Administrator’s Guide

Tmax 6

TMAXSOFT

Copyright

Restricted Rights Legend

Trademarks

Open Source Software Notice

Some modules or files of this product are subject to the terms of the following licenses: openssl-0.9.7.m, zlib-1.1.4, expat-2.0.0, netsnmp, DCE1.0, pthread, google-diff-match-patch, libevent, getopt.

Detailed Information related to the license can be found in the following directory: ${INSTALL_PATH}/license/oss_licenses

Document History

Product Version	Guide Version	Date	Remarks
Tmax 6	2.1.1	2018-06-11	-

Contents

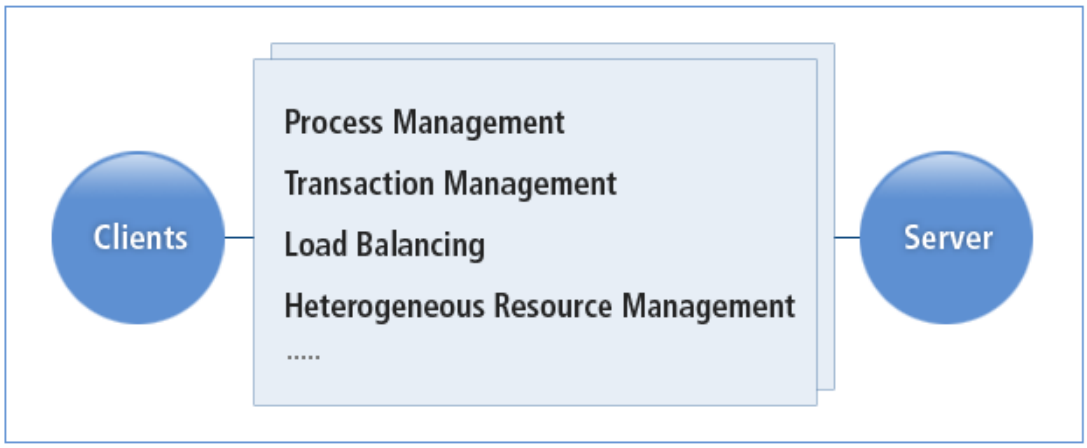
1. Introduction

This chapter describes the internal architecture, system architecture, administration, and directory structure of Tmax.

1.1. Overview

In a client/server computing environment, an increase in the number of clients has caused problems such as machine diversification, systems running a mixture of different operating systems and databases, data incompatibility, and deterioration of server performance.

Tmax, a transaction processing middleware solution, resolves such problems and provides functions such as process and transaction management, load balancing, and resource management in a heterogeneous environment.



Tmax Features

- **Process management**

  Tmax manages server instance (server process) loading, task distribution, and automatic process creation.

- **Transaction management**

  Tmax ensures the ACID properties of each transaction (Atomicity, Consistency, Isolation, and Durability).

- **Load balancing**

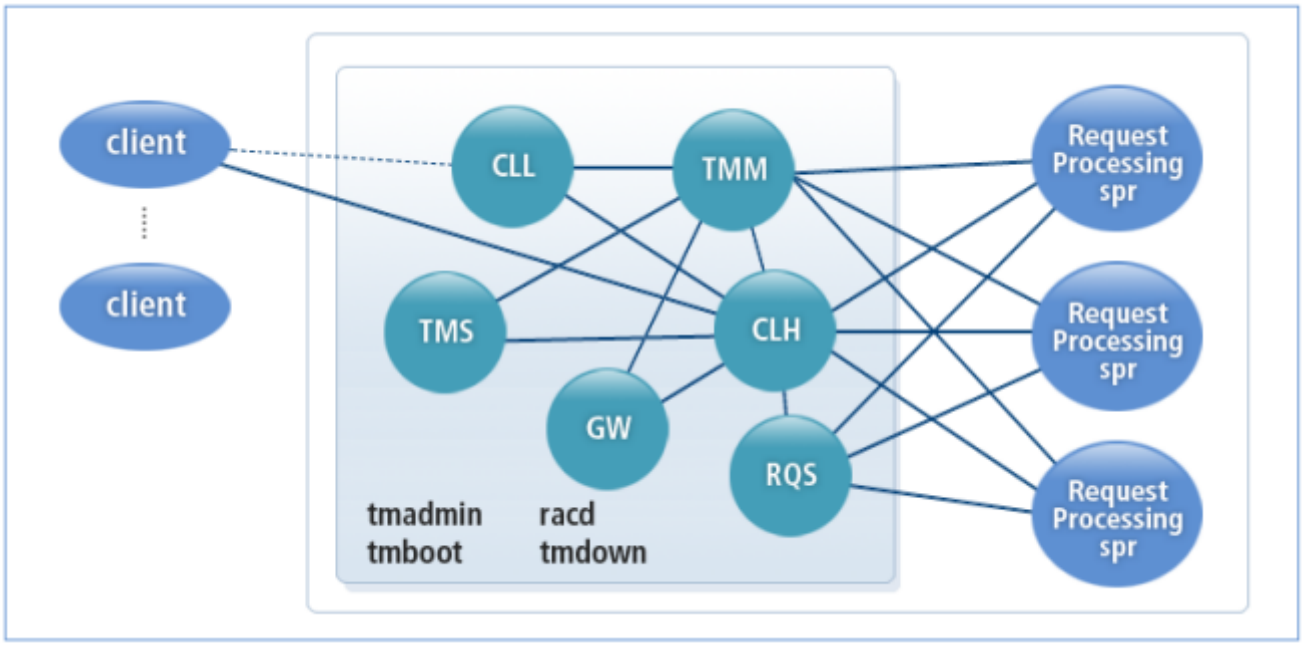
  Tmax provides optimal system performance and efficient processing.

- **Heterogeneous resource management**

  Tmax resolves various problems that arise from integrating heterogeneous platforms/resources.

1.2. Tmax Architecture

The Tmax system is made up of nine system operation processes (TMM, TMS, CLL, CLH, RQS, GW, CAS, TLM, and HMS) and four management tools (tmadmin, racd, tmboot, and tmdown).



Tmax Engine Architecture

- System Operation Processes
  - TMM (Tmax Manager)

    TMM, the Tmax Manager, manages the overall operations of Tmax. It manages system operation information and CLL, CLH, TMS, and RQS processes as well as various Tmax application processes. TMM is the first component to load when Tmax starts up, and the last to be terminated when Tmax shuts down.

    If a Tmax system contains multiple nodes, the TMM on each node maintains session connections with the others to monitor their heartbeats and provide fault tolerance for the server group. TMM is also responsible for logging each process and updating all shared data. TMM ensures normal system operation when a problem occurs that disrupts logging, such as lacking disk space, by temporarily suspending logging until the problem is resolved.

  - CLL (Client Listener)

    CLL, the Client Listener, forms an initial link between a client and Tmax system. The initial client-Tmax connection is handled by the CLL. When a client issues a service request, all service processing is done through an internal connection with CLH.

  - CLH (Client Handler)

    CLH, the Client Handler aka the client manager, serves as a mediator between client and server processes. CLH controls data flow by connecting to application server process, gateway, RQ process, and Tmax process. When a service is requested by a client, the CLH forwards the request to the appropriate application server process. Once it receives the

results from the application server process, it returns them to the client.

- TMS (Transaction Management Server)

  TMS, Transaction Management Server, is responsible for managing database and distributed transactions. While TMM, CLL, and CLH are critical processes that always gets loaded into memory, the TMS is only responsible for database management and distributed transaction processing. It is implemented by using the database libraries.

- RQS (Reliable Queue Server)

  The RQS, Reliable Queue Server, manages Tmax disk queues. Generally, all data stored within a system is at some level of risk, and the data stored in memory waiting to be processed can be lost in the event of a system failure. The RQS resolves this problem by storing data on the disk before and after service requests to preserve the stored data and resume services when the system is restarted.

  The RQS is only loaded when it is specified in the configuration file since an active RQS process can negatively impact the service processing performance.

- GW (Gateway)

  In a Tmax system that consists of multiple domains, the GW (Gateway) handles inter-domain communications. By using communication methods such as TCP/IP, SNA/LU0, SNA/LU6, and X.25, GW can facilitate communication and integration not only between different Tmax domains, but also between Tmax system and various other systems.

  Gateways are only loaded when they are specified in the configuration file. The actual name of a process may vary depending on the setting in the configuration file, and some gateways may require settings to be added to or modified in the configuration file. GW is an optional process that runs only when it is specified in the configuration file.

- CAS (Client Authentication Server)

  In Tmax systems that require security, the CAS performs level one and level two security checks for user authentication.

- TLM (Transaction Log Manager)

  TLM, Transaction Log Manager, monitors pending transactions and performs transaction logging, which was previously performed by TMM (earlier than 4.0).

- HMS (Hybrid Messaging System)

  HMS, Hybrid Messaging System, serves as the communication mediator that enables loose coupling between a sender and a receiver. Even though a sender and a receiver do not know each other, the messaging system enables them to communicate with one another with only virtual channel information of the destination. HMS ensures reliability in message transmission in order to allow delayed data processing by providing the ability to distinguish between the time the message is sent and received.

- Management Tools
  - tmadmin (Tmax Administration)

    Tmadmin is a management tool that can be used to dynamically check or modify the Tmax configuration file. It can also be used to monitor the behavior of server processes and to check the statuses of application service processes and queues.

  - racd (Remote Access Control Daemon)

    Racd is a daemon process that runs on each node (machine) in the system. When a Tmax system is comprised of multiple nodes in a single domain (group of nodes that share a single configuration file), racd can be used to enable the central monitoring and management of the entire system and nodes from a single node through tmadmin. It can also be used to ensure that changes in the configuration file are applied to all nodes of the domain without having to copy the file each time.

  - tmboot (Tmax System Boot)

    Tmboot is used to start up Tmax using the configuration file. It first loads the system processes, including TMM, CLL, and CLH, and then the RQ, TMS, and GW processes for additional environment configuration, and lastly it loads the application server processes. Tmboot can be used with various options to start Tmax.

  - tmdown (Tmax System Down)

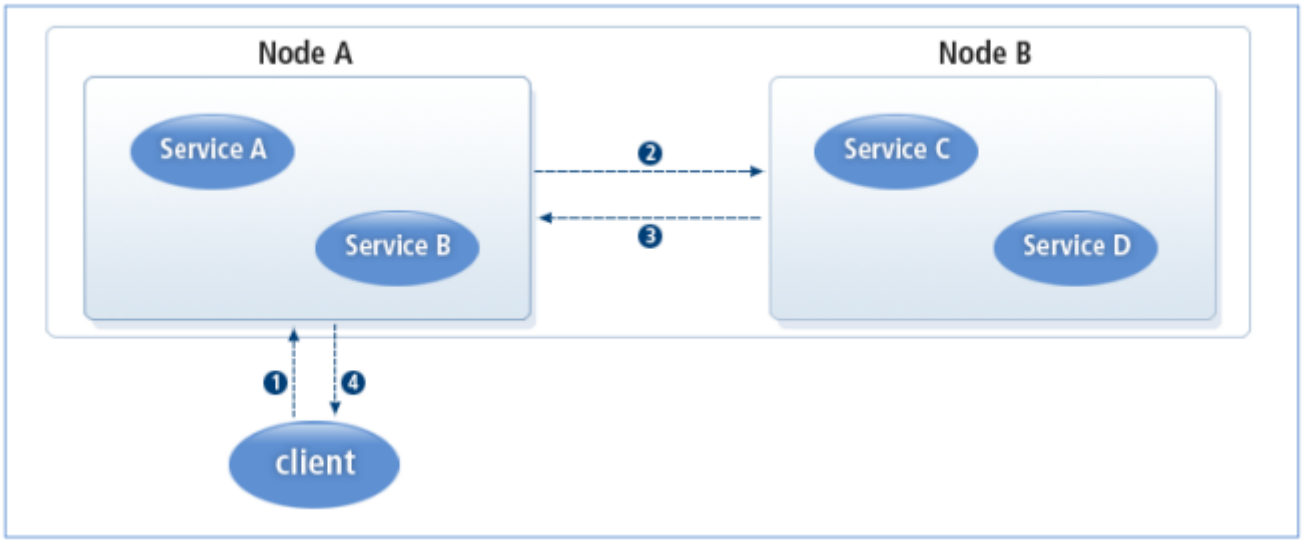
    Tmdown is used to shutdown Tmax using the configuration file. It first terminates application server processes, and then shuts down the Tmax management processes.

1.3. System Architecture

A **domain** is the highest modular unit of a Tmax system that uses a single configuration file. Domains can be composed of a single or multiple nodes, which are connected in a peer-to-peer relationship. In a domain with multiple nodes, the nodes communicate with each other at a constant time interval and share configuration information. Thus, all nodes of the domain possess information about all the other nodes and the overall domain status, which means that clients can access all services offered by the domain from any node.

Clients can connect to a particular node of the domain. If possible, a service requested by the client gets processed by the node that the client is connected to. However, if the service is provided by a different node, the node that the client is connected to will forward the request to that node. Once the request has been processed, the result will be returned to the client.

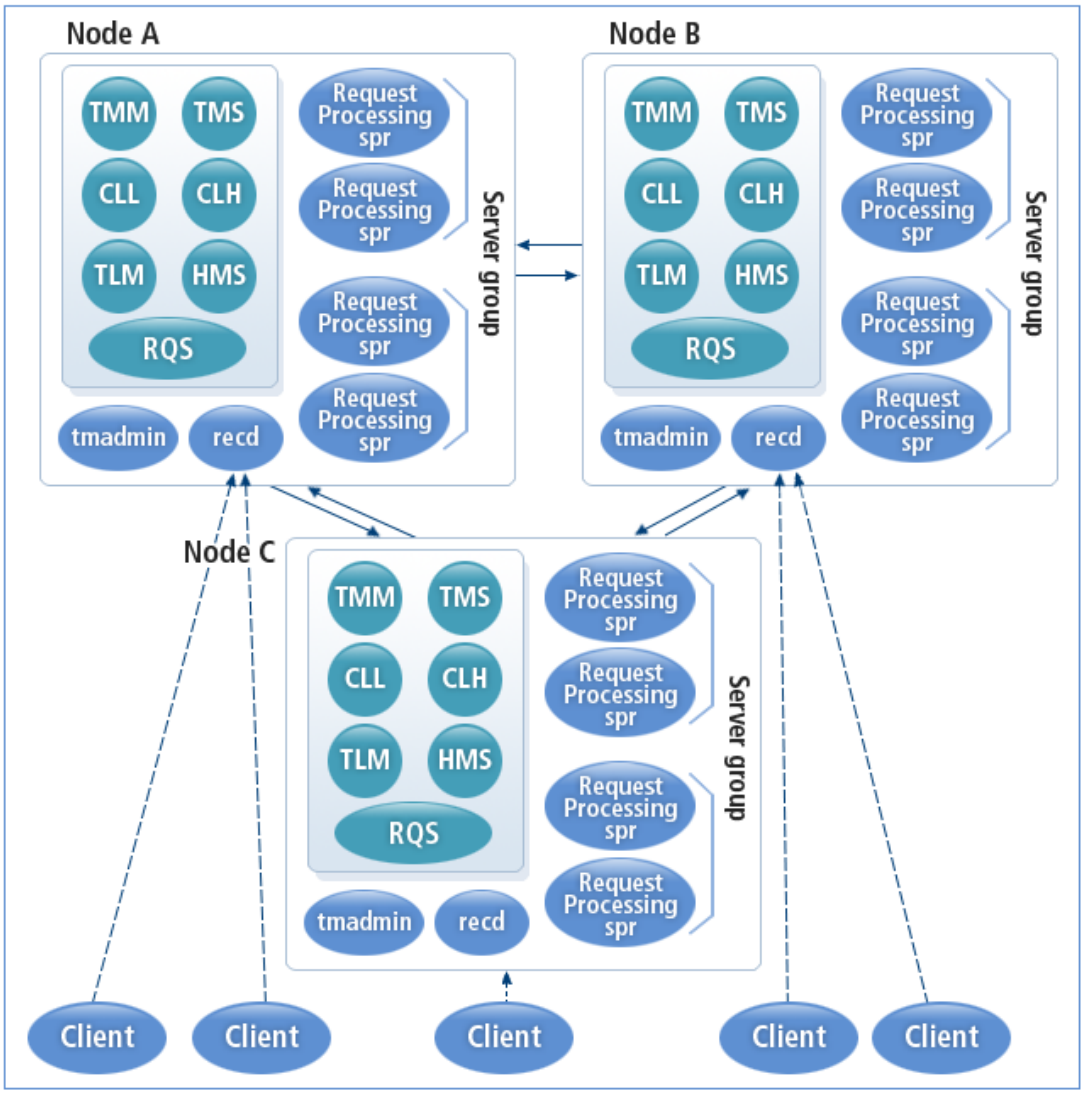
If the requested service is not provided by the node it is connected to, (refer to Passing a Request between Two Nodes) the request is processed as follows:



Passing a Request between Two Nodes

The node (Node A), which the client is connected to (1), communicates with the node that provides the service (Node B) to forward the request for processing (2). Node A will then receive the result from Node B (3) and then pass the result back to the client (4).

The following figure is a detailed architecture of Tmax system.



Tmax System Architecture

1.4. System Administration

Tmax system can be managed either statically or dynamically.

- **Static Management**

  Static management involves the manipulation of the **system configuration** file to perform administrative tasks before the system starts or without affecting the running system.

- **Dynamic Management**

  Dynamic management refers to manipulating the system while it is running to perform administrative tasks that may influence the state or performance of a running system, such as **starting or shutting down (tmboot/tmdown)**, **management through Tmax administration,** or the **modification of environment variables**.

1.4.1. Static Management

Static management of the system is done through environment configuration.

The following must be completed before configuring Tmax system.

- Creating Tmax Configuration File

  Before starting a Tmax system or configuring the environment, the configuration file for the system must be created. This is one of the key responsibilities of Tmax administrators. Once created, the Tmax configuration file will contain general information about a single domain that makes up an independent system unit, gateways that provide inter-domain connections or communication with legacy systems, nodes that make up the system, services of each node, server processes that provide services, and the logical groups of servers called a server group.

  For more information about creating the configuration file, refer to Environment Configuration.

- Configuration File Compilation

  Once the configuration file has been created, it will need to be compiled. If there are no errors in the configuration file, it will be compiled into a binary file. This binary file will be referred to whenever Tmax starts up or shuts down.

- Service Table Creation

  A complete configuration file is required before the service table can be created. The service table informs Tmax about the type and location of services available in the system. The service table should contain a separate table for each service defined in the configuration file. Executable programs must be linked to the service table in order for Tmax to recognize and execute them. The service table is used to compile program source codes written by developers to create server-side application programs.

1.4.2. Dynamic Management

The following describes how to dynamically manage system operation and system startup and shutdown.

- Startup and Shutdown

  After completing environment configuration, the administrator can start up Tmax. This involves starting various components of the Tmax engine, such as the TMM, CLL, and CLH (as well as TMS if database is involved, and RQS if RQ is involved) processes and the application server processes that will process client requests. It is important to note that normally servers are not started and terminated separately. They are managed by Tmax and are started and terminated when the Tmax engine components are started.

- System Operation Management

  The administrator can monitor the status of each system components while Tmax is running, and take appropriate actions if necessary. Dynamic management of the system is performed through **tmadmin**, an interactive monitoring program that can be accessed through the console.

  - Information Display

    Administrators can use tmadmin commands and options to collect information about clients (names and locations), servers (location, server group, availability, and execution count), server processes (location, status, processing time, execution count, and data backlog) and networking (connected nodes and partitioned nodes), as well as information on transactions and application program parameters.

  - Dynamic Modification of Tmax Configuration

    Administrators can dynamically modify Tmax system environment, including adjusting time-out values, priorities and load balancing, restarting dead processes, terminating processes, restarting terminated processes, and adding new services to an active server process.

Environment variables that are modified during system operation will revert back to the values defined in the configuration file when the system shuts down (tmdown) and is rebooted (tmboot).

1.5. Directory Structure

The following is the basic structure of the Tmax installation directory. This directory will be referred by several environment variables within the configuration file.

```
$Tmax HOME
+---- appbin
+---- bin
+---- config
+---- lib (or lib64 for 64-bit)
```

```
+---- license
|---- log
        +---- slog
        +---- tlog
        +---- ulog
+---- mod
+---- path
+---- run
|---- sample
        +---- client
        +---- tdl
        +---- fdl
        +---- sdl
        +---- server
+---- svct
+---- usrinc
+---- topinc
+---- cobinc
+---- tuxinc
+---- tcpgw
+---- tcpgwthr
+---- x25gw
+---- UninstallerData
+---- bk_appbin
```

$Tmax HOME

Tmax home directory. (System variable: TMAXDIR, configuration file item: TMAXDIR)

appbin

Contains server applications developed using Tmax. (Configuration file item: APPDIR)

bin

Contains Tmax commands and utilities.

config

Contains Tmax system configuration files.

lib (or lib64 for 64-bit)

Contains Tmax library files.

license

Contains license files.

log

Contains log files.

Subdirectory	Description
slog	Contains system log files. (Configuration file item: SLOGDIR)
ulog	Contains user log files. (Configuration file item: ULOGDIR)

Subdirectory	Description
tlog	Contains transaction information. (Configuration file item: TLOGDIR)

mod

Contains libraries to update when using TDL.

path

Used as a pipe for inter-process communication. (Configuration file item: PATHDIR)

run

Contains libraries with a version after executing tdlupdate when using TDL.

sample

Subdirectory	Description
client	Contains sample client program files.
tdl	Contains sample TDL program files.
fdl	Contains sample field key definition files (demo.f). (System variable: FDLFILE)
sdl	Contains sample structure definition file (demo.s). (System variable: SDLFILE)
server	Contains sample server program files.

svct

Contains service table files used to compile server applications.

usrinc

Contains Tmax header files.
demo.f defines Field Definition Language (FDL) fields, and demo.s defines Structure Definition Language (SDL) fields. You can create and use FDL and SDL fields for a project.

topinc

Contains header files used to migrate TOP END to Tmax.

cobinc

Contains COBOL header files.

tuxinc

Contains header files used to migrate Tuxedo to Tmax.

tcpgw

Contains TCPGW header files.

tcpgwthr

Contains TCPGW THR header files.

x25gw

Contains x25gw header files.

UninstallerData

Contains files used to uninstall Tmax.

bk_appbin (optional)

Contains a new process that is installed to replace an existing server process. Must be created by a user. (System variable: TMAX_BKAPPDIR)

2. Environment Variables

This chapter describes the environment variables used by Tmax system, and how to configure them.

2.1. Overview

There are two types of Tmax environment variables in the configuration file: system variables and server process variables. Most of the Tmax environment variables are used to access Tmax system. The same values should be specified for the environment variable in this chapter as for the items in the configuration file introduced in the next chapter. These variables will affect all server processes in the system.

On client-only machines, Tmax environment variables can be configured in one of the following two ways:

- In the shell's profile.
- As a file in the standard Tmax configuration file format.

Server process variables must be configured in a standard Tmax configuration file, and the file must be defined in the ENVFILE section of the configuration file.

2.2. Tmax Environment Variables

All clients and servers where Tmax is installed use the same environment variables, but there is no need to separately configure the environment variables of all Tmax nodes in the clients.

2.2.1. Tmax Server Configuration Variables

The following are environment variables configured on the server.

Variable	Description
TMAXDIR	Home directory of Tmax system (Configuration file item: TMAXDIR).
TMAX_BKAPPDIR	The directory, where a new process will be placed, when changing the existing server process.
TMAX_HOST_ADDR	IP address of the server where Tmax will run on. Loads are balanced as clients are randomly connected to servers. For more information, refer to Registering Multiple Servers.
TMAX_HOST_PORT	Port number for accessing Tmax (Configuration file item: TPORTNO).
TMAX_BACKUP_ADDR	IP Address of Tmax backup server.
TMAX_BACKUP_PORT	Port number of Tmax backup server.

Variable	Description
TMAX_RAC_PORT	Port number used by racd for centralized management of a system with distributed nodes (Configuration file item: RACPORT).
TMAX_ERR_MSG	Option to display error messages on the server console ("Y" or "N").
SDLFILE	File where structural information is stored.
FDLFILE	File where field key information is stored.
TMAX_PATHDIR	Environment file path that is referred to during tmdown (Configuration file item: PATHDIR).
TMAXHOME	To use the install directory and the work directory separately, set as follows. (Configuration file item: TMAXHOME) • TMAXHOME as the install directory: bin, lib, usrinc, tuxinc, topinc, cobinc, license • TMAXDIR as the work directory: config, path, log, svct
TMAX_SEMANTICS	Action to take according to the length of an FML buffer inside functions that read data (fbget, fbget_tu, fbget_tut, fbgetf, fbgetlast_tu, fbnext_tu). Enter the data length as the last argument of the function to verify the user-assigned buffer. This is used to prevent error when the size of the user-assigned buffer is smaller than the actual data size.
TMAX_TRACE	Option to enable run-time tracing of Tmax application execution.
TMAX_DEBUG	Name for the file to which a client's debugging and error logs will be recorded, instead of the screen. If set to filename.pid, the real name of the file is automatically set along with the client's process ID (PID). (Example: log.22672)
TMAX_STRING_NULL	Option to allocate a STRING type buffer by using the tpalloc or tprealloc API. If the user requests for service with data that fills up to the last one-byte space designated for NULL in the allocated buffer, a TPEINVAL (client) and TPESVCERR (server) error occurs. In such case, an additional one-byte buffer can be allocated and automatically set to a NULL character by setting this option to Y.
TMAX_HOST_IPV6	Set to 'Y' or 'IPV6' if an address set in the TMAX_HOST_ADDR variable uses the IPv6 protocol. Set to 'SDP' if the address uses InfiniBand Socket Direct Protocol (SDP).
TMAX_BACKUP_IPV6	Set to 'Y' or 'IPV6' if an address set in the TMAX_BACKUP_ADDR variable uses the IPv6 protocol. Set to 'SDP' if the address uses InfiniBand Socket Direct Protocol (SDP).
TMAX_WEBADM_IPV6	Option to use the IPv6 protocol when twagent (Tmax WebAdmin agent) listens from TMAX_WEBADM_PORT. Set to 'Y' or 'IPV6' to use IPv6. Set to 'SDP' to use InfiniBand Socket Direct Protocol (SDP).

Variable	Description
TMAX_RAC_IPV6	Option to use the IPv6 protocol when executing 'connect' or 'listen' from a racd daemon process that is used for integrated management of a distributed system. • Set to 'Y' or 'IPV6' to use IPv6, and set to 'SDB' to use InfiniBand Socket Direct Protocol (SDP). • Set to 'N' or no value to use IPv4. Only used when the [-k] option is used to start racd. SYSTEM_IPV6 setting will be used instead when a Tmax binary environment file is used to start racd.
TMAX_IPV6_LINK_IF	New network interface name to use if Tmax processes cannot recognize the interface index of a network interface name when executing 'connect' or 'listen' in an environment that uses the IPv6 protocol.
TMAX_APPLY_IPCPERM	When a server starts for the first time, permissions of log files that are written by specifying the -e and -o options are determined according to IPCPERM set in the DOMAIN or NODE section. • Set to 'Y' or 'y' to create the log file according to IPCPERM. • Set to 'N', 'n', or no value, it is not guaranteed that permissions of the log file created first follows IPCPERM.
TMAX_LOGLVL	Debug log level for tmm, modules that do not need to connect to tmm, and modules that are not connected to tmm yet. This variable has the same value as the value set in TMMLOGLVL under the DOMAIN section. Log output location can be set by using this variable along with TMAX_DEBUG. The default location of TMAX_DEBUG or higher log is stderr. General utilities output no message if TMAX_ERR_MSG is N or n. For modules that need to connect to tmm, the log output is determined by the TMAX_ERR_MSG environment variable before the tmm connection. After the tmm connection, all logs are outputted. In the DEBUG4 level, some gateways can output dump messages.
TMAX_TENC_CIPHER	Encryption method related to the SEED encryption/decryption utility that can be set in Tmax configuration file. If set to non-SEED or not set, existing encryption method is used.
TMAX_TENC_KEYFILE	Secret key location. Must be defined to use the SEED encryption algorithm.

TMAX_STRING_NULL is only supported for buffers allocated through tpalloc or tprealloc, and works similarly in both the server and client. Note that the variable must be set to Y.

The following is an example of setting TMAX_SEMANTICS. If a user specified buffer size is smaller than the actual data size, the data is truncated to the buffer size, and then written to the buffer.

```
TMAX_SEMANTICS = "FDL_LEN=TRU"
```

If the following is set, the FBENOSPACE error occurs when a user specified buffer size is smaller than the actual data size.

```
TMAX_SEMANTICS = "FDL_LEN=ERR"
```

If the following is set, a user specified length is ignored, and Tmax runs with the assumption that the user has allocated a buffer with sufficient length. This setting is valid only when the data length is specified as the last argument of the fbget type function. If the last argument is specified to 0 or NULL, Tmax runs with the assumption that the user allocated a buffer with sufficient length.

```
TMAX_SEMANTICS = "FDL_LEN=OLD"
```

2.2.2. Tmax Client Environment Variables

The following are environment variables configured on the client.

Variable	Description
TMAX_HOST_ADDR	Tmax system IP address.
TMAX_HOST_PORT	Tmax system port number.
TMAX_CONNECT_ TIMEOUT	Timeout value for connecting to Tmax (x.xxx seconds).
SDLFILE	File where structural information is stored.
FDLFILE	File where field key information is stored.
ULOG (or ULOGPFX)	Client side log path. If the path is not specified, the log will be recorded in ULOGDIR of the Tmax environment file.
TMAX_TRACE	Option to enable run-time tracing of Tmax application execution.

Variable	Description
TMAX_SEMANTICS	Action to take according to the length of an FML buffer inside functions that read data (fbget, fbget_tu, fbget_tut, fbgetf, fbgetlast_tu, and fbnext_tu). Refer to Tmax server configuration variables. The encryption function is provided for stable data transmission between a client and the server.
TMAX_CRYPT_ ALGORITHM	Encryption algorithm. "AES" can be specified. (Default: 3DES).
TMAX_DEBUG	Name for the file to which a client's debugging and error logs will be recorded, instead of the screen. If set to filename.pid, the real name of the file is automatically set along with the client's process ID (PID). (Example: log.22672)
TMAX_HOST_IPV6	Set to 'Y' or 'IPV6' if an address set in the TMAX_HOST_ADDR variable uses the IPv6 protocol. Set to 'SDP' if the address uses InfiniBand Socket Direct Protocol (SDP).
TMAX_BACKUP_IPV6	Set to 'Y' or 'IPV6' if an address set in the TMAX_BACKUP_ADDR variable uses the IPv6 protocol. Set to 'SDP' if the address uses InfiniBand Socket Direct Protocol (SDP).
TMAX_IPV6_LINK_IF	New network interface name to use if Tmax processes cannot recognize the interface index of a network interface name when executing 'connect' or 'listen' in an environment that uses the IPv6 protocol.
TMAX_ACTIVATE_ AUTO_TPSTART	Set to either Y or N. • Y: Automatically calls tpstart(NULL) to connect to Tmax if a client program does not explicitly call tpstart(), but calls a function such as tpcall() and tpacall(). (Default value) • N: If tpstart() is not called explicitly, the TPEPROTO error will occur. The following functions are affected by this variable. • tpcall(), tpacall(), tpcallsvg(), tpacallsvg(), tpconnect(), tpchkunsol(), tpsetunsol(), tpgetunsol(), tpsubscribe(), tpsubscribe2(), tpunsubscribe(), tppost(), tpbroadcast(), tpgetcliaddr(), tpgetpeername(), tpgetsockname(), tpsleep(), tpgetactivesvr(), tpqsvcstat(), tpqstat(), tpenq(), tpdeq(), tpreissue(), tpmcall(), tpsvgcall(), tpenq_ctl(), tpdeq_ctl(), tx_begin()

Variable	Description
PB_TPCALL_RETRY	Maximum number of attempted retries if a tpcall request fails due to a network error in the Power Builder environment. The following functions are affected by this variable. • pb_tpcall(), pb_tpfcall(), pb_tpcallw(), pb_btpcall(), pb_etpcall(), tuxcall(), tuxfcall32(), tp_call(), tp_call32(), tp_fcall(), tp_fcall32()
PB_TPCALL_RETRY_ TIMEOUT	BLOCKTIME per second for each request during a tpcall request in the Power Builder environment. For APIs which are affected by this environment variable, refer to the description about **PB_TPCALL_RETRY**.
PB_TUXCOMPAT	Provides compatibility with the Tuxedo library for using some of the APIs in the Power Builder environment. For detailed information, refer to *Tmax Programming Guide(4GL)*. The following functions are affected by this variable. • tuxcall(), tuxfcall32(), tp_call(), tp_call32(), pb_etpcall(), tp_fcall(), tp_fcall32()
FDLVERSION	FDL version. (Default value: 1) • 2: use the version with enhanced search performance and is not compatible with existing Tmax. This setting affects the scheduling of server applications and CLH that follows the ROUTING configuration. Configuration settings are not applied to clients, and the following environment variables must be configured. Set export FDLVERSION=2 or add "FDLVERSION=2" to the tmax.env environment file and then call tmaxreadenv(). When using fb* API, it returns the FBESETVER error if FDLVERSION setting does not match with its environment.
TMAX_LOGLVL	Debug log level for tmm, modules that do not need to connect to tmm, and modules that are not connected to tmm yet. This variable has the same value as the value set in TMMLOGLVL under the DOMAIN section.

