eLink TCP for TUXEDO gateway.

You must specify the OUTRECTYPE parameter when one of the circumstances described in the following table is true.

Circumstance	Explanation
The remote service returns an output record that is structured differently than the output buffer the local client program expects.	In this circumstance, the remote service returns a record that is structured differently than the client program's VIEW, X_C_TYPE, or X_COMMON buffer. For example, the structure members of the output record may be sequenced differently than the structure members of the output buffer.
The remote service returns an output record that differs in both type and structure from the output buffer the client program expects.	In this case, the remote service returns a particular record and the local client program expects a corresponding BEA TUXEDO FML buffer.

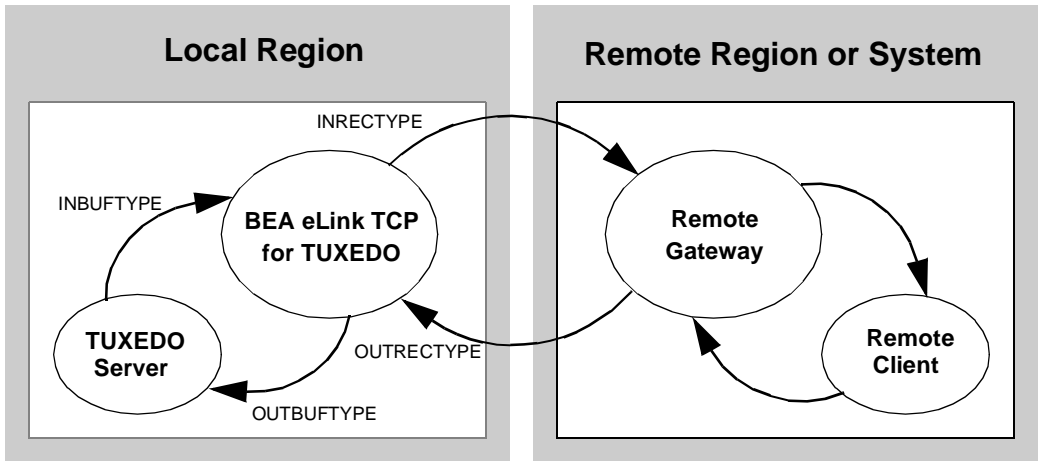
The OUTRECTYPE parameter may be omitted if the remote service returns an output record that is identical, in type and structure, to the output buffer the local client program expects.

Parameters for Remotely Originated Calls

This section takes a closer look at how eLink TCP for TUXEDO handles service calls that originate on remote computers, outside the local BEA TUXEDO region. Also, it explains how the INRECTYPE, INBUFTYPE, OUTBUFTYPE, and OUTRECTYPE parameters can be used to manage the conversion of buffers and records that flow between remote client programs and local services.

In Figure 3-2, a remote client program issues a service request that a remote eLink TCP gateway routes to the local eLink TCP for TUXEDO gateway. The gateway receives the request from the network and passes the request to a local BEA TUXEDO server.

Figure 3-2 How Parameters Are Mapped During Remotely Originated Calls



In this situation, the four configuration parameters that are shown in the figure have the following meanings:

- ◆ The **OUTRECTYPE** parameter describes the output record that the remote client sends to the eLink TCP for TUXEDO gateway.
- ◆ The **OUTBUFTYPE** parameter describes the BEA TUXEDO output buffer that is provided to the local server.
- ◆ The **INBUFTYPE** parameter describes the BEA TUXEDO input buffer that the local server returns to the eLink TCP for TUXEDO gateway.
- ◆ The **INRECTYPE** parameter describes the input record that the local eLink TCP for TUXEDO gateway returns to the remote client program.

Guidelines for Mapping Input Records to Input Buffers

The following sections provide detailed information explaining how to use the **INRECTYPE** and **INBUFTYPE** parameters for service calls that originate on remote systems (where remote client programs call local services).

The INBUFTYPE Parameter

The INBUFTYPE parameter is used to specify the type, and in some cases the structure, of the input buffer that the eLink TCP for TUXEDO gateway expects from a local server. eLink TCP for TUXEDO uses this information to map input buffers from local server programs to the appropriate kind of input records.

Because the gateway determines the type of incoming buffers automatically at runtime, this parameter is described here for conceptual completeness only.

The INRECTYPE Parameter

The INRECTYPE parameter is used to specify the type, and in some cases the format, of the input record that the local eLink TCP for TUXEDO gateway sends to the remote client.

You must specify the INRECTYPE parameter when one of the circumstances described in the following table is true.

Circumstance	Explanation
The remote client program requires an input record that is structured differently than the input buffer the local service provides.	In this circumstance, the remote client program sends a record that is structured differently than the local service's VIEW, X_C_TYPE, or X_COMMON buffer. For example, the structure members of the input record may be sequenced differently than the structure members of the input buffer.
The remote client program requires an input record that differs in both type and structure from the input buffer the local service provides.	In this case, the remote client program requires a particular record and the local service provides a corresponding BEA TUXEDO FML buffer.

You can omit the INRECTYPE parameter if the local server program sends an input buffer that is identical in type and structure to the input record the remote client expects.

Guidelines for Mapping Output Buffers to Output Records

The following sections provide detailed information explaining how to use the `OUTBUFTYPE` and `OUTRECTYPE` parameters for service calls that originate on remote computers (where remote client programs call local services and receive output from those services).

The `OUTBUFTYPE` Parameter

The `OUTBUFTYPE` parameter specifies the buffer type that the local eLink TCP for TUXEDO gateway provides to the local server.

The `OUTRECTYPE` Parameter

The `OUTRECTYPE` parameter is used to specify the type, and in some cases the format, of the output record a particular remote client program sends to the eLink TCP for TUXEDO gateway. eLink TCP for TUXEDO uses this information to convert output records from remote clients into buffers that local client programs can process.

You must specify the `OUTRECTYPE` parameter when one of the circumstances described in the following table is true:

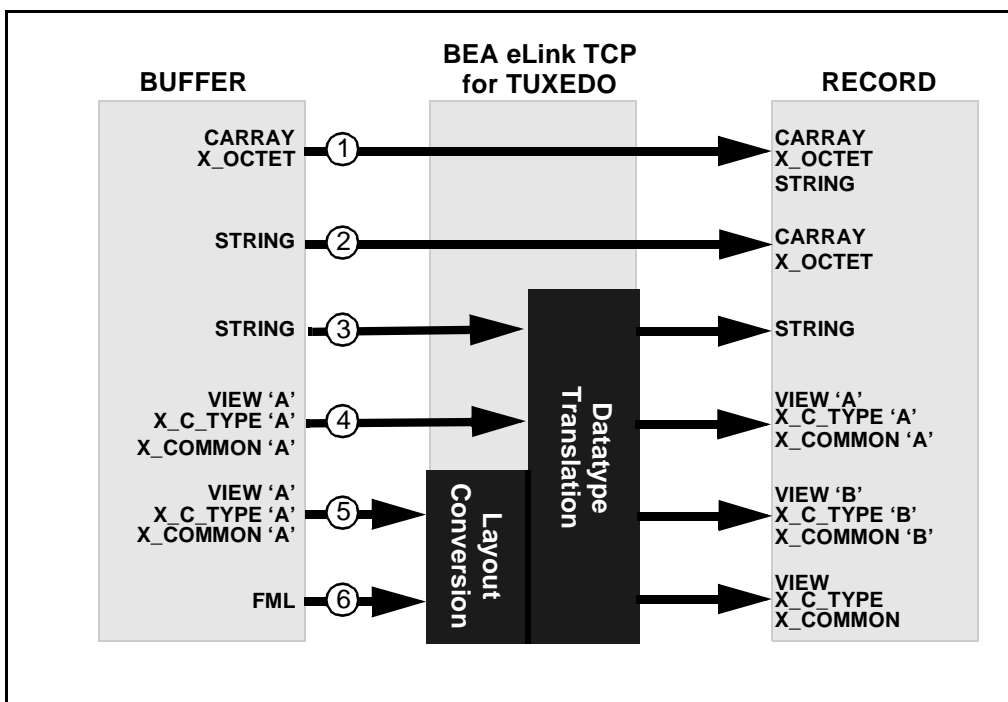
Circumstance	Explanation
The remote client program provides an output record that is structured differently than the local service's output buffer.	In this circumstance, the remote client program provides a record that is structured differently than the local service's <code>VIEW</code> , <code>X_C_TYPE</code> , or <code>X_COMMON</code> buffer. For example, the local server program may expect structure members to be sequenced differently.
The remote client program provides an output record that differs from the local service's output buffer in both type and structure.	In this case, the remote client outputs a record and the local client program expects a corresponding BEA TUXEDO FML buffer.

The OUTRECTYPE parameter may be omitted if the local service's output buffer is identical, in type and structure, to the record the remote client program provides.

Mapping Buffers to Records

Figure 3-3 shows all the possibilities for mapping buffers to records. The eLink TCP for TUXEDO gateway is responsible for mapping buffers to records, based on information it finds in the eLink TCP for TUXEDO configuration. This mapping occurs for TUXEDO client requests and TUXEDO server responses.

Figure 3-3 Buffer to Record Mappings



Setting the INBUFTYPE and INRECTYPE Parameters

Here are some comments about the mapping possibilities that are shown in Figure 3-3 and some suggestions for setting the INBUFTYPE and INRECTYPE parameters.

-
- 1 BEA TUXEDO CARRAY input buffers can be copied to CARRAY input records. A CARRAY buffer contains raw data that is not converted or translated. Set the INBUFTYPE parameter to CARRAY, and omit the INRECTYPE parameter.

CARRAY input buffers can also be copied to STRING input records. This creates a string that goes through no conversion and no translation. The resultant buffer is the length of the original CARRAY buffer. Since all characters are copied, if the CARRAY buffer contains null characters, it affects the buffer when later handled as a STRING. The INBUFTYPE parameter should be set to CARRAY and the INRECTYPE parameter should be set to STRING.
 - 2 BEA TUXEDO STRING input buffers can be mapped to CARRAY input records. No data conversion or translation is performed. The STRING buffer is copied through the leftmost null character only. Set the INBUFTYPE parameter to STRING and the INRECTYPE parameter to CARRAY.
 - 3 BEA TUXEDO STRING input buffers can be mapped to STRING input records. The buffer goes through data type translation (i.e., ASCII to EBCDIC). Set the INBUFTYPE parameter to STRING, and omit the INRECTYPE parameter.
 - 4 BEA TUXEDO VIEW input buffers can be mapped to identical BEA TUXEDO VIEW input records. In this situation, the data structure that the remote service expects is identical to the data structure the client program uses. There is no need to create a new VIEW definition. Instead, specify the input record type (VIEW) and the name of the existing VIEW definition (the one the client program currently uses) for both the INBUFTYPE parameter, and omit the INRECTYPE parameter.
 - 5 BEA TUXEDO VIEW input buffers can be mapped to VIEW input records—in any combination. However, in this situation, the data structure that the remote service expects (designated as VIEW ‘B’ mapping possibilities in Figure 3-3) differs from the data structure the client program uses (designated as VIEW ‘A’ in Figure 3-3). Consequently, you must
 1. Create a VIEW definition for the data structure that the remote service expects.
 2. Specify the desired record type and the name of this VIEW definition with the INRECTYPE parameter.
 3. Set the INBUFTYPE parameter to `VIEW:original-viewname`.
-

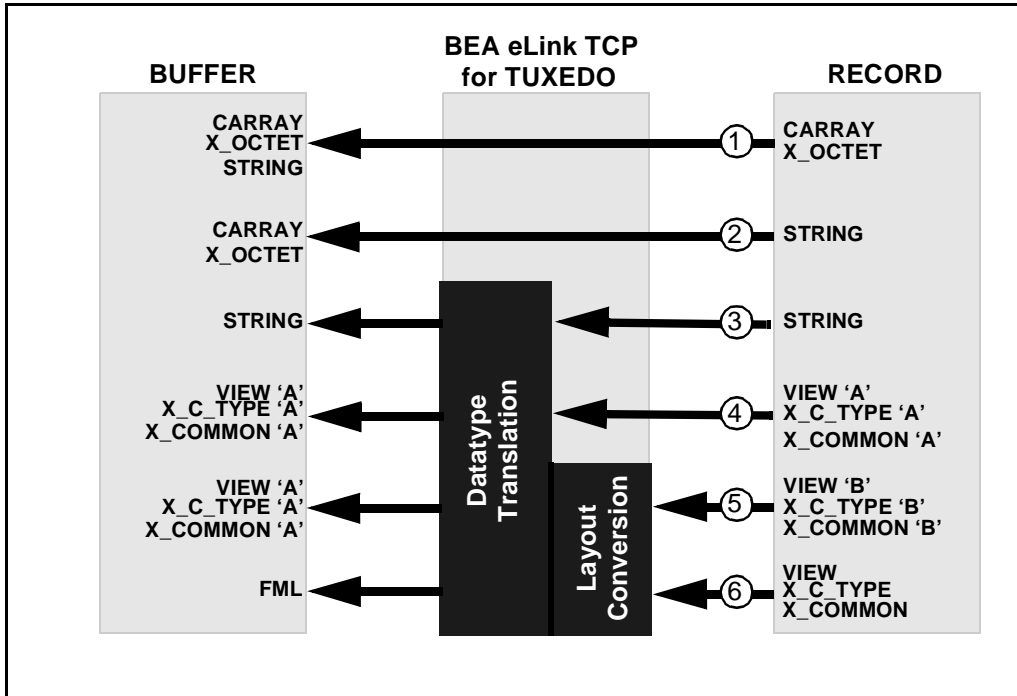
- 6 Before a BEA TUXEDO FML input buffer can be sent to a remote service that does not support FML, it must be mapped to one of the following input record types: VIEW, X_C_TYPE, or X_COMMON. Also, you must create a VIEW definition for the input data structure that the remote service expects. Set INBUFTYPE to FML and INRECTYPE to VIEW:*viewname*.

Note: In the DMCONFIG file, if FML or FML32 are specified as INBUFTYPE or OUTBUFTYPE, the type must be followed by a colon (:). (Example: INBUFTYPE="FML32:")

Mapping Records to Buffers

Figure 3-4 shows all the possibilities for mapping records to buffers. The eLink TCP for TUXEDO gateway is responsible for mapping records to buffers, based on information it finds in the eLink TCP for TUXEDO configuration. This mapping occurs for remote client requests and remote server responses.

Figure 3-4 Record to Buffer Mappings



Setting the OUTRECTYPE and OUTBUFTYPE Parameters

Here are some comments about the mapping possibilities that are shown in Figure 3-4 and some suggestions for setting the OUTRECTYPE and OUTBUFTYPE parameters (for service calls that originate locally).

- 1 BEA TUXEDO CARRAY output records can be copied to CARRAY output buffers. A CARRAY buffer contains raw data that is not converted or translated. Set the OUTBUFTYPE parameter to CARRAY. The OUTRECTYPE parameters need not be set.