Set TMAX_SEMANTICS to use the **encryption** function. For encryption, the encryption function must set for both the client and server. The client encryption function can be set in .profile as in the following. For information about encryption setting of the server, refer to "CRYPT in 3.2.1 DOMAIN".

<.profile>

```
export TMAX_SEMANTICS = "CRYPT_SW=Y"
```

To use TMAX_SEMANTICS in tmaxreadenv, use a colon (:) instead of an equal sign (=) inside double

quotes (").

<tmax.env>

```
TMAX_SEMANTICS = "FDL_LEN:ERR"
TMAX_SEMANTICS = "CRYPT_SW:Y"
```

2.3. Setting Tmax Environment Variables

The following describes how to set server and client environment variables.

2.3.1. Server Environment Variables

Environment variables for Tmax system must be configured according to the shell type. The variables must be configured in <.profile> for korn shell, in <.bash_profile> for bash shell, and in <.cshrc> for c shell.

Make sure that there are no blank spaces between the name of the variable and the equal sign ('=').

- korn shell and bash shell

  <.profile and .bash_profile>

  ```
  export TMAXDIR = /home/tmax
  export TMAX_BKAPPDIR = /home/tmax/bkappbin
  export TMAX_HOST_ADDR = 192.168.0.1
  export TMAX_HOST_PORT = 8888
  export TMAX_RAC_PORT = 3333
  export SDLFILE = /home/tmax/sample/sdl/tmax.sdl
  export FDLFILE = /home/tmax/sample/sdl/tmax.fdl
  export TMAX_PATHDIR = /home/tmax/path_new
  export TMAX_DEBUG = /home/tmax/sample/client/clidebug
  ```

- c shell

  <.cshrc>

  ```
  setenv TMAXDIR = /home/tmax
  setenv TMAX_BKAPPDIR = /home/tmax/bkappbin
  setenv TMAX_HOST_ADDR = 192.168.0.1
  setenv TMAX_HOST_PORT = 8888
  setenv TMAX_RAC_PORT = 3333
  setenv SDLFILE = /home/tmax/sample/sdl/tmax.sdl
  setenv FDLFILE = /home/tmax/sample/sdl/tmax.fdl
  setenv TMAX_PATHDIR = /home/tmax/path_new
  ```

```
setenv TMAX_DEBUG = /home/tmax/sample/client/clidebug
```

Tmax is installed by executing install.sh, which will automatically add the basic environment variables at the end of the shell environment file. Environment variables required for server processes must be configured in a text file, and the file must be set in the ENVFILE section of the NODE and SVRGROUP sections in Tmax environment file.

<Server Environment File>

```
LOGDIR = /tmp
USER_VARIABLE = test1
USER_VARIABLE = test2
USER_VARIABLE = test3
USER_VARIABLE = test4
USER_VARIABLE = test5
```

2.3.2. Client Environment Variables

The method for configuring the environment variables on client-only machines is different for each OS.

- UNIX

  Like server environment variables, variables are configured in the profile.

- DOS, Windows98, or Windows NT/2000

  Environment variables are configured in the <autoexec.bat> or in the system profile of the control panel. (**[Control panel] > [System properties] > [Advanced] > [Environment variables]**)

  <autoexe.bat>

  ```
  set TMAX_HOST_ADDR = 192.168.0.1
  set TMAX_HOST_PORT = 8888
  set TMAX_CONNECT_TIMEOUT = 3
  set SDLFILE = /home/tmax/sample/sdl/tmax.sdl
  set FDLFILE = /home/tmax/sample/sdl/tmax.fdl
  set TMAX_DEBUG = /home/tmax/sample/client/clidebug
  ```

Client environment variables must be defined in a text file, which the system will refer to.

<tmax.env>

```
[TEST]
TMAX_HOST_ADDR = 192.168.0.1
TMAX_HOST_PORT = 8888
TMAX_CONNECT_TIMEOUT = 3
SDLFILE = /home/tmax/sample/sdl/tmax.sdl
FDLFILE = /home/tmax/sample/sdl/tmax.fdl
ULOGPFX = /home/tmax/testlog
```

```
TMAX_DEBUG = /home/tmax/sample/client/clidebug

[REAL]
TMAX_HOST_ADDR = 192.168.0.2
TMAX_HOST_PORT = 1234
TMAX_CONNECT_TIMEOUT = 3
SDLFILE = /home/tmax/sample/sdl/tmax.sdl
FDLFILE = /home/tmax/sample/sdl/tmax.fdl
ULOGPFX = /home/tmax/reallog
TMAX_DEBUG = /home/tmax/sample/client/clidebug
```

For more information about setting client environment variables, refer to "3.1.37. tmaxreadenv" in *Tmax Reference Guide*.

2.4. Registering Multiple Servers

In the previous versions of Tmax, each client could only connect to one specific server, but versions 3.12.2 and later include load balancing feature to randomly connect each client to a server. To use multiple servers, configure the **TMAX_HOST_ADDR** setting.

To enable this function, the following must be set in the configuration file.

- IPv4

```
TMAX_HOST_ADDR = (host_address:portno|host_address2:portno2),
                 host_address3:portno3,host_address4:portno4
```

- IPv6

```
TMAX_HOST_ADDR = ([host_address]:portno|[host_address2]:portno2),
                 [host_address3]:portno3,[host_address4]:portno4
```

Backup addresses for the client to connect to are separated by a comma. In the previous example, the client will first attempt to connect to "host_address:portno|host_address2:portno2". If the connection attempt fails, the client will try to connect to "host_address3:portno3". If the second attempt also fails, then the client will try to connect to "host_address4:portno4".

The addresses in parenthesis separated by a pipe(|) have a load balancing relationship with one another. This means that a client will randomly attempt to connect to one of these addresses, and in case of a connection failure, the client will sequentially attempt to connect to those in the load balancing relationship. If all fails, it will then try to connect to the backup addresses.

The following is the description of TMAX_HOST_ADDR.

- Maximum length is 255 bytes.
- In the event that all attempts to connect to the addresses in TMAX_HOST_ADDR fail, the client will

then attempt to connect to the addresses specified by TMAX_BACKUP_ADDR.

- No spaces should be used for TMAX_HOST_ADDR value.
- Double quotation marks(" ") must be used when specifying environment variables with export or setenv in the shell, because the shell interprets parentheses and pipe character differently than other characters.

Example

The following example compares server configuration methods.

- Connecting the client by using existing settings

```
TMAX_HOST_ADDR = 192.1.1.1
TMAX_HOST_PORT = 9000
```

- Enabling load balancing that randomly connects the client to one of multiple addresses:

```
TMAX_HOST_ADDR = (192.1.1.1:9000|192.1.1.2:9001),192.1.1.3:9003,192.1.1.4:9004
```

- Randomly connecting the client to an address (IPv6 protocol environment)

  In an IPv6 protocol environment, addresses are placed inside square brackets([ ]) to distinguish a host address from a portno.

```
TMAX_HOST_ADDR = ([2011::10]:9000|[2011::20]:9001),[2011::100:30]:9003,[2011::100:40]:9004
```

3. Environment Configuration

This chapter describes how to set each environment variable.

3.1. Overview

The configuration file defines the environment of a single Tmax domain. A domain is a single independent Tmax system that can be made up of multiple nodes (machines). Domain settings include the maximum number of simultaneous users, time-out values, and service processing times.

Each node in the domain can be configured separately. Node settings include the maximum number of simultaneous users, the location of Tmax programs, and the location where application programs are processed.

Server groups can also be configured separately. A server group is made up of multiple server processes that provide application services, and it allows administrators to manage multiple server processes as a single unit. The server group settings include database access variables, load balancing variables, and fault tolerance variables.

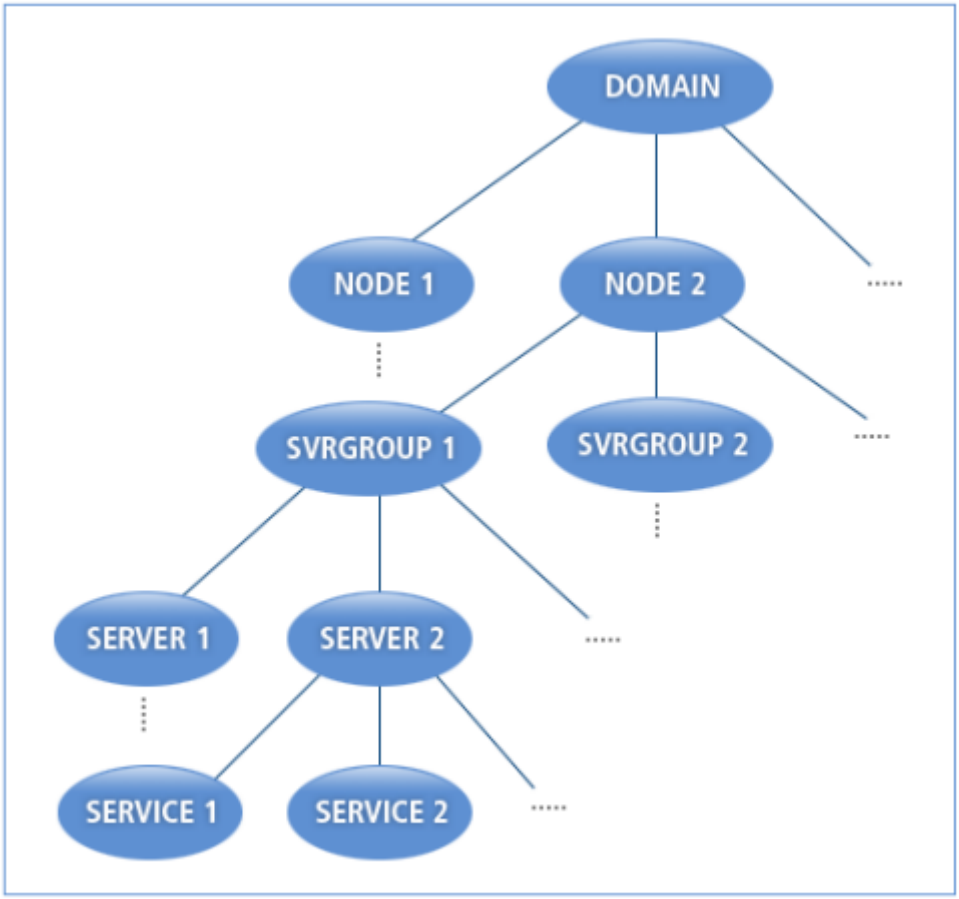
Configuration is also required for application server processes (command line options and the number of servers) that actually process client requests, services (service priorities and service timeout values), and routing (distributed processing across a server group based on the type and range of data). Gateway settings that allow inter-domain service processing can also be configured.

The configuration file contains a total of nine elements:

- DOMAIN
- NODE
- SVRGROUP
- SERVER
- SERVICE
- GATEWAY
- ROUTING
- RQ
- HMS

Elements are in hierarchical relationships of a tree structure. There is a domain at the root of the tree for the entire system environment. A domain contains one or more nodes, a node contains one or more server groups, a server group contains one or more servers, and an application server process contains one or more services.

The following figure shows the relationships.



Relationships between the Elements of Tmax System

3.1.1. Environment File Format

The nine elements of the Tmax environment file include: DOMAIN, NODE, SVRGROUP, SERVER, SERVICE, GATEWAY, ROUTING, RQ, and HMS. The first five elements must be configured by the administrator, but GATEWAY, ROUTING, RQ, and HMS are optional.

The following is an example of an environment file.

```
*DOMAIN
res1    SHMKEY = 77990, MINCLH = 1, MAXCLH = 3,
        MAXUSER = 100, TPORTNO = 8888, BLOCKTIME = 100

*NODE
tmax1   TMAXDIR = "/home/tmax",
        APPDIR = "/home/tmax/appbin",
        PATHDIR = "/home/tmax/path",
        SLOGDIR = "/home/tmax/log/slog",
        TLOGDIR = "/home/tmax/log/tlog",
        ULOGDIR = "/home/tmax/log/ulog"

*SVRGROUP
svg1    NODENAME = tmax1, DBNAME = ORACLE,
        OPENINFO = "Oracle_XA+Acc = P/tmax/tmax+SesTm = 60"
svg2    NODENAME = tmax1
```

```
*SERVER
svr1    SVGNAME = svg1, CLOPT ="-- -f aa -x bb"
svr2    SVGNAME = svg2

*SERVICE
SVC1    SVRNAME = svr1
SVC2    SVRNAME = svr2
```

Detailed Syntax

The following is the syntax of an element and an item.

- Element

  Each element is defined as follows:

  ```
  * Element
  Name        Item 1 = .··· , Item 2 = .··· , Item 3 = ....
  ```

- Item

  An item of each element is defined as follows. An item must be defined in the form of 'Item Name = Value'.

  ```
  Item = Type (default value)
         Range or Size
         Settings
  ```

  Each item value must be one of the following four types (item names cannot be used as values).

Item Value	Description
numeric	Number
string	abc type string
literal	"abc" type string
Y | N	Yes or No
Y | B | N	Yes, Both, or No

Configuration file must follow these guidelines:

- Each element name must be capitalized and begin with an asterisk (*).
- Each element name or item name must be placed in the first column.
- Only the element and item names can be in the first column.
- Each item of an element must be separated by a comma (,). If a comma is missing, it will indicate the end of the element and any subsequent items will be ignored.

- The order of defining elements is not fixed, and the order of elements is not important. Each element can be divided and defined in different places. For example, each node's server groups, servers, and services can be defined separately. Note that a setting must be only defined once.
- Empty spaces between items are ignored.
- Default values are used for undefined items.

3.1.2. Tmax Environment File

The environment file is accessed whenever Tmax starts or shuts down. The system will not operate if the configuration file is flawed or nonexistent. Thus, setting up the Tmax environment file is one of the most important responsibilities of the administrator.

The environment file is divided into the following nine sections.

- Basic settings

  Basic settings of Tmax system. Five most essential elements are DOMAIN, NODE, SVRGROUP, SERVER, and SERVICE. The user can additionally configure the GATEWAY element for inter-domain service processing, the ROUTING element for routing options, and other elements.

  Database, transaction, load balancing, fault tolerance, and security settings are based on the basic settings.

- Database settings

  Database related settings used for database management.

- Distributed transaction settings

  Transaction settings for distributed transaction processing in Tmax.

- Load balancing settings

  Settings that define load balancing between nodes. The element will also contain a detailed description of ROUTING.

- Reliable queue settings

  RQ settings for enabling uninterrupted processing of transactions after system recovery due to a system failure.

- HMS settings

  Settings for enabling delayed data processing by providing the ability to distinguish between the time the message is sent and received, and ensuring reliability for message transmission.

- Fault tolerance settings

  Settings for fault tolerance that allows service processing to be forwarded to another node when the current node fails.

- Security settings

  Security feature settings.

- Multi domain settings

  Inter-domain routing settings for a multi-domain system.

3.2. Basic Configuration

This section of the chapter will explain how to configure five of the nine elements (DOMAIN, NODE, SVRGROUP, SERVER, and SERVICE) described in the previous section, as well as the GATEWAY element for multi-domain service processing and the ROUTING element.

3.2.1. DOMAIN

The DOMAIN element configures the basic settings for an independent domain. If the system contains multiple domains, the GATEWAY element of the file will be used to configure settings for inter-domain service processing.

The DOMAIN element consists of the following settings:

- Shared-memory key value
- Number of startup CLH processes, and the maximum number of simultaneous CLH processes allowed
- Maximum number of clients that can simultaneously access the system
- Port number of the domain
- Period of time that a client waits for a response to a service request before being disconnected

The DOMAIN element settings are used by all the nodes of the system. Some of the settings can be redefined in the other four essential elements (for example, NODE). If a setting is defined multiple times in the configuration file, the bottom-most setting takes precedence (for example, NODE > DOMAIN).

The following is the basic outline of the DOMAIN element.

```
*DOMAIN
[DEFAULT : ]
Domain Name     SHMKEY = shared-memory-segment-key,
                [MAXUSER = 1 ~ MAX_INT,]
                [MINCLH = 1 ~ 10,]
                [MAXCLH = 1 ~ 10,]
                [TPORTNO = port-number,]
                [RACPORT = port-number,]
                [BLOCKTIME = timeout-value,]
                [CPC = channel-number]
```

```
[MAXFUNC = max-function-number,]
[LOGOUTSVC = logout-service-name,]
[CLICHKINT = interval-time-value,]
[IDLETIME = idle-time,]
[MAXSACALL = numeric,]
[MAXCACALL = numeric,]
[TXTIME = transaction-timeout-value,]
[NLIVEINQ = alive-check-interval,]
[NCLHCHKTIME = interval-time-value,]
[MAXCONV_NODE = numeric,]
[MAXCONV_SERVER = numeric,]
[SECURITY = ("NO_SECURITY") | "DOMAIN_SEC" | "USER_AUTH" |
            "ACL" | "MANDATORY" | "OTHER_AUTH" ,]
[OWNER = name,]
[IPCPERM = mask,]
[DOMAINID = numeric,]
[CMTRET = literal,]
[MAXNODE = numeric,]
[MAXSVG = numeric,]
[MAXSPR = numeric,]
[MAXSVR = numeric,]
[MAXSVC = numeric,]
[MAXCPC = numeric,]
[MAXTMS = numeric,]
[MAXROUT = numeric,]
[MAXROUTSVG = numeric,]
[MAXRQ = numeric,]
[MAXGW = numeric,]
[MAXCOUSIN = numeric,]
[MAXCOUSINSVG = numeric,]
[MAXBACKUP = numeric,]
[MAXBACKUPSVG = numeric,]
[MAXTOTALSVG = numeric,]
[MAXPROD = numeric,]
[GWCHKINT = interval-time-value,]
[GWCONNECT_TIMEOUT = interval-time-value,]
[TMMLOGLVL = tmm-log-level,]
[CLHLOGLVL = clh-log-level,]
[TMSLOGLVL = tms-log-level,]
[LOGLVL = server-log-level,]
[CRYPT=Y|(N),]
[CRYPT_ALGORITHM=("3DES")|"AES",]
[MAXTHREAD = numeric,]
[TDL = Y|(N),]
[MAXSESSION = numeric,]
[TXPENDINGTIME = pending-transaction-timeout,]
[CLIENT_PROTOCOL = ("IPV4")|",IPV6"|",SDP",]
[SYSTEM_PROTOCOL = ("IPV4")|",IPV6"|",SDP",]
[EXTSVR_PROTOCOL = ("IPV4")|",IPV6"|",SDP",]
[CLIENT_IPV6 = Y|(N),]
[SYSTEM_IPV6 = Y|(N),]
[EXTSVR_IPV6 = Y|(N),]
[CLL_BIND_IP = (Y)|N,]
[FDL_VERSION = 1 | 2,]
[MSGSIZEWARN = numeric,]
[MSGSIZEMAX = numeric,]
```

```
            [CASCPC = numeric]
```

Required Settings

- Domain Name = string
  - Range: a string of up to 63 characters.
  - The DOMAIN element configures settings for the overall Tmax environment.
- SHMKEY = numeric
  - Range: 32768 ~ 262143
  - Value of the pointer that points to a shared memory segment.

    Tmax has the following four main processes.

Process	Description
TMM	Manages other processes as Tmax Manager.
TMS	Manages databases and resources.
CLL	Manages client connections
CLH	Processes requests and responses between clients and server processes.

    SHMKEY configures shared memory key value for managing the information shared by the four processes. Tmax uses the four key values, SHMKEY, SHMKEY+1, SHMKEY+2, and SHMKEY+3, in the order listed.

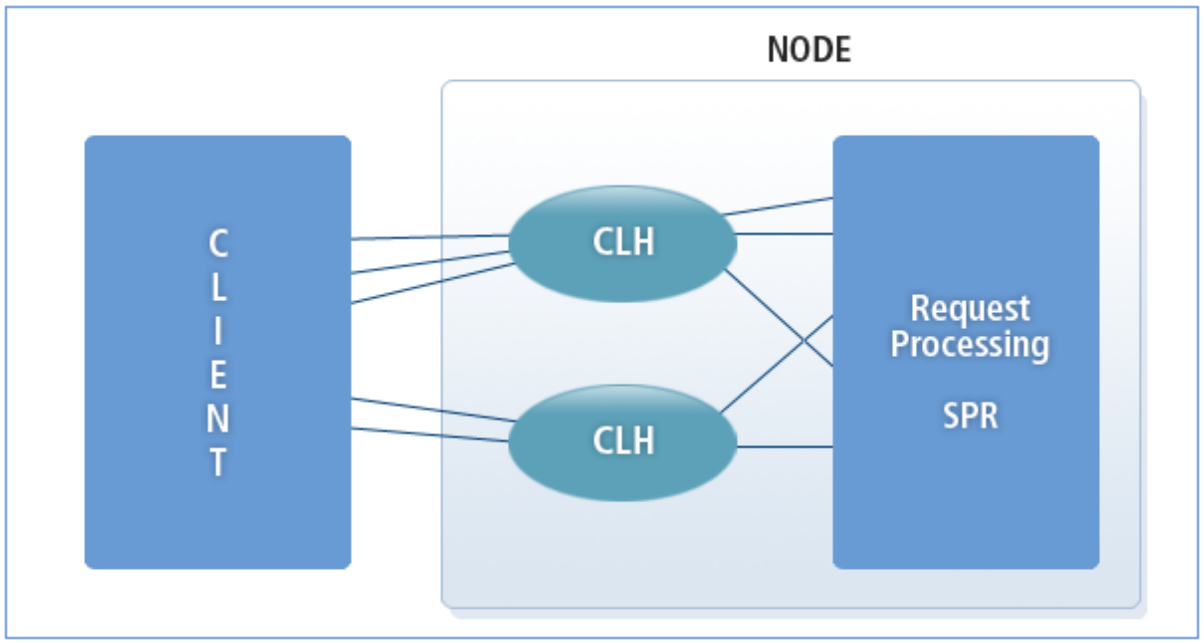
Make sure to verify that the key values are not used for other applications. In order to prevent conflicts, all key values, whether used or not, must be defined.

Optional Settings

- DEFAULT
  - Can be used in all elements of the environment file. For details, refer to "DEFAULT in 3.2.2 NODE".
- MAXUSER = numeric
  - Range: 1 ~ MAX_INT
  - The maximum number of clients that can be simultaneously connected to the domain. The actual number of connections allowed is MAXUSER +1.
- MINCLH = numeric
  - Range: 1 ~ 10

- Default: 1
- The minimum number of startup CLH processes.

  The CLH is used to forward client requests to the appropriate server process, and to return the results back to the client. Every client must be connected to CLH, but the number of clients that can connect to a single CLH is limited. Thus, a large domain with a large volume of clients should run more than one CLH instance by setting a suitable MINCLH value.



MINCLH = 2

- MAXCLH = numeric
  - Range: 1 ~ 10
  - Default: 10
  - The maximum number of active concurrent CLH processes allowed. When Tmax is started, MINCLH number of CLH processes are activated. Additional CLHs can be activated if the number of client requests exceeds the capacity of the active CLHs.

    The total number of active CLH instances cannot exceed the MAXCLH value. Currently, Tmax does not automatically increase the number of MAXCLH processes according to the number of client requests.

- TPORTNO = numeric
  - Default: 8888
  - Port number used by clients to connect with Tmax. Two port numbers are used: TPORTNO and TPORTNO+1.

    The TPORTNO value will be used for connections between clients and servers, while TPORTNO+1 will be used for inter-node communication. In a client/server environment, clients need to know the server address as well as the port number of Tmax.

  - The address (TMAX_HOST_ADDR) and port number (TMAX_HOST_PORT) must be set on the

client side. The port number must be set to TPORTNO. If the client and server are both on the same node, it is recommended to use the domain socket instead of the TCP/IP socket to achieve more efficient connections.

Servers are capable of receiving two types of client connections: TPORTNO is used for TCP/IP socket connections. If the client and server exist on the same node, stream pipe (cllrcad) is used according to the PATHDIR setting.

  - If the client and server are both on the same node, TMAX_HOST_PORT must be set to PATHDIR instead of TPORTNO. For example, set the following in .profile or tmax.env.

```
TMAX_HOST_PORT=/home/tmax/path
```

TPORTNO+1 can also be used for inter-node communication.

  - Nodes in the Tmax system are connected in a peer-to-peer relationship. By using the port number TPORTNO+1, subsequent nodes will be checked in the order they are configured. Port numbers less than 1024 should not be used since they are reserved for system use.
  - Since the port numbers, TPORTNO and TPORTNO+1, are used for communication between a client and Tmax as well as between nodes, check that these values are not used anywhere else. In addition, for multi-node systems, the number of specified port numbers must match the number of CLHs. There must be a separate port number for each CLH.
  - To use the IPv6 protocol, **CLIENT_PROTOCOL** must be set to 'IPV6'.

- RACPORT = numeric
  - Default: 3333
  - Port number used by racd for inter-node communication and centralized node management in a multi-node domain. The port number must be same as the TMAX_RAC_PORT port number. Port numbers less than 1024 should not be used since they are reserved for system use.
  - If the values specified here and in TMAX_RAC_PORT are different, only the TMAX_RAC_PORT value will be used. It is recommended to use a lower value for RACPORT than TPORTNO. Make sure that the port numbers specified in RACPORT are not used anywhere else.
- BLOCKTIME = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0 (seconds)
  - Fractional numbers can be set.
  - If no value is set, by default requests will infinitely wait for a response.
  - Time-out value for service requests. If a service request issued by a client fails to receive a response within the specified time frame, the request will time-out on the client side. However, the server process will continue to process the request.
  - To terminate the service on the server side, set the desired time-out period in **SVCTIME** of the **SERVICE element**. BLOCKTIME is strongly linked to SVCTIME. The shorter time value between

BLOCKTIME and SVCTIME is used as timeout for client tpcall. BLOCKTIME is the tpcall timeout value, and SVCTIME is the server process timeout value. In general, BLOCKTIME value is greater than SVCTIME value.

- CPC = numeric
  - Range: 1 ~ 128
  - Default: 1
  - The number of channels of CLH processes on a node in multi-node systems. In systems with large volumes of inter-node service requests or that process large volumes of distributed transactions, too few number of channels may slow down service processing. For optimal system performance, use an adequate number of parallel communication channels.
- MAXFUNC = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0
  - If not specified, the default value is set, and the function is not used.
  - Size of the function table require for TopEnd system functions in Tmax system when converting a TopEnd system to a Tmax system.
- LOGOUTSVC = service-name
  - Registers services to be processed in normal or abnormal state with the client logout process of Tmax server.
  - Used for normal client logouts when logging in or out of a service. Handles the case where clients are automatically logged out when a preset time (CLICHKINT, IDLETIME) expires.
  - Note that this variable only runs on Tmax clients. It does not run when using a general socket connection. For programs that use the service name, client IDs can be obtained by calling tpgetclid(). Refer to CD of tpsvcinfo structure to determine the cause of the logout.

Value	Description
TPNOAUTH(0)	Normal client logout
TPSYSAUTH(1)	Logout detected by Tmax (for example, network)
TPAPPAUTH(2)	Abnormal logout (for example, data reception failure)

- CLICHKINT = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0
  - If not specified, the default value is set, and the function is not used.
  - The time interval at which clients are monitored. CLICHKINT must be used in conjunction with IDLETIME to minimize resource waste.
  - If CLICHKINT is configured in the DOMAIN element, all clients connected to the domain are monitored. However, if it is configured in the NODE element, only those clients that are connected to the node are monitored.

- IDLETIME = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0
  - If not specified, the default value is set, and the function is not used.
  - Time period that clients can remain idle before being automatically disconnected from the system. If CLICHKINT is set to 5 and IDLETIME is set to 30, clients will be monitored every 5 seconds (CLICHKINT) and if they do not issue any service requests within 30 seconds (IDLETIME), they will be disconnected from Tmax system.
  - Clients will be monitored from the time tpstart() is called to make a connection, or when tpcall()/tpacall() is requested. The value of IDLETIME must be set to a value greater than BLOCKTIME and SVCTIME. If IDLETIME is set to a value less than either BLOCKTIME or SVCTIME value, clients will automatically be disconnected before they receive a response.
- MAXSACALL = numeric
  - Range: 1 ~ 1024
  - Default: 8
  - The number of times tpacall() can be requested by a server process without a tpgetrply() call. Since the server library follows each tpacall() call, the table needs to be created in the initial MAXSACALL that configures the table size.
- MAXCACALL = numeric
  - Range: 1 ~ 1024
  - Default: 16
  - The number of times tpacall() can be requested by a client process without a tpgetrply() call. Since the client library follows each tpacall(), the table needs to be created in the initial MAXCACALL that configures the table size.
- TXTIME = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0 (seconds)
  - Fractional numbers can be set.
  - If not specified, by default requests will infinitely wait for a response.
  - Time-out period for service processing in a transaction (in seconds). When the time-out period is exceeded, the transaction is cancelled.
  - Internal time-out setting that is applied to prevent infinite blocking that can be caused by DB or network failure during XA processing before/after service processing.
- NLIVEINQ = numeric
  - Range: 1 ~ MAX_INT
  - Default: 30 seconds
  - Node monitoring interval (in seconds) for a multi-node Tmax system. To ensure reliable inter-node communications at all times, the nodes of a domain are connected in a peer-to-peer

relationship, and each node regularly monitor the subsequent node in the order they are configured.

  - If the interval period is short, nodes are monitored frequently and node failures are detected quickly, but inter-node traffic may increase to an unacceptable level. On the other hand, if the interval period is long, nodes are monitored less frequently but inter-node traffic is reduced. Select an appropriate value by considering both the network traffic and need for quick error detection.

- NCLHCHKTIME = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0 (seconds)
  - Time-out period (in seconds) with no data flow that triggers CLHs of each node to send a monitoring signal to each other.
  - This feature must be used with care since after a time-out period with no data flow, the firewall will terminate all processes and this will affect CLH processes of Tmax.
- MAXCONV_NODE = numeric
  - Range: 1 ~ 2048
  - Default: 16
  - Maximum number of interactive services that can be simultaneously executed for each CLH. If all clients and server processes can simultaneously call up to 100 interactive services, use a value that is greater than or equal to 100.
  - If the number of services exceeds this value, a TPELIMIT error will occur.
- MAXCONV_SERVER = numeric
  - Range: 1 ~ 256
  - Default: 8
  - Similar to MAXCACALL and MAXSACALL, this variable specifies the maximum number of interactive services that can be simultaneously executed by a client or a server process.
  - If the number of services exceeds this value, a TPELIMIT error will occur.
- SECURITY = "NO_SECURITY" | "DOMAIN_SEC" | "USER_AUTH" | "ACL" | "MANDATORY" | "OTHER_AUTH"
  - Default: "NO_SECURITY"
  - Security setting whose value is one of the following: "NO_SECURITY", "DOMAIN_SEC", "USER_AUTH", and "OTHER_AUTH".

    For more details about Tmax security settings, refer to Security Settings.
- OWNER = user_name
  - Range: a string of up to 63 characters.
  - For details, refer to Security Settings.
- IPCPERM = numeric

    - Range: 0600 ~ 0777
    - Default: 0600 (If not set, the default value for file access control is 600.)
    - Inter-Process Communication Permission Mask (IPCPERM) specifies the control unit of the pipe file located in ${TMAXDIR}/path and in shared memory segment. It also sets the control unit of the system log file (${TMAXDIR}/log/slog).
    - In UNIX, the administrator can assign file access control (to read, record, and execute files) to each individual and group. Read and write accesses can be assigned to individuals, but not to groups.
    - This item is used to grant users the permission to start and terminate server processes. Without IPCPERM, users cannot start or terminate server processes.
    - This item sets the control unit of log files created by TMM, CLL, and CLH. It affects svclog and ulog files and log files created by an engine such as server log files that start according to the setting of ASQCOUNT and POD.
- DOMAINID = numeric
    - Range: 0 ~ 255
    - Default: 0
    - Key value that identifies the domain in a multi-domain environment. Required setting for multi-domain systems.
- CMTRET = Y | N
    - Default: Y
    - Refer to Distributed Transaction Settings.
- MAXNODE = numeric
    - Range: 1 ~ MAX_INT
    - Default: 32
    - Maximum number of nodes allowed per domain.
- MAXSVG = numeric
    - Range: 1 ~ MAX_INT
    - Default: 32
    - Maximum number of server groups allowed per node.
- MAXSPR = numeric
    - Range: 1 ~ MAX_INT
    - Default: 64
    - Maximum number of server processes that can be started on the node. It is calculated by taking the sum of maximum values of each server processes.
- MAXSVR = numeric
    - Range: 1 ~ MAX_INT
    - Default: 64

- Maximum number of servers that can be registered per node.

  For example, if there are three servers (svr1, svr2, and svr3) assigned to the node and the MAX value for each server is 50 (each server can generate up to 50 server processes), the MAXSPR is 150 and MAXSVR is 3.

- MAXSVC = numeric
  - Range: 1 ~ MAX_INT
  - Default: 512
  - Maximum number of services that can be assigned to each domain.
  - This value may be different from the result of executing "cfg -d" in tmadmin because this includes space reserved for dynamically adding services and nodes later.
- MAXCPC = numeric
  - Range: 0 ~ 65535
  - Default: 32
  - Maximum number of CPCs allowed per node.
- MAXTMS = numeric
  - Range: 1 ~ MAX_INT
  - Default: 32
  - Maximum number of active TMSs allowed per node.
- MAXROUT = numeric
  - Range: 1 ~ MAX_INT
  - Default: 16
  - Maximum number of services that can configure the ROUTING setting.

    For example, if 10 out of 30 services can configure the ROUTING setting, then set MAXROUT to 10.

- MAXROUTSVG = numeric
  - Range: 1 ~ MAX_INT
  - Default: 32
  - Maximum total number of groups (server groups or domain gateways) configured in the ROUTING setting of the SERVICE element.

    For instance, assume that the ROUTING setting of Service A is RT1 and that of Service B is RT2. If the number of groups set in RANGES of RT1 is 2, and that of RT2 is 4, the total number of ROUT SVGs is 6. In this case, MAXROUTSVG should be set to at least 6.

    ```
    *SERVICE
    A        ROUTING = RT1
    B        ROUTING = RT2
    ```

```
*ROUTING
RT1     RANGES = "'a':svg1,*:DGW1"
RT2     RANGES = "'a':svg1,'b':svg2,'c':DGW2,*:DGW1"
```

- MAXRQ = numeric
  - Range: 1 ~ MAX_INT
  - Default: 2
  - Maximum number of active RQs per node.
- MAXGW = numeric
  - Range: 1 ~ MAX_INT
  - Default: 2
  - Maximum number of active gateways per node.
- MAXCOUSIN = numeric
  - Range: 1 ~ MAX_INT
  - Default: 16
  - Maximum number of server groups that can configure the COUSIN setting. For example, if 10 out of 30 server groups can configure the COUSIN setting, set MAXCOUSIN to 10.
- MAXCOUSINSVG = numeric
  - Range: 1 ~ MAX_INT
  - Default: 32
  - Maximum total number of server groups that can configure the COUSIN setting of SVRGROUP element. For example, if the MAXCOUSIN value is 10 and COUSIN setting can be configured in two server groups, set MAXCOUSINSVG to 20.
- MAXBACKUP = numeric
  - Range: 1 ~ MAX_INT
  - Default: 16
  - Maximum number of server groups that can configure the BACKUP setting. For example, if 10 out of 30 server groups can configure the BACKUP setting, set MAXBACKUP to 10.
- MAXBACKUPSVG = numeric
  - Range: 1 ~ MAX_INT
  - Default: 32
  - Maximum total number of the server groups that can configure the BACKUP setting of SVRGROUP element. For example, if the MAXBACKUP value is 10 and each BACKUP setting can include two server groups, set MAXBACKUPSVG to 20.
- MAXTOTALSVG = numeric
  - Range: 1 ~ MAX_INT
  - Default: 64

    - Maximum number of server groups that can be configured.
- MAXPROD = numeric
    - Range: 1 ~ MAX_INT
    - Default: 0
    - If not set, the default value is used, and this function is not used.
    - This item indicates the maximum value of product name that can be specified as the item that is used to convert a system, which was made for TopEnd system, into the Tmax system. (for TopEnd use)
- GWCHKINT = numeric
    - Range: 1 ~ MAX_INT (seconds)
    - If a gateway to a backup node is available within a multi-domain system, service requests will be transferred to the backup node if a fault occurs in the main remote node. Service requests will maintain their connection to the backup node even after the main node has been recovered. The gateway to the backup node must be shut down before the service request can re-connect to the main node. However, if the GWCHKINT variable is properly configured, service requests will automatically reconnect to the main node after recovery.
    - Considerations
        - If no value is specified in GWCHKINT, service requests will not attempt to reconnect to the main node even after the main node has recovered.
        - After connecting to the recovered main node, the connection to the backup gateway must be disconnected manually. If not, connections to both the main node and the backup node are maintained. However, transactions are sent only to the main node.
- GWCONNECT_TIMEOUT = numeric
    - Range: 1 ~ MAX_INT (seconds)
    - The domain gateway tries to connect to the remote gateway at system startup.

      If the GWCHKINT item is not specified in the DOMAIN element of the configuration file and the machine where the remote gateway is set is currently not running, then it takes a long time to show a connection failure message to the user who requested the service.

      If the domain gateway does not attempt to reconnect when there are no requests, it will attempt to reconnect upon a service request, which will take a long time if the remote gateway is not ready. Such issue can be prevented by specifying the connection timeout.
    - If the GWCONNECT_TIMEOUT is not specified, default procedures will be used.
- TMMLOGLVL = string
    - Default: DETAIL
    - Specifies one of the following logging levels for the TMM.

Level	Description
COMPACT	E + F

Level	Description
BASIC	E + F + W
DETAIL	E + F + W + I
DEBUG1 ~ DEBUG4	Low internal debug log level (DEBUG4 gives the most detail). DEBUG1 ~ DEBUG4 are only available for debugging binary and library. (clh.dbg, tmm.dbg, libsvrd, libtmsd, libtmsthrd)

- CLHLOGLVL = string
  - Range: Same as that of the TMM
  - Specifies the logging level for the CLH.
- TMSLOGLVL = string
  - Range: Same as that of the TMM
  - Specifies the logging level for the TMS.
- LOGLVL = string
  - Specifies the logging level of the servers.
- CRYPT = Y | N
  - Default: N
  - Tmax 4 and later is capable of encrypting data transmitted between clients and the Tmax system to ensure extra protection. For this feature to be used, it must be configured for both the client and Tmax system.
  - To set this encryption function for Tmax system, set CRYPT to Y. To not use the encryption function, set CRYPT to N, default value. For information about encryption for a client, refer to the TMAX_SEMANTICS variable in Tmax Client Environment Variables.
  - Note
    - Encryption key information is exchanged when the client connects with the system and the encryption/decryption is performed as each service is processed. This, of course, will increase performance overhead.
    - To enable encryption function, /dev/random must first be installed. This '/dev/random' provided by the kernel will generate random keys for exchange when the client connects to the system. To use the encryption function, use the 'ls' command to check whether /dev/random has been installed. Refer to the system manual if it has not yet been installed.
    - Encryption is not supported for inter-node data exchange.
    - Encryption will not be supported in cases where the client accesses the system though $PATHDIR.
    - A TPECLOSE error will occur if an encryption-enabled client attempts to access an non-encryption-enabled server, and the client will be blocked.
- CRYPT_ALGORITHM = "3DES" | "AES"
  - Default: 3DES

- Encryption method. Either 3DES and AES.
- Confirm that the CRYPT_ALGORITHM value in the server configuration file and the TMAX_CRYPT_ALGORITHM value in the client configuration file are the same.
- CRYPT must be set to Y to use this item.

- MAXTHREAD = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0
  - If not specified, the default value is set, and the function is not used.
  - In Tmax 4 and later, TMS can be configured as a multi-thread transaction manager to improve transaction processing efficiency. To enable this feature, set this value to the maximum number of threads for TMS. Additionally, the TMSTYPE, TMSTHREAD, and TMSOPT variables of the SVRGROUP element must also be configured in conjunction with this variable.

    For a description of each variable, refer to TMSTYPE, TMSTHREAD, and TMSOPT in "3.2.3 SVRGROUP".

  - Note
    - To use this feature, the DBMS must also support multi-threading. (Currently, only ORACLE provides the support)
    - When using ORACLE, "Thread=true" must be added to the OPENINFO variable of the SVRGROUP element.

      ```
      OPENINFO="Oracle_XA+Acc=P/scott/tiger+SesTm=60+Thread=true"
      ```

    - To use this feature, 'libtmsthr' or 'libpthread' library, instead of "libtm," must be linked and compiled.
- TDL = Y | N
  - Default: N
  - Option to use (Y|N) the TDL (Tmax Dynamic Library) Implicit Version Protection will be used.

    For details about TDL, refer to *Tmax Programming Guide (Dynamic Library)*.

- MAXSESSION = numeric
  - Range: 1 ~ 65535
  - Default: 1024
  - Maximum number of sessions that HMS can create per node. For details, refer to HMS Settings.
- TXPENDINGTIME = numeric
  - Range: 1 ~ MAX_INT

- Default: 0
- Time-out value for pending transactions. If no response is issued from a pending transaction for this time period, the transaction will be rolled back. If no value is set, no checks are performed.

- CLIENT_PROTOCOL = "IPV4" | "IPV6" | "SDP"
  - Default: "IPV4"
  - Protocol used when creating a port for client connections. If multiple protocols can be used, they are set by using a comma (,) as a delimiter.

Value	Description
IPV4	Uses the IPv4 protocol to wait for client connection requests, and the client can connect to the system from IPv4 environment. Can be set for each node.
IPV6	Uses the IPv6 protocol to wait for client connection requests, and the client can connect to the system from both IPv4 and IPv6 environments.
SDP	Uses InfiniBand Socket Direct Protocol (SDP). Can be used along with IPV4 or IPV6.

- SYSTEM_PROTOCOL = "IPV4" | "IPV6" | "SDP"
  - Default: "IPV4"
  - Protocol used when creating a port for inter-node communication in a multi-node environment. If multiple protocols can be used, they are set by using a comma (,) as a delimiter.

Value	Description
IPV4	Uses the IPv4 protocol to wait for connection requests. Can be set for each node.
IPV6	Uses the IPv6 protocol to wait for connection requests when creating a port for another node.
SDP	Uses InfiniBand Socket Direct Protocol (SDP). Can be used along with IPV4 or IPV6.

- EXTSVR_PROTOCOL = "IPV4" | "IPV6" | "SDP"
  - Default: "IPV4"
  - Protocol used when creating a port for connection to the Extern Server from Tmax system. If multiple protocols can be used, they are set by using a comma (,) as a delimiter.

Value	Description
IPV4	Uses the IPv4 protocol to wait for connection requests. Can be set for each node.

Value	Description
IPV6	Uses the IPv6 protocol to wait for connection requests.
SDP	Uses InfiniBand Socket Direct Protocol (SDP). Can be used along with IPV4 or IPV6.

- CLIENT_IPV6 = Y | N
  - Default: N
  - Option to use the IPv6 protocol when creating a port for client connections.

Value	Description
Y	Use IPv6 protocol to wait for client connection requests, and the client can connect to the system from either IPv4 or IPv6 environment.
N	Use IPv4 protocol to wait for connection requests. A client can connect to the system only through IPv4. Setting can be configured per node.

It is recommended to use CLIENT_PROTOCOL instead of this item.

- SYSTEM_IPV6 = Y | N
  - Default: N
  - Option to use the IPv6 protocol when creating a port for inter-node communication in a multi-node environment.

Value	Description
Y	Use IPv6 protocol. A client can connect to the Tmax system in either IPv4 or IPv6 environment.
N	Use IPv4 protocol to wait for connection requests. Setting can be configured per node.

It is recommended to use SYSTEM_PROTOCOL instead of this item.

- EXTSVR_IPV6 = Y | N
  - Default: N
  - Option to use the IPv6 protocol when creating a port for connection to the Extern Server from Tmax system.

Value	Description
Y	Use IPv6 protocol.

Value	Description
N	Use IPv4 protocol. Setting can be configured per node.

It is recommended to use EXTSVR_PROTOCOL instead of this item.

- CLL_BIND_IP = Y | N
  - Default: Y
  - Option to use the interface of IP address configured in IP setting of the NODE element when creating a port for client connection.

Value	Description
N	If set to N, or no IP is specified, connections from all interfaces are allowed.

- FDL_VERSION = numeric
  - Default: 1
  - Range: 1 ~ 2
  - If set to 2, it uses the version with enhanced search performance and is not compatible with existing Tmax system.

    This setting affects server applications and the CLH scheduling that follows the ROUTING configuration section. When using fb* API, it returns the FBESETVER error if FDL_VERSION setting does not match with its environment.
- MSGSIZEWARN = numeric
  - Default: 1073741824
  - Range: 1024 ~ 1073741824
  - Maximum message size. If the size of a message from a client or server exceeds the value, a warning message is recorded in slog, but the message is processed.
- MSGSIZEMAX = numeric
  - Default: 1073741824
  - Range: 1024 ~ 1073741824
  - Maximum message size. If the size of a message from a client or server exceeds the value, a warning message is recorded in slog, and the connection is ended.

    If ROC is set to N, the server with the ended connection cannot reconnect to CLH. Therefore, set ROC to Y.
- CASCPC = numeric
  - Default: 1

- Range: 1 ~ 65535
- Number of CPCs between CAS processes and the node's CLH processes.
- Currently, **this item must be set to 1**.

Example

The following is an example of the DOMAIN element.

```
*DOMAIN
res1      SHMKEY = 77990, MAXUSER = 300,
          MINCLH = 2, MAXCLH = 3,
          TPORTNO = 8889, RACPORT = 3334,
          BLOCKTIME = 60, CLICHKINT = 5,
          IDLETIME = 30, LOGOUTSVC = logout,
          CLHLOGLVL = DEBUG2
```

3.2.2. NODE

The DOMAIN element defines variables used by the entire Tmax system, while the NODE element defines variables for each node.

The NODE element includes the following settings.

- Tmax home directory
- Directories that contain server application program executable files (server files)
- Directories for inter-process communication
- Directories where system messages are stored
- Directories where user messages are stored
- Directories where transaction information is stored
- Variables required to execute server application programs

Some settings in the NODE element can be redefined in the DOMAIN element. For details about these settings, refer to Re-definable Items.

The startup processes in Tmax system include TMM, CLL, CLH, TMS for database related systems, GATEWAY for multi-node domains, and server processes that actually process service requests. To start the processes, the system needs to find the directories with the executables. The NODE element should be configured with the directory paths of the executables. It also needs to configure directory paths needed for communication of Tmax processes and for information about various errors and warning messages.

Basic configuration of the NODE element is as follows:

```
*NODE
[DEFAULT: ]
```

```
Node Name      TMAXDIR = tmax-home-path ,
               APPDIR = application-path,
               [HOSTNAME = HOST_NAME,]
               [TMAXHOME = tmax-home-path,]
               [NODETYPE = SHM_RACD|(SHM_USER),]
               [PATHDIR = stream-pipe-path,]
               [SLOGDIR = system-log-path,]
               [TLOGDIR = transaction-log-path,]
               [ULOGDIR = user-log-path,]
               [DOMAINNAME = domain-name,]
               [CLHQTIMEOUT = 1 - MAX_INT,]
               [ENVFILE = environment-file-name,]
               [SHMKEY = shared-memory-segment-key,]
               [MAXUSER = 1 ~ ,]
               [MINCLH = 1 ~ 10,]
               [MAXCLH = 1 ~ 10,]
               [TPORTNO = port-number,]
               [TPORTNO2 = port-number,]
               [TPORTNO3 = port-number,]
               [TPORTNO4 = port-number,]
               [TPORTNO5 = port-number,]
               [RACPORT = port-number,]
               [IPCPERM = mask,]
               [IP = IP address,]
               [CRYPT_ALGORITHM=("3DES")|"AES",]
               [CLICHKINT = interval-time-value,]
               [IDLETIME = idle-time,]
               [TMMOPT = TMM-log-path,]
               [TLMOPT = TLM-transaction-log-file-size,]
               [CLHOPT = CLH-log=path,]
               [REALSVR = server-name,]
               [RSCPC = 1 ~ 128,]
               [MAXSVG = 1 ~ MAX_INT,]
               [MAXSPR = 1 ~ MAX_INT,]
               [MAXSVR = 1 ~ MAX_INT,]
               [MAXTMS = 1 ~ MAX_INT,]
               [MAXCPC = 0 ~ 128,]
               [MAXGWSVR = 1 ~ MAX_INT,]
               [MAXRQSVR = 1 ~ MAX_INT,]
               [MAXGWCPC = 1 ~ MAX_INT,]
               [AUTOBACKUP = (Y)|N,]
               [LOGOUTSVC = logout-service-name,]
               [TMAXPORT = port num1, ..., port number5,]
               [COMPRESSPORT = port num1, ..., port number5,]
               [COMPRESSSIZE = compress_size,]
               [RESTART = (Y)|N,]
               [MAXRSTART = numeric,]
               [GPERIOD = numeric,]
               [TMMLOGLVL = tmm-log-level,]
               [CLHLOGLVL = clh-log-level,]
               [TMSLOGLVL = tms-log-level,]
               [LOGLVL = server-log-level,]
               [EXTPORT = port number,]
```

```
[EXTCLHPORT = clh port number,]
[SMSUPPORT = Y | (N),]
[SMTBLSIZE = num,]
[CLLBLOCK = Y|(N),]
[CLLUNBLKPORT = Portno1, Portno2, ...,]
[CLLBLOCKTIME = num]
[CLLCONNLB=(RR)|LC,]
[CRYPTPORT=literal,]
[TRB = nodename,]
[MAXSESSION = 1 ~ MAX_INT,]
[SQKEY = 32768~262143 ,]
[SQSIZE = 4 ~ 4000000,]
[SQMAX = 2~MAX_INT-1,]
[SQKEYMAX = 1 ~ MAX_INT,]
[SQTIMEOUT =1 ~ MAX_INT,]
[CLIENT_PROTOCOL = ("IPV4")|",IPV6"|",SDP",]
[SYSTEM_PROTOCOL = ("IPV4")|",IPV6"|",SDP",]
[EXTSVR_PROTOCOL = ("IPV4")|",IPV6"|",SDP",]
[CLIENT_IPV6 = Y|(N),]
[SYSTEM_IPV6 = Y|(N),]
[EXTSVR_IPV6 = Y|(N),]
[CLL_BIND_IP = (Y)|N,]
[SVCLOG_FORMAT = svclog-format-string,]
[CASTHREAD = 1 ~ 65535,]
[CASOPT = literal,]
[SECURITY_LIB = literal,]
[CRYPT_LIB = literal,]
[TGOPT = literal]
```

Required Settings

- Node Name = string
  - Size: Up to 63 characters
  - Name of the physical machine. This value must match the name returned by the UNIX command, uname -n.

    Node name must be configured in the "/etc/hosts" file. Since a domain consists of one or more nodes, there must be at least one node configured in the NODE element.
- TMAXDIR = literal
  - Size: Up to 255 characters
  - Full path of the directory where Tmax is installed. The path must be the same as the value of TMAXHOME. If TMAXHOME is not specified, all Tmax related tasks are performed from the TMAXDIR directory.
- APPDIR = literal
  - Size: Up to 255 characters
  - Full directory path for server application program executable files.
  - Tmax system processes as well as application processes for client requests start and terminate with the system.

- **APPDIR** is also used as a working directory of the server. Core files created in a server program is stored in the APPDIR.

Optional Settings

- Default: item = value, ...
  - This variable can be defined in all elements of the configuration file (NODE, DOMAIN, ...).

    In the NODE element, it is useful for configuring items with the same value for all nodes in a multi-node configuration. When a default value is set using this label, it takes effect until the end of the element is reached or is overridden by another default setting.

    If the default settings defined for "DEFAULT:" are not reconfigured in a NODE element, they are applied to all the nodes.

  - If a default setting is reconfigured in a NODE element, the new value is applied to that node.
- HOSTNAME = literal
  - Size: Up to 255 characters
  - This item is used to assign an actual hostname when the hostname is different from the configured node name. By setting the HOSTNAME, multiple logical nodes can be configured on a single machine. Multiple node names can be used per hostname.
  - TMAXHOME, NODETYPE, RACPORT, TPORTNO, and SHMKEY items must be configured in the NODE element. For details, refer to the following "Example" of the NODE element.
- TMAXHOME = literal
  - Size: Up to 255 characters
  - This item can be used to separately configure the Tmax installation and working directories. TMAXHOME is the absolute path of the directory where Tmax is installed, and must be same as the value of the TMAXHOME environment variable. If TMAXHOME is configured, TMAXDIR is the working directory path. Otherwise, TMAXDIR is the path for both directories.

    The following is description of Tmax installation and working directories.

Item	Description
TMAXHOME	Installation directory that contains the following sub directories: bin, lib, usrinc, tuxinc, topinc, cblinc, and license.
TMAXDIR	Working directory that contains the following sub directories: config, path, log, and svct.

    If HOSTNAME is also configured, each TMAXDIR becomes a single logical node, and TMAXHOME is Tmax installation directory shared by all logical nodes. Thus, TMAXDIR, SHMKEY, and TPORTNO must be separately configured for each logical node of the physical node.

- NODETYPE = SHM_RACD | SHM_USER

- Default: SHM_RACD
- Method for managing individual nodes when there are multiple logical nodes on a single machine. Each local node of a machine must have a different RACPORT value. Where there are multiple local nodes, each local node must be set to a different RACPORT value.

  Note that this setting is only used for managing nodes that are on the same host node as the logical node that executes tmboot/tmdown/tmadmin/cfl. Nodes on another host node will be managed by using RACPORT setting regardless of the NODETYPE setting.

- This setting is used to determine whether to use RACD when executing tmboot/tmdown/tmadmin/cfl on node B, which is on the same host node as the logical node (node A) that has already started executing tmboot/tmdown/tmadmin/cfl.
- Management options are:

Value	Description
SHM_RACD	All of the logical nodes are managed using the RACPORT setting.
SHM_USER	Each logical node is managed separately without using the RACPORT setting.

- PATHDIR = literal
  - Size: Up to 255 characters
  - Default: ${TMAXDIR}/path
  - The directory path of the named stream pipe used for communication between Tmax processes. Communication methods include using a pipe (unnamed/named pipe), message queue, or shared memory.

    Tmax uses a shared memory and stream pipe for inter-process communication (refer to the description of SHMKEY in "3.2.1 DOMAIN"). A pathname is required to use a stream pipe.

  - PATHDIR is the directory path where pathnames, which are required for communication between Tmax processes through stream pipes, are created. Named pipes are created in this directory. If PATHDIR is not defined, pipes will be created in a new directory inside the directory configured in TMAXDIR.
  - If PATHDIR is set to another(not $TMAXDIR/path) directory, the system may not terminate properly when Tmax system shuts down. This is because the system uses the stream pipe(named pipe) in $TMAXDIR/path directory when the system terminates. In order to resolve this problem, the user should configure the environment variable, TMAX_PATHDIR, with the PATHDIR value in the configuration file. Now if the system terminates, it will shut down properly by referencing the environment variable, TMAX_PATHDIR.

    ```
    export TMAX_PATHDIR = /home/tmax/path_new
    ```

  - When Tmax system starts, previous configuration file in the config directory is copied to PATHDIR of the configuration file. This will prevent the system from not terminating properly

when the configuration file is modified after Tmax system starts. If you set the previous PATHDIR (the one that was referenced at startup) in the TMAX_PATHDIR, the previous configuration file will be created under this directory. This configuration file will not be modified even if the configuration file is modified after system startup.

- SLOGDIR = literal
  - Size: Up to 255 characters
  - Default: ${TMAXDIR}/log/slog
  - The absolute path of directory where system messages are logged. The logged messages include those generated by TMM, TMS, CLL, and CLH, as well as those generated internally by the system. The slog.date file for each date is created in this directory, and messages are saved to a file according to the date they are created.
  - If no directory is specified in SLOGDIR, a log/slog folder will be created inside the directory configured in TMAXDIR.

The directory path must exist in the system.

- TLOGDIR = literal
  - Size: Up to 255 characters
  - Default: ${TMAXDIR}/log/tlog
  - The absolute path where TXLOG file is created. The TXLOG file is used to log information about XA transactions. It is created as an unreadable binary file that is used to manage transaction information of Tmax engine. The file is created with a file system and TXLOG file logging is supported depending on the raw file.
  - If TLOGDIR is not defined, transaction information will be logged in the log/tlog folder inside the TMAXDIR directory.

The directory path must exist in the system.

  - During system boot, if a log file from the previous boot exists, the file is backed up as '.old', and a new transaction log file is created. The file cannot be larger than 16 MB, and the size can be modified by using the -l option of TLMOPT.
  - The size for a single xid in txlog is about 32 bytes. (txlog file size does not expand infinitely.)

- ULOGDIR = literal
  - Size: Up to 255 characters
  - Default: ${TMAXDIR}/log/ulog
  - The absolute path where user messages are logged.
  - Users, or developers, can use the userlog() function to log debugging or various error and warning messages.
  - The messages generated by userlog() function are saved in the ULOGDIR directory. The ulog

_date file for each date is created in this directory, and messages are saved to a file according to the date they are created. For more details about userlog(), refer to *Tmax Application Development Guide*.

- If no directory is set to ULOGDIR, messages will be logged in the log/ulog directory inside the TMAXDIR directory.

The directory path must exist in the system.

- DOMAINNAME = string
  - Size: Up to 63 characters
  - Since the system can consist of multiple domains, the domain that the node belongs to must be specified. The domain name must be configured in the DOMAIN element of the configuration file.
- CLHQTIMEOUT = numeric
  - Range: 1 ~ MAX_INT
  - Dequeues expired messages. Too many messages can be queued because of host and network issues. In this case, this item can be used to notify the queue status for a client to handle the status.
- ENVFILE = literal
  - Size: Up to 255 characters
  - Configuration file that contains UNIX environment variables used to configure an appropriate environment for executing server application programs. The server application programs run in the environment configured with the settings in the ENVFILE.
- TPORTNO[2-5] = numeric
  - Tmax system port number that clients can use to connect to servers. The port number is used as an alternative to TPORTNO.
  - A total of five ports can be used for connection if all ports from TPORTNO2 to TPORTNO5 are configured. Port numbers less than 1024 should not be used since they are reserved for system use.
  - TPORTNO and the next consecutive port number (TPORTNO+1) are reserved for communication between clients and Tmax or for inter-node communication. The port numbers must be unique, and the port number in TPORTNO should be different from those already configured.
- TMMOPT = literal
  - Size: Up to 255 characters
  - Command options that can be sent to the TMM process when it starts. Options before '--' are used by Tmax system, and those that come after '--' can be used by general users.
  - The following are the most commonly used options.

Option	Description
-e [file_name]	Writes standard errors that occur during TMM operation to the specified file.
-o [file_name]	Writes standard outputs that occur during TMM operation to the specified file.
- i | w | e | f	Logging level of a callback function called when SLOG occurs if the server process type (SVRTYPE) is EVT_SVR. • i: fatal, error, warn, info (The callback function is called for all events with the info or lower level.) • w: fatal, error, warn • e: fatal, error (Default) • f: fatal (The callback function is called only when a fatal error is recorded in slog.)
-t [second(s)]	Access timeout value of a server process started by ASQCOUNT. (Default: 10, unit: seconds) For example, TMMOPT = "-t 20"
-B [backlog value]	Backlog value of the listener port that handles connections with the server processes. A value between 1 and SOMAXCONN can be used. (Default: 1023) If the value is too small, multiple servers may fail to concurrently connect to TMM.
-A [Number of executions]	Number of connection requests from server processes that can be concurrently processed. This value can be dynamically configured during execution by using the setopt command in tmadmin. (Default: 100)

Option	Description
-F [Maximum number of concurrent executions]	Maximum number of processes that can be handled concurrently when creating new processes to execute additional processes or restart processes in TMM. This option can be used to make adjustments when the system performance or load conditions affect starting new processes. Use a value between 0 and the specified backlog value. The default value is 0.75 * backlog value. If set to 0, any number of new processes can be started immediately. This value can be dynamically configured during execution by using the setopt command in tmadmin. If set to 0, all waiting processes will be started. Calculate the number of running processes as follows: When a process is created upon execution request, add 1 to the number of currently running processes. Subtract 1 when the process connects with TMM and becomes registered. If exec() call fails in the running process or before it is registered, subtract 1 after checking the boottime of the -t option. If a certain process cannot start due to a missing execution file or dependent library error, the number of running processes remains the same during boottime. If there is a request to start a process when the number of running processes has reached -F option value, the request waits until the active process is registered and the count falls below the maximum.
-r [Interval]	Refer to the -k option. (Unit: seconds)
-k [Number of concurrent processes]	Number of concurrent processes that can try to establish a connection. When CLH restarts, TMM notifies to all server processes to connect to the CLH. If too many processes try to connect to the CLH at the same time, the servers can be in BLOCK status. To prevent this, use the -r and -k options, so that a limited number of processes can try to reconnect to the CLH. The number of processes is checked at the interval specified in the -r option. The -k option must be used along with the -r option. Extern servers are notified of an immediate connection request. They are not affected by these options. To allow a specific server to connect to the CLH, use the **nrc** command in tmadmin.

Option	Description
-g	Creates a separate thread for file output. If not set, file output is processed by the main thread. TMM records slog, trace log, and svclog to files. If TMAX_TRACE is used, many servers exist, service time is short, and there is heavy load, performance can decrease because TMM needs to write files. This option is prevent this. This option is related to the -m option.
-m	Number of requests to keep when file output performance is low. The default value is 10000. One request is about 8192 bytes. This option has a meaning only when used along with the -g option. If TMMOPT is set to "-g -m 10000", servers request to record log in slog or ulog, or request TPS exceeds possible file output, the specified number of requests are queued. Requests that arrive after the number of requests in the queue reaches the specified number are discarded. If this value is too big, TMM memory size is also too big, which causes an error when TMM manages ASQCOUNT or restarts because of abnormal termination.

- TLMOPT = literal
  - Size: Up to 255 characters
  - Command options that can be sent to the TLM process when it starts.
  - The following is the most commonly used option.

Option	Description
-l size	Log file size. (Default value: 16, maximum value: 2048, unit: MB)

- CLHOPT = literal
  - Command options that can be sent to the CLH process when it starts. Detailed options are same as that of TMMOPT.

Option	Description
-e file_name	Writes standard errors that occur during CLH operation to the specified file.
-o file_name	Writes standard outputs that occur during CLH operation to the specified file.

Option	Description
-h i | w | e | f	Logging level of a callback function called when SLOG occurs if the server process type (SVRTYPE) is EVT_SVR. • i: fatal, error, warn, info (The callback function is called for all events with the info or lower level.) • w: fatal, error, warn • e: fatal, error (Default) • f: fatal (The callback function is called only when a fatal error is recorded in slog.)
-L service name	If a service or a client that called the service terminates or reboots, the response message from service called by CLH is discarded, and a user designated service (loss service) can be called. (tpacacall with TPNOREADY | TPNOTRAN) Name of the loss service that is called when a response message is discarded. The service receives additional information through cltid of TPSVCINFO. • cltid.clientdata[1]: tperrno of discarded response • cltid.clientdata[2]: tpurcode of discarded response • cltid.clientdata[3]: Service index of discarded response (used as a parameter of the tpgetsvcname() function.)
-r count	Used to specify the maximum number of times read is attempted while CLH performs non-blocking I/O with a server or client. (Default: 100) It is recommended that the value is set to a number greater than or equal to the default value and less than 1000.
-w count	Used to specify the maximum number of times write is attempted while CLH performs non-blocking I/O with a server or client. (Default: 100) It is recommended that the value is set to a number greater than or equal to the default value and less than 1000.
-x 1	Writes TPENOENT errors to SLOG. If not specified, the errors are not written to SLOG and can be checked only from a client.
-s seconds	Maximum time for a client to wait for a TPSTART message after a connection is established to the client. (Default: 60 seconds)
-f	Immediately reconnects to CLH when a CLH channel is ended if NCLHCHKTIME is set.

Option	Description
-c	Makes CLH notify cousin server status to other nodes. This prevents an issue occurred because the statuses from TMM and CLH are different because of a gap between the time of TMM, CLH, and the server, but this causes heavy load. If not specified, TMM notifies the server status.
-k	Performs scheduling when a gateway and a service set as COUSIN restart in a remote domain. This option is used when domains are connected through tmaxgwnt and there is a service set as COUSIN in a multi-domain environment that uses both Tmax 5 and later and Tmax 4 SP3 Fix#9 and previous.