BEA TUXEDO CARRAY output records can also be copied to STRING output buffers. This creates a string that goes through no conversion and no translation. The resultant buffer is the length of the original CARRAY buffer. Since all characters are copied, if the CARRAY buffer contains null characters, it affects the buffer when later handled as a STRING. The OUTRECTYPE should be set to CARRAY and OUTBUFTYPE should be set to STRING.

-
- | | |
|---|--|
| 2 | BEA TUXEDO STRING output records can be mapped to CARRAY output buffers. There is no data conversion or translation performed. The STRING buffer is copied through the leftmost null characters only. Set OUTRECTYPE to STRING and OUTBUFTYPE to CARRAY. |
|---|--|
-
- | | |
|---|--|
| 3 | BEA TUXEDO STRING output records can be mapped to STRING output buffers. The buffer goes through datatype translation (i.e., ASCII to EBCDIC). OUTBUFTYPE and OUTRECTYPE parameters are set to STRING. |
|---|--|
-
- | | |
|---|--|
| 4 | BEA TUXEDO VIEW output records can be mapped to identical BEA TUXEDO VIEW output buffers. In this situation, the data structure that the remote service returns is identical to the data structure the local client program expects. There is no need to create a new VIEW definition. Instead, specify the VIEW buffer type and the name of the existing VIEW definition with the OUTBUFTYPE parameter. The OUTRECTYPE parameter can be set to VIEW: <i>viewname</i> , but it is not mandatory. |
|---|--|
-
- | | |
|---|---|
| 5 | <p>BEA TUXEDO VIEW output records can be mapped to VIEW output buffers—in any combination. However, in this situation, the data structure that the remote service returns (designated as VIEW ‘B’ in Figure 3-4) differs from the data structure the client program expects (designated as VIEW ‘A’ in Figure 3-4). To facilitate the conversion process, perform the following tasks:</p> <ol style="list-style-type: none">1. Create a VIEW definition for the data structure that the remote service returns.2. If the name given to the VIEW definition is different than the name that the remote service returns (that is, ATMI buffer subtype), specify the output record type and the name of VIEW ‘B’ with the OUTRECTYPE parameter. (By doing this, you override the value the eLink TCP for TUXEDO requester automatically detects.)3. Specify the output buffer type and the name of an existing view (VIEW ‘A’ in the figure) specified in the OUTBUFTYPE parameter. |
|---|---|
-
- | | |
|---|---|
| 6 | <p>BEA TUXEDO VIEW output records can be mapped to FML output buffers. To facilitate the conversion process, you must perform the following tasks:</p> <ol style="list-style-type: none">1. Create a VIEW definition that describes the data structure that the remote service returns.2. If the name given to the VIEW definition is different than the name that the remote service returns (that is, the ATMI buffer subtype), specify the output record type and the name of your VIEW definition with the OUTRECTYPE parameter. (By doing this, you override the value the eLink TCP for TUXEDO requester automatically detects.)3. Set the OUTBUFTYPE parameter to FML. <p>Note: In the DMCONFIG file, if FML or FML32 are specified as INBUFTYPE or OUTBUFTYPE, the type must be followed by a colon (:). (Example: INBUFTYPE="FML32:")</p> |
|---|---|
-

Creating VIEW Definitions to Facilitate Buffer Conversion

VIEW definitions are used to describe input and output records that are sent to and received from remote systems. They describe data elements and indicate how data elements are typed and sequenced. Based on these descriptions, eLink TCP for TUXEDO translates field data types as required to maintain transparency between dissimilar systems.

You should create VIEW definitions before you configure eLink TCP for TUXEDO. For complete information about VIEW definitions and related topics, see the *BEA TUXEDO Programmer's Guide*.

BEA eLink TCP for TUXEDO buffer and record conversion capabilities are extremely powerful and flexible. The key to maximizing these capabilities is to thoroughly understand the BEA TUXEDO VIEW definition mechanism.

VIEW definitions make it possible to specify composite data structures that can be used:

- ◆ On different kinds of machines
- ◆ With different programming languages

Preparing VIEW Definitions

After determining the input and output record layouts for the remote application programs you are working with, you need to:

- ◆ Create standard BEA TUXEDO VIEW definitions in files.
- ◆ Run the `viewc` or `viewc32` VIEW compiler.
- ◆ Set the `VIEWFILES`, `VIEWDIR`, `FIELDTBLS`, and `FLDTBLDIR` environment variables, using a BEA TUXEDO `ENVFILE` if necessary (so that eLink TCP for TUXEDO servers can locate binary VIEW files and field table files at run time).

After these tasks are complete, you can specify VIEW definitions in the `GWICONFIG` and `DMCONFIG` files (by associating names of VIEW definitions with the `INRECTYPE`, `OUTRECTYPE`, `INBUFTYPE`, and `OUTBUFTYPE` parameters, as required).

For detailed information about configuring eLink TCP for TUXEDO, see “Configuring BEA eLink TCP for TUXEDO.”

Note: FML fields must be specified for all VIEWs that eLink TCP for TUXEDO converts. In other words, any VIEW that you specify as an INRECTYPE, OUTRECTYPE, INBUFTYPE, or OUTBUFTYPE must be defined with appropriate FML fields (no dashes in the FNAME column of the VIEW definition). For the FML fields to match, you must compile these VIEWs without the -n option specified.

Translating Data

When a local client program sends data to (or receives data from) a service routine on a different kind of computer, eLink TCP for TUXEDO automatically translates data as required. Translation involves changing the representation of intrinsic data types by changing attributes such as word length and byte order.

BEA eLink TCP for TUXEDO automatically translates input and output data as required, following rules that are described in the following section. Read the information carefully before you create VIEW definitions (to facilitate buffer conversion) and configure eLink TCP for TUXEDO.

Basic rules for how eLink TCP for TUXEDO translates data are described in the following subsection. For detailed information about how eLink TCP for TUXEDO handles string and numeric data, refer to the “NULL Characters in String Length Calculations (C Programs)” section.

Data Translation Rules

Here are the data translation rules that eLink TCP for TUXEDO follows:

- ◆ CARRAY fields are passed untranslated as sequences of bytes.
- ◆ STRING and CHAR fields undergo ASCII-to-EBCDIC translation (if needed).
- ◆ SHORT and LONG fields are translated to S9(4) COMP and S9(9) COMP, respectively.
- ◆ FLOAT and DOUBLE fields translate to COMP-1 and COMP-2, respectively.

Warning: `dec_t` cannot be used with VIEW translations.

Note: BEA TUXEDO provides a field type named `dec_t` that supports decimal values within VIEWS. eLink TCP for TUXEDO translates these fields into machine independent representations of packed decimals. For example, `dec_t(m,n)` becomes `S9(2*m-(n+1))V9(n) COMP-3`. Therefore, a decimal field with a size of 8,5 corresponds to `S9(10)V9(5) COMP-3`.

The following table summarizes the relationships.

Remote Data Type	Description	View Field Type/Length
PIC X(n)	Alphanumeric characters	string / n
PIC X	Single alphanumeric character	char
PIC X(n)	Raw bytes	carray / n
PIC X	Single numeric byte	carray / 1
PIC S9(4) COMP	16-bit integer	short
PIC S9(9) COMP	32-bit integer	long
COMP-1	Single-precision floating point	float
COMP-2	Double-precision floating point	double
PIC S9((m+(n+1))/2)V9(n) COMP-3	Packed decimal	dec_t / m,n

NULL Characters in String Length Calculations (C Programs)

When you create VIEW definitions for input and output buffers that are used by C language applications, you must specify extra characters for terminating NULL characters that are used in string fields.

For example, when a local application program expects a 10-byte string in an output buffer, you would specify 11 for that field—10 for the string plus 1 for the terminating NULL character.

NULL Characters in String Length Calculations (COBOL Programs)

When you create VIEW definitions for input and output buffers that are used by COBOL language applications, do not specify extra characters for terminating NULL characters that are used in string fields.

For example, when a remote COBOL application program expects 10 characters in an input record, you would specify 10 for that field, not 10 plus 1 (for the terminating NULL character).

Note: Although eLink TCP for TUXEDO does not require strings to be NULL-terminated, it respects NULL termination. Therefore, when eLink TCP for TUXEDO detects a NULL (zero) character within a string, it does not process any subsequent characters. To pass full 8-bit data that contains embedded NULL values, use a CARRAY type field or buffer.

BEA eLink TCP for TUXEDO provides standard character translation from ASCII-to-EBCDIC and EBCDIC-to-ASCII. eLink TCP for TUXEDO automatically performs this translation on the string data type.

Converting Numeric Data

Numeric data can easily be converted into different data types, provided that you have enough range in the intermediate and destination types to handle the maximum value you need to represent.

For example, you can convert numeric values into strings (and the reverse). For example, while FML buffers do not directly support the `dec_t` type, you can place decimal values in string fields and map these to `dec_t` fields within VIEW definitions.

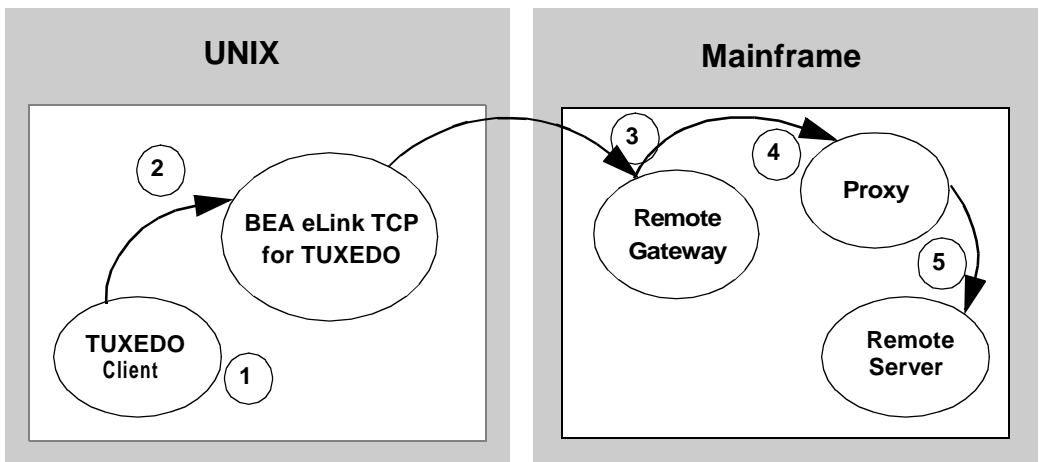
Enabling eLink TCP Security

The eLink TCP product supports a security feature that allows a requester from TUXEDO to pass a USERID requirement through the OTMA or CICS server interfaces for verification through RACF.

Security Checking from UNIX to Mainframe

Figure 3-5 depicts the process flow for security verifications from eLink TCP for TUXEDO on UNIX to a mainframe.

Figure 3-5 Security Checking for UNIX to Mainframe Transactions



1. Verify the user is a valid TUXEDO user. For valid users, access is given; for invalid users, access is rejected.
2. Verify user name (reviewing the `tpusr` file), group (reviewing the `tpgrp` file), and ACL (reviewing the `tpacl` file). If all three pass, the transaction request processes. If any one of the three are rejected, the transaction request stops and a security violation occurs.

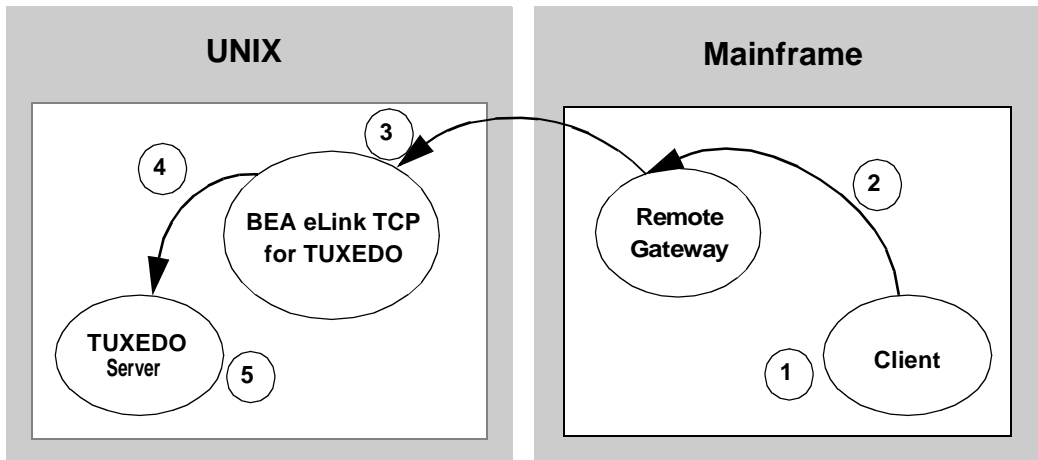
Note: The userids in these files must match in the BEA TUXEDO and the mainframe environments or a security violation occurs.

3. Accept the transaction request at the mainframe gateway based on the request coming from a *trusted source*. No password is passed.
4. Verify the user name against the security system (such as RACF). If the user name is valid, a proxy is spawned. If the user name is not valid, the request is rejected and a security violation occurs.
5. Complete the transaction request to the server if a proxy is spawned.

Security Checking from Mainframe to UNIX

Figure 3-6 depicts the process flow for security verifications from a mainframe to eLink TCP for TUXEDO on UNIX.

Figure 3-6 Security Checking for Mainframe to UNIX Transactions



1. Verify the user name against the security system (such as RACF). If the user name is valid, a proxy is spawned. If the user name is not valid, the request is rejected and a security violation occurs.
2. Pass the transaction request from the client to the mainframe gateway.

3. Accept the transaction request at the UNIX gateway based on the request coming from a *trusted source*. No password is passed.
4. Determine the `appkey` corresponding to the supplied user name. Decode the `appkey` to obtain the user and group numbers. Verify the user name against TUXEDO security. If the user name passes these checks, the transaction request is accepted. If the user name fails, the request is rejected and a security violation occurs.
5. The TUXEDO server will verify the user's access to the requested service by examining the access control list (ACL). If access is denied, an error is returned to the client. If access is granted, the server processes the request.

Setting Up Security

Complete the following tasks to enable the security feature.

- ◆ Code SECURITY in the BEA TUXEDO UBBCONFIG file. Refer to the *BEA TUXEDO Administration Guide* for more information.
- ◆ Set up user, group, and ACL files. Refer to the *BEA TUXEDO Administrator's Guide* for more information.
Note: The user information in these files must match in the BEA TUXEDO and the mainframe environments or a security violation occurs.
- ◆ Code the security parameter in your eLink TCP for TUXEDO configuration file (GWICONFIG). For GWICONFIG syntax and parameter definitions, refer to "Configuring BEA eLink TCP for TUXEDO."

Sample Security Files

Part of the process for setting up security for eLink TCP requires you to have user, group, and ACL files. The following sections include these sample files.

User Files

The following sample is a user file that includes user names, encrypted passwords, a userid number, group number, and a client name.