    - The following conditions are required to transmit a message to a service designated by CLH.
        1. The message must be a response to tpcall() or tpacall().
        2. Data must be included whether it is a normal response or an error response.
        3. The message may not be transmitted if CLH is terminated abnormally.
- REALSVR = literal
    - The name of the RDP server process, if using one.
    - There must be one real server per node. Clients can access all services of a node through the RDP server processes. The RSCPC value must be set to the number of channels that are used to receive results from other server processes. The number of CLH and RDP server processes must remain constant and the MIN and MAX values must be the same. In general, the ratio of the number of processes of RDP to CLH should be 2 to 1. The number of RDP processes must always be greater than that of CLH.
- RSCPC = numeric
    - Range: 1 ~ 128
    - Number of channels used by RDP processes to communicate with other servers.
- MAXSVG = numeric
    - Range: 1 ~ 128
    - Default: 32
    - Maximum number of server groups per node.
- MAXTMS = numeric
    - Range: 1 ~ 128
    - Default: 32
    - Maximum number of TMSs that can be invoked on the node.
- MAXGWSVR = numeric
    - Range: 1 ~ 128

- Default: 2
- While MAXGW is the maximum number of gateways that can be used on the node, MAXGWSVR is the maximum number of gateway server processes that can be invoked on the node. The default value is 2, which means that there are two gateways and each gateway can have one gateway server.

- MAXRQSVR = numeric
  - Range: 1 ~ 128
  - Default: 2
  - Maximum number of RQ server processes that can be invoked on the node. The default value is 2, which means that there are two RQs and each RQ can have one RQ server.
- MAXGWCPC = numeric
  - Range: 1 ~ 128
  - Default: 8
  - Maximum number of connections between a gateway and a CLH.
  - CPC with a server is basically composed of a pair of channels (input/output). If CPC and the number of gateways are both set to 2, MAXGWCPC must be set to at least eight connections. The sum of MAXGWCPC must also be less than the value of MAXCPC.

    To increase the MAXGWCPC value, the MAXCPC value must also be increased.
- AUTOBACKUP = Y | N
  - Default: Y
  - Option to automatically activate the backup server in the event of server failure.
- TMAXPORT = "port num1, ... , port num5"
  - Alternate port number for TPORTNO and TPORTNO[2-5]. It is recommended to use TMAXPORT instead of TPORTNO[2-5] when multiple port numbers are used.
- COMPRESSPORT = "port num1, ... , port num5"
  - Port numbers used by clients to communicate with Tmax system using the data compression function. If no port numbers are set, the data compression function is not used during communication. Note that the port numbers must be those that are managed by Tmax.

    If TMAXPORT of the NODE element is specified, use one of the port numbers of TMAXPORT setting. If TMAXPORT is not specified, use the TPORTNO value of the DOMAIN element. Only the clients who are connected to the specified port can use the data compression function.
  - This feature is not recommended for use in a general network environment since overhead for data compression is greater than network transmission delay. Data compression is usually used when sending a relatively large volume of data under poor network conditions.

    For example, data compression can be used when a client is exchanging tens of KB of data via a modem.
  - The following tables display the amount of time required for transmitting 1.5 MB of data

using a 100 Mbps Local LAN (first two tables) and a 56 KB modem (third table). The results show that it is best to use the compression function when transmitting a large volume of text data via a modem.

- Local network client

	Compressed	Uncompressed
mp3 download	10	3
	10	3
	10	4
	10	3
Text data download	8	1
	8	0(0.85)
	8	0(0.85)
	8	0(0.85)

- Modem client

	Compressed	Uncompressed
mp3 download	352	350
	309	307
	328	331
	349	351
Text data download	80	137
	89	153
	77	154
	77	154

<Transmitted data size>

	Actual	Compressed
mp3	1,459,095 bytes	1,442,869 bytes
txt	1,445,184 bytes	313,057 bytes

1. To use compression, the link libz.a(so) file must be linked to client programs during compilation.
2. The current version of Tmax does not support data compression when transmitting data between real servers, between nodes, or between domains.

- COMPRESSSIZE = numeric
  - Minimum size (unit: bytes) of data that can be compressed when using the compression function.
  - Must be used in conjunction with COMPRESSPORT.
  - In the following example, Tmax system uses data compression when communicating with the client through port 9999. In the example, data compression will only be used if the data size is larger than 10000 bytes. If the data size is smaller than this value or the client is connected to port 8888, data will not be compressed.

```
TMAXPORT = "8888, 9999", CompressPort = "9999" CompressSize = 10000
```

- RESTART = Y | N
  - Default: Y
  - Option to reload CLH, CAS, CLL, and TLM upon abnormal termination.
- MAXRSTART, GPERIOD
  - Settings for fault tolerance features. For details, refer to Fault Tolerance Settings.
- EXTPORT = numeric
  - Tmax supports EXTERN server functions to use functions of Tmax configuration file from an external process, a non-Tmax process.

    The server process TYPE must be set to EXTSVR. Two additional variables, EXTCLHPORT and EXTPORT, must be configured in the NODE element of the configuration file. Configure the listen port number of TMM in EXTPOT.
- EXTCLHPORT = literal
  - Listen port number of CLH. If not specified by the system administrator, the system will assign a value.
  - The following is an example of configuring EXTERN SERVER.

```
*NODE
tmaxh3        ...
              EXTPORT = 9000, EXTCLHPORT = 9010

*SERVER
alinkjmapp    SVGNAME = svg1,
              MIN = 1, MAX = 1,
              CPC = 10,
              SVRTYPE=EXTSVR

*SERVICE
JMAPPER       SVRNAME = alinkjmapp,
              SVCTIME = 30
```

- SMSUPPORT = Y | N

- Default: N
- Option to use SysMaster Trace function.

Value	Description
Y	Supports SysMaster Trace
N	Does not support SysMaster Trace.

- SysMaster Trace function uses the GID of a running service to trace system tasks. Only TCS and UCS servers that can use GID are supported. GID can be used for requests sent through TCP_GATEWAY(CUSTOM_GW) are supported.

  The following describes the GID structure (12 bytes).

	Size	Description
GID0	4 bytes	Unique id for each client in the system. ('cli id' for WebtoB) Identifies clients through domain id, node id, hth #, and slot id.
GID1	4 bytes	First three bytes indicate the seq # and the last byte is for a unique product ID.
SEQNO	4 bytes	First two bytes indicate the branch # for synchronous messaging, and the last two bytes indicate the seq # for all messages.

- SMTBLSIZE = numeric
  - Range: 1024~MAX_INT
  - Default: 50000
  - Maximum number of SysMaster traces that can be stored per CLH. Only required if SMSUPPORT has been set to Y.
- CLLBLOCK = Y | N
  - Default: N
  - Option to use the CLL Block.

Value	Description
Y	CLL will be blocked when a client connects to the Tmax system.

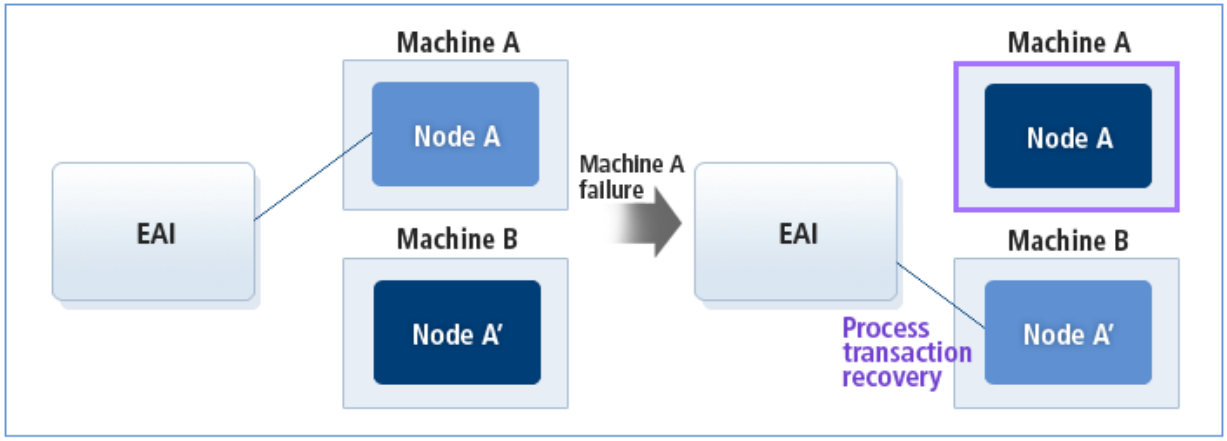
  - If there is a client request when Tmax service is not completely started up, the client is allowed to access the Tmax system if CLL is running, but it will receive the TPENOREADY error.

    In order to prevent such problem, the administrator must have the client request blocked in CLL. This will prevent CLL from receiving service requests before the service is ready.

  - When the service is running and ready, you can cancel the CLL blocking function by using the tmadmin command.
- CLLUNBLKPORT = "Portno1, Portno2 ..."

    - Exception port numbers for the CLL Block. Select among the port numbers set in TMAXPORT.
    - Even if the CLL Block is set, you can specify exception ports for the CLL Block in the Tmax configuration file, and the clients that connect through these port numbers will not be blocked.
- CLLBLOCKTIME = numeric
    - Default: 0
    - CLL Block time (seconds) for if CLLBLOCK=Y. If the block time expires, the CLL Block is released.
    - If this value is not set when the block time is less than 0 or CLLBLOCK=Y, the CLL Block must be released using tmadmin.
- CLLCONNLB = RR | LC
    - Default: RR
    - Method used by CLL to send a client connection to CLH.
    - If set to CLLCONNLB = LC, Least Connection method is used to make the next connection to the CLL with the least number of connections.
    - If CLLCONNLB is not set or is substituted by some other value, the default method (Round Robin) is used to make the next connection.
- CRYPTPORT= Port number list
    - If this value is set in the NODE element, only the client data that were accessed through these ports are encrypted. This allows encryption at the session level which provides a more flexible encryption function.
    - Note:
        - Each port number item must be one of the TMAXPORT items.
        - Encryption can only be used when CRYPT=Y is set in the DOMAIN element.
        - If this setting is not configured in the NODE element after setting CRYPT=Y in the DOMAIN element, CRYPTPORT is set to all port numbers in TMAXPORT.
        - Multiple ports can be assigned.
- TRB = string
    - A Back-up Node (TRB Node) that can process transaction recovery when a node is temporarily unavailable due to machine failure in a Multi-Node & Multi-Domain environment. This setting can help with the low availability problem caused by pending transactions.
    - As in the following figure, preconfigure TRB Node A' for the Node A.

      When machine A fails, the TRB Node A' will start up and automatically recover any pending transaction.



TRB

- TRB item and Primary Node name must be configured in the Back-up Node (TRB Node).
- TRB Node must have one server group. Database information and one TMS must be configured in the server group. To use a Gateway, the TRB item must also be configured in the GATEWAY element.
- Note:
  - Before starting up the TRB Node, you must copy the tlog (TXLOG, GWTXLOG) of the failed Primary Node to the TRB Node.
  - TXLOG and GWTXLOG are in a binary format that is dependent on the machine in use and compatible only with its kind. It is not compatible if the CPU (Little endian / Big Endian) or binary mode (32bit / 64bit) is different.
  - TRB Node must be able to access the Database accessed by the failed Node.
  - Since IP/PORT of the failed Node and TRB Node will be different, the EAI must be able to accept both.
  - TRB Node only executes transaction recovery function. Services are not processed.
  - Basically, startup & termination of TRB Node must be controlled manually. An APM Solution can be used to automate node failure detection and startup/termination of the TRB Node.
- Startup/Termination of TRB Node

  The –B option is used to start up and terminate TRB Node in a way that is different from other nodes.

```
$tmboot –B <nodename>
```

  TRB is configured as follows:

```
*NODE
#Primary node
NODE1     TMAXDIR = /data/tmax

#TRB node
```

```
NODE2     TMAXDIR = /data2/tmax, TRB = NODE1
```

- MAXSESSION = numeric
  - Range: 1 ~ 65535
  - Default: 1024
  - Maximum number of sessions that can be created in an HMS of the node. For details, refer to HMS Settings.
- SQKEY=numeric
  - Range: 32768~262143
  - Key value that refers to the shared memory segment used as Tmax Session Queue repository.
- SQSIZE=numeric
  - Range: 4 ~ 4000000 (Unit: KB)
  - Size of the shared memory segment used as SQ repository.
- SQMAX=numeric
  - Range: 2 ~ MAX_INT-1 (Unit: KB)
  - Maximum number of SQs that can be created in the node. Must be an even number. If you use an odd number, it will be automatically decreased by 1.
- SQKEYMAX=numeric
  - Default: 1~ MAX_INT
  - Maximum number of keys that can be saved in each SQ.
- SQTIMEOUT=numeric
  - Default: 1~MAX_INT (seconds)
  - Timeout for SQ data keeping. When timeout is reached, data will be automatically deleted.
- IP = literal
  - Size: Up to 255 characters
  - IP address of the node.
  - Can be used if the IP address cannot be deduced from the host name when attempting to connect to another node.
  - Set SYSTEM_PROTOCOL to 'IPV6' to use an IPv6 address.
  - Used as IP address of an interface to bind to when creating a port for client connection (related setting: CLL_BIND_IP).
- CLIENT_PROTOCOL = "IPV4" | "IPV6" | "SDP"
  - Default: "IPV4"
  - Protocol used when creating a port for client connections. If multiple protocols can be used, they are set by using a comma (,) as a delimiter.

Value	Description
IPV4	Uses the IPv4 protocol to wait for client connection requests, and the client can connect to the system from IPv4 environment. Can be set for each node.
IPV6	Uses the IPv6 protocol to wait for client connection requests, and the client can connect to the system from both IPv4 and IPv6 environments.
SDP	Uses InfiniBand Socket Direct Protocol (SDP). Can be used along with IPV4 or IPV6.

- SYSTEM_PROTOCOL = "IPV4" | "IPV6" | "SDP"
  - Default: "IPV4"
  - Protocol used when creating a port for inter-node communication in a multi-node environment. If multiple protocols can be used, they are set by using a comma (,) as a delimiter.

Value	Description
IPV4	Uses the IPv4 protocol to wait for connection requests. Can be set for each node.
IPV6	Uses the IPv6 protocol to wait for connection requests when creating a port for another node.
SDP	Uses InfiniBand Socket Direct Protocol (SDP). Can be used along with IPV4 or IPV6.

- EXTSVR_PROTOCOL = "IPV4" | "IPV6" | "SDP"
  - Default: "IPV4"
  - Protocol used when creating a port for connection to the Extern Server from Tmax system. If multiple protocols can be used, they are set by using a comma (,) as a delimiter.

Value	Description
IPV4	Uses the IPv4 protocol to wait for connection requests. Can be set for each node.
IPV6	Uses the IPv6 protocol to wait for connection requests.
SDP	Uses InfiniBand Socket Direct Protocol (SDP). Can be used along with IPV4 or IPV6.

- CLIENT_IPV6 = Y | N
  - Default: N
  - Option to use the IPv6 protocol when creating a port for client connection.

Value	Description
Y	Use IPv6 protocol. A client can connect to the Tmax system in either IPv4 or IPv6 environment.
N	Use IPv4 protocol. A client can connect to the system only through IPv4.

It is recommended to use CLIENT_PROTOCOL instead of this item.

- SYSTEM_IPV6 = Y | N
  - Default: N
  - Option to use the IPv6 protocol when creating a port for inter-node communication in a multi-node environment.

Value	Description
Y	Use IPv6 protocol when creating a port used for connection requests from other nodes.
N	Use IPv4 protocol.

It is recommended to use SYSTEM_PROTOCOL instead of this item.

- EXTSVR_IPV6 = Y | N
  - Default: N
  - Option to use the IPv6 protocol when creating a port for connection to the extern server from Tmax system.

Value	Description
Y	Use IPv6 protocol.
N	Use IPv4 protocol.

It is recommended to use EXTSVR_PROTOCOL instead of this item.

- CLL_BIND_IP = Y | N
  - Default: Y
  - Option to use an interface of the IP address set in the NODE element when creating a port for client connection.

Value	Description
N	Connections from all interfaces are allowed.

- SVCLOG_FORMAT = literal
  - Default: N
  - Allows for users to modify svclog items. Specify a string such as "$(CLIENTIP) $(RESULT)".

String	Item
$(CLIENTIP)	Client IP
$(SBUFTYPE)	Request buffer type
$(RBUFTYPE)	Return buffer type
$(RESULT)	Processing result
$(MEM)	Memory usage
$(SCPU)	System CPU usage
$(UCPU)	User CPU usage
$(TID)	tid

- CASTHREAD = 1 ~ 65535
  - Number of worker threads for encryption/decryption, authentication, and qualification when third-party security system is used for a CAS process. For more information about security settings, refer to Third-party Settings.
- CASOPT = literal
  - Can be up to 255 characters
  - Command options passed to a CAS process when the process starts if third-party security system is used for the process. For more information about security settings, refer to Third-party Settings.
- SECURITY_LIB = literal
  - Third-party authentication library path if the library is used instead of the default authentication library provided by Tmax. For more information about security settings, refer to Third-party Settings.
- CRYPT_LIB = literal
  - Third-party encryption library path if the library is used instead of the default encryption library provided by Tmax. For more information about security settings, refer to Third-party Settings.
- TGOPT = literal
  - Can be up to 255 characters
  - Command options sent to the a TmaxGrid process when the process starts. Options before '--' are used by Tmax system, and those that come after '--' can be used by general users.

◦ The following are available user options.

Option	Description
-e file_name	Writes standard errors that occur during TmaxGrid operation to the specified file.
-o file_name	Writes standard outputs that occur during TmaxGrid operation to the specified file.

Re-definable Items

Some of items of the NODE element can be defined in the DOMAIN element, and redefined in the NODE element of each node. If they are not redefined in the NODE element, values specified in the DOMAIN element are applied to all the nodes. Otherwise, the value redefined in the NODE element will be applied to the node and its syntax and content are the same as the DOMAIN element.

The following are the expression format and relevant table.

```
Item name = Value
```

Item Name	Value	Default Value
SHMKEY	numeric	
MINCLH	numeric	1
MAXCLH	numeric	10
TPORTNO	numeric	8888
RACPORT	numeric	3333
MAXUSER	numeric	
IPCPERM	numeric	0600
MAXSPR	numeric	64
MAXSVR	numeric	64
MAXCPC	numeric	32
LOGOUTSVC	string	
TMMLOGLVL	string	DETAIL
CLHLOGLVL	string	DETAIL
TMSLOGLVL	string	DETAIL
LOGLVL	string	DETAIL
TDL	Y|N	N
CLIENT_PROTOCOL	"IPV4" | "IPV6" | "SDP"	"IPV4"
SYSTEM_PROTOCOL	"IPV4" | "IPV6" | "SDP"	"IPV4"

Item Name	Value	Default Value
EXTSVR_PROTOCOL	"IPV4" | "IPV6" | "SDP"	"IPV4"
CLIENT_IPV6	Y|N	N
SYSTEM_IPV6	Y|N	N
EXTSVR_IPV6	Y|N	N
CLL_BIND_IP	Y|N	Y
MSGSIZEWARN	numeric	1073741824
MSGSIZEMAX	numeric	1073741824

Example

The following are examples of the NODE section.

- Example 1 (a physical node)

```
*NODE
DEFAULT:
       MINCLH = 2, MAXCLH = 3
tmax1  TMAXDIR = "/home/tmax",
       APPDIR = "/home/tmax/appbin",
       PATHDIR = "/home/tmax/path",
       SLOGDIR = "/home/tmax/log/slog",
       TLOGDIR = "/home/tmax/log/tlog",
       ULOGDIR = "/home/tmax/log/ulog",
       TMMOPT = "-o /home/tmax/log/slog/tmmo.log -e /home/tmax/log/slog/tmme.log",
       CLHOPT = "-o /home/tmax/log/slog/clho.log -e /home/tmax/log/slog/clhe.log",
       ENVFILE = "/home/tmax/server/start",
       TMAXPORT = "8850, 9000, 9001, 9002, 9003",
       CompressPort = "9000, 9001",
       CompressSize = 10240,
       RESTART = Y,
       MAXRSTART = -1,
       GPERIOD = 100
```

- Example 2 (logical nodes on a single machine)

```
*NODE
DEFAULT:
        HOSTNAME = "tmaxs1",
        TMAXHOME = "/user/QA/tmax",
        NODETYPE = SHM_USER

tmaxs1  TMAXDIR = "/user/QA/tmax",
        APPDIR  = "/user/QA/tmax/appbin",
        PATHDIR = "/user/QA/tmax/path",
        TLOGDIR = "/user/QA/tmax/log/tlog",
        ULOGDIR = "/user/QA/tmax/log/ulog",
        SLOGDIR = "/user/QA/tmax/log/slog"
```

```
NODE1   TMAXDIR = "/user/QA/proj1",
        APPDIR  = "/user/QA/proj1/appbin",
        PATHDIR = "/user/QA/proj1/path",
        TLOGDIR = "/user/QA/proj1/log/tlog",
        ULOGDIR = "/user/QA/proj1/log/ulog",
        SLOGDIR = "/user/QA/proj1/log/slog",
        TPORTNO = 8952, SHMKEY = 76995,
        RACPORT = 3334

NODE2  TMAXDIR = "/user/QA/proj2",
       APPDIR  = "/user/QA/proj2/appbin",
       PATHDIR = "/user/QA/proj2/path",
       TLOGDIR = "/user/QA/proj2/log/tlog",
       ULOGDIR = "/user/QA/proj2/log/ulog",
       SLOGDIR = "/user/QA/proj2/log/slog",
       TPORTNO = 8993, SHMKEY = 76999,
       RACPORT = 3335, NODETYPE = RACD
```

In the previous example, the name of the machine (or host) is given as tmaxs1 (HOSTNAME), and the three logical nodes of the host are named tmaxs1, NODE1, and NODE2. TMAXHOME is the Tmax installation directory shared by all three logical nodes, and it contains the bin, lib, usrinc, tuxinc, topinc, and cblinc folders. TMAXDIR is the working directory for each node and contains the config, path, license, log, and svct folders. If tmboot is executed for tmaxs1, tmboot (as well as tmdown/tmadmin) can be directly executed on NODE1 without using RACD since its node type is SHM_USER. However, NODE2 must execute tmboot through RACD since its node type is SHM_RACD.

Each logical node uses TMAXDIR to find its node number, and connects with TMM to execute "tmboot/tmdown/tmadmin/cfl/gst" or read or write from/to a file.

To compile the configuration file, cfl references config (${TMAXDIR}/config) of the $TMAXDIR directory. To create a service table in the local node, gst references "${TMAXDIR}/svct".

- Example 3 (logical nodes on multiple machines)

```
*NODE
Default:
        HOSTNAME = "tmaxs1",
        TMAXHOME = "/user2/QA/tmax32",
        NODETYPE = SHM_USER

tmaxs1  TMAXDIR = "/user2/QA/tmax32",
        APPDIR  = "/user2/QA/tmax32/appbin",
        PATHDIR = "/user2/QA/tmax32/path",
        TLOGDIR = "/user2/QA/tmax32/log/tlog",
        ULOGDIR = "/user2/QA/tmax32/log/ulog",
        SLOGDIR = "/user2/QA/tmax32/log/slog"

NODE1   TMAXDIR = "/user2/QA/proj1",
        APPDIR  = "/user2/QA/proj1/appbin",
        PATHDIR = "/user2/QA/proj1/path",
        TLOGDIR = "/user2/QA/proj1/log/tlog",
        ULOGDIR = "/user2/QA/proj1/log/ulog",
        SLOGDIR = "/user2/QA/proj1/log/slog",
```

```
        TPORTNO = 8952, SHMKEY = 76995,
        RACPORT = 4335
Default:
        HOSTNAME = tmaxc1,
        TMAXHOME = "oracle/QA/tmax",
        NODETYPE = SHM_RACD

tmaxc1  TMAXDIR = "/oracle/QA/tmax",
        APPDIR  = "/oracle/QA/tmax/appbin",
        PATHDIR = "/oracle/QA/tmax/path",
        TLOGDIR = "/oracle/QA/tmax/log/tlog",
        ULOGDIR = "/oracle/QA/tmax/log/ulog",
        SLOGDIR = "/oracle/QA/tmax/log/slog",
        TPORTNO = 8893, SHMKEY = 76980,
        RACPORT = 4335

NODE3   TMAXDIR = "/oracle/QA/proj1",
        APPDIR  = "/oracle/QA/proj1/appbin",
        PATHDIR = "/oracle/QA/proj1/path",
        TLOGDIR = "/oracle/QA/proj1/log/tlog",
        ULOGDIR = "/oracle/QA/proj1/log/ulog",
        SLOGDIR = "/oracle/QA/proj1/log/slog",
        TPORTNO = 9984, SHMKEY = 76999,
        RACPORT = 4336
```

In the previous example there are two hosts (machines), tmaxs1 and tmaxc1, set to the HOSTNAME. There are four logical nodes on the two hosts. Tmaxs1 and NODE1 are on host tmaxs1, and tmaxc1 and NODE3 are on host tmaxc1.

The NODETYPE defined for the nodes tmaxc1 and NODE3 on host tmaxc1 is SHM_RACD. This will cause tmboot/tmdown/tmadmin to execute through the RACD between tmaxc1 and NODE3 on tmaxc1. If the NODE executing tmboot is NODE1 on host tmaxs1, then RACD is used for communication regardless of the NODETYPE of tmaxc1 and NODE3 on tmaxc1 host.

The RACD is initiated by using TMAXDIR and RACPORT. To bypass the inconvenience of changing these two items every time you need to initiate the RACD, use the -i and –l option commands with racd to easily identify the node the RACD belongs to. This is because if the NODETYPE is SHM_RACD, then each logical node has to execute one RACD.

The following are examples of using the TRB Node.

- Multi node

```
*DOMAIN
dom1      SHMKEY =78755,
          RACPORT = 3366, . . .

*NODE
tmaxh4     TMAXDIR = "/data1/tmaxqas/tmax",
           APPDIR = "/data1/tmaxqas/tmax/appbin",

tmaxi1     TMAXDIR = "/data/tmaxqas/tmax",
           APPDIR = "/data/tmaxqas/tmax/appbin",
```

```
tmaxh4b    TMAXDIR = "/data/QA/tmax",
           APPDIR  = "/data/QA/tmax/appbin",
           HOSTNAME = "tmaxi1", SHMKEY = 78630, TPORTNO = 8630,
           TRB = tmaxh4, RACPORT = 3155

tmaxi1b    TMAXDIR = "/EMC01/QA/tmax",
           APPDIR  = "/EMC01/QA/tmax/appbin",
           HOSTNAME = tmaxh4, SHMKEY = 78950, TPORTNO = 8520
           TRB = tmaxi1,
           RACPORT = 3355

*SVRGROUP
svg1       NODENAME = tmaxi1, DBNAME = ORACLE,
           OPENINFO = "...",
           TMSNAME  = tms_ora1

svg5       NODENAME = tmaxh4, DBNAME = ORACLE,
           OPENINFO = "..."
           TMSNAME  = tms_ora5

#TRB NODE
svgb1      NODENAME = tmaxh4b, DBNAME = ORACLE,
           OPENINFO = "...",
           TMSNAME  = tms_orab1,
           MINTMS = 1, MAXTMS = 1  #only 1 tms needed

svgb2      NODENAME = tmaxi1b, DBNAME = ORACLE,
           OPENINFO = "..."
           TMSNAME  = tms_orab2,
           MINTMS = 1, MAXTMS = 1

*SERVER
svr2301TX  SVGNAME = svg1,
           MIN = 5, MAX = 5, MAXRSTART = -1
svr2305TX  SVGNAME = svg5,
           MIN = 5, MAX = 5, MAXRSTART = -1
```

- Multi domain example (2 Domains)

  <Multi domain (Dom1)>

```
*DOMAIN
dom1    SHMKEY =78351,
        DOMAINID = 10, ...

*NODE
tmaxh4   TMAXDIR = "/data1/tmaxqas/tmax",
         APPDIR = "/data1/tmaxqas/tmax/appbin",

tmaxh4b  TMAXDIR = "/data/QA/tmax",
         APPDIR  = "/data/QA/tmax/appbin",
         TRB = tmaxh4, HOSTNAME = tmaxi1, ...

*SVRGROUP
svg5     NODENAME = tmaxh4, DBNAME = ORACLE,
         OPENINFO = "...",
         TMSNAME  = tms_ora5
```

```
#TRB NODE
svgb      NODENAME = tmaxh4b, DBNAME = ORACLE,
          OPENINFO = MINTMS=1, MAXTMS=1,  # only 1 tms needed
          TMSNAME = tms_orab1

*SERVER
svr2305TX   SVGNAME = svg5, MIN = 5, MAX = 5, MAXRSTART = -1

*SERVICE
SVC2301TX_1     SVRNAME = gw2301X
SVC2305TX_1     SVRNAME = svr2305TX

*GATEWAY
gw2301X        GWTYPE = TMAX, PORTNO = 4789,
               NODENAME = tmaxh4

               # Domain B's gw2301TX
               RGWADDR = "192.168.1.13",
               RGWPORTNO = 4789,

               # Domain B's gw2302TX (TRB)
               BACKUP_RGWADDR = "192.168.1.43",
               BACKUP_RGWPORTNO = 5789,

gw2302X        GWTYPE = TMAX, PORTNO = 5789,
               NODENAME = tmaxh4b,

               # Domain B's gw2301TX
               RGWADDR = "192.168.1.13",
               RGWPORTNO = 4789,

               # Domain B's gw2302TX (TRB)
               BACKUP_RGWADDR = "192.168.1.43",
               BACKUP_RGWPORTNO = 5789,

               TRB = gw2301X
```

<Multi domain (Dom2)>

```
# Nodeno 1
*DOMAIN
domB      SHMKEY = 78651,
          DOMAINID = 20, ...

*NODE
tmaxi1     TMAXDIR = "/data/tmaxqas/tmax",
           APPDIR = "/data/tmaxqas/tmax/appbin",

tmaxi1b    TMAXDIR = "/EMC01/QA/tmax",
           APPDIR  = "/EMC01/QA/tmax/appbin",
           TRB = tmaxi1, HOSTNAME = tmaxh4

*SVRGROUP
svg1       NODENAME = tmaxi1, DBNAME = ORACLE,
           OPENINFO = "...",
           TMSNAME  = tms_ora1
```

```
# TRB NODE
svgb1      NODENAME = tmaxi1b, DBNAME = ORACLE,
           OPENINFO = "..." ,
           TMSNAME  = tms_orab1, MINTMS = 1, MAXTMS = 1


*SERVER
svr2301TX       SVGNAME = svg1, MIN = 5, MAX = 5, MAXRSTART = -1

*SERVICE
SVC2301TX_1     SVRNAME = svr2301TX
SVC2305TX_1     SVRNAME = gw2301X

*GATEWAY
gw2301X        GWTYPE = TMAX, PORTNO = 4789,
               NODENAME = tmaxi1,

               # Domain A's gw2301X
               RGWADDR = "192.168.1.43",
               RGWPORTNO = 4789,

               # Domain A's gw2302X (TRB)
                BACKUP_RGWADDR = "192.168.1.13",
                BACKUP_RGWPORTNO = 5789,


gw2302X        GWTYPE = TMAX, PORTNO = 5789,
               NODENAME = tmaxi1b,

               # Domain A's gw2301X
                RGWADDR = "192.168.1.43",
                RGWPORTNO = 4789,

                # Domain B's gw2302X (TRB)
                BACKUP_RGWADDR = "192.168.1.13",
                BACKUP_RGWPORTNO = 5789,
                TRB = gw2301X
```

- Multi domain example (3 Domains)

  <Multi domain (Dom1)>

```
*DOMAIN
dom1         SHMKEY = 78351,
             RACPORT = 3155,
             DOMAINID = 10,
#------------------------------------------------------------------------
*NODE

tmaxh4       TMAXDIR = "/data1/tmaxqas/tmax",
             APPDIR = "/data1/tmaxqas/tmax/appbin",

tmaxh4b1     TMAXDIR = "/data/QA/tmax",
             APPDIR  = "/data/QA/tmax/appbin",
             TRB = tmaxh4, SHMKEY = 86655, TPORTNO = 8450,
             HOSTNAME = tmaxi1
```

```
tmaxh4b2     TMAXDIR = "/user1/tmaxqam/tmax",
             APPDIR  = "/user1/tmaxqam/tmax/appbin",
             TRB = tmaxh4, SHMKEY = 86655, TPORTNO = 8450,
             HOSTNAME = ibm51

*SVRGROUP
svg1         NODENAME = tmaxh4, DBNAME = ORACLE,
             OPENINFO = "...",
             TMSNAME  = tms_ora1

# TRB NODE
svgb1        NODENAME = tmaxh4b1, DBNAME = ORACLE,
             OPENINFO = "...",
             TMSNAME  = tms_orab1, MINTMS = 1, MAXTMS = 1

svgb2        NODENAME = tmaxh4b2, DBNAME = ORACLE,
             OPENINFO = "...",
             TMSNAME  = tms_orab2, MINTMS = 1, MAXTMS = 1

*SERVER
svr2301TX    SVGNAME = svg1, MIN = 5, MAX = 5, MAXRSTART = -1
*SERVICE
SVC2301TX_1  SVRNAME = svr2301TX
SVC2305TX_1  SVRNAME = gw2301X
SVC2309TX_1  SVRNAME = gw2302X

*GATEWAY
# Gateway for the domain B
gw2301X      GWTYPE = TMAX, PORTNO = 4010,
             NODENAME = tmaxh4,

             #Domain A's gw2301X
             RGWADDR = "192.168.1.13",
             RGWPORTNO = 4010,

             # Domain B's gw2301Xb (TRB)
             BACKUP_RGWADDR = "192.168.1.31",
             BACKUP_RGWPORTNO = 5010,

             # Domain B's gw2302Xb (TRB)
             BACKUP_RGWADDR2 = "192.168.1.43",
             BACKUP_RGWPORTNO2 = 6010,

gw2301Xb     GWTYPE = TMAX, PORTNO = 5010,
             NODENAME = tmaxh4b1, #192.168.1.13
             RGWADDR = "192.168.1.13",
             RGWPORTNO = 4010,
             BACKUP_RGWADDR = "192.168.1.31",
             BACKUP_RGWPORTNO = 5010,
             BACKUP_RGWADDR2 = "192.168.1.43",
             BACKUP_RGWPORTNO2 = 6010,
             TRB = gw2301X

gw2301Xb2    GWTYPE = TMAX, PORTNO = 6010,
             RGWADDR = "192.168.1.13",
             RGWPORTNO = 4010,
             BACKUP_RGWADDR = "192.168.1.31",
             BACKUP_RGWPORTNO = 5010,
```

```
            BACKUP_RGWADDR2 = "192.168.1.43",
            BACKUP_RGWPORTNO2 = 6010,
            NODENAME = tmaxh4b2, #192.168.1.31
            TRB = gw2301X

# Gateway for Domain C
gw2302X     GWTYPE = TMAX, PORTNO = 4020,

            #Domain C's gw2302X
            RGWADDR = "192.168.1.31",
            RGWPORTNO = 4020,

            # Domain C'sgw2302Xb (TRB)
            BACKUP_RGWADDR = "192.168.1.43",
            BACKUP_RGWPORTNO = 5020,

            # Domain C's gw2302Xb2 (TRB)
            BACKUP_RGWADDR2 = "192.168.1.13",
            BACKUP_RGWPORTNO2 = 6020,
            NODENAME = tmaxh4,

gw2302Xb    GWTYPE = TMAX, PORTNO = 5020,
            RGWADDR = "192.168.1.31",
            RGWPORTNO = 4020,
            BACKUP_RGWADDR = "192.168.1.43",
            BACKUP_RGWPORTNO = 5020,
            BACKUP_RGWADDR2 = "192.168.1.13",
            BACKUP_RGWPORTNO2 = 6020,
            NODENAME = tmaxh4b1,  #192.168.1.13
            TRB = gw2301X

gw2302Xb2   GWTYPE = TMAX, PORTNO = 6020,
            RGWADDR = "192.168.1.31",
            RGWPORTNO = 4020,
            BACKUP_RGWADDR = "192.168.1.43",
            BACKUP_RGWPORTNO = 5020,
            BACKUP_RGWADDR2 = "192.168.1.13",
            BACKUP_RGWPORTNO2 = 6020,
            NODENAME = tmaxh4b2,  #192.168.1.31
            TRB = gw2301X
```

3.2.3. SVRGROUP

Tmax manages servers that process client requests in groups. Servers can be grouped based on their location, the type of database they use, or their logical relationships. A SVRGROUP is a basic unit of Tmax system used in transaction processing, load balancing, fault tolerance functions, and routing.

SVRGROUP element configures the following:

- Node that each server group belongs to.
- Database access information for the server group.
- Options for distributed transaction processing.

Some items of SVRGROUP are configured in the NODE element of the configuration file, but can be

redefined in the SVGROUP element. For details about such items, refer to "Re-definable Items".

The following are the basic settings of the SVRGROUP element.

```
*SVRGROUP
[DEFAULT :]
SVRGROUP Name        NODENAME = node-name
                    [APPDIR = path,]
                    [ULOGDIR = path,]
                    [SVGTYPE = TMAX|EXTSVG ,]
                    [COUSIN = group-name,]
                    [BACKUP = group-name,]
                    [LOAD = numeric,]
                    [DBNAME = name-of-database,]
                    [OPENINFO = string,]
                    [CLOSEINFO = string,]
                    [MINTMS = numeric,]
                    [MAXTMS = numeric,]
                    [TMSNAME = name-of-tms,]
                    [ENVFILE = path,]
                    [XAOPTION = xaoption,]
                    [CPC = numeric,]
                    [RESTART = (Y)|N,]
                    [MAXRSTART = numeric,]
                    [GPERIOD = numeric,]
                    [TMSLOGLVL = tms-log-level,]
                    [LOGLVL = server-log-level,]
                    [TMSRECOVERY = (Y)|N,]
                    [TMSDEP = tms_name,...,]
                    [TMSRANGE = (DOMAIN)|NODE,]
                    [TMSTYPE = (STD) | STD_MT,]
                    [TMSTHREAD = numeric,]
                    [TMSOPT = arguments,]
                    [TMSXATIME = numeric,]
                    [DUMMY = Y | (N),]
                    [HMSNAME = string,]
                    [HMSINDEX = numeric,]
                    [HMSMAXTHR = numeric,]
                    [HMSMAXDBTHR = numeric,]
                    [HMSMAXBULKTHR = numeric,]
                    [HMSMAXBULKSIZE = numeric,]
                    [HMSOPT = literal,]
                    [HMSSUBSCFG = string,]
                    [HMSMSGLIVE = numeric,]
                    [HMSPORT = numeric,]
                    [HMSHEARTBEAT = numeric,]
                    [HMSGQINT = numeric]
                    [SVCLOG_FORMAT = svclog-format-string,]
                    [RQSOPT = literal]
```

Required Items

- SVRGROUP Name = string

- Size: Up to 63 characters
- Logical name of the server group. Must be unique in the SVRGROUP element.
- This variable will be referred to from the SVGNAME item of the SERVER element.

- NODENAME = string
  - Size: Up to 63 characters
  - Node where the server group exists. Must be a node that is defined in the NODE element. If the node name includes a hyphen (-), it must be enclosed in double quotation marks.

Optional Items

- Default: item = value, ...
  - Refer to DEFAULT in "3.2.2 NODE".
- SVGTYPE = string
  - Default: TMAX
  - Server group type. One of TMAX, RQMGR, EXTSVG, and TXRQMGR. The default value is TMAX. Refer to *Tmax Programming Guide (MultipleRM)* for detailed information about MTMAX and STMAX.
  - EXTSVG is same as EXTERN SERVER. It is selected when using the EXTERN TMS to support WAS(Web Application Server) and 2PC. When using EXTSVG, the OPENINFO item must also be configured. Even though its value is not significant, set the OPENINFO element to " ".

    <Example of using EXTSVG Server Group>

```
*NODE
tmaxh3          ...
                EXTPORT = 9000, EXTCLHPORT = 9010

*SVRGROUP
Extsvg          NODENAME = "tmaxh4",
                SVGTYPE = "EXTSVG", TMSNAME = TMSWAS
                OPENINFO = " "

*SERVER
alinkjmapp      SVGNAME = Extsvg1,
                MIN = 1, MAX = 1, CPC = 20,
                SVRTYPE = EXTSVR

*SERVICE
JMAPPER         SVRNAME = alinkjmapp,
                SVCTIME = 30
```

- COUSIN = literal
  - Size: Up to 8000 characters
  - Groups that can share a process or perform routing with. These sever groups can exist on a same node or on different nodes. Both COUSIN and BACKUP items can be configured, but

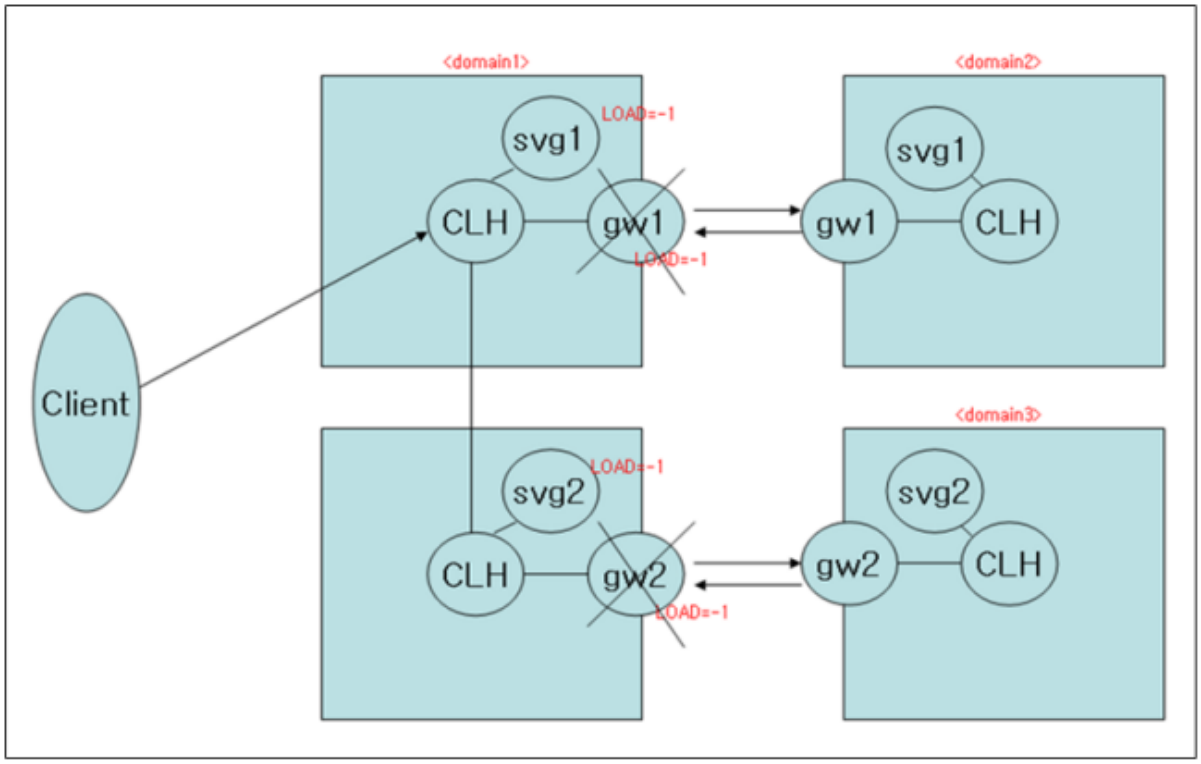
they must be set to different server groups.

The server groups set in COUSIN are in charge of routing, and the server groups set in BACKUP handle failures. Gateway servers can also be included in COUSIN.

- BACKUP = literal
  - Size: Up to 8000 characters
  - Backup group name used to provide seamless services even in case of a failure.
- LOAD = numeric
  - LOAD = -1

    Use Static Routing. Client service requests will only be routed to the server groups of the node to which they are connected. If LOAD is set to -1 (for local use only), the domain gateways in COUSIN are not called. Thus, LOAD cannot be set to -1 in the domain gateway's COUSIN.

    This option cannot be used for transaction and non-transaction gateways.



If LOAD is Set to -1

- Example

```
*SVRGROUP
svg1      NODENAME = tmaxh4,
          COUSIN = "svg2, gw1, gw2", LOAD = -1
svg2      NODENAME = tmaxh2, LOAD = -1

*GATEWAY
gw1      NODENAME = tmaxh4, LOAD = -1
gw2      NODENAME = tmaxh2, LOAD = -1
```

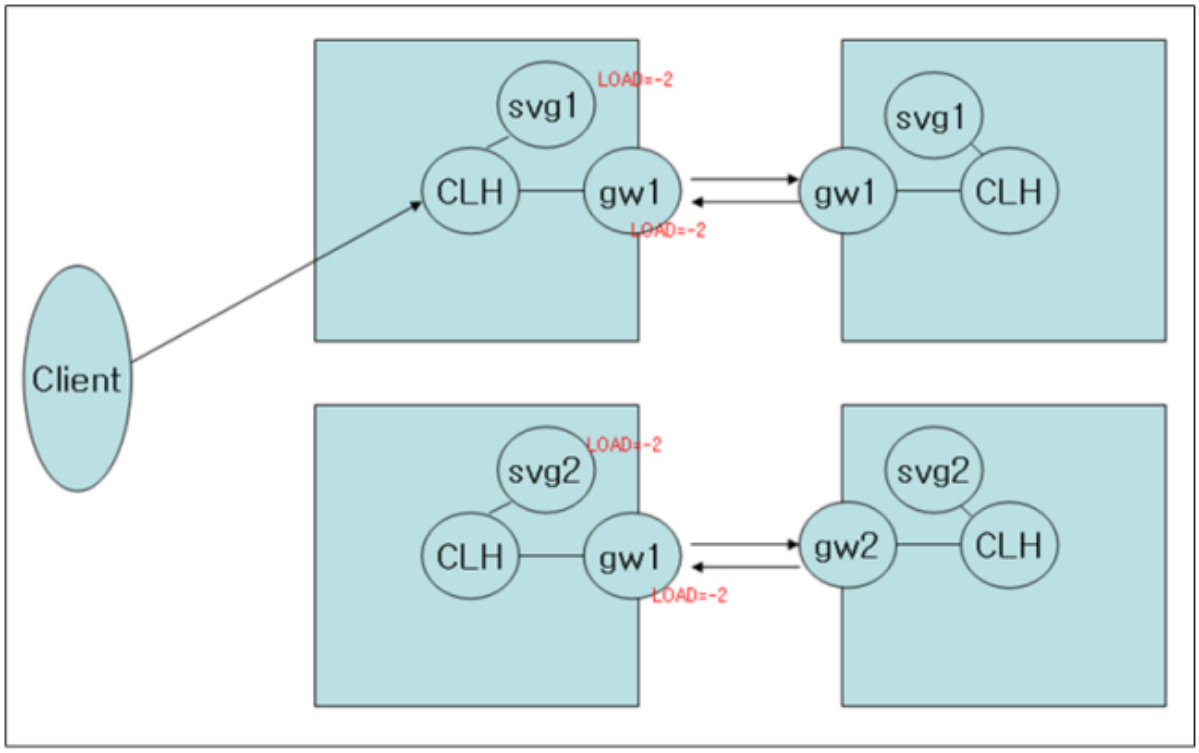
◦ LOAD = -2

Configure each gateway in a multi node environment and create a separate client connection to each node. Then, by having the initial node, to which the client connects, process the requested service can help decrease the work load.

When you use the process local first function in a multi-node environment grouped by COUSIN, the server group of the node that the client is connected to will process the request one-on-one. The gateway of the node is used to make a request to another domain.

When you make a call to a certain server group using tpcallsvg and tpacallsvg, the request can be processed by all server groups even if the local node first function(LOAD=-2) is set. Moreover, if the server group is terminated, the service is not replaced by another server. Instead, the user receives a TPENOREADY error.

For a transaction gateway, XA server group must be used as general server group (dummy server group) for configuring COUSIN. LOAD must be set to -2 for all server groups and gateways grouped as COUSIN.



If LOAD is Set to -2

▪ Example of setting a non-transaction gateway

```
*SVRGROUP
svg1       NODENAME = tmaxh4,
           COUSIN = "svg2, gw1, gw2", LOAD = -2
svg2       NODENAME = tmaxh2, LOAD = -2

*GATEWAY
gw1       NODENAME = tmaxh4, GWTYPE = TMAXNONTX, LOAD = -2
gw2       NODENAME = tmaxh2, GWTYPE = TMAXNONTX, LOAD = -2
```

- Example of setting a transaction gateway

```
*SVRGROUP
svg1      NODENAME = tmaxh4,
          DBNAME = ORACLE,
         OPENINFO = "Oracle_Xa+Acc=P/scott/tiger+SesTm=60+LogDir=
         /data1/tmaxqam/tmax/log/tlog/xalog",
         TMSNAME = tms_ora,
          COUSIN = "svg2, gw1, gw2", LOAD = -2
svg2      NODENAME = tmaxh2,
          DBNAME = ORACLE,
         OPENINFO = "Oracle_Xa+Acc=P/scott/tiger+SesTm=60+LogDir=
         /data1/tmaxqam/tmax/log/tlog/xalog",
         TMSNAME = tms_ora,
          LOAD = -2

*GATEWAY
gw1       NODENAME = tmaxh4, GWTYPE = TMAX, LOAD = -2
gw2       NODENAME = tmaxh2, GWTYPE = TMAX, LOAD = -2
```

◦ LOAD = -3

Use when a backup is required between server groups in COUSIN.

The BACKUP item of the SVRGROUP element is not a backup configuration for server groups set in COUSIN but a backup configuration for the server group. For instance, let's assume that a server group A is configured with COUSIN="B,C" and BACKUP="D". If all the nodes where A, B, and C belong to are terminated, backup scheduling to D is attempted only when the node that A belongs to fails. LOAD=-3 can be used to failover to server group D when all servers that belong to A, B, and C server groups are disabled.

- Example

```
A COUSIN = "B,C,D", LOAD = -2
B LOAD = -2
C LOAD = -2
D LOAD = -3
```

With the previous configuration, scheduling is attempted only to A, B, and C. If A, B, and C all fail, scheduling is attempted to D. Normally, scheduling uses server groups with a LOAD value other than -3, but when there is failure scheduling uses server groups with LOAD=-3. In this case, the scheduling follows the local first load balancing (LOAD=-2).

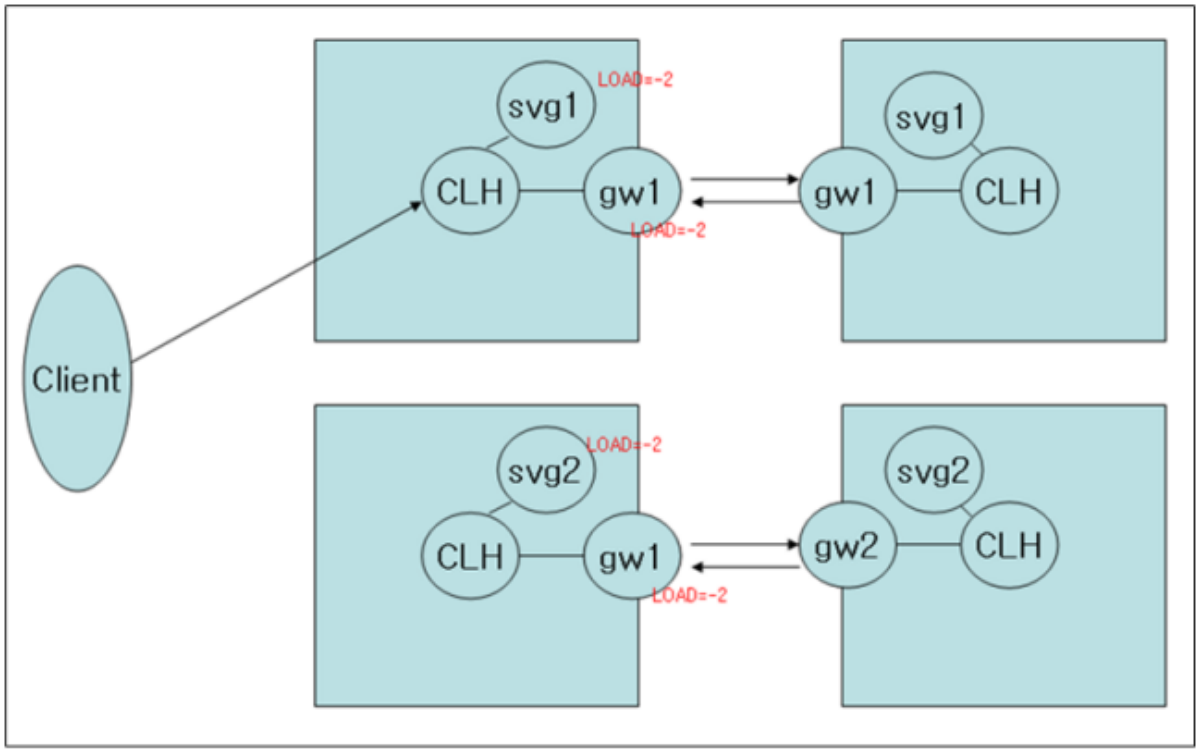
◦ LOAD = 0

Use dynamic routing for load balancing. Tmax routes client requests to the server group with the required service and the lowest work load at the time.

◦ LOAD > 0

Use Rule based routing. Client service requests are distributed on the target server group

based on the specified LOAD ratio. Can only be used for a group specified in COUSIN. Load balancing for Tmax server group can be configured in the LOAD and ROUTING settings, and both must be set to the same value. If the service is configured in the ROUTING element, it can only be scheduled through ROUTING and the LOAD setting is ignored. Thus, it is impossible to perform scheduling by grouping the server groups of each ROUTING in the COUSIN setting.



If LOAD > 0

- Example

```
*SVRGROUP
svg1      NODENAME = tmaxh4,
          COUSIN = "svg2, gw1, gw2", LOAD = 1
svg2      NODENAME = tmaxh2, LOAD = 1

*GATEWAY
gw1      NODENAME = tmaxh4, LOAD = 1
gw2      NODENAME = tmaxh2, LOAD = 1
```

- DBNAME = string
  - Size: Up to 63 characters
  - Unique DB name required if TMS is configured.
- OPENINFO = literal
  - Size: Up to 255 characters
  - Used to connect the server group to the database through TMS. Initializes the DB connection and must be defined by using the syntax required by each DB. Do not use "@@" because the

characters are recognized as an encrypted statement.

- To encrypt some settings, use ***** (5 asterisks) in place of the setting. When using cfl, a password will be prompted for the encrypted parts.
- Partial encryption

```
OPENINFO = "ORACLE_XA+Acc=P/scott/*****+SesTm=100"
```

- Full encryption

```
OPENINFO = "*****"
```

- Encrypted statement

  Encrypted statements can be used by using tencrypt.

- CLOSEINFO = literal
  - Size: Up to 255 characters
  - Required if TMS, which is a group that integrates to the DB, is configured. Used to terminate the connection with the DB. Must be defined by using the syntax required by each DB. This item is not required by most DBs except for Informix.
- MINTMS = numeric
  - Range: 1~16
  - Default: 2
  - Number of TMS processes that will be loaded in the server group when Tmax system starts up if TMS is configured.
- MAXTMS = numeric
  - Range:1~16
  - Default: 3
  - Maximum number of active TMS processes in the server group at any one time.
- TMSNAME = string
  - Size: Up to 63 characters
  - Name of the TMS process that will manage the database for the server group if TMS is configured.
- ENVFILE = literal
  - Size: Up to 255 characters
  - Required to pass values using environment variables to servers that are part of the server group. It can also be used to connect to multiple homogeneous databases on the same node.
- XAOPTION = literal

    - Only required for a limited number of DBMSs. For DBMSs like DB2 that only support dynamic registration, set it to 'Dynamic'.
    - If the DBMS is similar to Sybase or earlier versions of Informix and uses 'int' (not 'long') for the xaswitch field even though it has a 64-bit engine, set it to 'XASWITCH32'.
- CPC = numeric
    - Range: 1 ~ 128
    - Number of channels with CLH processes. Used only when SVGTYPE is RQMGR. For details, refer to Reliable Queue Settings.
- RESTART = Y | N
    - Default: Y
    - Option to reload TMS processes if they are abnormally terminated.
- MAXRSTART, GPERIOD
    - Options to configure Tmax fault tolerance feature. For details, refer to Fault Tolerance Settings.
- TMSRECOVERY = Y | N
    - Default: Y
    - Option to use transaction recovery.
    - Tmax 4 and later use the Transaction Recovery Function defined in X/OPEN DTP. When TMS is restarted, Tmax obtains an XID list of pending transactions from the RM and performs transaction recovery on the list. Transaction recovery function is supported in a single node or multi node environment, or a multi-domain system that uses Tmax Domain Gateways for communication.
    - Since recovery is performed when the entire group is started/terminated, recovery can be performed by each TMS group. A command option is added to tmboot/tmdown to start/terminate a specific TMS. For more details about the command, refer to *Tmax Reference Guide*.
    - Permissions to query the Pending list must be given to RM to use this feature.
- TMSDEP = "tmsname1,..."
    - When different server groups of a single node share a database, dependencies can be established to prevent recovery from executing while TMS is running.

      As in the following example, if three server groups share one DB, tms1 can be configured to perform recovery for both tms2 and tms3 to prevent duplicate recovery processing. For stable recovery, recovery is not performed while tm2 or tm3 is running.

```
svg1 TMSNAME = tms1,
     TMSDEP = "tms2, tms3"

svg2 TMSNAME = tms2,
     TMSRECOVERY = N

svg3 TMSNAME = tms3,
```

```
    TMSRECOVERY = N
```

- TMSRANGE = DOMAIN | NODE
  - Default: DOMAIN
  - When server groups on different nodes share a single database, TMSRANGE can be set to NODE so that recovery is performed separately for each node. This will reduce inter-node traffic and increase stability of the recovery function.
- TMSTYPE = STD | STD_MT
  - Default: STD
  - TMS type used.

Value	Description
STD	To use general TMS, set TMSTYPE to STD. Must use libtms.so library.
STD_MT	To use multi-threaded TMS, set TMSTYPE to STD_MT. Must use libtmsthr.so library.

- TMSTHREAD = numeric
  - Number of working threads per TMS process.
- TMSOPT = "arguments"
  - Location and name of the TMS log file. Its usage is the same as in TMMOPT and CLHOPT, but the -J option of CLOPT in the SERVER section is additionally supported.
- TMSXATIME = numeric
  - Unit: seconds
  - XA processing that takes longer than TMSXATIME is logged.
  - If XA processing, such as xa_prepare, xa_commit, and xa_rollback, takes unusually long time in TMS, this may indicate a DB failure. Logging is provided for cause analysis of the failure in the future.
  - Log format:

```
CLH2160 XA function name(xid) processing was delayed (elapsed time) by tms(server group no.)
       : Return value of XA function
CLH2161 XA function name(xid) processing may be stopped due to closed tms(server group no.)
```

- DUMMY = Y | N
  - Default: N
  - Option to temporarily use a server group name as "__dummy".
  - If set to Y, TMS of the group is not started up when executing tmboot.

The DUMMY setting of the SVRGROUP element is not inherited by the SERVER element, and so DUMMY must be separately configured for each SERVER element.

  - Can be useful when a dummy server group setting, such as Domain Gateway Cousin, is needed. For details, refer to Domain Gateway COUSIN Setting.
  - If not set, the following error occurs during tmboot.

```
(F) BOOT0014 exec error : /data1/tmaxkjh/tmax/appbin/tms_tbr [BOOT0029]: No such file or
directory
```

- HMSNAME = string
  - Size: Up to 63 characters
  - Name of HMS process. To use HMS, SVGTYPE must be set to HMS. For more details about the HMS setting, refer to *Tmax HMS User Guide*.
- HMSINDEX = numeric
  - Range: 0 ~ 65535
  - Unique value in the domain that is required for processing an HMS message.
- HMSMAXTHR = numeric
  - Range: 0 ~ 65535
  - Number of threads that do not perform storage processing for HMS messages.
- HMSMAXDBTHR = numeric
  - Range: 0 ~ 65535
  - Number of threads that perform storage processing for HMS messages.
- HMSMAXBULKTHR = numeric
  - Range: 1 ~ 65535
  - Number of threads that perform batch storage processing for HMS messages.
- HMSMAXBULKSIZE = numeric
  - Range: 2 ~ 65535
  - Maximum number of HMS messages that can be simultaneously processed for batch storage processing.
- HMSOPT = literal
  - Size: Up to 255 characters
  - Command options sent to an HMS process when the process starts.

Option	Description
-e file_name	Writes standard errors that occur during HMS operation to the specified file.

Option	Description
-o file_name	Writes standard outputs that occur during HMS operation to the specified file.

- HMSSUBSCFG = string
  - Size: Up to 255 characters
  - Path and name of a configuration file where Durable Subscriber is defined.
  - The durable subscriber is automatically registered when HMS starts up.
- HMSMSGLIVE = numeric
  - Range: 0 ~ 65535 (unit: hours)
  - An interval to delete persistent messages that HMS accumulates in a storage.
- HMSPORT = numeric
  - Port number used by HMS to communicate with other HMSs in a cluster.
- HMSHEARTBEAT = numeric
  - Range: 0 ~ 65535 (unit: seconds)
  - Heartbeat check interval. Clustered HMSs regularly exchange heartbeat messages to detect any network failure among them.
- HMSGQINT = numeric
  - Range: 0 ~ 65535 (unit: milliseconds)
  - Interval for updating status information required for HMS queue clustering.
  - Must be set to use a clustered queue. Data is more evenly distributed in the queue if the interval is shorter, but network load is heavier. Set a value accordingly.
- RQSOPT = literal
  - Can be up to 255 characters
  - Command options sent to an RQS process when the process starts.

Option	Description
-e file_name	Writes standard errors that occur during RQS operation to the specified file.
-o file_name	Writes standard outputs that occur during RQS operation to the specified file.

Re-definable Items

- APPDIR = literal
  - Size: Up to 255 characters
  - Paths of the directories that contain the executable files for server application programs.
  - APPDIR of the NODE element sets the paths of all applications running on the node, and it

can be redefined in the SVRGROUP to set application file paths for the server group.

- ULOGDIR = literal
  - Size: Up to 255 characters
  - Paths of the directories where user messages are logged. Can be redefined for each server group. The method of logging user messages is same as ULOGDIR of the NODE element.

The specified path must exist in the system.

- TMSLOGLVL = string
  - Log level of TMS.
- LOGLVL = string
  - Log level of the server.
- SVCLOG_FORMAT = literal
  - Allows for users to modify svclog items.

Example

The following is an example of using the SVRGROUP element.

```
*SVRGROUP
svg1          NODENAME = tmax,
              APPDIR = "/home/tmax/appbin",
              ULOGDIR = "/home/tmax/appbin/svg1_log"

svgora        NODENAME = tmax, DBNAME = ORACLE,
              OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm = 60,
              DbgFl = 0x01 + LogDir=/tmp",
              TMSNAME = svg1_tms

svginfx       NODENAME = tmax, DBNAME = INFORMIX,
              OPENINFO = "stores7; USER = {userid};
              PASSWD = {password}",
              TMSNAME = info_tms
```

LogDir in svgora specifies the path where xa log is saved. With INFORMIX, if the account for booting Tmax system is same as the one for logging in Informix, you may omit the USER, PASSWD as in OPENINFO="stores7".

Verification of server group setting that is set as COUSIN

You can output the errors during the cfl stage to prevent the following configuration of server groups that are configured as COUSIN.

- Reconfigured as the BACKUP of another server group (Complex setting)

```
*SVRGROUP
svg1            NODENAME = "tmaxh4",
                COUSIN = "svg2, svg3"
svg2            NODENAME = "tmaxh4",
                BACKUP = "svg3"
svg3            NODENAME = "tmaxh2"
```

The following compilation error occurs:

```
(E) CFL3008 server group svg3 is defined as both COUSIN and BACKUP [CFL0309]
```

- Reconfigured as a COUSIN of another server group (Multiple inheritance)

```
*SVRGROUP
svg1            NODENAME = "tmaxh4",
                COUSIN = "svg3"
svg2            NODENAME = "tmaxh4",
                COUSIN = "svg3"
svg3            NODENAME = "tmaxh2"
```

The following compile error occurs:

```
(E) CFL3008 server group svg3 is defined as duplicate COUSIN [CFL0310]
```

Multi Backup Setting

The Multi Backup function is provided by expanding backup functions that are provided as fault tolerance measures.

```
svg1 NODENAME = @HOSTNAME@, BACKUP = "svg2,svg3,svg4"
svg2 NODENAME = @RMTNAME@
svg3 NODENAME = @RMTNAME2@
svg4 NODENAME = @RMTNAME3@
```

To provide a more stable recovery function, Multi Backup function enables failover to another backup server when the original backup server fails.

One or more server group(s) can be specified in the Backup item. To set multiple backup servers, use a comma(,) as separator. If there is a failure on the original server node, the transactions will failover to the first backup node. And if there is a failure on the first backup node, the transactions will failover to the second backup node.

When the original node recovers while the backup node is processing, the fail back function can be used to allow the original node to process the transactions again. When the first backup node recovers while the second backup node is processing, the transactions will fail back to the first backup node, and when the main node recovers, the transactions will be processed by the original

node.

3.2.4. SERVER

The DOMAIN, NODE, and SVRGROUP elements of the configuration file contain variables that are referenced internally by Tmax system. The SERVER and SERVICE elements on the other hand contain definitions and settings for the services that are requested by clients. The application servers and the services that they process must be configured in these two elements of the configuration file to provide information about them.

Since Tmax cannot process new services until they are defined in the configuration file, they must be added to both the SERVER and SERVICE elements.

All server application programs managed by Tmax system will have to be registered in the SERVER element. The servers defined in the SERVER element will be loaded into memory when Tmax starts up and removed from memory when Tmax shuts down.