Listing 3-1 Sample User (tpusr) File

```
#illen:w2ZMOKeJmiU0M:1:0:TPCLTNM,someguy::
#illen:0YzvQeqzcNz56:1:0:TPCLTNM,*::
#eke:x3vG37eOqh0XE:2:0:TPCLTNM,*::
#illen:0YzvQeqzcNz56:1:1:TPCLTNM,*::
#illen:0YzvQeqzcNz56:1:2:TPCLTNM,*::
john:x3vG37eOqh0XE:2:1:TPCLTNM,*::
jim:0YzvQeqzcNz56:1:1:TPCLTNM,*::
richard:IxqosKHu5Q3BA:3:1:TPCLTNM,*::
JDOE:zBMWVUBNNBVgo:4:0:TPCLTNM,*::
smith:ULfrJzAeyGAD2:5:0:TPCLTNM,*::
```

Lines that begin with the pound sign (#) are users that have been changed or deleted by `tpusrmod` or `tpusrdel`.

Group File

The following sample is a group file that specifies the names and indexes of groups.

Note: The `tpgrp` file is only necessary when specifying ACL or MANDATORY_ACL modes for security. If you specify USER_AUTH for security, you can assign users to groups, but they do not correlate to the groups used for security by the remote system.

Listing 3-2 Sample Group (tpgrp) File

```
good::1:
bad::2:
```

ACL File

The `tpacl` file correlates a group and the services to which that group has access. In the `tpacl` file, the first field specifies what is protected, the second field specifies the type of object being protected (specified in the first field), and the third field specifies the group that has access to the object.

In the following example, only users in group 1 (john, jim, richard) can access TOLOWER, and only users in group 2 can access TOUPPER.

Note: The `tpacl` file is only necessary when specifying ACL or MANDATORY_ACL modes for security.

Listing 3-3 Sample ACL (tpacl) File

```
TOLOWER:SERVICE:1:
TOUPPER:SERVICE:2:
```

Considerations for Error Handling

There are several ways that local client programs can learn about application errors that occur on remote systems. For example:

- ◆ Application failures can be communicated through error indicators that remote systems send to the local eLink TCP for TUXEDO gateway.
- ◆ Service routines can include information about failures in output records that are returned to client programs.

4 Configuring BEA eLink TCP for TUXEDO

The following configuration files must be set up prior to running BEA eLink for Mainframe TCP for TUXEDO (hereafter referenced as eLink TCP for TUXEDO):

- ◆ UBBCONFIG
- ◆ GWICONFIG
- ◆ DMCONFIG

This document explains the following tasks for configuring eLink TCP for TUXEDO:

- ◆ Updating the BEA TUXEDO UBBCONFIG File
- ◆ Specifying Parameters in the GWICONFIG File
- ◆ Defining Gateway Configurations in the DMCONFIG File

Updating the BEA TUXEDO UBBCONFIG File

As with any TUXEDO server, you must establish a server group for eLink TCP for TUXEDO by adding entries for eLink TCP for TUXEDO gateway in the UBBCONFIG file for the local region. Specifically, you must add records to:

- ◆ The *GROUPS section
- ◆ The *SERVERS section

See the *BEA TUXEDO Administrator's Guide* for more information about the UBBCONFIG file. For information about the UBBCONFIG, refer to “Updating the *GROUPS Section to Establish a Server Group,” and “Updating the *SERVERS Section.”

Updating the *GROUPS Section to Establish a Server Group

To establish a server group for eLink TCP for TUXEDO, you must add the following items to the *GROUPS section of the BEA TUXEDO UBBCONFIG file:

- ◆ A name for the eLink TCP for TUXEDO server group using the *groupname* variable
 - ◆ The logical machine name for the system on which eLink TCP for TUXEDO is installed using the *LMID* parameter
- Note:** For Windows NT, the LMID must be in uppercase
- ◆ A group number for the eLink TCP for TUXEDO server group using the *GROUPNO* parameter

Syntax

The syntax of the configuration file entry is as follows:

Listing 4-1 Syntax for *GROUPS Section of UBBCONFIG File

```
groupname LMID=logical_machine_identifier  
GROUPNO=group_number
```

The variable definitions follow.

Variable	Description
<i>groupname</i>	Specifies the name you select for the eLink TCP for TUXEDO server group.
<i>logical_machine_identifier</i>	Specifies the logical machine identifier for the system on which eLink TCP for TUXEDO is installed.
<i>group_number</i>	Specifies the number you assign to the eLink TCP for TUXEDO server group

Example

Here is an example of a UBBCONFIG entry that establishes a server group:

Listing 4-2 Establishing a Server Group

```
NODE2GATE    LMID=NODE2
              GROUPNO=1
```

Updating the *SERVERS Section

This section explains how to specify eLink TCP for TUXEDO servers in the BEA TUXEDO configuration.

BEA eLink TCP for TUXEDO provides the gateway you need to list in the SERVERS section of the BEA TUXEDO UBBCONFIG configuration file:

For each eLink TCP for TUXEDO server group, you can specify one gateway. There must also be entries for domain administration (DMADM) and gateway administration (GWADM) servers.

Syntax

The syntax of each configuration file entry is as follows.

Listing 4-3 Syntax for *SERVER Section of UBBCONFIG File

```
DMADM          SVRGRP=groupname SRVID=integer
GWADM          SVRGRP=groupname SRVID=integer
GWIDOMAIN      SVRGRP=groupname SRVID=integer
                CLOPT=" -A  "
```

The following table describes the parts of the syntax.

Part	Description
GWIDOMAIN	Specifies the name of the eLink TCP for TUXEDO gateway.
DMADM	Specifies the name of the BEA TUXEDO supplied domain administration server.
GWADM	Specifies the name of the BEA TUXEDO supplied gateway administration server.
<i>groupname</i>	Specifies the name you select for the eLink TCP for TUXEDO server group. This is the group with which the server in question is associated.
<i>integer</i>	Specifies the number you assign to the eLink TCP for TUXEDO server.
CLOPT=" -A "	Specifies the command line option string that is used to initialize eLink TCP for TUXEDO when it is invoked. This is the default for BEA TUXEDO server processes.

For more information about BEA TUXEDO servers and related configuration parameters, see the *BEA TUXEDO Administrator's Guide*.

Other Options for Configuring Servers

As with other TUXEDO servers, you can use the full range of BEA TUXEDO system server boot options with eLink TCP for TUXEDO servers.

The `-r` option may be particularly useful. Using this timestamp recording option, together with the `txrpt` report command, you can analyze remote service response times. Of course, the process of gathering statistics creates overhead. Consequently, you should use this option selectively.

The `RQADDR` parameter is also helpful. It enables you to simplify system administration tasks by assigning intuitive names to eLink TCP for TUXEDO server queues. To do so, specify the `nodename`, `LMID` or `SRVGRP` value as part of the queue name (as shown in the following example). Each server entry must have a unique `RQADDR` value. Also, you cannot use the `RQADDR` parameter if the BEA TUXEDO `MIN` or `MAX` parameters are set to values that are greater than 1.

Here are two final points about boot options:

- ◆ Boot options must be specified with the `CLOPT` parameter (before the `CLOPT` double-dash separator).
- ◆ Because eLink TCP for TUXEDO dynamically advertises services that are listed in its initialization file, you should not advertise services by using the `-s` option.

For more information about these and other boot options, see `servopts(5)` in the *BEA TUXEDO Administrator's Guide*.

Example

Here is an example of a UBBCONFIG entry that configures a eLink TCP for TUXEDO gateway.

Listing 4-4 Configuring a eLink TCP for TUXEDO Gateway

```
SERVERS
DMADM      SRVGRP=LIAISON  SRVID=1
GWADM      SRVGRP=REQIMS   SRVID=2
GWIDOMAIN  SRVGRP=REQCICS  SRVID=3
           CLOPT="-A"
```

Specifying Parameters in the GWICONFIG File

The GWICONFIG file is the mechanism that system administrators use to configure eLink TCP for TUXEDO. The particular file the gateway uses is determined by the environment variable GWICONFIG. The configuration file is similar to the TUXEDO Transaction Manager UBBCONFIG file, both in structure and in composition. This makes it possible for experienced BEA TUXEDO administrators to configure eLink TCP for TUXEDO without extensive training.

Note: GWICONFIG is a generic filename. You are free to choose other filenames just as with the TUXEDO UBBCONFIG file and DMCONFIG file. Be sure that the name of the GWICONFIG file is specified in the GWICONFIG environment variable. Also, the GWICONFIG file should be saved to the application directory.

The GWICONFIG file is divided into the following required sections:

◆ ***GLOBAL**

Describes certain general characteristics of all eLink TCP for TUXEDO gateways.

◆ ***NATIVE**

Describes all native systems. Use the gateway name, specified in the `GWI_GWNAME` environment variable, to distinguish different gateways in the same configuration file.

◆ ***FOREIGN**

Describes all foreign systems.

◆ ***LOCAL_SERVICES**

Describes local services that are accessible in the BEA TUXEDO domain through the eLink TCP for TUXEDO gateway. Each local service is linked to a native system using the `NATIVE` parameter. For each local service, eLink TCP for TUXEDO uses the TCP port number and IP address of the corresponding native system to establish a TCP listening endpoint.

◆ *REMOTE_SERVICES

Describes remote services that are accessible from the BEA TUXEDO domain. Each remote service describes a TCP service on a foreign system. For each remote service, the FOREIGN parameter defines the name of the foreign system. eLink TCP for TUXEDO uses the TCP port number and IP address of the foreign system to connect to the remote TCP service.

Warning: The GWICONFIG file *must* contain the required sections in the order shown in Listing 4-5.

Listing 4-5 Sample GWICONFIG File

```
*GLOBAL
                                NWDEVICE="/dev/tcp"
                                CONNECT_TIME=20
#                               IDLE_TIME=90
                                OUTREQ_TIME=20
                                LATENCY=-2
                                SECURE=N
                                MULTIPLEX=2
                                DFLTWRAP="TPS"
                                DFLTTYPE="MVS"

*NATIVE
LOCAL                          IPADDR="//beasun2"
                                TCP_PORT=9002
                                IDLE_TIME=20
#                               MULTIPLEX=4

*FOREIGN
RIGHTY                          WRAP="TPS"
                                TYPE="MVS"
                                IPADDR="//beasun2"
                                TCP_PORT=9004
                                MULTIPLEX=2
                                IDLE_TIME=30
                                RMTACCT="zeke"
                                PASSWORD="maple"

HP10                            WRAP="TPS"
                                TYPE="MVS"
                                IPADDR="//dalhp10"
                                TCP_PORT=0x7000
                                RMTACCT="zeke"
                                PASSWORD="maple"
#                               IDLE_TIME=60
```

```
MIKE          WRAP="TPS"
              TYPE="MVS"
              IPADDR="//dalvs3"
              TCP_PORT=9001
              MULTIPLEX=6

CURTIS        WRAP="TPS"
              TYPE="MVS"
              IPADDR="//dalvs2"
              TCP_PORT=9002
              CICS=Y

*LOCAL_SERVICES
TUXTOUPPER    NATIVE="LOCAL"
#            SECURE=Y

ECHO          NATIVE="LOCAL"

*REMOTE_SERVICES
TUXTOLOWER    FOREIGN="RIGHTY"
#TUXTOLOWER    FOREIGN="HP10"
              OUTREQ_TIME=20

BEASVR07      FOREIGN="MIKE"
              OUTRECTYPE="VIEW:weird"

TST1V         FOREIGN="HP10"
```

Defining the *GLOBAL Section of the GWICONFIG File

The following sections describe parameters that are associated with the *GLOBAL section of the GWICONFIG file. These parameters describe global characteristics for the gateway.

Syntax

The format of the *GLOBAL section of the GWICONFIG file follows.

Listing 4-6 Syntax for *GLOBAL Section of GWICONFIG File

```
[ # A comment (from "#" to end of line)]

*GLOBAL
[ NWDEVICE=TCP_device ]
[ CONNECT_TIME=n]
[ IDLE_TIME=n]
[ OUTREQ_TIME=n]
[ LATENCY=n|1 ]
[ SECURE=Y|N ]
[ MULTIPLEX=n|1 ]
[ DFLTWRAP=wrapper name ]
[ DFLTTYPE=translation type ]
```

Required Parameters

There are no required parameters for the *GLOBAL section of the GWICONFIG file.

Optional Parameters

This section describes the optional parameters that you set in the *GLOBAL section.

NWDEVICE="*TCP_device*"

specifies the network device to be used for communication with remote gateways. The default value for NWDEVICE is `"/dev/tcp"`.

CONNECT_TIME=*n* seconds

specifies the number of seconds the gateway will wait to establish a connection. The default is no timeout.

IDLE_TIME=*n* seconds

specifies the number of seconds a connection can be idle before timing out. The default is no idle timeout.

OUTREQ_TIME=*n* seconds

specifies the default timeout value, in seconds, for requests sent to foreign gateways. There is no default for this parameter.

LATENCY=*n* seconds

specifies the number of seconds to be deducted from the timeout value sent to a remote gateway for a TUXEDO client request. This increases the likelihood that the remote gateway detects a timeout before the local gateway. The default is 1.

SECURE=Y | N

specifies whether the eLink TCP for TUXEDO gateway supplies user information to local and remote services.

If **SECURE**=Y, then the eLink TCP for TUXEDO gateway supplies user information to remote services and applies remote user information to local services using the `appkey`.

If **SECURE**=N, then the eLink TCP for TUXEDO gateway does not supply user information to remote services and does not apply remote user information to local services using the `appkey`. The default is N.

Note: If **SECURE**=N and the TUXEDO domain is set with **SECURITY**=ACL in the `UBBCONFIG` file, a request to a local service can fail even if ACLs are in place.

MULTIPLEX=*n*

specifies the maximum number of outstanding requests per connection that the local gateway can support. The default is 1.

DFLTWRAP=*wrapper name*

specifies the name of the default wrapping library to use for wrapping and unwrapping messages for machines without a `*FOREIGN` section, or without a `WRAP` parameter in the `*FOREIGN` section. A corresponding wrapper object `WRAP<wrapper name>` must exist. The wrapper name `TPS` is used in most cases. Specify `TPSD` if data area security is used. For more information about data area security, refer to “Data Area Security.” The default is `TPS`.

DFLTTYPE=system type

specifies the default foreign system type for encoding and decoding message buffers. If you specify `TYPE` in the `*FOREIGN` section, that `TYPE` definition overrides this default value. **DFLTYPE**=MVS is the default value. For normal C-to-COBOL encoding, specify **DFLTYPE**=“MVSC” to enable COBOL data encoding.

Note: For more information about **DFLTYPE** values `MVS` and `MVSC`, refer to “COBOL Data Encoding.”

Defining the *NATIVE Section of the GWICONFIG File

The following sections describe parameters that are associated with the *NATIVE section of the GWICONFIG file. These parameters are specific to the local system. You can specify multiple native systems in the same configuration file allowing multiple gateway processes to access the same configuration file. This makes a single repository of connectivity services. The link between the gateway process and the native system entry is made through the GWINAME environment variable.

Syntax

The format of the *NATIVE section of the GWICONFIG file follows.

Listing 4-7 Syntax for *NATIVE Section of GWICONFIG File

```
[ # A comment (from "#" to end of line)]
*NATIVE
    <GATEWAY_NAME>
[ IPADDR=ip_address ]
[ TCP_PORT=port number ]
[ IDLE_TIME=n ]
[ MULTIPLEX=n ]
[ POLL_TIME=n ]
[ MAXCONNECT=n ]
```

Required Parameters

The required parameter described in this section must be set for each local system that you specify in the *NATIVE section.

<GATEWAY_NAME>

a 1-78 alphanumeric character string that represents the gateway identification passed in the GWINAME environment variable.

Optional Parameters

The optional parameters described in this subsection may be set for each local system that you specify in the *NATIVE section.

IPADDR=*ip_address*

specifies the IP address for the local system. The IPADDR can be in the hexadecimal format 0xaaaaaaaa, the dotted decimal format *//#. #. #. #*, or the DNS format *//host[.domainname]*. If you do not specify this parameter, the IP address of the local host machine is looked up and used.

TCP_PORT=*port number*

specifies the local port number used for listening for services. This parameter is optional if no local services are advertised. If you do not specify TCP_PORT, a listener port is not created.

IDLE_TIME=*n*

specifies the number of seconds a connection can remain idle before being disconnected.

MULTIPLEX=*n*

specifies the number of outstanding requests per connection that the local gateway can support.

POLL_TIME=*n*

specifies the polling timeout (in microseconds) to be used in polling for TUXEDO messages. The range of values for this parameter is 100,000-10,000,000. The default timeout is 250,000 microseconds.

MAXCONNECT=*n*

specifies the number of connections into the gateway from remote hosts. If the remote system attempts more connections than this parameter specifies, the remote systems will be disconnected. The default is no maximum.

Defining the *FOREIGN Section of the GWICONFIG File

The *FOREIGN section of the GWICONFIG file contains parameters that collectively describe foreign systems.

Syntax

The format of the *FOREIGN section of the GWICONFIG file is as follows:

Listing 4-8 Syntax for *FOREIGN Section of GWICONFIG File

```
[ # A comment (from "#" to end of line)]
*FOREIGN
  <SYSTEM_NAME>
  IPADDR=ip_address
  [TYPE=system_type]
  [WRAP=wrapper name]
  [TCP_PORT=port number]
  [MULTIPLEX=n sessions]
  [IDLE_TIME=n seconds]
  [RMTACCT="userid" ]
  [PASSWORD="password" ]
  [CICS=Y | N]
  [CICSHAND=<name>]
  [MAXCONNECT=n]
  [CONNSYNC=Y|N]
  [CONNECT_TIME=n]
  [CICSDATA="string" ]
```

Required Parameters

The required parameters described in this subsection must be set for each foreign system that you specify in the *FOREIGN section.

<*SYSTEM_NAME*>

a 1-78 alpha-numeric character string that represents the foreign system name.

IPADDR=*ip_address*

specifies the IP address for the remote system. The IPADDR can be in the hexadecimal format 0xaaaaaaaa, the dotted decimal format //#. #. #. #, or the DNS format //host[. domainname].

Optional Parameters

The optional parameters described in this subsection may be set for each foreign system that you specify in the *FOREIGN section.

TYPE=*system type*

specifies the foreign system type for encoding and decoding the TUXEDO buffers (application data). TYPE values of MVS and MVSC support

C-to-COBOL or COBOL data encoding. If you do not specify TYPE, the value in DFLTYPE in the *GLOBAL section is used.

Note: For more information about MVS and MVSC TYPE values, refer to “COBOL Data Encoding.”

WRAP=*wrapper name*

specifies the name of the wrapping entry to use for wrapping and unwrapping messages for this host. A corresponding wrapper object wrap<wrapper name> must exist. The wrapper name TPS is used in most cases. Specify TPSD if data area security is used. For more information about data area security, refer to “Data Area Security.” If WRAP is not specified in the *FOREIGN section, the value for DFLTWRAP in the *GLOBAL section is used.

TCP_PORT=*port number*

specifies the port number of the foreign gateway. This parameter is optional if remote services are not defined for this foreign system. The default is no port number.

MULTIPLEX=*n* sessions per connection

specifies the maximum number of sessions per connection that the local gateway can support. The default is 1 session per connection.

IDLE_TIME=*n* seconds

specifies the number of seconds a connection can remain idle before being disconnected. The default is no idle timeout.

CICS=Y | N

specifies whether to send control information to the IBM TCP/IP listener for use with the eLink TCP for CICS gateway. The default is N.

Warning: If CICS=Y and you are not using IBM TCP/IP or your remote gateway is not eLink TCP for CICS, the transaction will not process correctly.

CICSHAND=*<name>*

specifies the name of the handler transaction to be passed to the IBM TCP/IP listener for use with eLink TCP for CICS. The default is BEAH.

RMTACCT=*"userid"*

specifies the userid for gateway-level security on the foreign system. The default is " ".

PASSWORD=*"password"*

specifies the password associated with the userid for gateway-level security on the foreign system. The default is " ".

MAXCONNECT=*n*

specifies the maximum number of connections to the specified host. The default is no limit.

CONNSYNC=Y|N

specifies whether to force the gateway to establish connections to the specified host in a synchronous manner. The default is N.

CONNECT_TIME=*n*

specifies the number of seconds the gateway will wait to establish a connection. If you do not specify this parameter, the value of CONNECT_TIME in the *GLOBAL section is used.

CICSDATA=*"string"*

specifies a string to be passed to the IBM TCP/IP listener for use with the eLink TCP for CICS gateway. The default is " ".

Defining the *LOCAL_SERVICES Section of the GWICONFIG File

The *LOCAL_SERVICES section of the GWICONFIG file contains parameters for each local service specified in the DMCONFIG file. Each service entry name matches the remote name of the service in the *DM_LOCAL_SERVICES section.

Syntax

The format of the *LOCAL_SERVICES section of the GWICONFIG file follows:.

Listing 4-9 Syntax for *LOCAL_SERVICES Section of GWICONFIG File

```
[ # A comment (from "#" to end of line)]
*LOCAL_SERVICES
  <SERVICE_NAME>
    NATIVE=native system
```

```
[ INRECTYPE="foreign_incoming_buffer_type" ]  
[ OUTRECTYPE="foreign_outgoing_buffer_type" ]  
[ SECURE=Y | N ]  
[ CONV=Y | N ]
```

Required Parameters

The required parameters described in this section must be set for each service you specify in the *LOCAL_SERVICES section.

<SERVICE_NAME>

a 1-78 alphanumeric character string that represents the local service name that matches the service name value in the *DM_LOCAL_SERVICES section of the DMCONFIG file.

NATIVE=*"native_system"*

specifies the corresponding native system or gateway.

Optional Parameters

The optional parameters described in this section may be set for each service you specify in the *LOCAL_SERVICES section.

INRECTYPE=*"foreign_incoming_buffer_type"*

specifies the foreign buffer type for messages to remote clients. If you do not specify INRECTYPE, the default is no type. In this case, the type of the buffer will be unchanged.

OUTRECTYPE=*"foreign_outgoing_buffer_type"*

specifies the foreign buffer type for messages from remote clients. If you do not specify OUTRECTYPE, the default is to match the INRECTYPE value.

SECURE=Y | N

specifies whether the eLink TCP for TUXEDO gateway applies remote user information to local services.

If SECURE=Y, then the eLink TCP for TUXEDO gateway applies remote user information to local services using the appkey.

If SECURE=N, then the eLink TCP for TUXEDO gateway does not apply remote user information to local services using the appkey. The default is N.

Note: If SECURE=N and the TUXEDO domain is set with SECURITY=ACL in the UBBCONFIG file, a request to a local TUXEDO service can fail even if ACLs are in place.

CONV=Y | N

specifies whether service is conversational. Conversational mode is not currently supported, so Y will return an error message and the gateway will not start. The default is N.

Defining the *REMOTE_SERVICES Section of the GWICONFIG File

The *REMOTE_SERVICES section of the GWICONFIG file contains parameters for each remote service specified in the DMCONFIG file. Each service entry name matches the remote name of the service in the *DM_REMOTE_SERVICES section.

Syntax

The format of the *REMOTE_SERVICES section of the GWICONFIG file is as follows:

Listing 4-10 Syntax for *REMOTE_SERVICES Section of GWICONFIG File

```
[ # A comment (from "#" to end of line)]
*REMOTE_SERVICES
  <SERVICE_NAME>
  FOREIGN=system_name
  [ INRECTYPE="foreign_outgoing_buffer_type"]
  [ OUTRECTYPE="foreign_incoming_buffer_type"]
  [ OUTREQ_TIME=n]
  [ SECURE=Y | N]
  [ CONV=Y | N]
```

Required Parameters

The required parameters described in this section must be set for each service you specify in the *REMOTE_SERVICES section.

<SERVICE_NAME>

a 1-78 alphanumeric character string that represents the local service name that matches the RNAME value in the *DM_REMOTE_SERVICES section of the DMCONFIG file. If RNAME is not specified, this name matches the TUXEDO service name.

FOREIGN=*"foreign_system"*

specifies the corresponding foreign system or gateway.

Optional Parameters

The optional parameters described in this section may be set for each service you specify in the *REMOTE_SERVICES section.

INRECTYPE=*"foreign_outgoing_buffer_type"*

specifies the foreign buffer type for messages to remote servers. If you do not specify INRECTYPE, the default is no type. In this case, the type of the buffer will be unchanged.

OUTRECTYPE=*"foreign_incoming_buffer_type"*

specifies the foreign buffer type for messages from remote servers. If you do not specify OUTRECTYPE, the default is to match the INRECTYPE value.

OUTREQ_TIME=*n*

specifies the timeout value, in seconds, for requests sent to this service. If you do not specify this parameter, the value for OUTREQ_TIME in the *GLOBAL section is used. If OUTREQ_TIME is not specified in this section or the *GLOBAL section, an error message occurs.

SECURE=Y | N

specifies whether the eLink TCP for TUXEDO gateway supplies local user information to remote services.

If SECURE=Y, then the eLink TCP for TUXEDO gateway supplies user information to remote services.

If SECURE=N, then the eLink TCP for TUXEDO gateway does not supply user information to remote services.

CONV=Y | N

specifies whether the service is conversational. Conversational mode is not currently supported, so Y will return an error message and the gateway will not start. The default is N.

Defining Gateway Configurations in the DMCONFIG File

The following paragraphs describe the significant parameters within specific sections of the DMCONFIG file that define new gateway configurations.

***DM_LOCAL_DOMAINS Section**

This section identifies local domains and their associated gateway groups. The section must have an entry for each gateway group (local domain). Each entry specifies the parameters required for the domain gateway processes running in that group.

Syntax

The format of the *DM_LOCAL_DOMAINS section of the DMCONFIG file follows.

Listing 4-11 Syntax for *DM_LOCAL_DOMAINS Section of DMCONFIG File

```
LDOM    required parameters [optional parameters]
```

LDOM is an *identifier* value used to name each local domain and must be unique within a particular configuration. In the description of the *DM_LOCAL_SERVICES section, *LDOM* is the identifier that connects local services with a particular gateway group.

Required Parameters

The following parameters are required.

GWGRP = *identifier*

specifies the name of the gateway server group (the name provided in the TUXCONFIG file) representing this local domain. There is a one-to-one relationship between a DOMAINID and the name of the gateway server group, that is, each GWGRP must have its own, unique DOMAINID.

TYPE = *identifier*

is used for grouping local domain into classes. TYPE can be set to TDOMAIN or any other domain gateway type. The TDOMAIN value indicates that this local domain can only communicate with another TUXEDO System/Domain. For use with eLink TCP for TUXEDO, specify TYPE=IDOMAIN. Domain types must be defined in the \$TUXDIR/udataobj/DMTYPE file.

DOMAINID = *string*

is used to identify the local domain. DOMAINID must be unique across both local and remote domains. The value of *string* can be a sequence of characters (for example, "BA.CENTRAL01"), or a sequence of hexadecimal digits preceded by "0x" (for example, "0x0002FF98C0000B9D6"). DOMAINID must be 32 octets or fewer in length. If the value is a string, it must be 31 characters or fewer.

Optional Parameters

The following optional parameters describe resources and limits used in the operation of domain gateways:

AUDITLOG = *string*

specifies the name of the audit log file for this local domain. The audit log feature is activated from the `dmadmin` command and records all the operations within this local domain. If the audit log feature is active and this parameter is not specified, the file `DMmmddyy.LOG` (where *mm*=month, *dd*=day, and *yy*=year) is created in the directory specified by the \$APPDIR environment variable or the APPDIR keyword of the MACHINES section of the TUXCONFIG file.

BLOCKTIME = *numeric*

specifies the maximum wait time allowed for a blocking call. The value sets a multiplier of the SCANUNIT parameters specified in the TUXCONFIG file. The value SCANUNIT * BLOCKTIME must be greater than or equal to SCANUNIT and less than 32,768 seconds. If this parameter is not specified, the default value is set to the value of the BLOCKTIME parameter specified in the TUXCONFIG file. A timeout always implies a failure of the affected request. Notice that the timeout specified for transactions in the TUXCONFIG is always used when the request is issued within a transaction.

DMTLOGDEV = *string*

specifies the TUXEDO file system that contains the Domain transaction log (DMTLOG) for this machine. The DMTLOG is stored as a TUXEDO System VTOC table on the device. If this parameter is not specified, the domain gateway group is not allowed to process requests in transaction mode. Local domains running on the same machine can share the same DMTLOGDEV file system, but each local domain must have its own log (a table in the DMTLOGDEV) named as specified by the DMTLOGNAME keyword.

DMTLOGNAME = *identifier*

specifies the name of the domain transaction log for this domain. This name must be unique when the same DMTLOGDEV is used for several local domains. If not specified, the default is the string “DMTLOG”. The name must be 30 characters or less.

DMTLOGSIZE = *numeric*

specifies the numeric size, in pages, of the Domain transaction log for this machine. It must be greater than 0 and less than the amount of available space on the TUXEDO file system. If not specified, the default is 100 pages.

MAXDATALEN = *numeric*

specifies a maximum amount of data (in bytes) that can be sent to or from any services advertised by this local domain. There is no limit if this parameter is not specified.

MAXRDOM = *numeric*

specifies the maximum number of connections (or dialogs if the domain is of type *OSITP*) allowed per gateway. There is no limit if this parameter is not specified.

MAXRDTRAN = *numeric*

specifies the maximum number of domains that can be involved in a transaction. It must be greater than 0 and less than 32,768. If not specified, the default is 16.