The following are defined in the SERVER element:

- Server group that the server belongs to.
- Commands used to launch server processes.
- Maximum and minimum number of allowed server processes.
- The number of queues required for the dynamic management of server processes.
- Option to use interactive mode.
- Option to use automatic restart and allowed number of restarts (Refer to Fault Tolerance Settings.)
- Maximum and minimum number of threads of Multi Thread/Multi Context server process, and thread stack size.

Some items defined in the SVRGROUP element can be re-defined in the SERVER element. For information about the items, refer to "Re-definable Items".

The following is the basic syntax of the SERVER element.

```
*SERVER
[DEFAULT : ]
Server Name          SVGNAME = server-group-name
                    [CLOPT = literal,]
                    [MIN = numeric,]
                    [MAX = numeric,]
                    [CONV = Y | (N) | B ,]
                    [ASQCOUNT = numeric,]
                    [MAXQCOUNT = max-queue-count,]
                    [SVRTYPE = (STD) | UCS | CUSTOM_GATEWAY | STD_DYN |
                             UCS_DYN | REALSVR | REALSVR_MT | EXTSVR,]
                    [CPC = channel-number,]
                    [MINTHR = numeric,]
                    [MAXTHR = numeric,]
```

```
                    [STACKSIZE = numeric,]
                    [ULOGDIR = user-log-path,]
                    [RESTART = (Y) | N,]
                    [MAXRSTART = numeric,]
                    [GPERIOD = numeric,]
                    [TARGET = string,]
                    [AUTOTRAN = Y | (N) | B ,]
                    [SCHEDULE = (FA)/RR,]
                    [LIFESPAN = (IDLE_DOWN) | IDLE_0 | IDLE_sec,]
                    [LOGLVL = server-log-level,]
                    [DUMMY = Y | (N) ,]
                    [MAX_USE_COUNT = numeric,]
                    [CTX_EREPLY = (Y) | N ,]
                    [MULTICLH = (Y) | N ,]
                    [AUS = Y | (N) ,]
                    [TMAPM = (Y) | N ,]
                    [MAC = Y | (N) ,]
                    [ROC = Y | (N) ,]
                    [SVRQTIMEOUT = numeric,]
                    [INBOUNDCPC = numeric]
                    [SVCLOG_FORMAT = svclog-format-string]
```

Required Items

- Server Name = string
  - Size: Up to 63 characters
  - Server name and also the executable name of the server application program. Each server name must be unique and must be defined only once in the SERVER element of the configuration file. In multi-domain systems, a server name must be unique in each domain.
- SVGNAME = string
  - Size: Up to 63 characters
  - Server group that the server belongs to.
  - The server group must already be defined in the SVRGROUP element. By mapping a server to a server group, the node it runs on and the resource manager (database) that it uses are also known. It can also be used to acquire parameters necessary for opening the resource manager.
  - When operating in XA mode, Tmax manages all database transactions and oversees 2PC processing. To facilitate this, the OPENINFO field must be properly configured in the SVRGROUP element as previously described.

    When Tmax is operating in XA mode, users are not involved when the server connects to the database. However, if Tmax is not operating in XA mode, OPENINFO must not be included in the SVRGROUP element and the individual users must manage server-database connections.

Optional Items

- Default: item = value, ...

- Refer to DEFAULT in "3.2.2 NODE".

- CLOPT = literal
  - Size: Up to 255 characters
  - Command options that are passed to the server processes when they start up. Options that are defined before "--" are used by the system, and options that are defined after "--" can be used by users.
  - Major system options include:

Option	Description
-e file_name	Writes standard errors that occur while server process is running by creating a file named 'server process name_file name'. Standard errors use the fprintf(stderr, format, args) function.
-o file_name	Writes standard outputs that occur while server process is running by creating a file named 'server process name_file name'. Similar to the general printf() function and the standard error output option, this option uses the fprintf(stdout, format, args) function.
-u uid	uid registered in the user file. The three-level security function (service access control) ensures that services will be available if a non-Tmax client such as UCS usermain or gateway requests for an ACL service and its user_id is specified in the CLOPT setting of the SERVER element. For example, if svr_usc in usermain sends a request for an ACL service (SVC_ACL), the user can set the CLOPT setting of svr_ucs to CLOPT="-u user_id" in the SERVER element.
-r	Option to prevent the RESTART value from increasing when a server process is forced to restart after a service times out. Applied only when RESTART is set to Y. In a UNIX platform, the parameter of tpreturn() in tpsvctimeout() must be set to TPEXIT.
-l -L value	For possible values that can be used with the -L option, refer to the description following this table. -L <value> can be set regardless of -l option, but logs are not written to svclog if -l option is not used.
- l	Log service performance results. Log will be saved in the '$ULOGDIR svclog.mmddyyyy' file every 5 minutes. Tmax provides a program named svcrpt for analyzing the log.

Option	Description
- B	In versions earlier than Tmax 4 SP3 Fix#2, if requests were scheduled to one server process simultaneously in a multi CLH environment, CLH Queue Timeout does not occur which delays request processing. This issue has been resolved. However, an exception must be implemented for batch processing. Also note that if the [–B] option is specified in the CLOPT setting of the SERVER element and requests are scheduled to the server process, Queue Timeout is ignored.
- X	The ORA-24761 error occurs when processing a query for a previously rolled back transaction in Oracle. If the user ignores the error and continues processing the query, it is processed as a local transaction resulting in an integrity issue. This problem is caused by user code, but the [–X] option can be used to restart the server process if the result of xa_end() is not XA_OK. The XA server resets an XA channel if xa_end() fails in tpreturn(). The [–X] option can be used to terminate the server process after outputting a fatal error message (service code: CSC5608) when xa_end() fails.
- c	In the previous versions of Tmax, if a date is modified while the server is running, a standard output file of userlog is not deleted even if the delete command is executed. In the current version, the [-c] option can be used to close and delete a log file by checking for date changes.
- q	Used for RDPMT server (SVRTYPE=REALSVR_MT) to move data from the TPSENDTOCLI queue to the WRITE queue when executing tpflush(). It is not recommended for use because it has no memory limit.
- m	If the memory used by user server program exceeds the size set in the -m option, terminate and restart the server program. (unit: Byte) When the server restarts, TMM outputs "sever closed due to MAX_USE_COUNT or MEM_USAGE(CLOPT:-m)".
-x	Used with the TARGET option to request for a service using a substitute service name by using "-x existing service name: the service name to replace". Use a comma (,) to replace multiple service names. If an undeclared service name or duplicate service names are configured, an error can occur while a server is starting up.
-J seconds	Maximum time to wait before restarting the server process after xa_open fails. It is not counted for MAXRSTART.

Option	Description
-i	Determines actions of tp_sleep() and tp_usleep() called by usermain() of UCS servers. Basically, if this option is specified, TMM events are detected, and if not, the events are not detected. usermain() uses tpschedule() to process service requests and handle events such as tmdown. If usermain() uses tp_speep() that waits for a CLH event, calling tpschedule() can be delayed. While usermain() waits because of tp_sleep(), it does not detect TMM events even when a request such as tmdown is received. However, if this option is used, usermain() detects a TMM event and tpschedule() can process the event. If the return value of tp_sleep() is 2 or higher in application logic, tpschedule() must be called because the return value means that a TMM event occurred.

- The following is an example of using the -x option.

```
*SERVER
svr2  SVGNAME = svg1, MIN = 1
svr2_1  SVGNAME = svg1, MIN = 1, TARGET = "svr2",
       CLOPT="-x TOUPPER:TOUPPER1,TOLOWER:TOLOWER1"

*SERVICE
TOUPPER  SVRNAME = svr2
TOLOWER  SVRNAME = svr2
TOUPPER1 SVRNAME = svr2_1
TOLOWER1 SVRNAME = svr2_1
```

- The following is an example of using the -e and -o options.

```
 CLOPT = "-e err1 –o out1 -- abc"
```

Option	Description
-e err1	Save standard errors, which occur while server process(tpcals) is running, to the tpcals_err1 file in the ULOGDIR directory.
-o out1	Save standard output, which occur while server process(tpcals) is running, to the tpcals_out1 file in the ULOGDIR directory.

- The following shows how to set the value with -L option of the CLOPT setting.

```
*SERVER
svr1     CLOPT = "-l –L SL_DFLT", ULOGDIR = "/EMC01/tmax/log/ulog"
svr2     CLOPT = "-l –L SL_NODE", ULOGDIR = "/EMC01/tmax/log/nlog"
svr3     CLOPT = "-l –L SL_SVG", ULOGDIR = "/EMC01/tmax/log/glog"
svr4     CLOPT = "-l –L SL_SVR", ULOGDIR = "/EMC01/tmax/log/slog"
```

Value	Description
SL_DFLT	Same as default.
SL_NODE	Create svclog in the ULOGDIR of NODE element.
SL_SVG	Create svclog in the ULOGDIR of SVRGROUP element. If there's no ULOGDIR setting, it is created in the ULOGDIR of NODE element.
SL_SVR	Create svclog in the ULOGDIR of SERVER element. If there's no ULOGDIR setting, it is created in the ULOGDIR of NODE element.

- If the -r option is set, the following log is recorded in SLOG, and the server restart count is not affected.

```
(W) SVR3032 service timeout error : SVC25 [SVR0403]
(I) SVR3022 SVR (svr25) is down due to tpreturn(TPEXIT) at svc timeout handler. [SVR0305]
(I) TMM0211 General Information : server closed due to TIMEOUT : SVR, pid = 15040
[TMM0161]
(I) TMM3004 SVR (svr25) is restarted the 5th time [TMM0149]
```

◦ MACRO Commands1

If the following macro is defined in the SERVER element, the file name is determined at server startup.

MACRO	Description
$(SVR)	Server name
$(SVRI)	Server index
$(SPRI)	Server process index
$(SPRMIN)	MIN field of the SERVER element
$(SPRMAX)	MAX field of the SERVER element
$(SPRN)	Server process sequential number (0 - $(SPRMAX))
$(DATE)	Date of the MMDDYYYY format
$(TIME)	Time of the HHMMSS format
$(PID)	Process ID

◦ MACRO Commands2

If the following macro is defined in the SERVER element, the file name is determined immediately before tpsvrinit(), tpsvrdone() and the service functions are called for TCS server. For UCS server, It is determined immediately before tpsvrinit() and tpsvrdone() are called and tpschedule() is returned.

MACRO	Description
$(CDATE)	Date in MMDDYYYY format
$(CTIME)	Time in HHMMSS format
$(CYEAR)	Year in YYYY format
$(CMONTH)	Month in MM format (01 ~ 12)
$(CMONTHS)	Month in "Jan, Feb, ... Dec" format
$(CDAY)	Day of the month, DD format (01 ~ 31)
$(CWDAY)	Day of the week, D format (1 ~ 7)
$(CWDAYS)	Day of the week, "Mon, Tue, ... Sun" format
$(CYDAY)	Day of the year, DDD format (001 ~ 366)
$(CHOUR)	Current hour, HH format (00 ~ 23)
$(CMINUTE)	Current minute, MM format (00 ~ 59)
$(CSECOND)	Current second, SS format (00 ~ 59)

    - This MACRO is used to generate different types of log files (for example, by date).

      Example MACRO setting:

      ```
      CLOPT="-o $(SVR).$(DATE).out –e $(SVR).$(DATE).err"
      ```

      A log file will be created in the directory specified by the ULOGDIR field of [server name].03312001.out and [server name].03312001.err files. To modify a file name of standard error or standard output, use an absolute path name like '-e /usr/tmax/log/ulog/test.log'. The 'abc' option after '--' is passed to the tpsvrinit() function of the server program as argv[1] for the user.

- MIN = numeric
    - Range: 1 ~ MAX_INT
    - Default: 1
    - Number of server processes that is created when Tmax starts up.
    - Unlike the conventional client/server model where one server process is loaded per client, Tmax manages and maintains a set number of active server processes. The CLH checks the status of server processes whenever a client issues a request and assigns the request to idle server processes to maximize system performance.
    - Since this value determines the default number of active server processes, it is important that the administrator sets MIN to a value appropriate for the system. If the number of server processes available for handling frequently requested services is too low, the wait time for processing the request will increase and system performance will decrease. On the other hand, if an excessive number of server processes are idle, system resources will be wasted and the performance of the system will decrease.

- MAX = numeric
  - Range: 1 ~ MAX_INT
  - Default: 1
  - Maximum number of server processes that can be generated on the server.

    While MIN specifies the number of server processes that are initially loaded, MAX sets the maximum number of server processes that can be loaded to handle requests. This value includes the total number of active server processes at any one time.

    Initially as many server processes as MIN are loaded, and additional processes can be added up to the MAX value if load increases as client requests increase. Since this value has a strong impact on the system performance, care must be taken in setting the value. If a forked server process is used, set MAX to 0.

- CONV = Y | N | B
  - Default: N
  - Option to use a conversational communication method.
  - There are three types of server communication methods in Tmax: synchronous, asynchronous and conversational. Services that use conversational communication must be registered with a server that is set to conversational communication. If set to a value other than "Y", the server can only facilitate synchronous or asynchronous communication.
  - If set to CONV=B, the server process becomes the scheduling process. B represents BOTH, which supports both conversational (Y) and non-conversational (N) types. It can also be used for inter-domain communication through the domain gateway.
- ASQCOUNT = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0
  - Number of accumulated requests in the queue that automatically activates new server processes.

    When the number of queued requests exceeds this value, additional server processes will be activated, up to the value set in MAX. If not set, additional server processes are not automatically activated.

  - If the number of server processes exceeds the MIN value, the status of the server processes are monitored and those that have been idle a given period of time are terminated to reduce the count. This option is not applied to POD type server processes.
- MAXQCOUNT = numeric
  - Range: 1 ~ MAX_INT
  - Default: -1
  - If not set, the default value is used and MAXQCOUNT is not used.
  - Maximum number of requests that can be queued before additional requests are rejected by the server.

- If the number of requests exceeds the value set in MAXQCOUNT, incoming requests will not be placed in the queue and an error message (tperrno = 26) is sent immediately to the requesting client. MAXQCOUNT is useful for systems that process large volumes of service requests at peak times, such as banks and public offices.

- SVRQTIMEOUT = numeric
  - Range: -1, 0 ~ MAX_INT
  - Default: -1
  - Dequeues expired messages. Can be set for each server. Too many messages can be queued because of host and network issues. In this case, this item can be used to notify the queue status for a client to handle the status.
  - The following are available values.

Value	Description
-1	The CLHQTIMEOUT value in the NODE section is used.
0	Does not dequeue expired messages.
1 ~ MAX_INT	Dequeues expired messages.

- SVRTYPE = STD | UCS | CUSTOM_GATEWAY | STD_DYN | UCS_DYN | REALSVR | REALSVR_MT | EXTSVR | STD_MT
  - Default: STD
  - The following are the service types processed by the server.

Value	Description
UCS	While general service processes handle requests and send result when they occur, UCS server processes implement main() so that data can be sent to each user without a request. The stock quote service is an example of a UCS server process.
REALSVR	REALSVR processes outperform UCS servers in terms of process usage and processing speed when small amounts of data are frequently sent to a large number of clients (more than 10 times per second). Only one RDP server can exist on each node.
REALSVR_MT	Supports multithreading.
CUSTOM_GATEWAY	Use with gateways for an external link such as Hostlink or TN3270 gateway.
EXTSVR	Non-Tmax processes can use server functions registered in the Tmax configuration file. For details about the configuration file, refer to EXTPORT and EXTCLHPORT in "3.2.2 NODE".

Value	Description
STD_DYN, UCS_DYN	Can dynamically increase a MAX value specified in CFL of the SERVER element. Set STD_DYN in TCS, and UCS_DYN in UCS. Only these server types can dynamically modify a MAX value by using the set command of tmadmin. Set to STD_DYN or UCS_DYN in SVRTYPE of the SERVER element. If change in SPRI is checked using the si command after modifying MAX, SPRI order will be out of sequence. When the MAX value is modified, a SPRI that was already used on a server can be used on a different server. <Limitation> This option is not supported in FD_SET = 16384 version. MAX cannot be less than MIN. To reduce the MAX value, SPRs that will be removed must be in the down state. For example, if initially SPRIs are 8193, 8194, 8195, and 8192, then 8195 and 8192 must be shut down in order to reduce MAX to 2.
STD_MT	Supports Multi Thread/Multi Context functions. A thread pool is configured internally and each thread provides services. Hence, a server process can handle multiple service requests concurrently. Also, a user can create additional threads at the application level in addition to the thread pool provided by the library. Service thread context can be shared in the newly created thread. Requires MINTHR, MAXTHR, STACKSIZE, and CPC configuration.

- CPC = numeric
  - Range: 1 ~ 128
  - Default: 1
  - Number of parallel communication channels between a UCS server and a CLH process. In situations where a single channel cannot handle the amount of requests, multiple communication channels can be used in parallel to increase processing speed.
  - Only applies to UCS, UCS_DYN, CUSTOM_GATEWAY, REALSVR, REALSVR_MT, STD_MT server types.
  - Required for STD_MT server type.

    This value is related to the MAXCLH and MAXTHR items. MAXCLH x CPC value must be equal to or greater than the MAXTHR.

    For instance, if MAXCLH is 1 and MAXTHR is 8, CPC must be at least 8, and if MAXCLH is 2 and

MAXTHR is 8, CPC must be at least 4. This is because the maximum number of service requests that can be processed simultaneously by one server process is CPC x MAXCLH. Hence, MAXTHR does not need to be greater than CPC x MAXCLH. If not set, the value, MAXTHR / MAXCLH, is used.

- INBOUNDCPC = numeric
  - Range: 1 ~ 128
  - Default: 1 if CPC is not set or the CPC value if CPC is set
  - The number of parallel channels from CLH processes to UCS server processes.
  - Available only for UCS and UCS_DYN.
  - If not set, the number of channels from CLH to UCS is the same as that of channels from UCS to CLH. Setting this option can reduce waste of resources and does not directly affect performance.
- MINTHR = numeric
  - Range: 0 ~ MAXTHR
  - Default: 0
  - Minimum number of service threads that are initially created when a STD_MT server process starts. Required for STD_MT server.
  - If set to 0, a service request from CLH is processed after a thread is created.
- MAXTHR = numeric
  - Range: 1 ~ 1000
  - Default: 1
  - Maximum number of service threads managed by a thread pool of the STD_MT server process. Required for STD_MT server.
  - Creating and adding threads whenever there is a service request can degrade performance. Create threads only up to the maximum number, which must be set after careful consideration.
  - A total of MAXTHR + 1 threads including the main thread can be running. The main thread does not handle service requests but manages service threads and requests. If the value is 1, it is recommended to use a general server library since only one service thread is created.
  - The MAXTHR item is related to the CPC and MAXCLH items. Refer to the **'CPC'** item for detailed information.
- STACKSIZE = numeric
  - Range: 0 ~ MAX_INT
  - Default: 0 (Unit: KB)
  - Stack size of the service threads in the STD_MT server process. If not set, the default size of the OS is used.
- RESTART = Y | N
  - Default: Y

- Option to reload the server if it is terminated abnormally.

- MAXRSTART, GPERIOD
  - Fault tolerance related options. For more information, refer to Fault Tolerance Settings.
- TARGET = literal
  - A server name is a name of a server executable file, and TARGET is used to redefine a server process name. Server process refers to a process that executes the server executable file in APPDIR.
  - By setting TARGET as in the following, tmadmin can check for the actual server name and server process name by using ps.

    Example) SVGNAME = svrgrp, MIN =1, MAX = 1, TARGET = svr2

```
/user1/starbj/tmax/sample/server> tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---
$$1 tmaxs1 (tmadm): st -p
CLH 0:
------------------------------------------------
svr_name  svgname  spr_no  status  count  avg  svc
------------------------------------------------
SVR      svrgrp     32     RDY      0   0 .00  -1

/user2/starbj/tmax/sample/server> ps
starbj 28702 1 0 11:13:03 pts/8 0:00 cll -b 28700
starbj 28704 1 0 11:13:03 pts/8 0:00 svr2
 -b 28700 -S SVR -s SVR -d -1
starbj 28701 1 0 11:13:03 pts/8 0:00 tmm -b 28700
starbj 28703 1 0 11:13:03 pts/8 0:00 clh -b 28700
```

  - Example configuration file for TARGET.

```
*SERVER
original        SVGNAME = svg0,
copied1         SVGNAME = svg0,
                TARGET  = "original",
                CLOPT   = "-e $(SVR).err -o $(SVR).out -x ORIGINAL:COPIED1"
copied2         SVGNAME = svg0,
                TARGET  = "original",
                CLOPT   = "-e $(SVR).err -o $(SVR).out -x ORIGINAL:COPIED2"
*SERVICE
ORIGINAL        SVRNAME = original
COPIED1         SVRNAME = copied1
COPIED2         SVRNAME = copied2
```

    On copied1 and copied2 servers, server processes are initiated with the name 'original'. In general, a server name and a server process name are the same in Tmax, but if TARGET is used, the names are different. Thus, when Tmax starts up, three processes named 'original' starts.

    If a service request is sent to COPIED1 and COPIED 2, the target is changed to original by [-x]

option in the CLOPT setting. A client can request a service to COPIED1 and COPIED2 separately. The server and service count are also incremented when checking in tmadmin.

TARGET is useful when checking a separate COUNT for each object that calls the tpcall within the same program.

If using the tmdown -S command to modify the original server, only the original server shuts down and its version may not match with versions of the copied servers. Thus, it is recommended to shut down all servers that use the TARGET when executing tmdown.

- AUTOTRAN = Y | N | B
  - Default: N
  - If a service is dynamically added to Tmax system using 'mksvr' while it is running, by default this value is set to 'N'.
  - The AUTORAN setting in the SERVER element will automatically load the transactions for services that have been added dynamically. This affects the setting of dynamically added service. For more information, refer to AUTOTRAN in "3.2.3. SERVICE".
  - If AUTORAN is defined in both the SERVICE and SERVER elements, the SERVICE element setting is used.
  - Option B is used for compatibility with AUTOTRAN in old versions (~3.8.8). It works in the same way as Y, except that the transaction operates as a local transaction instead of a global transaction.
    - Versions that use outdated AUTOTRAN: Versions earlier than 3.8.9

      If the XA server is called without tx_begin, a transaction is automatically started (Default: AUTOTRAN=Y), and it operates as a local transaction.

      For an external call, transactions are not linked to each together, and it works like a TPNOTRAN call.

    - Versions that use current AUTOTRAN: 3.8.9 version or later

      If AUTOTRAN is set to Y, global transactions are automatically started, and if there is an external call, they are automatically linked. Since the default value has been changed to 'N', the value must be explicitly set after an upgrade.

- SCHEDULE = FA | RR
  - Default: FA
  - Processing order of transactions among server processes when the MIN value of the server is greater than 2.
  - Setting SCHEDULE

Value	Description
FA(First Available)	Allocate tasks to the process with the smaller number (higher priority).
RR(Round-Robin)	Allocate tasks evenly among the processes.

- LIFESPAN = IDLE_DOWN | IDLE_0 | IDLE_sec
  - Default: IDLE_DOWN
  - Lifespan of the following additional server processes (more than MIN value).
    - POD server processes
    - Additional server processes that have been activated through ASQCOUNT
    - Additional server processes that have been manually activated by the tmboot -s command
  - Server processes started by POD (when MIN=0)
    - MIN = 0, MAX >= 1, or LIFESPAN=IDLE_DOWN

      No server processes are loaded (MIN=0) when Tmax starts. If a client issues a service request, a server process is loaded (count=1) and it remains active until a user terminates it.

    - MIN=0, MAX>=1, LIFESPAN=IDLE_0

      When a client issues a service request, a server process is loaded (count=1). After processing the request, the server process is automatically terminated if there are no requests to process.

    - MIN=0, MAX>=1, LIFESPAN=IDLE_sec

      When a client issues a service request, a server process is loaded. After processing the request, it waits for further requests for the specified time period (sec). If no new requests are issued during the time period, the server process is terminated.

  - Server processes started through ASQCOUNT:
    - MIN=1, MAX>=2, ASQCOUNT=2, LIFESPAN=IDLE_DOWN

      One server process is loaded (MIN=1) when Tmax starts. If there are 2 or more client requests in the queue, another server process is started to handle the next request. The newly added server process remains active until a user terminates it.

    - MIN=1, MAX>=2, ASQCOUNT=2, LIFESPAN=IDLE_0

      If there are 2 or more client requests in the queue, another server process is started to handle the next request. This additional server process is immediately terminated if there are no requests waiting to be processed which leave just one active server process.

    - MIN=1, MAX>=2, ASQCOUNT=2, LIFESPAN=IDLE_sec

      If there are 2 or more client requests in the queue, another server process is started to

handle the next request. After processing the request, it waits for further requests for the specified time period (sec). If no new requests are issued during the time period, the server process is terminated.

- DUMMY = Y | N
  - Default: N
  - Temporarily uses "__dummy" as the server name.
  - If DUMMY is set to Y in the SERVER element, the server process is not started during tmboot.
  - This item is useful when a dummy server setting, such as Domain Gateway Cousin, is required. For more details, refer to Domain Gateway COUSIN Setting.
  - If not set, the following error occurs during tmboot.

```
(F) BOOT0014 exec error : /data1/tmaxkjh/tmax/appbin/tbrtest [BOOT0029]: No such file or
directory
```

- AUS = Y | N
  - Default: N
  - Allow Unregistered Services
  - When a server is started after a service has been deleted from the configuration but not from the application (not rebuilt), the server is restarted as many times as MAXRSTART.
  - If set to AUS=Y in the SERVER element, the server is started ignoring this setting.
- CTX_EREPLY = Y | N
  - Default: Y
  - Only used for a UCS server. Other server types cannot use this option.
  - If a server process is terminated by tmdown -s, tmdown -s -i, or abnormal shutdown before tprelay() is called and after the UCS server calls tpsavectx() and saves the client information, the caller does not receive a response for the request. If set to 'Y', when the server process is being terminated an error message is sent to the client for requests that have not been handled by tprelay().
  - If the CTX_EREPLY option is set to default, when the server process is being terminated an error response is sent to the caller for requests that are being processed. This way, other processes on the server are not affected. This setting operates normally when the client information (CTX_T) created by tpsavectx() call is only managed internally in the server process.

    But in an environment where multiple processes share the client information through IPC, when a specific process is abnormally terminated an error message may still be sent even if another process can execute tprelay(). Set to 'N' to prevent an error message from being sent.
- MULTICLH = Y | N
  - Default: Y

    - If set to Y, a server process can concurrently receive requests from multiple CLHs when MINCLH >= 2. A server process handles concurrent service requests on a first-come-first-served basis. After the process completes the current request with the tpreturn call, it processes a request from another CLH.
    - If set to N, each server process can receive and handle service requests from a CLH. This prevents multiple CLHs from simultaneously requesting services to one server process. Each server process receives requests from an assigned CLH even though overload occurs on other CLHs which is inefficient in terms of performance.
    - Precautions if set to N
        - If the MIN value of the server is set, the value must be MINCLH or greater.
        - If the MAX value of the server is set, the value must be MAXCLH or greater. The MAX value does not need to be a multiple of the MAXCLH value.
        - Each server process is assigned a clh number, which it will be responsible for, in the ascending order of SPRI number. In other words, the first server process is responsible for the first CLH, the second one for the second CLH, and so on. If the server process order number exceeds MAXCLH, the assignment restarts from the first CLH.

          If the MINCLH and MAXCLH settings are different and CLH has not started when MAXCLH is reached, the server processes assigned to the inactive CLHs cannot receive any requests. Hence, when a server process is started through tmboot or tmm, server processes with SPRI assigned to an inactive CLH are not started. To start server processes with SPRI assigned to a CLH, the CLH must be running.

        - If the MINCLH and MAXCLH settings are different and an additional CLH is started, the tmboot command must be issued to start the server processes assigned to the CLH. Otherwise, the TPENOREADY error is returned when a service is requested from a client connected to the CLH.
        - When this setting is changed from the default value to 'N', the MIN and MAX settings must also be checked and adjusted to appropriate values. If MULTICLH is set to 'N' without changing the MIN and MAX settings, the number of processes that each CLH can schedule is reduced and may cause requests on a particular CLH to accumulate in the queue.
- TMAPM = Y | N
    - Default: Y
    - Option to use tmapm for each service.
    - As a user function, it is recommended not to use tmapm when using a user defined function (tpsetsvctimeout()) to readjust the timeout value.
- MAC = Y | N
    - Default: N
    - MAC(Master Alive Check) is used to terminate a server when TMM is terminated to prevent it from becoming a zombie process.
    - If set to MAC=Y, the server is terminated when TMM is terminated abnormally.
- ROC = Y | N

    - Default: N
    - ROC(Reconnect Outbound CPC) is used to reconnect to CPC when a server is disconnected from CLH.
    - If set to ROC=Y, reconnection is attempted when CLH is alive and the socket is disconnected.
- MAX_USE_COUNT = numeric
    - Default: 0
    - Number of times that the server calls a service before restarting the server.
    - If this setting triggers a server restart, TMM outputs "sever closed due to MAX_USE_COUNT or MEM_USAGE(CLOPT:-m)."

Re-definable Items

- LOGLVL = string
    - Log level of the server.
- ULOGDIR = literal
    - Path name of the directory where user messages are stored.
    - Users like Tmax application programmers can easily log messages for debugging and error and warning messages by using the userlog() function. The messages that are outputted by the userlog() function are stored in the ULOGDIR directory. Files called 'ulog_date' are created in the directory, and the messages are logged by date in each file.

      If not set, messages are stored in the log/ulog directory under the directory specified in TMAXDIR. The system must exist in the specified path.

For details about the userlog function, refer to *Tmax Application Development Guide*.

- SVCLOG_FORMAT = literal
    - Allows for users to modify svclog items.

Example

The following is an example of using the SERVER element.

```
*SERVER
svr1     SVGNAME = svg1,
         CLOPT = "-e err1 -o out1 -- svr1 1",
         MIN = 1,MAX = 10, CONV = N

svr2     SVGNAME = svg1,
         CLOPT = "-e err1 -o out1 -- svr1 2",
         MIN = 1, MAX = 10,
         CONV = N,
```

```
        SVRTYPE = UCS,
        CPC = 5
```

3.2.5. SERVICE

Services supported by Tmax must be registered in the SERVICE element. Even if the servers and services have been properly configured and are running, services cannot be processed until they have been added to the configuration file.

The following items can be defined for each service:

- Server(s) that process the service
- Priority level of the service
- Timeout value for service processing
- Routing information

The following is the syntax for the SERVICE element.

```
[DEFAULT : ]
Service Name        SVRNAME = server-process-name or gateway-name
                    [FUNCNAME = module-name,]
                    [PRIO = priority-value,]
                    [ROUTING = rout-name,]
                    [SVCTIME = timeout-value,]
                    [EXPORT = (Y)|N,]
                    [AUTOTRAN = Y|(N)|B,]
                    [TXTIME = transaction-timeout-value]
```

Required Items

- Service Name = string
  - Size: Up to 63 characters
  - Function name (service routine name) of the service. The name must be unique within the SERVICE element. In a multi-domain environment, the Service Name must be unique across all domains in the system.
- SVRNAME = string / literal
  - Size: Up to 63 characters
  - Name of the server that processes the service.
  - Use the executable file name of the server program that contains the service routine. The server must be defined in the SERVER element. If the server name includes any special characters, this must be wrapped in double quotes as in "Server Process Name". If multiple services are provided by a single server process, the same server process name can appear in the SERVICE element multiple times.

Optional Items

- Default: item = value, ...
  - Refer to DEFAULT in "3.2.2 NODE".
- FUNCNAME = string
  - File name of the module when the service name is different from the module name.
- PRIO = numeric
  - Range: 1 ~ 100
  - Default: 50
  - Priority level of the service. A service with a higher number is given higher priority. For example, a service with a PRIO value of 100 has the highest priority.
- ROUTING = string
  - Data routing setting. For more information, refer to Load Balancing Settings.
- SVCTIME = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0
  - Fractional numbers can be set.
  - If not set, services wait indefinitely (default value).
  - Max time allowed for service processing. The service processing must be finished within this time period.

    If the processing time exceeds SVCTIME, the server process will stop processing the service and will send an error message to the client. While BLOCKTIME in the DOMAIN element starts from the time that the client issues the requests (for example, through tpcall() or tpacall()), SVCTIME starts from the time when the CLH passes the request to the server process and the server process begins to process the service. If SVCTIME is set to 0, time limit is infinite.

  - If service times out while using a database session, the next service cannot be processed since the session cannot be forced to disconnect from the database. Tmax resolves such problem in the following way. If service timeout occurs, a user can restart the server by using TPRETURN(TPEXIT, ...) in the tpsvctimeout routine which causes the MAXRSTART count in the SERVER element to be incremented.

    It is also recommended to use the XA mode on Windows for application development. To use non-XA mode, set a sufficient number for RESTART count. SVCTIME only includes the processing time and excludes the time in the queue.

    For more information about tpsvctimeout, refer to *Tmax Reference Guide*.

- EXPORT = Y | N
  - Default: Y

- Option to process service requests from another domain in a multi domain environment.
- If EXPORT is set to N and a request is sent from another domain via a gateway, a TPESECURITY error occurs.