MAXTRAN = *numeric*

specifies the maximum number of simultaneous global transactions allowed on this local domain. It must be greater than or equal to 0 and less than or equal to the MAXGTT parameter specified in the TUXCONFIG file. If not specified, the default is the value of MAXGTT.

MAXSENDLEN = *numeric*

specifies the maximum length (in bytes) of messages sent or received by this local domain. If this parameter is set, all messages sent or received are broken up into packets of no more than MAXSENDLEN bytes. There is no limit if this parameter is not specified.

*DM_REMOTE_DOMAINS Section

This section identifies the known set of remote domains and their characteristics.

Syntax

The format of the *DM_REMOTE_DOMAINS section of the DMCONFIG file is as follows.

Listing 4-12 Syntax for *DM_REMOTE_DOMAINS Section of DMCONFIG

RDOM required parameters

RDOM is an *identifier* value used to identify each remote domain known to this configuration and must be unique within the configuration.

Required Parameters

The following parameters are required.

TYPE = *identifier*

is used for grouping remote domain into classes. TYPE can be set to TDOMAIN or any other domain gateway type. The TDOMAIN value indicates that this remote domain can only communicate with another TUXEDO System/Domain. The OSITP value indicates that this remote domain communicates with another TP domain via the OSI-TP protocol. For use with eLink TCP for TUXEDO, specify TYPE=IDOMAIN.

DOMAINID = *string*

is used to identify a remote domain. DOMAINID must be 32 octets or fewer in length. If the value is a string, it must be 31 characters or fewer. DOMAINID must be unique across remote domains. The value of *string* can be a sequence of characters or a sequence of hexadecimal digits preceded by "0x".

Optional Parameters

There are no optional parameters for this section.

*DM_ACCESS_CONTROL Section

This section is optional in the DMCONFIG file and specifies the access control lists used by local domain.

Syntax

The format of the *DM_ACCESS_CONTROL section of the DMCONFIG file follows.

Listing 4-13 Syntax for *DM_ACCESS_CONTROL Section of DMCONFIG

```
ACL_NAME    required parameters
```

ACL_NAME is a (*identifier*) name used to identify a particular access control list; it must be 15 characters or less in length.

Required Parameters

The following parameter is required.

`ACLIST = identifier [,identifier]`

where an ACLIST is composed of one or more remote domain names (RDOM) separated by commas. The wildcard character (*) can be used to specify that all the remote domains defined in the *DM_REMOTE_DOMAINS section can access a local domain.

Optional Parameters

There are no optional parameters for this section.

*DM_LOCAL_SERVICES Section

This section provides information on the services exported by each local domain. This section is optional and if it is not specified then all local domains defined in the *DM_LOCAL_DOMAINS section accept requests to all of the services advertised by the TUXEDO System/Domain application. If this section is defined then it should be used to restrict the set of local services that can be requested from a remote domain.

Syntax

The format of the *DM_LOCAL_SERVICES section of the DMCONFIG file follows.

Listing 4-14 Syntax for *DM_LOCAL_SERVICES Section of DMCONFIG File

```
service      [optional parameters]
```

service is the (*identifier*) local name of the exported service, and it must be 15 characters or fewer in length.

This name corresponds to a name advertised by one or more servers running with the local TUXEDO System/Domain application. Notice that exported services inherit the default or special properties specified for the service in an entry in the SERVICES section of the TUXCONFIG file. Some of these parameters are: LOAD, PRIO, AUTOTRAN, ROUTING, BUFTYPE, and TRANTIME.

Required Parameters

There are no required parameters for *DM_LOCAL_SERVICES.

Optional Parameters

The following parameters are optional.

ACL = *identifier*

specifies the name of the access control list (ACL) to be used by the local domain to restrict requests made to this service by remote domains. The name of the ACL is defined in the *DM_ACCESS_CONTROL section. If this parameter is not specified, then access control is not performed for requests to this service.

LDOM = *identifier*

specifies the name identifying the local domain exporting this service. If this keyword is not specified, then all the local domains defined in the *DM_LOCAL_DOMAINS section accept requests to this local service.

RNAME = *string*

specifies the name exported to remote domains. The remote domains use this name for request to this service. If this parameter is not specified, the local service name is supposed to be the name used by any remote domain. For eLink TCP, this should match the service name in the GWICONFIG.

INBUFTYPE= "*local_incoming_buffer_type*"

specifies the TUXEDO buffer type for incoming messages. For conversational services, the same buffer type must be used for all incoming messages.

OUTBUFTYPE= "*local_outgoing_buffer_type*"

specifies the TUXEDO buffer type for outgoing messages.

*DM_REMOTE_SERVICES Section

This section provides information on services “imported” and available on remote domains.

Syntax

The format of the *DM_REMOTE_SERVICES section of the DMCONFIG file follows.

Listing 4-15 Syntax for *DM_REMOTE_SERVICES Section of DMCONFIG

```
service      [optional parameters]
```

service is the (*identifier*) name used by the local TUXEDO System/Domain application for a particular remote service. Remote services are associated with a particular remote domain.

Required Parameters

There are no required parameters for the *DM_REMOTE_SERVICES section.

Optional Parameters

The following parameters are optional.

CONV = { Y | N }

specifies whether the remote service is a conversational service. Use Y to specify the remote service is a conversational service. Use N to specify the remote service is not a conversational service. The default value is N.

LDOM = *identifier*

specifies the name of a local domain in charge of routing requests to this remote service. The gateway group associated with the local domain advertises *service* in the TUXEDO System/Domain Bulletin Board. If this parameter is not specified, then all the local domains are able to accept

requests to this remote service. The service request is redirected to a remote domain of the same type (see the following definition for RDOM keyword).

RDOM = *identifier*

specifies the name of the remote domain responsible for the actual execution of this service. If this parameter is not specified and a routing criteria (see the following definition for ROUTING keyword) is not specified, then the local domain assumes that any remote domain of the same type accepts this service and it selects a known domain (a domain to which a connection already exists) or remote domain from the *DM_REMOTE_DOMAINS section.

RNAME = *string*

specifies the actual service name expected by the remote domain. If this parameter is not specified, the remote service name is the same as the name specified in *service*. For eLink TCP, this should match the service name in the GWICONFIG file.

ROUTING = *identifier*

when more than one remote domain offers the same service, a local domain can perform data-dependent routing if this optional parameter is specified. The *identifier* specifies the name of the routing criteria used for this data-dependent routing. If not specified, data-dependent routing is not done for this service. *identifier* must be 15 characters or less in length. If multiple entries exist for the same service name but with different RDOM parameters, the ROUTING parameter should be the same for all of these entries.

TRANTIME = *integer*

specifies the default time-out value in seconds for a transaction automatically started for the associated service. The value must be greater than or equal to 0 and less than 2147483648. The default is 30 seconds. A value of 0 implies the maximum time-out value for the machine.

INBUFTYPE = *"local_outgoing_buffer_type"*

specifies the TUXEDO buffer type for outgoing messages or buffers passed as input to the remote service.

OUTBUFTYPE = *"local_incoming_buffer_type"*

specifies the TUXEDO buffer type for incoming messages or buffers returned as output from the remote service.

*DM_ROUTING Section

This section is optional in the DMCONFIG file and provides information for data-dependent routing of service requests using FML, VIEW, X_C_TYPE, and X_COMMON typed buffers.

Syntax

The format of the *DM_ROUTING section of the DMCONFIG file follows.

Listing 4-16 Syntax for *DM_ROUTING Section of DMCONFIG File

```
CRITERION_NAME      required parameters
```

CRITERION_NAME is the (*identifier*) name of the routing entry that was specified on the services entry. *CRITERION_NAME* must be 15 characters or less in length.

Required Parameters

The following parameters are required.

FIELD = *identifier*

specifies the name of the routing field. It must be 30 characters or less. This field is assumed to be a field name that is identified in an FML field table (for FML buffers) or an FML view table (for VIEW, X_C_TYPE, or X_COMMON buffers). The FLDTBLDIR and FIELDTBLS environment variables are used to locate FML field tables, and the VIEWDIR and VIEWFILES environment variables are used to locate FML view tables.

RANGES = *string*

specifies the ranges and associated remote domain names (RDOM) for the routing field. *string* must be enclosed in double quotes. The format of *string* is a comma-separated ordered list of range/RDOM pairs.

A range is either a single value (signed numeric value or character string in single quotes), or a range of the form “lower - upper” (where lower and upper are both signed numeric values or character strings in single quotes).

Note that “lower” must be less than or equal to “upper.” To embed a single quote in a character string value (as in O’Brien, for example), it must be preceded by two backslashes (‘O\\’Brien’).

The value MIN can be used to indicate the minimum value for the data type of the associated FIELD; for strings and arrays, it is the null string; for character fields, it is 0; for numeric values, it is the minimum numeric value that can be stored in the field.

The value MAX can be used to indicate the maximum value for the data type of the associated FIELD; for strings and arrays, it is effectively an unlimited string of octal-255 characters; for a character field, it is a single octal-255 character; for numeric values, it is the maximum numeric value that can be stored in the field.

Thus, “MIN - -5” is all numbers less than or equal to -5 and “6 - MAX” is all numbers greater than or equal to 6. The meta-character “*” (wildcard) in the position of a range indicates any values not covered by the other ranges previously seen in the entry; only one wildcard range is allowed per entry and it should be last (ranges following it are ignored).

The routing field can be of any data type supported in FML. A numeric routing field must have numeric range values and a string routing field must have string range values.

String range values for string, array, and character field types must be placed inside a pair of single quotes and cannot be preceded by a sign. Short and long integer values are a string of digits, optionally preceded by a plus or minus sign. Floating point numbers are of the form accepted by the C compiler or `atof()`: an optional sign, then a string of digits optionally containing a decimal point, then an optional e or E followed by an optional sign or space, followed by an integer.

When a field value matches a range, the associated RDOM value specifies the remote domain to which the request should be routed. A RDOM value of “*” indicates that the request can go to any remote domain known by the gateway group.

Within a range/RDOM pair, the range is separated from the RDOM by a “:”.

`BUFTYPE = ~type1[:subtype1[, subtype2 . . .]];type2[:subtype3[, . . .]]` . . . ~

is a list of types and subtypes of data buffers for which this routing entry is valid. The types are restricted to be either FML, VIEW, X_C_TYPE, or

X_COMMON. No subtype can be specified for type FML and subtypes are required for the other types (“*” is not allowed). Duplicate type/subtype pairs cannot be specified for the same routing criterion name; more than one routing entry can have the same criterion name as long as the type/subtype pairs are unique. This parameter is required. If multiple buffer types are specified for a single routing entry, the data types of the routing field for each buffer type must be the same.

If the field value is not set (for FML buffers), or does not match any specific range and a wildcard range has not been specified, an error is returned to the application process that requested the execution of the remote service.

Optional Parameters

There are no optional parameters for the *DM_ROUTING section.

Sample DMCONFIG File

The following sample is a DMCONFIG file and must be set up prior to running the eLink TCP for TUXEDO product.

Listing 4-17 Sample DMCONFIG File

```
#
# Copyright (c) 1996 BEA Systems, Inc
# All Rights Reserved
#
# THIS IS UNPUBLISHED PROPRIETARY SOURCE CODE OF
# BEA Systems, Inc.
# The copyright notice above does not evidence any
# actual or intended publication of such source code.
#
#
# *DM_RESOURCES
#
#
# *DM_LOCAL_DOMAINS
LOCAL          GWGRP=GROUP
                TYPE=IDOMAIN
                DOMAINID="LOCAL"
```

```
#
*DM_REMOTE_DOMAINS
REMOTE          TYPE=IDOMAIN
                DOMAINID="REMOTE"

#
*DM_LOCAL_SERVICES
ECHOX           RNAME="ZECHOSV4"
TOLOWER         RNAME="TUXTOLOWER"
                INBUFTYPE=string
                OUTBUFTYPE=string

#
*DM_REMOTE_SERVICES
TOUPPER         RDOM=REMOTE
                LDOM=LOCAL
                RNAME="TUXTOUPPER"

ECHOX           RDOM=REMOTE
                LDOM=LOCAL
                RNAME="BEASVR07"
                INBUFTYPE="VIEW:myview"
                OUTBUFTYPE="FML"

NORMAL          RDOM=REMOTE
                LDOM=LOCAL
                RNAME="TST1V"
```

5 Starting BEA eLink TCP for TUXEDO

The following topics provide information about starting the BEA eLink for Mainframe TCP for TUXEDO (hereafter referenced as eLink TCP for TUXEDO) product:

- ◆ Setting Environment Variables
- ◆ Invoking eLink TCP for TUXEDO
- ◆ Administering the Gateways

Setting Environment Variables

Before you attempt to use eLink TCP for TUXEDO, you must set the TUXDIR and PATH environment variables as the following example illustrates.

Listing 5-1 Setting ROOTDIR and PATH Environment Variables

```
TUXDIR=/usr/tuxedo; export TUXDIR
PATH=$PATH:$TUXDIR/bin; export PATH
GWICONFIG=$APPDIR/gwiconfig;export GWICONFIG
BDMCONFIG=$APPDIR/bdmconfig;export BDMCONFIG
TUXCONFIG=$APPDIR/tuxconfig;export TUXCONFIG
```

You should set TUXDIR to the actual path where your BEA TUXEDO and eLink TCP for TUXEDO software is installed. Set APPDIR to the application directory, similar to the UBBCONFIG file. You may also need to set the LANG environment variable if you have generated custom mapping tables or message catalogs. Some platforms may require that LANG always is set. Consult your operating system documentation for the appropriate LANG value.

Invoking eLink TCP for TUXEDO

Perform a `tmloadcf` and `dmloadcf` to load the UBBCONFIG and the DMCONFIG files. Invoke eLink TCP for TUXEDO using the `tmboot` command to boot the BEA TUXEDO system. Be sure to configure eLink TCP for TUXEDO prior to using the `tmboot` command. For details, see “Getting Ready to Configure BEA eLink TCP for TUXEDO,” and “Configuring BEA eLink TCP for TUXEDO.”

Listing 5-2 Invoking eLink TCP for TUXEDO Using `tmboot`

```
tmboot -y
```

Administering the Gateways

BEA TUXEDO has a set of `tmadmin` and `dmadmin` commands for the administration of the eLink TCP for TUXEDO gateways. For detailed information about these commands, refer to the *BEA TUXEDO Administrator's Guide* and the *BEA TUXEDO Domain User Guide*.

A Operational Considerations

Operational considerations are permanent limitations on using a feature. The following operational considerations apply to BEA eLink for Mainframe TCP for TUXEDO (hereafter referenced as eLink TCP for TUXEDO):

Note: In the following discussion, a local application program is one that resides within the immediate BEA TUXEDO administrative region. A remote application program is one that resides outside the immediate BEA TUXEDO administrative region.