- AUTOTRAN= Y | N | B
  - Default: N
  - Option to automatically launch a global transaction for a service request that is received while in non-transaction mode.
  - It is different from AUTOTRAN in versions earlier than Tmax 3.8.8.

    In the old versions, AUTOTRAN is set to Y by default, and a local transaction begins without executing tx_begin. In the new version, Tmax 3.8.8, AUTOTRAN is set to N by default, and a transaction is not automatically started.

    If not set, tx_begin must be used to start a transaction. If set to Y, when the current service requests for another service they are grouped into a global transaction. If set to Y, and tx_begin is used to start a transaction, the TPESVCERR error occurs.

  - The following is an example of the configuration file and the tmadmin tool:

    < config.m>

```
*SVRGROUP
svg2            NODENAME = "tmax1"

### tms for Oracle ###
svg3            NODENAME = "tmax1",    DBNAME = ORACLE,
                OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm=60",
                TMSNAME  = tms_ora

*SERVER
svr2            SVGNAME = svg2
### servers for Oracle sample program###
fdltest         SVGNAME = svg3

*SERVICE
### services for fdltest ###
FDLINS          SVRNAME = fdltest
FDLSEL          SVRNAME = fdltest
FDLUPT          SVRNAME = fdltest
FDLDEL          SVRNAME = fdltest
TOUPPER         SVRNAME = svr2
```

    <tmadmin>

```
tmax@tmax1;/user/tmax/tmax/config>tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmax1 (tmadm): cfg -s
   ------------------------------------------------
   svc_name    prio(pr)  autotran  svctime(st)  routno  svrname      svgno
```

```
   ------------------------------------------------
FDLINS          50      YES         0           -1      fdltest         2
FDLUPT          50      YES         0           -1      fdltest         2
FDLSEL          50      YES         0           -1      fdltest         2
FDLDEL          50      YES         0           -1      fdltest         2
TOUPPER         50       NO         0           -1        svr2          0

$$2 tmax1 (tmadm): q
ADM quit for node (tmax1)
```

- TXTIME = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0
  - For more information, refer to TXTIME in "3.2.1 DOMAIN".

Example

The following is an example of using the SERVICE element.

```
* SERVICE
TOUPPER         SVRNAME = svr1,
                PRIO = 1,
                SVCTIME = 60
TOLOWER         SVRNAME = svr1,
                PRIO = 2
```

3.2.6. GATEWAY

Unlike the DOMAIN, NODE, SVRGROUP, SERVER, and SERVICE elements, which must be configured, the GATEWAY element is optional and is only required for multi-domain systems.

If a system consists of multiple nodes in a single domain, it can be difficult to manage the system and the large volume of inter-node communication may degrade system performance. In this situation, it would be preferable to divide the nodes into separate domains and link the domains together into a single system through a gateway. Multiple GATEWAY elements can be defined in a domain, but each node can only access one gateway.

The gateway allows a client request to be passed to domains that it is not directly connected to. Multiple nodes can be assigned to a single domain in the GATEWAY element. A gateway can be assigned to only one node. Once a GATEWAY element has been configured, appropriate processes (based on gateway type) are loaded into the system.

The following items must be configured in the GATEWAY element.

- Node where the gateway process runs on
- TCP/IP information for gateway communications

- Gateway type

The following is the syntax for the GATEWAY element.

```
[DEFAULT : ]
Gateway Name        NODENAME = node-name,
                   PORTNO = port-number,
                   RGWADDR = remote-ip-addr,
                   RGWPORTNO = remote-port-number,
                   GWTYPE = {TMAX | TMAXNONTX | SNACICS | OSITP | JEUS
                          | JEUS_ASYNC | TUXEDO | TUXEDO_ASYNC | WSGW | XAGW }
                   [MAXINRGW = numeric(32),]
                   [CPC = channel-number,]
                   [COUSIN = gateway-name,]
                   [BACKUP = gateway-name,]
                   [BACKUP_RGWADDR = remote-ip-addr,]
                   [BACKUP_RGWPORTNO = remote-port-number,]
                   [BACKUP_RGWADDR2 = Backup-Tuxedo-ipaddr2,]
                   [BACKUP_RGWPORTNO2 = Backup-Tuxedo-domaingw-portno2,]
                   [BACKUP_RGWADDR3 = Backup-Tuxedo-ipaddr3,]
                   [BACKUP_RGWPORTNO3 = Backup-Tuxedo-domaingw-portno3,]
                   [LOCAL_PROTOCOL = "IPV4" | "IPV6" | ",SDP",]
                   [RGW_PROTOCOL = "IPV4" | "IPV6" | ",SDP",]
                   [RGW_B1_PROTOCOL = "IPV4" | "IPV6" | ",SDP",]
                   [RGW_B2_PROTOCOL = "IPV4" | "IPV6" | ",SDP",]
                   [RGW_B3_PROTOCOL = "IPV4" | "IPV6" | ",SDP",]
                   [LOCAL_IPV6 = Y|(N),]
                   [RGW_IPV6 = Y|(N),]
                   [RGW_B1_IPV6 = Y|(N),]
                   [RGW_B2_IPV6 = Y|(N),]
                   [RGW_B3_IPV6 = Y|(N),]
                   [TIMEOUT = second,]
                   [DIRECTION = (BIDIR) | IN | OUT,]
                   [CLOPT = string,]
                   [RESTART = (Y)|N,]
                   [MAXRSTART = numeric,]
                   [GPERIOD = numeric,]
                   [LOAD = numeric,]
                   [PTIMEOUT = 1~MAXINT/2,]
                   [PTIMEINT = 1~MAXINT/2,]
                   [GWCHKINT = numeric,]
                   [GWCONNECT_TIMEOUT = numeric,]
                   [NLIVEINQ = numeric,]
                   [TRB = nodename]
```

Required Items

- Gateway Name = string
  - Size: Up to 63 characters
  - Logical name of the gateway. Must be unique within the GATEWAY element.
- NODENAME = literal

- Size: Up to 255 characters
- Name of the node where the gateway process runs on.
- Node name must already be defined in the NODE element.

- PORTNO = numeric
  - 'listen' port number used by local gateway processes. Enables a remote gateway process to connect to a local gateway process. The port number must be configured in the RGWPORTNO item. Port numbers less than 1024 must not be used since they are reserved for system use.
  - Since port numbers are essential for communication between gateways, make sure not to use the port numbers that are used by other system functions.
- RGWADDR = literal
  - Size: Up to 255 characters
  - Name or IP address of the node where the remote gateway process, to which the local gateway wants to connect, is running on.
  - If using the node name, it must be registered in the "/etc/hosts" file. The local gateway processes use both the IP address in RGWADDR and the port number in RGWPORTNO to connect to a remote gateway.
- RGWPORTNO = numeric
  - Port number through which the local gateway process connects to a remote gateway process.
  - This number is a listen port number of a remote gateway process. Port numbers less than 1024 must not be used since they are reserved for system use.

Check that the RGWPORTNO is available for gateway communication.

- GWTYPE = string
  - Type of the remote gateway to which the local gateway process is to connect.
  - The following values are available.

Value	Description
TMAX	Tmax is the remote system.
TMAXNONTX	Use multiplexing on inter-domain channels to enhance the efficiency of CPC in situations where the remote system is Tmax and inter-domain transactions are used. This improves the response time and decreases the overhead since processes occupy the gateway while waiting for results.
TUXEDO	Tuxedo is the remote system. Communicates with a domain gateway of Tuxedo through a synchronous channel.
TUXEDO_ASYNC	Tuxedo is the remote system. Communicates with a domain gateway of Tuxedo through an asynchronous channel.
JEUS	JEUS is the remote system.

Value	Description
JEUS_ASYNC	Used for asynchronous I/O communication with Async WebT on WAS(Web Application Server). While a remote system must be JEUS if GWTYPE is set to JEUS, it can connect to other Web Application Servers if set to JEUS_ASYNC. For more information, refer to *Tmax WebTAsync User Guide*.
WSGW	Can use Tmax services as Web services through bi-directional communication.
XAGW	Used to process data from Tmax XA Library (libtxa.so). Guarantees 2PC between heterogeneous systems.
SNACICS	Available for SNA LU6.2. It is a Hostlink gateway for transaction communication.

Optional Items

- Default: item = value, ...
  - Refer to DEFAULT in "3.2.2 NODE".
- MAXINRGW = numeric
  - Range: 1 ~ 128
  - Default: 32
  - Number of channels that can be used concurrently in the gateway.
  - This is only used when TMAXNONTX is configured in GWTYPE. If not set, the gateway will only connect to and receive service requests from up to 32 domains.
- LOCAL_PROTOCOL = "IPV4" | "IPV6" | ",SDP"
  - Default: "IPV4"
  - Protocol used when a local gateway listens. If multiple protocols can be used, they are set by using a comma (,) as a delimiter.

Value	Description
IPV4	Uses the IPv4 protocol to wait for connection requests.
IPV6	Uses the IPv6 protocol to wait for connection requests.
SDP	Uses InfiniBand Socket Direct Protocol (SDP). Can be used along with IPV4 or IPV6.

- RGW_PROTOCOL = "IPV4" | "IPV6" | ",SDP"
  - Default: "IPV4"
  - Protocol used when a local gateway connects to a remote gateway. If multiple protocols can be used, they are set by using a comma (,) as a delimiter.

Value	Description
IPV4	Uses the IPv4 protocol to try to establish connections.
IPV6	Uses the IPv6 protocol to try to establish connections.
SDP	Uses InfiniBand Socket Direct Protocol (SDP). Can be used along with IPV4 or IPV6.

- LOCAL_IPV6 = Y | N
  - Default: N
  - Option to use the IPv6 protocol when local gateway listens.

It is recommended to use LOCAL_PROTOCOL instead of this item.

- RGW_IPV6 = Y | N
  - Default: N
  - Option to use the IPv6 protocol when a local gateway uses IPV6 to connect to a remote gateway.

It is recommended to use RGW_PROTOCOL instead of this item.

- CPC = numeric
  - Range: 1 ~ 128
  - Default: 1
  - Number of parallel communication channels between CLH process of both domains when using the gateway.
  - In situations where high volumes of data need to be passed through the gateway process, parallel communication using multiple channels can speed up the processing. For gateways set as TMAXNONTX, it is sufficient to set CPC to 2 or 3.
- TIMEOUT = numeric
  - Range: 1 ~ MAX_INT
  - Default: 0
  - If not set, services wait indefinitely (default value).
  - Max time allowed for inter-domain tpcall / tpacall.

    If set to the default value, time limit is infinite. If the delay is less than 30 seconds, TIMEOUT does not occur since the gateway checks timeout every 30 seconds which is different from a server.
- COUSIN = literal
  - Size: Up to 8000 characters

- Names of related gateways when a service needs to be distributed through multiple gateways (load balancing). Can use the names of gateways on the same node or different nodes. Both COUSIN and BACKUP items can be configured, but they must be set to different gateways.
- The gateways set in COUSIN are in charge of routing, and the gateways set in BACKUP handle failures.
- Can only use the gateway names registered in the GATEWAY element. Server groups cannot be used.
- The following is an example of the environment file.

  The gateways, gw1 and gw2, are set to COUSIN. If there is a GWSVC1 service request, it is distributed and processed according to the LOAD configuration.

  < config.m>

```
*SERVICE
GWSVC1       SVRNAME = gw1
GWSVC2       SVRNAME = gw1
GWSVC3       SVRNAME = gw1

*GATEWAY
gw1          NODENAME = node1, GWTYPE = TMAXNONTX, PORTNO = 10001,
             RGWADDR = "192.168.1.1",   RGWPORTNO = 10002,
             LOAD = 1, CPC = 5, CLOPT = "-i",
             COUSIN = "gw1"

gw2          NODENAME = node2, GWTYPE = TMAXNONTX, PORTNO = 10001,
             RGWADDR = "192.168.1.2",   RGWPORTNO = 10002,
             LOAD = 1, CPC = 5, CLOPT = "-i"
```

- LOAD = numeric
  - Refer to LOAD in "3.2.3. SVRGROUP".
- BACKUP = *gateway name*
  - Size: Up to 7999 characters
  - Name of the backup gateway to be used in the event of main gateway failure. It must be a gateway set in the GATEWAY element of the Tmax configuration file.
- BACKUP_RGWADDR = *remote address*
  - Size: Up to 255 characters
  - IP address or node name of the backup remote gateway.
  - If using a node name, it must also be defined in the /etc/hosts file.
- BACKUP_RGWPORTNO = *remote-port-no*
  - Port number of the backup remote gateway.
  - Used for listening. Port numbers less than 1024 must not be used since they are reserved for system use.
  - BACKUP, BACKUP_RGWADDR, and BACKUP_RGWPORTNO settings are all used for fault

tolerance purposes.

Since the port number, BACKUP_RGWPORTNO, is used for communication between gateways, check that it is not used anywhere else.

- RGW_B1_PROTOCOL = "IPV4" | "IPV6" | ",SDP"
  - Default: "IPV4"
  - Protocol used when a local gateway connects to a backup remote gateway. If multiple protocols can be used, they are set by using a comma (,) as a delimiter. The remote gateway's address must be set in BACKUP_RGWADDR and its port number must be set in BACKUP_RGWPORTNO.
Value	Description
IPV4	Uses the IPv4 protocol to try to establish connections.
IPV6	Uses the IPv6 protocol to try to establish connections.
SDP	Uses InfiniBand Socket Direct Protocol (SDP). Can be used along with IPV4 or IPV6.

- RGW_B1_IPV6 = Y | N
  - Default: N
  - Option to use the IPv6 protocol when local gateway connects to a backup remote gateway.
  - Used when connecting to a remote gateway whose address is BACKUP_RGWADDR and the port is BACKUP_RGWPORTNO.

It is recommended to use RGW_B1_PROTOCOL instead of this item.

- BACKUP_RGWADDR2, BACKUP_RGWPORTNO2 = *remote address*
  - Size: Up to 255 characters
  - Used only for Tuxedo gateways (GWTYPE = TUXEDO | TUXEDO_ASYNC). Up to three backup Tuxedo gateways can be assigned by using BACKUP_RGWADDR2-BACKUP_RGWPORTNO2 and BACKUP_RGWPORT3-BACKUP_RGWPORTNO3.
- RGW_B2_PROTOCOL, RGW_B3_PROTOCOL = "IPV4" | "IPV6" | ",SDP"
  - Default: "IPV4"
  - Used only for Tuxedo gateway (GWTYPE = TUXEDO | TUXEDO_ASYNC). Determines the protocol used when connecting to the second or third backup remote gateway. If multiple protocols can be used, they are set by using a comma (,) as a delimiter. The second remote gateway's address must be set in BACKUP_RGWADDR2 and its port number must be set in BACKUP_RGWPORTNO2, and the third remote gateway's address must be set in BACKUP_RGWADDR3 and its port number must be set in BACKUP_RGWPORTNO3.
  -

Value	Description
IPV4	Uses the IPv4 protocol to try to establish connections.
IPV6	Uses the IPv6 protocol to try to establish connections.
SDP	Uses InfiniBand Socket Direct Protocol (SDP). Can be used along with IPV4 or IPV6.

- RGW_B2_IPV6, RGW_B3_IPV6 = Y | N
  - Default: N
  - Option to use the IPv6 protocol to connect to second or third backup remote gateway. Used only for Tuxedo gateways (GWTYPE = TUXEDO | TUXEDO_ASYNC)
  - Used when connecting to a remote gateway whose address and port number are BACKUP_RGWADDR2 and BACKUP_RGWPORTNO2 or BACKUP_RGWADDR3 and BACKUP_RGWPORTNO3 respectively.

It is recommended to use RGW_B2_PROTOCOL and RGW_B3_PROTOCOL instead of these items.

- DIRECTION = BIDIR | IN | OUT
  - Default: BIDIR
  - Only used when GWTYPE is set to TMAX or TMAXNONTX.
  - Options are:

Value	Description
BIDIR	Bi-directional request processing.
IN	Only incoming requests into the domain are processed through the gateway. A TPESECURITY error will be generated if an outgoing request attempts to use the gateway.
OUT	Only outgoing requests from the domain are processed through the gateway. A TPESECURITY error will be generated if an incoming request attempts to use the gateway.

- CLOPT = string
  - Size: Up to 255 characters
  - Gateway command options that can be sent to the gateway when it starts up. Options preceding '--' are used by the system, and options following '--' can be used by the users.
  - Common user options are:
    - [-h Tmax Version Number]

      This option is used to enable connection between a domain using Tmax 4 and later and a domain using older versions of Tmax. The version number of Tmax used by the target

domain must be specified in the configuration file of the domain for Tmax 4 and later (this must be in the format: 'xxxyyzz').

The gateway communicates with the target domain according to the version specified in the configuration file. If the Tmax version of the target domain is set incorrectly, various errors will occur such as service block error. When TYPE=JEUS (JEUSGW), CLOPT must be set to "–h 1".

For Tmax 3.14.4, use the following setting.

```
CLOPT = "-h 031404"
```

- [-c checktime]

  Regularly checks the checktime and attempt to reconnect if disconnected from CLH. If the blocktime is 5 seconds, the connection to CLH is checked every 5 seconds (if there is no request from RGW or CLH) even if -c option is used to set the checktime in GW to 15 seconds.

- [-i]

  Connect to a channel independent from the service request. If not set, it operates as before. The gateway will attempt to reconnect to the remote domain when it is disconnected because of system or network failure in the remote domain.

  In the past version, the gateway waits for GWCONNECT_TIMEOUT and tries to reconnect when there is a request. The new function separates the reconnection attempt from the service request. The gateway does not attempt to reconnect when a request made, but instead it sends a response according to the current status. This option provides faster error response to service requests that occurs when the gateway is disconnected from the remote domain.

  If the gateway is disconnected from the remote domain when a request is made, it immediately sends the TPENOREADY error. After checking that the gateway is disconnected from the remote domain, the TPESYSTEM error is sent for requests that have already been sent to the remote domain. In the past versions, TPETIME error is sent in such situations. When connected to the remote domain, a request is sent. Reconnection is attempted independent of the service request and the smaller value between NLIVEINQ and BLOCKTIME is used as the retry interval. If GWCHKINT is set, it is used as the retry interval.

  For fast response to service requests, it is important to set the value of BLOCKTIME according to the NLIVEINQ. If the BLOCKTIME is too short, the user may receive the TPETIME error before detecting the channel disconnection. And if the NLIVEINQ is too short, connection to the Gateway is attempted too frequently, which may cause the TPETIME error to occur instead of the TPENOREADY or TPESYSTEM error for most requests during network failure.

  Reversely, if NLIVEINQ is too long, it will send the TPENOREADY error for most requests,

but it will take too much time to detect the network failure. For IRT Gateway, this option must be used to independently connect to a channel. This is because when the IRT COUSIN setting is used, reconnection attempts cannot be made since the service request is not sent to the problematic Gateway.

IRT COUSIN is supported for TMAXNONTX, TMAX, JEUS, JEUS_ASYNC, TUXEDO, and TUXEDO_ASYNC gateways only.

- [-R]

  Transaction Recovery function is not supported in Tmax 3.x versions and JTmax, and hence pending transactions may not be processed. To resolve this issue in Tmax 5, the processing method for pending transactions must be configured in advance.

  Options are:

Value	Description
IGN	Does not commit/rollback pending transactions. Thus, DB must process commit/rollback.
COM	Commit all pending transactions.
RBK	Rollback all pending transactions.

- [-r]

  Transaction Recovery function support.

Classification	Description
GWTYPE=JEUS or JEUS_ASYNC	Use the Transaction Recovery function for transaction between WebT-JEUSGW and also between JEUSGW-JTmax. This option is only available for WebT version with Transaction Recovery function support.
GWTYPE=TUXEDO or TUXEDO_ASYNC	Tuxedo domain ID that is transmitted when connecting to Tmax for authentication. If a gateway with a different domain ID tries to connect, an error occurs and access is denied.

Classification	Description
GWTYPE = TMAX or TMAXNONTX	This option is used to enable transaction recovery when establishing a connection between domains that use different versions of Tmax. The Tmax version of the target domain must be specified as 'xxxyyzz'. (For example, Tmax 3.8.15 : 030815) If set, gateway function varies depending on the Tmax engine version used by the target domain. XA related operations are different for each of the following Tmax version divisions. The -r option must be used if the Tmax version of the target domain is in a different division from the current version. • Division 1: 2.0 ~ 3.8.7 • Division 2: 3.8.8 ~ 3.8.13 • Division 3: 3.8.14 and later

- [-a]

  Domain ID that is transmitted when connecting to Tuxedo. The ID must be configured in the Tuxedo configuration file.

- [-l]

  Does not write logs when a gateway starts, which reduces the volume of logs. If this option is not specified and a gateway often restarts, the volume of logs becomes too large.

- [-p configuration file path]

  The results and messages of handling a domain gateway requests are logged for each case defined in the specified configuration file. To use this option, LOGLVL of the gateway must be set to DEBUG4. LOGLVL can be dynamically changed using the chlog command in tmadmin. Since the log is output to stdout, the log file name must be specified using the -o option.

  Configure each line with a type [on | off] in the specified configuration file. Each type is a number between 0 and 7 representing the following cases. Set a type to ON to enable logging for the case.

Type	Description
0	When receiving a request (call/acall) message from CLH
1	When receiving a response (call/acall) message from CLH
2	When receiving a request (call/acall) message from a remote gateway

Type	Description
3	When receiving a response (call/acall) message from a remote gateway
4	When sending a request (call/acall) from CLH
5	When sending a response (call/acall) from CLH
6	When sending a request (call/acall) from a remote gateway
7	When sending a response (call/acall) from a remote gateway

The following is an example of a configuration file that enables logging for all cases.

```
0 on
1 on
2 on
3 on
4 on
5 on
6 on
7 on
```

The following is the log format.

If a request is processed successfully, 0 is recorded. If it fails, 1 is recorded. Other items are set to an appropriate value, and the message is recorded in hexadecimal format on the subsequent line.

```
time [service_name] call type(0~7):success(0 or 1):data type(xx):data
size(xx):errno(xx):urcode(xx)
message
```

- [-k]

  Performs scheduling when a gateway and a service set as COUSIN restart in a remote domain.

  This option is used when domains are connected through tmaxgwnt and there is a service set as COUSIN in a multi-domain environment that uses both Tmax 5 and later and Tmax 4 SP3 Fix#9 and previous. Do not use this option in a multi-domain environment that uses only Tmax 5 and later versions.

◦ Considerations

- If TMAXGW -h is specified to Tmax 3.x (for example, 031404), and the -R option is not set, it is set to the default value, RBK.
- If the -R option is not set while JEUSGW -h1 is set, it is set to the default, RBK.
- JEUSGW -r (recovery support) and the -R option cannot be set together.

- RESTART = Y | N
  - Default: Y
  - Option to restart the gateway if it is terminated abnormally.
- MAXRSTART, GPERIOD
  - These two items are used to configure fault tolerance. Refer to Fault Tolerance Settings.
- PTIMEOUT = numeric
  - Range: 1 ~ MAX_INT/2
  - Default: -1
  - Maximum time that GW keeps the pending list. The maximum time must be less than or equal to the value of TIMEOUT divided by 2. (TIMEOUT >=PTIMEOUT)
  - Retransmit the pending transaction (prepare-done/commit/rollback) from the Domain Gateway to prevent any pending transactions even when a decision has not been received.

    When Gateway1 and Gateway2 are connected through 2pc, if Gateway2 receives a prepare() and then does not receive a commit() or rollback(), Gateway2 will send prepare-done() to Gateway1. In this case, the PTIMEOUT and PTIMEINT settings can be used to prevent having a pending transaction when a decision is not received due to a network problem (or abnormal termination) between Gateways.
- PTIMEINT = numeric
  - Range: 1 ~ MAX_INT/2
  - Default: -1
  - Specify the time interval for GW to retransmit a pending transaction (prepare-done, commit, rollback).
  - Maximum time must be less than or equal to the value of PTIMEOUT.
- GWCHKINT = numeric
  - Range: -1 ~ MAX_INT
  - Default: -1
  - Interval for making a re-connect attempt to the main sever for failback while the gateway is connected to a backup.

Value	Description
-1	Refer to the DOMAIN section.
0	Failback is not used.

- GWCONNECT_TIMEOUT = numeric
  - Range: -1 ~ MAX_INT
  - Default: -1
  - Maximum waiting time for GW to connect to a remote gateway.

Value	Description
-1	Refer to the DOMAIN section.
0	Default value of 5 seconds.

- NLIVEINQ = numeric
  - Range: -1 ~ MAX_INT
  - Default: -1
  - Interval for sending a message to check if the channel that connects a gateway to a remote gateway is available.

Value	Description
-1	Refer to the DOMAIN section.
0	Not used.

- TRB = string
  - For details, refer to TRB in "3.2.2 NODE".

Example

The following is an example of using the GATEWAY element.

```
*GATEWAY
gw1             NODENAME = "tmax1",
                PORTNO = 2222,
                RGWADDR = "192.168.23.1",
                RGWPORTNO = 2225,
                GWTYPE = TMAX,
                CPC = 10,
                CLOPT = "-r 030815"
```

The following is an example of using the VIEW type of Tuxedo.

```
*DOMAIN
...
*NODE
ibm5l       TMAXDIR = "...",
            APPDIR  = "...",

*SVRGROUP
svg1        NODENAME = "ibm5l"

*SERVER
svr_sdl     SVGNAME = svg1

*SERVICE
TOUPPER_SDL     SVRNAME = svr_sdl
#svc for tuxedo
```

```
TUX_TOUPPERSDL     SVRNAME = TUXGW

*GATEWAY
TUXGW          GWTYPE = TUXEDO,
               PORTNO = 9521,
               RGWADDR = "192.168.1.43",
               RGWPORTNO = 9511,
               CLOPT = "-a TUXGW1 -v",
               NODENAME = ibm5l,
               CPC = 20,
               TIMEOUT = 30
```

The following is an example of using GATEWAY to give processing priority to local nodes.

```
*DOMAIN
tmax1          SHMKEY = @SHMEMKY@,
               MINCLH = 1, MAXCLH = 3,
               TPORTNO = @TPORTNO@,
               BLOCKTIME = 25, RACPORT = @TRACPORT@

*NODE
@HOSTNAME@     TMAXDIR = "@TMAXDIR@",
                . . .
@RMTNAME@      TMAXDIR = "@RMTDIR@",
                . . .

*SVRGROUP
svg1           NODENAME = "@HOSTNAME@",
               COUSIN = "svg2, svg3, svg4, gw1, gw2",
               LOAD = -2
svg2           NODENAME = "@HOSTNAME@", LOAD = -2
svg3           NODENAME = "@RMTNAME@", LOAD = -2
svg4           NODENAME = "@RMTNAME@", LOAD = -2

*SERVER
svr2           SVGNAME = svg1

*SERVICE
TOUPPER        SVRNAME = svr2

*GATEWAY
gw1            GWTYPE = TMAXNONTX, PORTNO = 7788,
               RGWADDR = "@GATENAME@",
               RGWPORTNO = 6688,
               NODENAME = @HOSTNAME@,
               CPC = 2, LOAD = -2
gw2            GWTYPE = TMAXNONTX, PORTNO = 7788,
               RGWADDR = "@GATENAME2@",
               RGWPORTNO = 6688,
               NODENAME = @RMTNAME@,
               CPC = 2, LOAD = -2
```

As you can see in the previous example, when client connects to HOSTNAME and calls TOUPPER service, the servers that belong to svg2, gw1, and svg1 will process the request in a round robin manner. (1st request: svg2, 2nd request: gw1, 3rd request: svg1, 4th request: svg2, ...) If there is a

failure with the server for svg1, svg2 and gw1 will process the request in order. When all three servers fail, the server group of another node specified as a COUSIN will process the request. When client connects to the RMTNAME, the servers in svg3, svg4, and gw2 will process the request in a round robin manner.

Local-first load balancing function of the domain gateway is only supported by gateways for non-XA connections.

3.2.7. ROUTING

The ROUTING element defines the routing settings used between the nodes of a domain, or between the domains in a multi-domain system. It is an optional element. For more information, refer to "data dependent routing" of Load Balancing Settings.

3.2.8. RQ

This is an optional element. For more information, refer to Reliable Queue Settings.

3.2.9. HMS

This is an optional element. For more information, refer to HMS Settings.

3.2.10. Basic Configuration Example

The following is an example of a basic Tmax configuration file.

```
*DOMAIN
res             SHMKEY = 77990, MAXUSER = 300, MINCLH = 3,
                MAXCLH = 5, TPORTNO = 8899, BLOCKTIME = 60
*NODE
tmax1           TMAXDIR = "/home/tmax",
                APPDIR = "/home/tmax/appbin",
                PATHDIR = "/home/tmax/path",
                SLOGDIR = "/home/tmax/slog",
                ULOGDIR = "/home/tmax/ulog",
                ENVFILE = svr_env
tmax2           TMAXDIR = "/system/tmax",
                APPDIR = "/system/tmax/server",
                SLOGDIR = "/system/tmax/slog",
                ULOGDIR = "/system/tmax/ulog",
                ENVFILE = app_env
*SVRGROUP
svg1            NODENAME = tmax1
svg2            NODENAME = tmax2
svg3            NODENAME = tmax2, cousin = svg4
svg4            NODENAME = tmax2
```

```
*SERVER
svr1            SVGNAME = svg1, CLOPT="-e err1 -- apple", MIN = 3, MAX = 5
svr2            SVGNAME = svg2, MIN = 4, MAX = 5
svr3            SVGNAME = svg3,
                CLOPT = "-e $(SVR).$(DATE).err –o
                $(SVR).$(DATE).out"
*SERVICE
svc1            SVRNAME = svr1, PRIO = 100, SVCTIME = 40
svc2            SVRNAME = svr2
svc3            SVRNAME = svr3
```

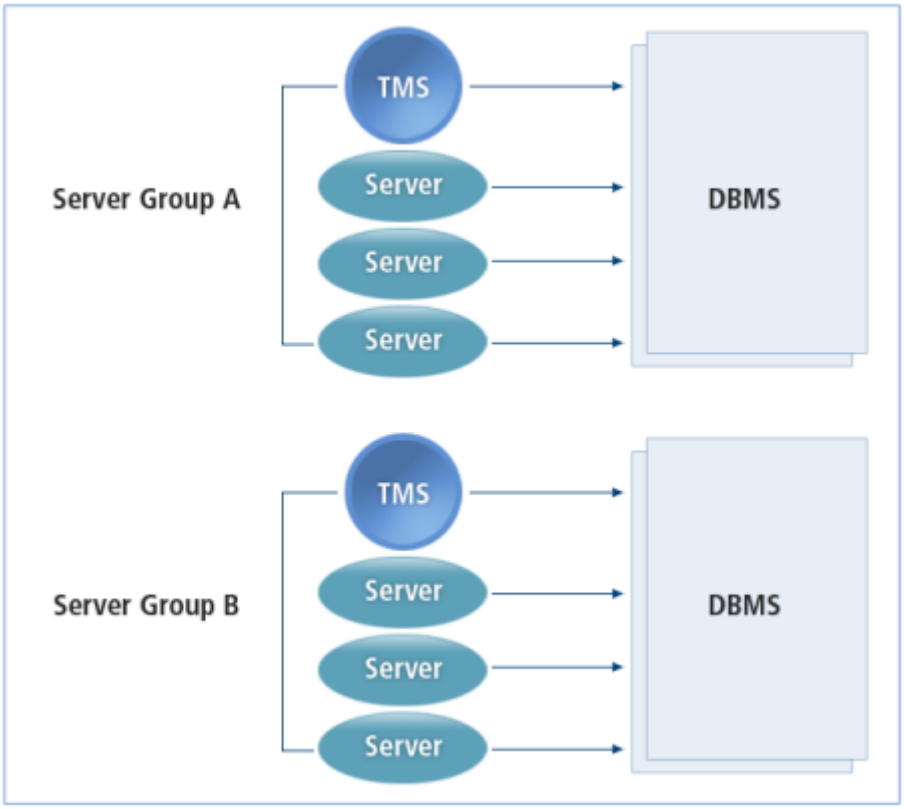
3.3. Database Configuration Settings

To process database related services, the appropriate database settings must be added to the configuration file. Once this information has been properly registered, connecting to and disconnecting from the resource manager, as well as opening and closing the database, will be managed by Tmax, and distributed transaction processing will be enabled.

Server groups play an important role in database configuration. A server group is defined by both the nodes where the servers of the group exist as well as the database type that they will access.

Servers in the same server group all use the same DBMS. Two servers that use different DBMSs cannot belong to the same server group. Thus, the server group is the basic unit for database management.

Databases are managed by the TMS (Transaction Management Server), a functional process of Tmax. When required, the TMS will be loaded into each server group to perform database management functions and manage transaction processing. The TMS will be further explained in Distributed Transaction Settings.



Relationships between TMS, server groups, and databases

Database information must be configured in the SVRGROUP element.