- ◆ BEA eLink TCP for TUXEDO does not support conversational communication.
- ◆ BEA eLink TCP for TUXEDO supports only nontransactional communication.
- ◆ Local client and server programs which use the `tpsprio()` function set the priority at which service requests are dequeued by eLink TCP for TUXEDO. This does not affect any prioritization on the remote system.
- ◆ Local client programs cannot use the `tpbroadcast()` function to send unsolicited messages to remote client programs (and the reverse).
- ◆ Local services cannot use the `tpbroadcast()` or `tpnotify()` functions to send messages to remote client programs (and the reverse).
- ◆ When local client and server programs use the `tpgprio()` function to determine the priority of a remote service, the priority of a local eLink TCP for TUXEDO requester is returned.
- ◆ For background information about these operational considerations, see “Understanding How BEA eLink TCP for TUXEDO Works.”

B Data Area Security

BEA eLink for Mainframe TCP for TUXEDO (hereafter referenced as eLink TCP for TUXEDO) uses data area security which is a specialized security protocol for the following cases:

- ◆ User information is propagated across multiple TUXEDO domain boundaries
- ◆ A remote or local service requires a user's LTERM information

In these cases, a client's USERID, group name, and LTERM can be specified in the data area of a request. For TUXEDO clients, user information specified in the data area is verified by the remote gateway in the usual manner. For remote clients, remote user information is placed in the data area fields by the local gateway to be used by TUXEDO services. In this case, the remote client does not have to populate these fields, but must allocate space for them in the data area.

Enabling Data Area Security

Complete the following tasks to enable data area security.

1. Add fields to the user's data area on the local and remote hosts. These fields are passed to and from the mainframe host. For the field formats, refer to Listing B-1.
2. Set WRAP=TPSD in the *FOREIGN section corresponding to the remote host in the GWICONFIG file. For syntax and parameter definitions for the *FOREIGN section of the GWICONFIG file, refer to "Defining the *FOREIGN Section of the GWICONFIG File."
3. Populate the data area with the user information before sending a request to a remote service.

4. The remote user's information is populated into the data area when a request is received for a local service.

Note: If using a VIEW data format, allocate the extra fields before the application data as defined in Listing B-1. If using the STRING data format, allocate 24 additional bytes at the beginning of the string to be used for the security fields.

Format

The user data area fields in C use the following format.

Listing B-1 Syntax for C User Data Area Fields

```
struct da_security {  
    char uname[8]; /*user name*/  
    char group[8]; /*user group*/  
    char lterm[8]; /*terminal id*/  
    /*user data is appended here*/  
}
```

C Error and Informational Messages

This document contains a description of error and informational messages that can be encountered while using BEA eLink for Mainframe TCP for TUXEDO (hereafter referenced as eLink TCP for TUXEDO).

1000	ERROR: Memory allocation error
	Description: An attempt to allocate memory failed.
	Action: Check available memory.
	See Also: None.
1001	ERROR: Cannot load shared module <modname>: <error text>
	Description: An attempt to load the listed shared library failed. The reason is noted in the error text.
	Action: Verify that the shared library exists.
	See Also: None.
1002	ERROR: Cannot get symbol '<symbol>' in module '<modname>'
	Description: The listed symbol was not found in the listed shared module.
	Action: Contact your TUXEDO System Technical Support.

	See Also:	None.
1003	ERROR: No encoding type defined for <gwname>. No encoding done	
	Description:	The gateway configuration file does not specify a TYPE parameter for the listed FOREIGN gateway or a DFLTTYPE parameter in the GLOBAL section.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1004	ERROR: Encoding type '<type>:<subtype>' for type '<etype>' failed.	
	Description:	Allocating a typed buffer for encoding output failed
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1005	ERROR: Decode failed due to allocation error	
	Description:	The output data buffer was not created correctly with tmalloc().
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1006	ERROR: No encoding type defined for <gwname>. No decoding done	
	Description:	The Domain Gateway library software does not have an encoding type defined in shared memory for the listed gateway.
	Action:	Verify the TYPE parameter for the gateway in the gateway configuration file.
	See Also:	None.
1007	ERROR: Decoding type <type:subtype> for type <rtype> failed	
	Description:	The Domain Gateway library software detected an error while getting decoding information for the listed type, subtype, and remote type.

	Action:	Previous logged error messages should provide the exact reasons for the failure. Contact your TUXEDO System Technical Support.
	See Also:	None.
1008	ERROR: GWICONFIG env var not found!	
	Description:	Retrieving the environment variable GWICONFIG failed.
	Action:	Set and export the GWICONFIG environment variable.
	See Also:	None.
1009	ERROR: <filename> config file not found!	
	Description:	The named configuration file cannot be found.
	Action:	Verify that the file which the GWICONFIG environment variable points to exists.
	See Also:	None.
1010	ERROR: Parse of <filename> failed, exiting	
	Description:	An attempt to parse the gateway configuration file failed.
	Action:	Verify the format of the gateway configuration file.
	See Also:	None.
1011	ERROR: <errcount> errors found while parsing <filename>, exiting.	
	Description:	Errors occurred while parsing the gateway configuration file.
	Action:	Verify the format of the gateway configuration file
	See Also:	None.
1012	INFO: Dumping configuration to stdout	
	Description:	The current configuration is dumped to the standard output
	Action:	No action required.
	See Also:	None.

1013	ERROR: Sending error to <gwname>: <error text>
Description:	The error condition described is being sent to the listed gateway.
Action:	Contact your TUXEDO System Technical Support.
See Also:	None.
1014	ERROR: Invalid action detected <action>
Description:	The Domain Gateway library software detected an invalid action while trying to encode a buffer.
Action:	This is an internal error with no associated user action. If the error persists, contact your TUXEDO System Technical Support with the exact error message.
See Also:	None.
1015	ERROR: Data encoding failed
Description:	The Domain Gateway library software detected an error encoding data.
Action:	Previous logged error messages should provide the exact reasons for the failure. Contact your TUXEDO System Technical Support.
See Also:	None.
1016	ERROR: Invalid wrap switch index
Description:	The wrap switch index in shared memory is not valid.
Action:	Contact your TUXEDO System Technical Support.
See Also:	None.
1017	ERROR: Buffer wrapping failed
Description:	The buffer wrapping function returned an error.
Action:	Contact your TUXEDO System Technical Support.
See Also:	None.

1018	ERROR: Bad wrapping protocol
	Description: The wrapping protocol in shared memory is not GWI_PROTOCOL.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1019	ERROR: Cannot unwrap buffer
	Description: The buffer unwrapping function returned an error.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1020	ERROR: Unable to obtain local service (<service>) information from shared memory
	Description: Information about the local service could not be found in shared memory.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1021	ERROR: Request for unknown service '<service>' from <gwname>
	Description: Information about the listed service could not be found in shared memory.
	Action: Verify the DMCONFIG file and insure dmloadcf has been run on it.
	See Also: None.
1022	ERROR: Reply received for invalid session
	Description: The Domain Gateway received a reply for an invalid session.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.

1023	ERROR: Reply received for inactive session
Description:	The Domain Gateway received a reply for an inactive session.
Action:	Contact your TUXEDO System Technical Support.
See Also:	None.
1024	ERROR: Unable to obtain remote service (<service>) information from shared memory
Description:	Information about the remote service could not be found in shared memory.
Action:	Verify the DMCONFIG file and insure dmloadcf has been run on it.
See Also:	None.
1025	ERROR: Data decoding failed
Description:	The Domain Gateway library software detected an error while decoding a received message buffer.
Action:	Previous logged error messages should provide the exact reasons for the failure. Contact your TUXEDO System Technical Support.
See Also:	None.
1026	INFO: <field name> : <value>
Description:	The field name and its value in a fielded buffer are logged.
Action:	No action required.
See Also:	None.
1027	INFO: Type <type> : <value>
Description:	The field type and its value in a fielded buffer are logged.
Action:	No action required.
See Also:	None.

1028	ERROR: Cannot open <view>:<name>
	Description: Could not find the listed view file.
	Action: Verify the VIEWDIR and VIEWFILES environment variables.
	See Also: None.
1029	ERROR: FML incompatible field in <view>:<name>
	Description: The listed view contains the packed decimal data type. Views containing this type cannot be used for FML-to-view or view-to-view conversions.
	Action: Remove packed decimal fields from the input FML or view, or send the same view structure to the local gateway that will be sent to the remote gateway.
	See Also: None.
1030	ERROR: talloc for <buffer> failed
	Description: Allocation of an TUXEDO typed buffer failed.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1031	ERROR: Fvstof failed for <view>:<name>
	Description: The conversion of a C structure to a fielded buffer failed.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1032	INFO: <product>, licensed to <licensee>
	Description: Informational message describing license information
	Action: None required
	See Also: None.
1033	ERROR: talloc for <view>:<name> failed
	Description: Allocating a typed buffer for the listed view failed.

	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1034	ERROR: Connect errno <errcode> (<error text>)	
	Description:	The Domain Gateway library software detected an error while trying to open a connection on a socket. The reason is noted in the error text.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1035	ERROR: Unable to open device "<dev>" network error code <errcode>	
	Description:	Cannot open the given network device.
	Action:	Verify the NWDEVICE in the gateway configuration file. Contact your TUXEDO System Technical Support.
	See Also:	None.
1036	ERROR: Unable to open connection endpoint; network error code <errcode>	
	Description:	Cannot open a network connection.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1037	INFO: Unable to set signals on file descriptor	
	Description:	The Domain Gateway library software detected an error while trying to turn on signaling for a file descriptor.
	Action:	No action required. Incoming data will be detected by poll rather than asynchronously.
	See Also:	None.
1038	ERROR: Unable to connect to <host>; network error code <errcode>	
	Description:	The Domain Gateway library software detected an error while trying to open a connection on a socket.

	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1039	ERROR: Connection failed to <gwname>	
	Description:	The Domain Gateway library software detected an error while trying to connect to the listed gateway.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1040	ERROR: Connect reply failed; network error code <errcode>	
	Description:	The Domain Gateway library software detected an error while trying to open a connection on a socket.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1041	ERROR: Connection shutdown failed	
	Description:	The Domain Gateway library software detected an error while trying to shutdown a connection on a socket.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1042	ERROR: Orderly release confirmation failed	
	Description:	The Domain Gateway library software detected an error while trying to shutdown a connection on a socket.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1043	ERROR: Invalid sub-state GWI_ACT_<substate> in gwi_nwclose()	
	Description:	This message should never occur. The program code or defines are wrong if it does.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.

1044	WARN: Receive error: t_errno <errcode>
Description:	The Domain Gateway library software detected an error while trying to receive a message from the network.
Action:	This is an internal error with no associated user action. If the error persists, contact your TUXEDO System Technical Support with the exact error message.
See Also:	None.
1045	WARN: Receive error(TLOOK) <errcode>
Description:	The Domain Gateway library software detected an error while trying to receive a message from the network.
Action:	This is an internal error with no associated user action. If the error persists, contact your TUXEDO System Technical Support with the exact error message.
See Also:	None.
1046	WARN: Error on connection <conn> to <gwname>: <error text> (<errcode>)
Description:	The Domain Gateway library software detected an error while trying to receive a message from the network.
Action:	This is an internal error with no associated user action. If the error persists, contact your TUXEDO System Technical Support with the exact error message.
See Also:	None.
1047	WARN: Connection <conn> closed by <gwname>
Description:	While trying to receive a message from the network, the Domain Gateway library software detected that the listed connected had been closed by the listed gateway.
Action:	This is an internal error with no associated user action. If the error persists, contact your TUXEDO System Technical Support with the exact error message.
See Also:	None.

1048	ERROR: Unable to send service request
Description:	The Domain Gateway library software detected an error while trying to send a message to the network.
Action:	This is an internal error with no associated user action. If the error persists, contact your TUXEDO System Technical Support with the exact error message.
See Also:	None.
1049	ERROR: Unable to send service request "<service>"
Description:	The Domain Gateway library software detected an error while trying to send a message to the network.
Action:	This is an internal error with no associated user action. If the error persists, contact your TUXEDO System Technical Support with the exact error message.
See Also:	None.
1050	ERROR: Unable to get remote service information from shared memory
Description:	Information about the remote service could not be found in shared memory.
Action:	Verify the DMCONFIG file and insure dmloadcf has been run on it.
See Also:	None.
1051	ERROR: No free fd structures available
Description:	After using all allocated slots in the fd table, reallocation of more space for additional fd slots failed.
Action:	Check available memory.
See Also:	None.
1052	ERROR: Can't get remote service name for <service>
Description:	The remote service name for the listed service could not be found in shared memory.

	Action:	Verify the DMCONFIG file and insure dmloadcf has been run on it.
	See Also:	None.
1053	ERROR: No remote service entry for <service>	
	Description:	No remote service entry for the listed service was found in shared memory.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1054	ERROR: Cannot suspend action	
	Description:	The Domain Gateway library software was unable to suspend the action while trying to set up a network connection.
	Action:	This is an internal error with no associated user action. If the error persists, contact your TUXEDO System Technical Support with the exact error message.
	See Also:	None.
1055	WARN: CICS init string failed for <gwname>.	
	Description:	The Domain Gateway library software detected an error while trying to send a CICS initialization string to the listed gateway.
	Action:	This is an internal error with no associated user action. If the error persists, contact your TUXEDO System Technical Support with the exact error message.
	See Also:	None.
1056	INFO: Connection <conn> initialized to <gwname>, ID:<data>	
	Description:	The Domain Gateway library software successfully initialized a connection to the gateway.
	Action:	No action required.
	See Also:	None.

1057	INFO: Multiplex count for connection <i>conn</i> is count
	Description: The multiplex count for the listed connection is given.
	Action: No action required.
	See Also: None.
1058	INFO: Listen port <i><port></i> (<i><gateway></i>) established
	Description: The listen port for the gateway has been established.
	Action: No action required.
	See Also: None.
1059	ERROR: Cannot change action into msg_failure
	Description: The action cannot be changed to msg_failure.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1060	ERROR: Cannot change action into msg_reply
	Description: The action cannot be changed to msg_failure.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1062	ERROR: Unexpected action state <i><state></i>
	Description: This message should never occur. The program code or defines are wrong if it does.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1063	WARN: Conversation not found - closing connection
	Description: The ID of the connection was not found in the shared memory conversation table.
	Action: No action required.

	See Also:	None.
1064	ERROR: No more actions	
	Description:	Trying to create an action to time out a conversation failed.
	Action:	No action required.
	See Also:	None.
1065	ERROR: Network initialization failure	
	Description:	(none)
	Action:	(none)
	See Also:	None.
1066	ERROR: Cannot allocate appropriate array for poll	
	Description:	The Domain Gateway library software was unable to allocate the pollfd array.
	Action:	This is an internal error with no associated user action. If the error persists, contact your TUXEDO System Technical Support with the exact error message.
	See Also:	None.
1067	ERROR: Run-time environment setting failure	
	Description:	The Domain Gateway library software was unable to initialize the run-time environment.
	Action:	Previous logged error messages should provide the exact reasons for the failure. Contact your TUXEDO System Technical Support.
	See Also:	None.
1068	ERROR: No free listen structures available	
	Description:	Obtaining a free listen structure failed.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.

1069	ERROR: Cannot listen for host <gwname>
	Description: The Domain Gateway library software was unable to resolve the IP address of the listed gateway.
	Action: Previous logged error messages should provide the exact reasons for the failure. Contact your TUXEDO System Technical Support.
	See Also: None.
1070	ERROR: Unable to establish listening endpoint; network error code <errcode>
	Description: The Domain Gateway library software was unable to open a socket connection.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1071	ERROR: Unable to establish listen port <port> (<gateway>)
	Description: The Domain Gateway library software was unable to establish a listen socket.
	Action: Verify that the configured listen port is not in use by another process. Contact your TUXEDO System Technical Support.
	See Also: None.
1073	ERROR: /DOMAIN gateway could not close a NW connection
	Description: The Domain Gateway library software was unable to close a socket connection.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1074	ERROR: Unlicensed TUXEDO System Binary
	Description: The files in your TUXEDO System installation do not contain the expected software license information for eLink TCP 3.0.

	Action:	Reinstall the eLink TCP 3.0 System software, using the license token and serial number supplied with the distribution media. Make sure not to terminate the installation program prematurely, because the license information is processed at the end.
	See Also:	None.
1075	ERROR: Cannot alloc conv switch	
	Description:	Creating a conversation switch in shared memory failed.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1076	ERROR: Cannot alloc wrap switch	
	Description:	Creating a wrap switch in shared memory failed.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1077	ERROR: Couldn't load configuration	
	Description:	The Domain Gateway library software was unable to load the configuration.
	Action:	Verify the format of the gateway configuration file.
	See Also:	None.
1078	ERROR: No gateway defined in config file.	
	Description:	The gateway configuration file does not contain a gateway definition.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1079	ERROR: Couldn't find <gwname> gateway (GWINAME) in NATIVE section	
	Description:	The listed gateway was not found in the NATIVE section of the gateway configuration file.

	Action:	Correct the gateway configuration file.
	See Also:	None.
1080	ERROR: Couldn't find env var GWINAME. Cannot guess which gateway.	
	Description:	The gateway configuration file contains more than one gateway entry and the environment variable GWINAME is not set.
	Action:	Set the GWINAME environment variable.
	See Also:	None.
1081	ERROR: Couldn't find host name. Cannot guess gateway address.	
	Description:	The Domain Gateway library software detected an error while trying to get the name of the current host.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1082	ERROR: Service <service> is SECURE, Tuxedo security disabled	
	Description:	The listed service requires security, but Tuxedo security is disabled.
	Action:	Either turn on Tuxedo security with the SECURITY resource, or remove the SECURE parameter from the gateway configuration file for this service.
	See Also:	None.
1084	ERROR: Cannot establish accept endpoint	
	Description:	The Domain Gateway library software was unable to open a socket connection.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.

1085	ERROR: Cannot accept new connection
Description:	The Domain Gateway library software was unable to accept a connection on a socket.
Action:	Contact your TUXEDO System Technical Support.
See Also:	None.
1086	ERROR: Cannot get free cnx structure for new connection
Description:	The Domain Gateway library software was unable to allocate shared memory for a new connection structure.
Action:	Contact your TUXEDO System Technical Support.
See Also:	None.
1087	ERROR: Can't get remote address
Description:	Could not convert the remote internet address to a binary address.
Action:	Previous logged error messages should provide the exact reasons for the failure. Contact your TUXEDO System Technical Support.
See Also:	None.
1088	INFO: Accepted new connection <conn> from <gwname>
Description:	A new connection from the remote gateway has been accepted.
Action:	No action required.
See Also:	None.
1089	ERROR: Poll returned error (num_cnx=<conn>)
Description:	The Domain Gateway library software detected an error while polling the connection for events.
Action:	Contact your TUXEDO System Technical Support.
See Also:	None.

1090	ERROR: No free sessions available
	Description: The Domain Gateway library software was unable to allocate shared memory for a new session structure.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1091	WARN: POLL returned event <i><event></i>, cnx_idx <i><index></i> fd <i><descriptor></i>
	Description: POLL returned an event other than POLLIN or POLLOUT.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1092	ERROR: Invalid ses_idx <i><index></i> for act <i><action></i>
	Description: The action structure references an invalid session index.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1093	ERROR: gw_nw_recv: Invalid fd <i><descriptor></i> for act <i><action></i>
	Description: The pollfds structure references an invalid file descriptor.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1094	WARN: Deleting late reply from local service <i><service-name></i>
	Description: A late reply has arrived from a TUXEDO service. Because the request has already timed out, the reply is discarded.
	Action: No action is required. If an excessive number of timeouts occur, adjustments may be needed in the configuration.
	See Also: None.
1095	WARN: Cleaning up connection <i><conn></i>
	Description: The file descriptor, action, and connection structures for the connection are being removed from shared memory.

	Action:	Previous logged error messages should provide more information. Contact your TUXEDO System Technical Support.
	See Also:	None.
1096	WARN: Network on receive from <gwname>, closing connection	
	Description:	The Domain Gateway library software detected an error while trying to receive a message from the gateway.
	Action:	Previous logged error messages should provide more information. Contact your TUXEDO System Technical Support.
	See Also:	None.
1097	ERROR: Error message from <gwname>:Tuxedo code <errcode> (<error text>): <message>	
	Description:	Received an error message from the remote gateway.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1098	WARN: Late response received for session <conn>/<session>, unlocking	
	Description:	Received a response on a file descriptor which had timed out.
	Action:	No action required.
	See Also:	None.
1099	INFO: Stale message found	
	Description:	Received a message on a session with no associated action.
	Action:	No action required.
	See Also:	None.
1100	ERROR: Data conversion failed for local request '<service>'	
	Description:	The Domain Gateway library software detected an error while trying to decode a message.

	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1101	ERROR: Unexpected diagnostic <diag> returned from server	
	Description:	An unexpected diagnostic flag was returned from the server.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1102	ERROR: Data conversion failed for local service '<service>'	
	Description:	The Domain Gateway library software detected an error while trying to encode a message.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1103	INFO: Connection <conn> initialized from <gwname>	
	Description:	The Domain Gateway library software received a connection request from the remote gateway.
	Action:	No action required
	See Also:	None.
1104	ERROR: Unable to identify local service, rname '<remoteservice>'	
	Description:	The service requested by the listed remote service was not found in shared memory.
	Action:	Verify the DMCONFIG file and insure dmloadcf has been run on it.
	See Also:	None.
1105	ERROR: Unable to map local service RNAME <remote service>	
	Description:	The service requested by the listed service name was not found in shared memory.
	Action:	Verify the DMCONFIG file and insure dmloadcf has been run on it.

	See Also:	None.
1106	ERROR: Unable to identify user <username>	
	Description:	The listed user is not authorized for the requested service.
	Action:	Verify the <code>tpusr</code> file in <code>APPDIR</code> .
	See Also:	None.
1107	ERROR: Protocol error : server cannot use GWI_OP_DISCON	
	Description:	A remote service is sending a disconnect.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1108	ERROR: Invalid opcode on receive	
	Description:	The Domain Gateway library software received an invalid opcode.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1110	INFO: Closing connection <conn>	
	Description:	The listed connection is being closed.
	Action:	No action required
	See Also:	None.
1111	ERROR: gwi_mk_error called with bad act_idx!	
	Description:	The Domain Gateway library software detected that an error occurred between a send and receive. The connection will be closed.
	Action:	No action required.
	See Also:	None.

1112	ERROR: Too many unrecoverable errors occurred - deleting action
	Description: The Domain Gateway library software detected more than 2 errors while trying to complete the action.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1113	ERROR: Unrecoverable error occurred on send of reply request - deleting action
	Description: Cannot send a reply to a call made with TPNOREPLY.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1114	ERROR: Local request '<service>' timed out, returning error response
	Description: The request for the local service timed out.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1115	WARN: Locking session <conn>/<session> awaiting late response
	Description: The request for the local service timed out.
	Action: No action required.
	See Also: None.
1116	ERROR: Local request '<service>' failed, returning error response
	Description: The request for the local service failed.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1117	ERROR: Error on receive from <gwname>
	Description: The Domain Gateway library software detected an error when receiving a request from a remote gateway.
	Action: Contact your TUXEDO System Technical Support.

	See Also:	None.
1118	ERROR: Unrecoverable error occurred on unknown receipt(2) - deleting action	
	Description:	The current action has no associated session.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1119	ERROR: Unrecoverable error occurred on send of reply - deleting action	
	Description:	The Domain Gateway library software detected an error when sending a reply.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1120	ERROR: Unrecoverable error occurred - deleting GWEV_MSG_FAILURE action (type <action>)	
	Description:	The Domain Gateway library software detected an unrecoverable error.
	Action:	Previous logged error messages should provide more information. Contact your TUXEDO System Technical Support.
	See Also:	None.
1121	"ERROR: Unrecoverable error occurred - changing action (type <action>) into GWEV_MSG_FAILURE	
	Description:	The Domain Gateway library software detected an unrecoverable error.
	Action:	Previous logged error messages should provide more information. Contact your TUXEDO System Technical Support.
	See Also:	None.

1122	ERROR: Unrecoverable error occurred - deleting action (<i>type <action></i>)
	Description: The Domain Gateway library software detected an unrecoverable error.
	Action: Previous logged error messages should provide more information. Contact your TUXEDO System Technical Support.
	See Also: None.
1123	INFO: Request id not found
	Description: The action with the given request ID was not found in shared memory.
	Action: No action required.
	See Also: None.
1124	ERROR: convid not found
	Description: The session with the given conversation ID was not found in shared memory.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1125	ERROR: Invalid network address <<<i>addr</i>>>
	Description: The given network address is not valid.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1126	ERROR: Invalid host name <<<i>host</i>>>
	Description: Could not get the network address for the listed host.
	Action: Contact your TUXEDO System Technical Support.
	See Also: None.
1127	ERROR: Bad ip address format <<<i>addr</i>>>
	Description: The format of the listed ip address is invalid.

	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1128	ERROR: Input string too long <<addr>>	
	Description:	The address is too long to fit in the memory allocated for it.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1129	ERROR: Timeout on remote service <service> on <gateway>	
	Description:	The Domain Gateway library software detected a timeout on the listed remote service.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1130	ERROR: Timeout on local service <service>, returning error to <gateway>	
	Description:	The Domain Gateway library software detected a timeout on the listed local service.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1131	ERROR: Connect time-out on connection <conn> to <gateway>	
	Description:	The Domain Gateway library software detected a timeout while attempting to connect to the remote gateway.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.
1132	ERROR: Idle time-out on connection <conn> to <gateway>	
	Description:	The Domain Gateway library software detected a timeout on the listed connection to the remote gateway.
	Action:	Contact your TUXEDO System Technical Support.
	See Also:	None.

1133	ERROR: Parse error: <i><error text></i> line <i><lineno></i> column <i><col></i> (after <i><keyword></i>) at <i><yytext></i>
	Description: The Domain Gateway library software encountered an error while parsing the gateway configuration file.
	Action: Correct the gateway configuration file.
	See Also: None.
1134	ERROR: No NWDEVICE specified
	Description: There was no NWDEVICE specified in the gateway configuration file.
	Action: Correct the gateway configuration file.
	See Also: None.
1135	ERROR: NATIVE name <i><gateway></i> already in use
	Description: The listed gateway name is specified more than once in the NATIVE section of the gateway configuration file.
	Action: Correct the gateway configuration file.
	See Also: None.
1136	WARN: Unreasonable POLL_TIME <i><time></i> ignored for NATIVE gateway
	Description: The gateway configuration file specifies a poll time which is less than 100,000 or greater than 10,000,000.
	Action: No action required. The poll time will be set to 250,000.
	See Also: None.
1137	ERROR: REMOTE_SERVICE <i><service></i> has no OUTREQ_TIME, no default in GLOBAL
	Description: The gateway configuration file does not have an OUTREQ_TIME specified for a remote service
	Action: Correct the gateway configuration file.
	See Also: None.

1138	ERROR: FOREIGN name <i><gateway></i> already in use
Description:	The listed gateway name is specified more than once in the FOREIGN section of the gateway configuration file.
Action:	Correct the gateway configuration file.
See Also:	None.
1139	ERROR: Missing mandatory TYPE parameter for FOREIGN <i><gateway></i>
Description:	The gateway configuration file does not have a TYPE parameter for the listed gateway.
Action:	Correct the gateway configuration file.
See Also:	None.
1140	ERROR: Missing mandatory WRAP parameter for FOREIGN <i><gateway></i>
Description:	The gateway configuration file does not have a WRAP parameter for the listed gateway.
Action:	Correct the gateway configuration file.
See Also:	None.
1141	ERROR: Missing mandatory IP_ADDR parameter for FOREIGN <i><gateway></i>
Description:	The gateway configuration file does not have an IP_ADDR parameter for the listed gateway.
Action:	Correct the gateway configuration file.
See Also:	None.
1142	ERROR: LOCAL_SERVICE name <i><service></i> already in use
Description:	The listed service name is specified more than once in the LOCAL_SERVICE section of the gateway configuration file.
Action:	Correct the gateway configuration file.

	See Also: None.
1143	ERROR: LOCAL_SERVICE <service> NATIVE <gateway> has no TCP_PORT
	Description: There is no TCP_PORT parameter for the local gateway providing the listed service in the gateway configuration file.
	Action: Correct the gateway configuration file.
	See Also: None.
1144	ERROR: LOCAL_SERVICE <service> attached to no NATIVE
	Description: The listed local service has no NATIVE parameter in the gateway configuration file.
	Action: Correct the gateway configuration file.
	See Also: None.
1145	ERROR: REMOTE_SERVICE name <service> already in use
	Description: The listed service name is specified more than once in the REMOTE_SERVICE section of the gateway configuration file.
	Action: Correct the gateway configuration file.
	See Also: None.
1146	ERROR: Missing mandatory FOREIGN parameter for REMOTE_SERVICE <service>
	Description: The gateway configuration file does not have a FOREIGN parameter for the listed service.
	Action: Correct the gateway configuration file.
	See Also: None.
1147	ERROR: REMOTE SERVICE <service> FOREIGN <gateway> has no TCP_PORT
	Description: There is no TCP_PORT parameter for the remote gateway providing the listed service in the gateway configuration file.

	Action:	Correct the gateway configuration file.
	See Also:	None.
1148	ERROR: REMOTE_SERVICE <service> attached to no FOREIGN	
	Description:	The listed remote service has no FOREIGN parameter in the gateway configuration file.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1149	ERROR: Mandatory <section> section missing.	
	Description:	The gateway configuration file must contain the GLOBAL, FOREIGN, NATIVE, LOCAL_SERVICES, and REMOTE_SERVICES sections.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1150	ERROR: Duplicate <section> section.	
	Description:	The gateway configuration file contains multiple entries for the listed section.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1151	ERROR: Section <section> out of order	
	Description:	The listed section is not in the proper order in the gateway configuration file.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1152	INFO: Section order must be *GLOBAL then *NATIVE/*FOREIGN then *LOCAL_SERVICES/*REMOTE_SERVICES	
	Description:	The gateway configuration file must have the following order of sections: GLOBAL, NATIVE/FOREIGN, LOCAL_SERVICES/REMOTE_SERVICES.

	Action:	No action required.
	See Also:	None.
1153	WARN: Bad subtype <subtype> truncated to <max> characters	
	Description:	The gateway configuration file specifies a subtype name which is too long.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1154	WARN: Bad type <type> truncated to <max> characters	
	Description:	The gateway configuration file specifies a type name which is too long.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1155	WARN: SECURE= accepts Y or N, not <parameter>	
	Description:	The gateway configuration file contains an invalid entry for the SECURE parameter.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1156	WARN: DSECURITY= accepts Y or N, not <parameter>	
	Description:	The gateway configuration file contains an invalid entry for the DSECURITY parameter.
	Action:	Correct the gateway configuration file.
	See Also:	None.
1157	WARN: DUMP= accepts Y or N, not <parameter>	
	Description:	The gateway configuration file contains an invalid entry for the DUMP parameter.
	Action:	Correct the gateway configuration file.
	See Also:	None.

1158	WARN: CONNSYNC= accepts Y or N, not <parameter>
Description:	The gateway configuration file contains an invalid entry for the CONNSYNC parameter.
Action:	Correct the gateway configuration file.
See Also:	None.
1159	WARN: CICS= accepts Y or N, not <parameter>
Description:	The gateway configuration file contains an invalid entry for the CICS parameter.
Action:	Correct the gateway configuration file.
See Also:	None.
1160	ERROR: Remove obsolete line in <filename> : "REPLY = parameter"
Description:	The gateway configuration file contains a REPLY parameter, which is obsolete.
Action:	Remove the REPLY parameter from the gateway configuration file.
See Also:	None.
1161	ERROR: CONV=Y, conversational mode not supported
Description:	The gateway configuration file specifies CONV=Y for a local service. This is currently not supported.
Action:	Correct the gateway configuration file.
See Also:	None.
1162	WARN: CONV= accepts Y or N, not <parameter>
Description:	The gateway configuration file contains an invalid entry for the CONV parameter.
Action:	Correct the gateway configuration file.
See Also:	None.

1163	ERROR: Bad unwrapping protocol
Description:	The unwrapping protocol in shared memory is not GWI_PROTOCOL.
Action:	Contact your TUXEDO System Technical Support.
See Also:	None.

D COBOL Data Encoding

An additional encoding library, `ConvMVSC`, has been included for TUXEDO clients using COBOL data types. This library, `ConvMVSC`, is similar to the default library, `ConvMVS`, but differs in the following ways:

- ◆ For the `STRING` data type, both `ConvMVSC` and `ConvMVS` perform ASCII-to-EBCDIC and EBCDIC-to-ASCII translation.
 - ◆ For strings sent to a remote gateway, both libraries will perform ASCII-to-EBCDIC conversion, and forward the string to the mainframe.
 - ◆ For string received from a remote gateway, the `ConvMVS` library will perform EBCDIC-to-ASCII translation, truncate any trailing space characters and add a NULL terminator. The `ConvMVSC` library will perform the translation, but will not truncate spaces or add the terminator.
- ◆ For the `VIEW` data type, the `ConvMVS` and `ConvMVSC` libraries treat all field types except `STRING` the same.
 - ◆ For `STRING` fields within a view sent to a remote gateway, `ConvMVS` will perform ASCII-to-EBCDIC conversion, and append space to 'pad' the string with space characters to the size of the field. `ConvMVSC` will perform the character conversion, but will not perform any 'padding'.
 - ◆ For `STRING` fields within a view received from a remote gateway, `ConvMVS` will perform EBCDIC-to-ASCII conversion, truncate any trailing space characters, and add a NULL terminator. The `ConvMVSC` library will perform the character conversion, but will not truncate spaces or add the terminator.

Using the COBOL Data Encoding Library

There are two methods for enabling the COBOL data-encoding library:

- ◆ COBOL data encoding for all services
- ◆ COBOL data encoding for messages to and from a specific host

Encoding for All Services

If COBOL data encoding is desired for every service in the gateway, set the parameter `DFLTTYPE="MVSC"` in the `*GLOBAL` section of the `GWICONFIG` file.

Listing D-1 Encoding for All Services

```
*GLOBAL  
  
DFLTTYPE="MVSC"
```

Encoding Messages To and From a Specific Host

To enable COBOL data encoding for messages to and from a specific host, set the parameter `TYPE="MVSC"` for that hosts `*FOREIGN` entry in the `GWICONFIG` file.

Listing D-2 Encoding for Messages To and From a Specific Host

```
*FOREIGN  
  
HOST_NAME  
TYPE="MVSC"
```

Index

Symbols

- *DM_ACCESS_CONTROL
 - optional parameters 4-24
 - required parameters 4-24
 - syntax 4-23
- *DM_LOCAL_DOMAINS
 - optional parameters 4-20
 - required parameters 4-20
- *DM_LOCAL_SERVICES
 - optional parameters 4-25
 - required parameters 4-25
 - syntax 4-24
- *DM_REMOTE_DOMAINS
 - optional parameters 4-23
 - required parameters 4-22
 - syntax 4-22
- *DM_REMOTE_SERVICES
 - optional parameters 4-26
 - required parameters 4-26
 - syntax 4-26
- *DM_ROUTING
 - optional parameters 4-30
 - required parameters 4-28
 - syntax 4-28

A

- access control list
 - format 4-23
 - local domains 4-23
 - using wildcards 4-24

- ACL file,enabling security 3-26
- audit log
 - local domain 4-20

B

- BEA eLink TCP for TUXEDO
 - See eLink TCP for TUXEDO
- BEA TUXEDO
 - See TUXEDO
- blocking 4-21
- boot options 4-5
- buffer types
 - TUXEDO ATMI 1-6
 - X/Open standard XATMI 1-6
- buffers
 - configuration parameters for mapping 3-5
 - data-dependent routing 4-28
 - definition 3-3
 - received from local programs 3-4

C

- CARRAY buffer type 1-6
- CICS
 - GWICONFIG
 - *FOREIGN section 4-14
 - CICS on IBM MVS 1-1
 - CICSDATA
 - GWICONFIG
 - *FOREIGN section 4-15

CICSHAND

GWICONFIG

*FOREIGN section 4-14

classes

grouping local domains 4-20

remote domains 4-23

CLOPT 4-5

COBOL

data types D-1

COBOL data encoding

all services D-2

messages to and from a host D-2

configuration

setting parameters for 3-13

configuring

servers 4-5

CONNECT_TIME 4-9

GWICONFIG

*FOREIGN section 4-15

connections

maximum per gateway 4-21

CONNSYNC

GWICONFIG

*FOREIGN section 4-15

CONV

GWICONFIG

*LOCAL_SERVICES section 4-17,
4-19

*REMOTE_SERVICES section 4-
19

conversational communication A-1

converting

input/output data 3-3

ConvMVSC

COBOL data encoding library D-1

D

data

maximum amount 4-21

data encoding

COBOL library D-1

data translation 3-19

data types

COBOL D-1

dequeuing by eLink TCP for TUXEDO,
prioritization for A-1

DFLTTYPE 4-10

DFLTWRAP 4-10

DMCONFIG

identifying local domains 4-19, 4-22

parameters 4-19

DMTLOG 4-21

Domain transaction log 4-21

page size 4-21

domains

local 4-19

remote 4-22

E

eLink TCP for TUXEDO

architecture 1-2

control information 1-4

converting input/output data 3-3

definition 1-1

limitations on using A-1

starting 5-1

eLink TCP for TUXEDO configuration file

See GWICONFIG

environment variables, setting 5-1

errors

messages C-1

F

FML buffer type 1-6–1-7

FOREIGN 4-18

G

GATEWAY_NAME 4-11

gateways

- associated to local domains 4-19
- maximum connections 4-21
- server group names 4-20

group file,enabling security 3-25

GWICONFIG

- *FOREIGN** section 4-12
 - description 4-6
- *GLOBAL** section
 - description 4-6
- *LOCAL_SERVICES** section 4-15
 - description 4-6
- *NATIVE** section
 - defining 4-11
 - description 4-6
- *REMOTE_SERVICES** section
 - defining 4-17
 - description 4-7
- buffer conversion 3-4
- environmental differences 3-1
- FML buffer support 1-7
- initializing 2-2
- setting up 4-6

I

IDLE_TIME

GWICONFIG

- *FOREIGN** section 4-14
- *GLOBAL** section 4-9
- *NATIVE** section 4-12

IMS/DC on IBM MVS 1-1

IMS/TM on IBM MVS 1-1

INBUFTYPE 3-10

- buffer conversion 3-5

DMCONFIG

- *DM_LOCAL_SERVICES** section 4-25
- *DM_REMOTE_SERVICES** section 4-25, 4-27

GWICONFIG

- *REMOTE_SERVICES** section 4-27
- mapping buffers to records 3-7
- mapping possibilities 3-13
- mapping records to buffers 3-11

informational messages C-1

input/output data

- converting 3-3

INRECTYPE

GWICONFIG

- *LOCAL_SERVICES** section 4-16
- *REMOTE_SERVICES** section 4-18
- mapping 3-10
- mapping buffers to records 3-7
- mapping possibilities 3-13
- mapping records to buffers 3-11
- record conversion 3-5

IPADDR

GWICONFIG

- *FOREIGN** section 4-13
- *NATIVE** section 4-12

L

LATENCY 4-10

local domain

- access control list 4-23
- audit log 4-20
- class groups 4-20
- identifying 4-19
- maximum amount of data 4-21
- maximum message length 4-22
- naming 4-19
- number of simultaneous global transactions 4-22
- routing requests to remote service 4-26
- services exported 4-24
- syntax for ***DM_LOCAL_DOMAINS** 4-19

log

transaction 4-21
LTERM
 exchanging information B-1

M

MACONNECT
 GWICONFIG
 *NATIVE section 4-12
mapping
 input buffers to input records 3-7, 3-13
 input records to input buffers 3-10
 locally originated calls 3-6
 output buffers to output records 3-12
 output records to output buffers 3-8, 3-15
 remotely originated calls 3-9
MAXCONNECT
 GWICONFIG
 *FOREIGN section 4-15
messages
 COBOL data encoding D-2
 error C-1
 informational C-1
 maximum length for local domain 4-22
MULTIPLEX
 GWICONFIG
 *FOREIGN section 4-14
 *GLOBAL section 4-10
 *NATIVE section 4-12

N

NATIVE 4-16
non-transactional communication A-1
notation conventions xii–xiii
NWDEVICE 4-9

O

OLTP systems supporting eLink TCP for
 TUXEDO gateways 1-1

OUTBUFTYPE 3-5, 3-8, 3-10, 3-12, 3-16
 DMCONFIG
 *DM_LOCAL_SERVICES section
 4-25
 *DM_REMOTE_SERVICES
 section 4-27
OUTRECTYPE
 GWICONFIG
 *LOCAL_SERVICES section 4-16
 *REMOTE_SERVICES section 4-
 18
 mapping 3-10
 mapping buffers to records 3-12
 mapping output records to output buffers
 3-8
 mapping possibilities 3-16
 record conversion 3-5
OUTREQ_TIME 4-9

P

parameters
 DMCONFIG 4-19
PASSWORD
 GWICONFIG
 *FOREIGN section 4-15
PATH 5-1
POLL_TIME
 GWICONFIG
 *NATIVE section 4-12

R

records
 configuration parameters for mapping 3-
 5
 definition 3-3
remote domain
 classes 4-23
 data-dependent routing 4-27
 format for *DM_REMOTE_SERVICES

- 4-26
- identifying 4-22, 4-23
- imported services 4-26
- RDOMs in access control list 4-24
- restricting services 4-25
- service execution 4-27
- syntax for *DM_REMOTE_DOMAINS 4-22
- remote service
 - routing request to 4-26
- remote services
 - dynamic advertising of 2-2
- Requesters
 - local priority A-1
- requests
 - routing to remote service 4-26
- RMTACCT
 - GWICONFIG
 - *FOREIGN section 4-14
- routing
 - data-dependent 4-27
 - format for *DM_ROUTING 4-28
 - requests to remote service 4-26
 - service requests for typed buffers 4-28
- RQADDR 4-5

S

SECURE

- GWICONFIG
 - *GLOBAL section 4-10
 - *LOCAL_SERVICES section 4-16
 - *REMOTE_SERVICES section 4-18

security

- data area B-1
- description 3-22
- enabling 3-24
- mainframe to UNIX 3-23
- passing LTERM information B-1
- sample ACL file 3-26

- sample group file 3-25
- sample user file 3-25
- UNIX to mainframe 3-22
- servers
 - configuring 4-5
- service calls
 - routing through eLink TCP for TUXEDO 1-3
- SERVICE_NAME
 - GWICONFIG
 - *LOCAL_SERVICES section 4-16
 - *REMOTE_SERVICES section 4-18
- services
 - COBOL data encoding D-2
 - execution by remote domain 4-27
 - exported by local domains 4-24
 - format for *DM_LOCAL_SERVICES 4-24
 - imported on remote domains 4-26
 - restricting requests by remote domains 4-25
- starting eLink TCP for TUXEDO 5-1
- STRING buffer type 1-6
- support
 - customer xv
 - documentation xv
- SYSTEM_NAME 4-13

T

TCP_PORT

- GWICONFIG
 - *FOREIGN section 4-14
 - *NATIVE section 4-12

- tadmin command 1-8, 5-2
- tpacl,sample file 3-26
- tpbroadcast() function A-1
- tpgprio() function A-1
- tpgrp,sample file 3-25
- tpnotify() function A-1

tpsprio() function A-1
tpusr,sample file 3-25
transaction log
 name 4-21
transactions
 simultaneous global 4-22
translation of data 3-19
TUXDIR,setting environment variables 5-1
TUXEDO
 administration, effects of eLink TCP for
 TUXEDO on 1-8
 applications, effects of eLink TCP for
 TUXEDO on 1-6
 ATMI buffer types 1-6
 buffer types 1-6
 Bulletin Board 1-4
 configuration file
 See UBBCONFIG
TYPE
 GWICONFIG file
 *FOREIGN section 4-13
typed buffers
 conversion of 1-7
 data-dependent routing 4-28

U

UBBCONFIG 4-1
 updating 4-1
 updating *GROUPS section 4-2
 updating *SERVERS section 4-3
user data area fields B-2
user file,enabling security 3-25

V

VIEW
 buffer type 1-6
 definitions 1-7, 2-3, 3-18

W

wildcards,access control lists 4-24
WRAP 4-14

X

X/Open standard XATMI buffer types 1-6
X_C_TYPE buffer type 1-6
X_COMMON buffer type 1-6
X_OCTET buffer type 1-6