3.3.1. SVRGROUP

To properly enable Tmax's database management features, the configuration file must be modified to include information about opening and closing the database used by each server group, as well as other information about the TMS of each server group.

To enable the database management features, the following items must be specified for each server group within the SVRGROUP element.

```
*SVRGROUP
SVRGROUP Name       [DBNAME = DBMS-name,]
                    [OPENINFO = DBMS-open-information,]
                    [CLOSEINFO = DBMS-close-information,]
                    [TMSNAME = TMS-process-name,]
                    [MINTMS = TMS-process-min-number,]
                    [MAXTMS = TMS-process-max-number]
```

- DBNAME = string
  - Size: Up to 63 characters
  - Name of the database to be used by the server group. Use standard DBMS names, such as ORACLE, INFORMIX, and SYBASE.

- OPENINFO = string
  - Size: Up to 255 characters
  - Database information related to server group.
  - If using the same resource manager (a database), using one server group for management can raise resource efficiency and enable distributed transaction processing. The OPENINFO syntax for this item will vary depending on the resource manager vendor (such as Oracle and Informix). If you are unsure of what to enter, refer to the relevant database manual.
    - Oracle database

      ```
      OPENINFO="Oracle_XA+Acc=P/scott/tiger+SesTm=60"
      ```

      "Oracle_XA" represents the Oracle database and XA interface, and "Acc=P/scott/tiger" represents the account and password for database access.

    - Informix database

      OPENINFO for Informix database only requires the database name.

      ```
      OPENINFO= Database Name
      ```

- CLOSEINFO = literal (NULL)
  - Size: Up to 255 characters
  - Default: NULL
  - Must include information required by the resource manager to close the database of the server group.
  - This item is usually omitted or set to NULL. Informix database requires that CLOSEINFO be set to a null character and it may be omitted for Oracle databases.

    <Server group that uses Oracle>

    ```
    *SVRGROUP
    svg1            NODENAME = tmax1,
                    DBNAME = ORACLE,
                    OPENINFO = "Oracle_XA+Acc = P/scott/tiger+SesTm= 60"
    ```

    <Server group that uses Informix>

    ```
    *SVRGROUP
    svg2            NODENAME = tmax2,
                    DBNAME = INFORMIX,
                    OPENINFO = "test",
                    CLOSEINFO = ''''
    ```

- TMSNAME = string
  - Size: Up to 63 characters
  - Name of the TMS that will manage database transactions for the server group.
  - If OPENINFO for the database has been configured, the TMSNAME setting must also be set. The TMS process is required for managing the database of the server group from the systems that use the database. In order for Tmax to manage a database, the database open/close information must be configured, and TMSNAME must be defined for each server group to start the TMS process. The TMS process is a transaction manager that processes transactions through XA linked to the database. It is created by linking the libtms.a file in the $TMAXDIR/lib directory to the database library of the DBNAME defined in the SVRGROUP element.
- MINTMS = numeric
  - Range: 1 ~ 32
  - Default: 2
  - Number of TMS processes that can be started for the server group. This setting will influence transaction processing management more than database management. The default value is two TMS processes per server group, which is enough for only database management.
- MAXTMS = numeric
  - Range: 1 ~ 32
  - Default: 3
  - Number of additional TMS processes that can be loaded dynamically as needed. For more information, refer to Distributed Transaction Settings.

3.3.2. Database Configuration Example

The following is an example of a basic Tmax configuration file containing both the basic settings and database management settings.

```
*DOMAIN
res       SHMKEY = 77990,
          MAXUSER = 300 , MINCLH = 3 , MAXCLH = 5 ,
          TPORTNO = 8899, BLOCKTIME = 60

*NODE
tmax1     TMAXDIR = "/home/tmax" ,
          APPDIR = "/home/tmax/appbin",
          PATHDIR = "/home/tmax/path",
          SLOGDIR = "/home/tmax/slog",
          ULOGDIR = "/home/tmax/ulog",
          TLOGDIR = "/home/tmax/tlog"
tmax2     TMAXDIR = "/system/tmax",
          APPDIR = "/system/tmax/server",
          SLOGDIR = "/system/tmax/slog",
          ULOGDIR = "/system/tmax/ulog",
          TLOGDIR = "/system/tmax/tlog",
          ENVFILE = app_env
```

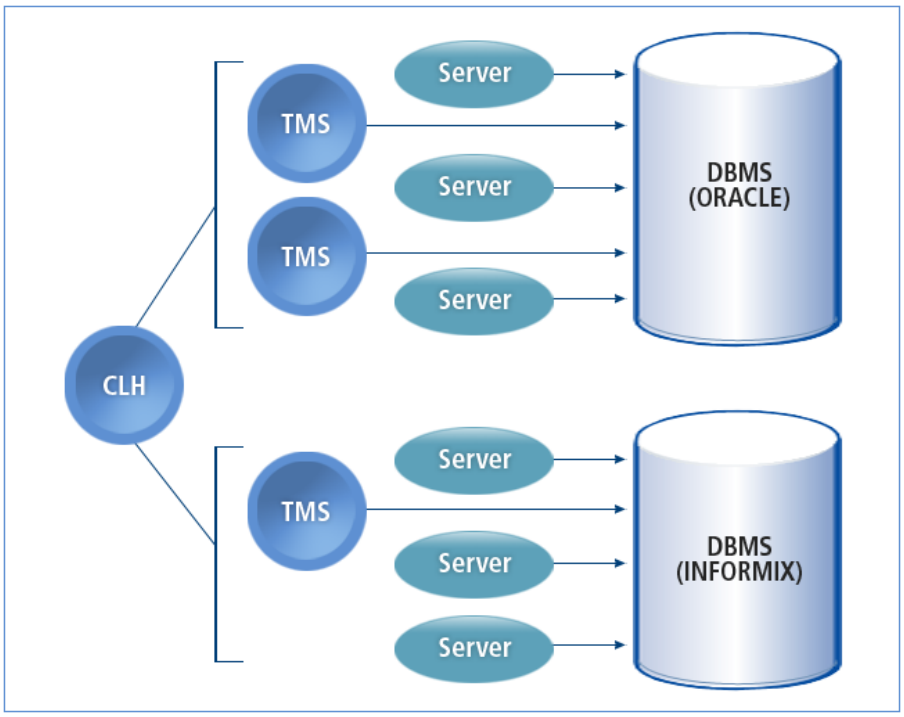
```
*SVRGROUP
svg1      NODENAME = tmax1, DBNAME = ORACLE,
          OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm=60",
          TMSNAME = ora_ tms, MINTMS = 3, MAXTMS = 5
svg2      NODENAME = tmax2, DBNAME = INFORMIX,
          OPENINFO = "infodb",
          TMSNAME = info_tms, CLOSEINFO = ""

*SERVER
svr1      SVGNAME = svg1 ,
          CLOPT="-e err1 -- apple", MIN = 3,
          MAX = 5
svr2      SVGNAME = svg2 , MIN = 4 , MAX = 5
svr3      SVGNAME = svg2,
          CLOPT = "-e $(SVR).$(DATE).err –o
          $(SVR).$(DATE).out"

*SERVICE
svc1      SVRNAME = svr1 ,
          PRIO=100, SVCTIME = 40
svc2      SVRNAME = svr2
svc3      SVRNAME = svr3
```

3.4. Distributed Transaction Settings

One of the main functions of a TP Monitor (Transaction Processing Monitor) is enabling distributed transaction processing. This is when several transactions involving several heterogeneous database systems are grouped and processed as a single global transaction. To enable distributed transaction processing, database management settings must be properly configured. TMS processes must be started for each server group as shown in the following figure.



Distributed Transaction Architecture

The CLH manages commit and rollback of sub-transactions within each distributed transaction. Distributed transactions will only be committed once all sub transactions have been committed. If any one of the sub-transactions is rolled back, the entire transaction will also be rolled back.

The management of commit and rollback statements for updating the database is conducted by the TMS within each server group. Items required to properly enable distributed transactions must be set within the DOMAIN, NODE and SVRGROUP elements.

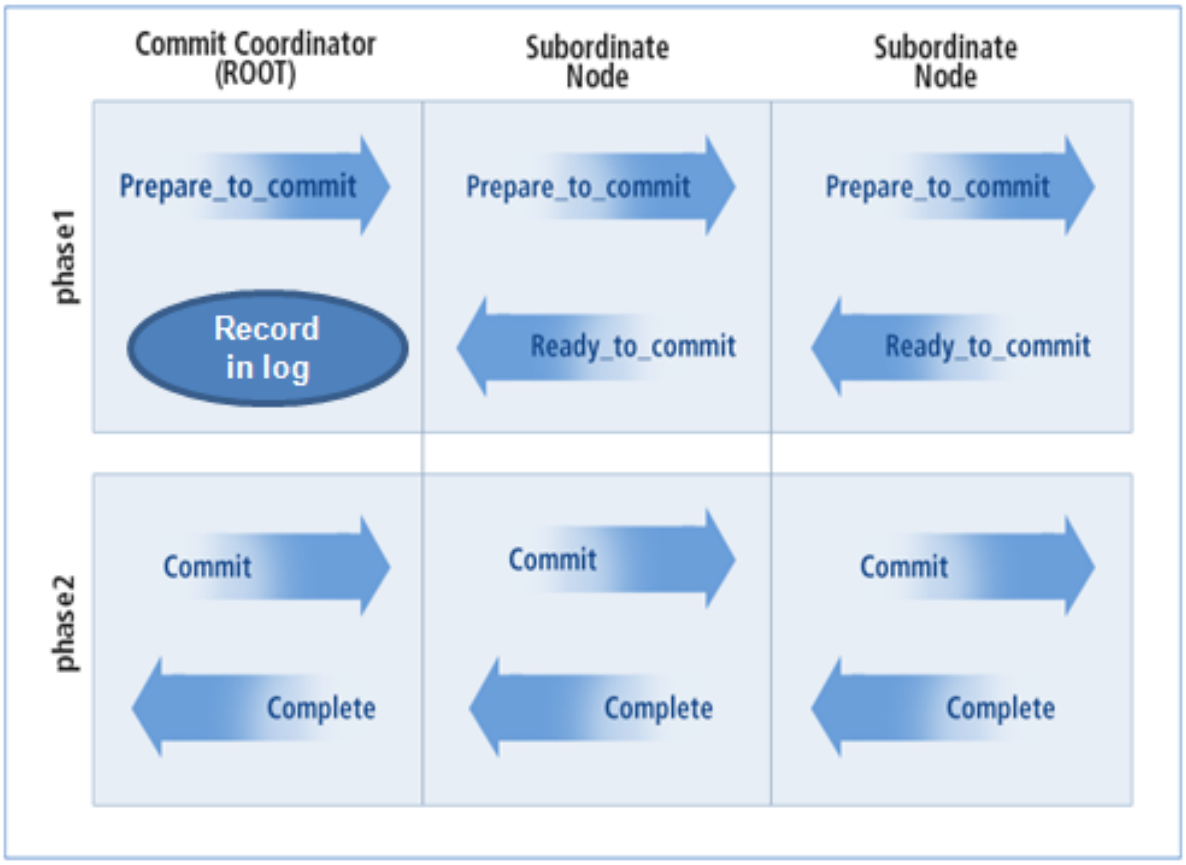
3.4.1. DOMAIN

The DOMAIN element contains items for specifying the two-phase commit (2PC) and transaction time-out values of distributed transactions.

Additionally, the following database related items are included within the DOMAIN element.

```
*DOMAIN
Domain Name      [CMTRET = (Y) | N,]
                [TXTIME = transaction-timeout-value]
```

- CMTRET = Y | N
  - Default: Y
  - Setting for the two-phase commit (2PC) for transaction processing. 2PC transactions consist of two phases, preparation and commit/rollback. Tmax supports 2PC for distributed transaction implementation.
    - Phase 1 (preparation): The transaction's processing state is verified with each DBMS to determine whether it has committed or rolled back.
    - Phase 2 (commit / rollback): Depending on the result of phase 1, the transaction is committed if all related transactions have been committed, or rolled back if any related transaction has been rolled back.



Two-Phase Commit

There are two modes for 2PC. In the complete return mode, the service returns after completing all transactions using 2PC protocol and logging all rollbacks. In the return immediately mode, for faster processing the service returns immediately after issuing a commit/rollback command in Phase 2. Either mode can be selected according to the system environment.

- CMTRET determines whether to do a complete return.

Value	Description
Y	Complete return.
N	For faster processing, service returns immediately after issuing a commit/rollback command.

- TXTIME = numeric
  - Range: 0 ~ MAX_INT
  - Default: 0
  - Timeout value for processing transactions, starting from when a transaction begins with tx_begin() to when the transaction ends with tx_commit() or tx_rollback(). If tx_commit() or tx_rollback() is not complete when the timeout expires, the transaction will be automatically rolled back.

3.4.2. NODE

In addition to the configuration required for database management, the NODE element contains items that specify the files for logging distributed transaction processes of each node.

The following are the items in the NODE element required for distributed transaction processing.

```
*NODE
Node Name     [TLOGDIR = transaction-log-path]
```

- TLOGDIR = literal
  - Size: Up to 255 characters
  - Default: TMAXDIR
  - To handle a mid-transaction failure, all distributed transactions are logged. TLOGDIR is used to specify the file where transaction information will be stored.

    This log file will be saved as an ordinary file since Tmax does not support raw files for logs. The general file defined in TLOGDIR with any name will only be used for logging transactions in situations with light transaction processing workload. To shorten response times in situations with heavy transaction processing workload, the specified raw file is used to set the device name to TLOGDIR without going through the UNIX file system. The device will only be used for logging.

The specified path must exist within the system in use.

Example

A log file in the TLOGDIR path is saved in the binary format. The database transaction log can be saved by setting the trace flag in OPENINFO within the server group element.

```
TLOGDIR = "/home/tmax/log/tlog"
```

Add the following to the xa_NULLmmdd.trc file in the /tmp directory.

```
OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm = 60, DbgFl = 0x01 + LogDir=/tmp"
```

3.4.3. SVRGROUP

Most of the items required for configuring the database environment of each server group must be set within the SVRGROUP element. For more information, refer to Database Configuration Settings.

The basic settings of the SVRGROUP element is as follows:

```
*SVRGROUP
SVRGROUP Name       [DBNAME = name-of-database,]
                    [OPENINFO = string,]
                    [CLOSEINFO = string,]
                    [TMSNAME = name-of-tms,]
                    [MINTMS = numeric,]
                    [MAXRSTART = numeric,]
                    [MAXTMS = numeric]
```

- DBNAME = string
  - Name of the database used by the server group.
- OPENINFO = literal
  - Information required for opening the database.
- CLOSEINFO = literal
  - Default: NULL
  - Information required for closing the database.
- TMSNAME = string
  - Name of the TMS process.
- MINTMS = numeric
  - Minimum number of TMS processes for the server group.
  - This item is not important in terms of database management. However, it is very important in transaction processing. If there are too few TMS processes (one or two) when the transaction workload is high, commit/rollback procedures will generate a large resource overhead. Thus it is important to specify an appropriate number of TMSs to effectively manage resources and throughput.
- MAXRSTART = numeric
  - Fault tolerance setting. For details, refer to Fault Tolerance Settings.
  - Within the SVRGROUP element, this setting refers to the MAXRSTART of a TMS or RQ.
- MAXTMS = numeric
  - Maximum number of additional TMS processes that can be loaded within the server group. As the transaction workload increases, additional TMS processes can be loaded up to MAXTMS.

Example

The following is an example Tmax configuration file with distributed transaction settings.

```
*DOMAIN
res       SHMKEY = 77990, MAXUSER = 300 ,
          MINCLH = 3 , MAXCLH = 5 ,
          TPORTNO = 8899 ,
          BLOCKTIME = 60 ,
          CMTRET = N , TXTIME = 50
```

```
*NODE
tmax1     TMAXDIR = "/home/tmax",
          APPDIR = "/home/tmax/appbin",
          PATHDIR = "/home/tmax/path",
          SLOGDIR = "/home/tmax/log/slog",
          ULOGDIR = "/home/tmax/log/ulog",
          TLOGDIR = "/home/tmax/log/tlog "
tmax2     TMAXDIR = "/system/tmax",
          APPDIR = "/system/tmax/server",
          SLOGDIR = "/system/tmax/slog",
          ULOGDIR = "/system/tmax/ulog",
          TLOGDIR = "/system/tmax/log/tlog"

*SVRGROUP
svg1      NODENAME = tmax1, DBNAME = ORACLE ,
          OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm=60,
          DbgFl=0x01 + LogDir=/tmp",
          TMSNAME = ora_tms,
          MINTMS = 3, MAXTMS = 5
svg2      NODENAME = tmax2, DBNAME = INFORMIX,
          OPENINFO = "infodb", CLOSEINFO = "",
          TMSNAME = info_tms,
          MINTMS = 3, MAXTMS = 5
svg3      NODENAME = tmax2

*SERVER
svr1      SVGNAME = svg1,
          CLOPT = "-e err1 -- apple", MIN = 3, MAX = 5
svr2      SVGNAME = svg2, MIN = 4, MAX = 5,
          CLOPT = "-e $(SVR).$(PID).err –o $(SVR).$(PID).out"
svr3      SVGNAME = svg3
          CLOPT = "-e $(SVR).$(DATE).err –o $(SVR).$(DATE).out"

*SERVICE
svc1      SVRNAME = svr1, PRIO = 100, SVCTIME = 40
svr2      SVRNAME = svr2
svc3      SVRNAME = svr3
```

3.5. Load Balancing Settings

Load balancing is used by Tmax to optimize system performance and resource management during peak periods.

Load Balancing can be set to operate in one of the following three modes.

- System Load Management (SLM)

  Service requests are routed to a node based on the system status and the node's performance.

- Data Dependent Routing (DDR)

  Service requests are routed to a node based on the data in the request.

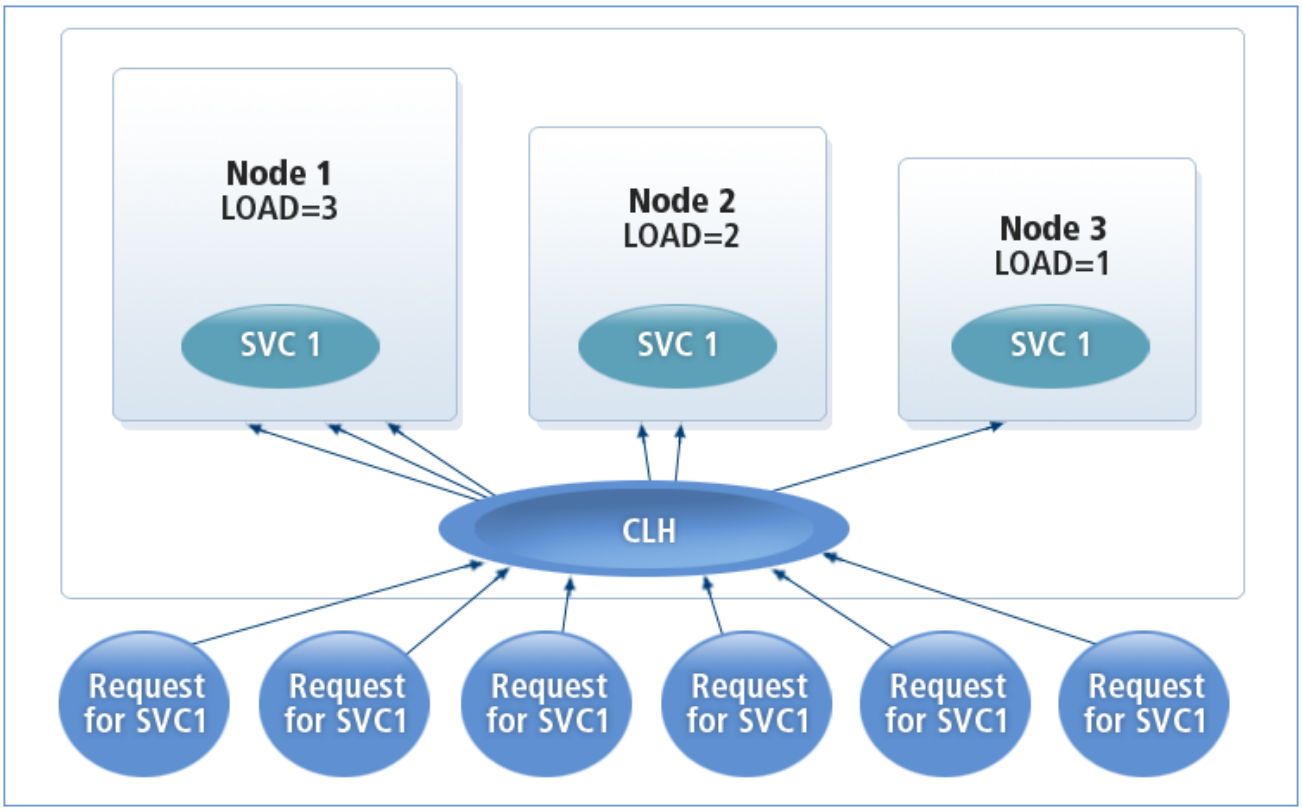
- Dynamic Load Balancing (DLM)

When service requests are concentrated on a particular node, further requests will be sent to other nodes, maximizing the overall system performance.

For information about items that are not described in this section, refer to Basic Configuration.

3.5.1. System Load Management

Different volumes of service requests will be assigned to a node based on the system status and the node's performance. This type of load balancing must be used in situations where a single service can be processed by multiple nodes. System load setting must be defined for each server group.



System Load Management

SVRGROUP

To implement System Load Management, the following items must be defined within the SVRGROUP element.

```
*SVRGROUP
SVRGROUP Name                [COUSIN = group-name,]
                             [LOAD = load-value]
```

- COUSIN = literal
  - Size: Up to 255 characters
  - This is an item that specifies the name of server group when the Load Balancing function is

used on 2 or more nodes.

- Name of each server group must be registered in the SVRGROUP element. The Load Balancing function of Tmax system is processed in server group units. But when Load Balancing is processed across multiple nodes, each node name must be specified differently in each server group. The node name must already be configured in the NODE element of configuration file. When defining multiple server groups, separate them by using a comma(,). The required server program must exist in the APPDIR of each node. To use performance-based load balancing, a load value must be set for each server group.

- LOAD = numeric
  - Range: -1 ~ MAX_INT
  - Default: 1
  - Represents the server group's service performance level. A higher value represents higher performance level.
  - Set to a high value for a high-performing node and a low value for a low-performing node.

    For system-based load balancing, set the COUSIN item to the server group and set an appropriate LOAD value for each server group. To disable distributed processing on a node, LOAD must be set to –1. If LOAD is not set or set to 0, load balancing will be automatically performed based on the system load.

Example

The following is an example of a Tmax configuration file for copying SVRGROUP.

- Tmax configuration file for copying SVRGROUP in a multi-node environment

```
*DOMAIN
site1     SHMKEY = 77990, MAXUSER = 400,
          MAXCLH = 5, TPORTNO = 8899, BLOCKTIME = 60,
          CMTRET = N, TXTIME = 50

*NODE
NODE1     TMAXDIR = "/home/tmax",
          APPDIR = "/home/tmax/appbin",
          PATHDIR = "/home/tmax/path",
          SLOGDIR = "/home/tmax/log/slog",
          ULOGDIR = "/home/tmax/log/ulog",
          TLOGDIR = "/home/tmax/log/tlog "
NODE2     TMAXDIR = "/system/tmax",
          APPDIR = "/system/tmax/appbin",
          SLOGDIR = "/system/tmax/log/slog",
          ULOGDIR = "/system/tmax/log/ulog",
          TLOGDIR = "/system/tmax/log/tlog"
NODE3     TMAXDIR = "/system/tmax",
          APPDIR = "/system/tmax/appbin",
          SLOGDIR = "/system/tmax/log/slog",
          ULOGDIR = "/system/tmax/log/ulog",
          TLOGDIR = "/system/tmax/log/tlog"

*SVRGROUP
```

```
SVG1      NODENAME = NODE1, COUSIN = "SVG2,SVG3"·····..(1)
          LOAD = 3 , DBNAME = ORACLE,
          OPENINFO ="ORACLE_XA+Acc=P/scott/tiger+SesTm=60",
          TMSNAME = tms1, MINTMS = 3
SVG2      NODENAME = NODE2, ·····(2)
          LOAD = 2
SVG3      NODENAME = NODE3, ·····(3)
          LOAD = 1, DBNAME = INFORMIX ,
          OPENINFO = "infodb", CLOSEINFO = "",
          TMSNAME = tms2
          ......

*SERVER
SVR1      SVGNAME = SVG1, MIN = 5 , MAX = 10
SVR2      SVGNAME = SVG1
          ......

*SERVICE
SVC_A     SVRNAME = SVR1
SVC_B     SVRNAME = SVR1
SVC_C     SVRNMAE = SVR2
```

- Tmax configuration file for copying SVRGROUP in a single node environment

```
*DOMAIN
site1     SHMKEY = 77990, MAXUSER = 400,
          MAXCLH = 5, TPORTNO = 8899, BLOCKTIME = 60,
          CMTRET = N, TXTIME = 50

*NODE
NODE1     TMAXDIR = "/home/tmax",
          APPDIR = "/home/tmax/appbin",
          PATHDIR = "/home/tmax/path",
          SLOGDIR = "/home/tmax/log/slog",
          ULOGDIR = "/home/tmax/log/ulog",
          TLOGDIR = "/home/tmax/log/tlog "

*SVRGROUP
SVG1      NODENAME = NODE1, COUSIN = "SVG2"
          LOAD = 3 , DBNAME = ORACLE,
          OPENINFO ="ORACLE_XA+Acc=P/scott/tiger+SesTm=60",
          TMSNAME = tms1, MINTMS = 3
SVG2      NODENAME = NODE1, LOAD = 2
     ......

*SERVER
SVR1      SVGNAME = SVG1, MIN = 5 , MAX = 10
SVR2      SVGNAME = SVG1
          ......

*SERVICE
SVC_A     SVRNAME = SVR1
SVC_B     SVRNAME = SVR1
SVC_C     SVRNMAE = SVR2
```

3.5.2. Data Dependent Load Balancing

In Data Dependent Routing (DDR), service requests are assigned to server groups according to the range of data values.

DDR can only be used for services that use STRUCT, STRING, CARRAY, or FIELD KEY type buffers. To properly implement DDR, server group cousins must be configured within the SVRGROUP element, and routing settings must be defined in the SERVICE and ROUTING elements.

SVRGROUP

To use DDR, the following items must be defined in the SVRGROUP element.

```
* SVRGROUP
SVRGROUP Name              [COUSIN = group-name]
```

- COUSIN = literal
  - Size: Up to 255 characters
  - DDR can only be used when multiple server groups provide the same service. Hence, the original SVRGROUP needs to be copied to a different node according to the COUSIN setting. For more information, refer to the description of COUSIN in SVRGROUP.

SERVICE

To use DDR, the following items must be defined in the SERVICE element.

```
* SERVICE
SERVICE Name               [ROUTING = routing-name]
```

- ROUTING = string
  - Size: Up to 63 characters
  - Logical routing name.
  - This must be unique within the SERVICE element, and will be used in the ROUTING element, which needs to be defined for a service to use DDR (this is explained next). Additionally, the typed buffer name, and field used for routing and its value range must be set with the server group name.

ROUTING

DDR can be implemented across the nodes of a domain, or across the domains of a multi-domain system by using the value of a specific field or member of the typed buffer.

ROUTING, set inside the SERVICE element, is used for configuring inter-node DDR within a domain. In

this case, in addition to the ROUTING name, a STRUCT, STRING, or CARRAY typed buffer or a field buffer must also be defined. The member or field name of the buffer and its value range must also be set with the server group name. When DDR is used across domains in a multi-domain environment, the GATEWAY name must be specified in lieu of the server group name.

To use DDR, the following items must be defined in the ROUTING element.

```
* ROUTING
ROUTING Name        FIELD = BufferType/field-name,
                  RANGES = "Range1: Group1, Range2: Group 2, ....",
                  MATCH = (FIRST)|MULTIPLE
```

- ROUTING Name = string
  - Size: Up to 63 characters
  - Routing name defined in ROUTING item of the SERVICE element. Because DDR is implemented for each service, the ROUTING name must match the one defined in the SERVICE element.
- FIELD = string
  - Size: Up to 2000 characters
  - This item is configured according to the field data type (STRUCT, FIELD, STRING, or CARRAY type) of the services that use data dependent routing.
  - The following is the configuration syntax for each of the four data types.
    - STRUCT type buffer

      The STRUCT type buffer is a user-defined C struct, equivalent to the X_C_TYPE and X_COMMON buffers defined in X/OPEN. The name and contents of the struct must be declared in advance, and the file required by Tmax must be provided. Tmax references the environment variable, SDLFILE, to identify the contents of the struct.

      ```
      STRUCT/subtype/fieldname
      ```

Item	Description
subtype	Name of the struct
fieldname	Member of the struct used for routing

    - FIELD type buffer

      The buffer's FIELD item and field type must be defined in advance and the file required by Tmax must be provided. Tmax references the environment variable, FDLFILE, to identify the contents of the buffer.

      ```
      FIELD/fieldname
      ```

Item	Description
fieldname	Name of the field to be used with routing.

- STRING and CARRAY type buffers

  DDR is also supported for STRING and CARRAY type data. Specify "CARRAY" or "CARRAY/offset/length" to use CARRAY data and specify "STRING" or "STRING/offset/length" to use STRING data. There are two types of Data Dependent Routing: matching and range. Refer to "example" for more details about the difference between the two. Image type data can be stored in a CARRAY type buffer, but the data must be printable.

```
STRING or STRING/offset/length
CARRAY or CARRAY/offset/length
```

  For a description of offset and length, refer to the following "STRING/CARRAY: offset" and "STRING / CARRAY: length" sections.

- STRING / CARRAY: offset = numeric(0)
  - Starting position of the input data that will be used for DDR.

    Both offset and length must be set to use only part of the data specified by the offset and length for DDR. For instance, STRING/0/4 represents offset of "0" and length of "4". If the input data was "Tmaxsoft", only "Tmax" which is 4 bytes in length starting from offset position of 0, will be used for comparison in performing DDR. This item is only used with CARRAY / STRING type DDR.

  - Offset and length must only be used for the range type DDR and not for the matching type because the offset and length items specify only the input data and are not related to the data specified in RANGES.

    In the following example, only "abcd", 4 bytes counted from the offset (0), of the string "aaaaaaa" will be used as comparison criteria for DDR. Since "aaaa" (the comparison criteria for the input data) and "aaaaaaa" (specified in RANGES) do not match, the data will be processed by SVG2. Note that, in this case, all incoming data will be processed unconditionally by SVG2.

```
ROUTE_1   FIELD = "STRING/0/4",
          RANGES = "aaaaaaa":SVG1, *:SVG2
```

  - Offset and length are available when using the range type DDR.

    In the following example (Range type DDR), if the input data is "aaaaaaa", only the "aaaa" part of the input data (4 bytes counted from the offset 0) will be used as comparison criteria for DDR. Since the string, "aaaa" is smaller than the range specified in SVG1, it will be processed by SVG2. If the input data is "bbbb", it falls in the range specified by SVG1, and hence it will be

processed by SVG1.

```
ROUTE_1    FIELD = "STRING/0/4",
           RANGES = "aaaaaaa"-"ccccccc":SVG1, *:SVG2
```

- STRING / CARRAY: length = numeric
  - As explained already, offset specifies the starting point of the element within the input data used for DDR criteria, and length specifies the length of element from the offset. Length must be used in conjunction with offset.
  - If length is not specified, Tmax will use the input data from offset until NULL, or the end of the data.
- RANGES = "Range: Group1, Range: Group2, …."
  - Size: Up to 2047 characters
  - Destination of client service requests based on the value of the field specified in FIELD.
  - Values specified must conform to the following.
    - For routing field, int, short, long, float, double, string, and char can be assigned.
    - For an integer or real number, a single value or a range, specified by a lower and an upper value, can be used.
    - For a matching type, strings are compared for a full match. A range type can be expressed by a lower and an upper value.
    - Positive(+) and negative(-) signs can be used.
    - Characters such as MIN (minimum value) and MAX (maximum value) can be used.
    - In a range specified as "A - B", "A" cannot be larger than "B".
    - An asterisk (*) can be used as a wild card. This is used to represent those values that are not included in other ranges. The asterisk wild card must always be defined to account for values not included in other ranges. However, the asterisk wild card can be omitted when both MIN and MAX are defined.
    - The server group name must be specified after the colon (:) which comes after the range. The server group specified here must be one that is set as a COUSIN within the SVRGROUP element.
    - The entire string must be enclosed in double quotation marks.
    - When configuring cross-domain routing, the domain gateway name must be used instead of the server group name.
    - If a string is used for routing, the value must be enclosed in single quotations.
- MATCH = (FIRST)/MULTIPLE
  - When not using the MATCH option

```
*ROUTING
rout1    FIELD = "STRING, CARRAY, FIELD/INPUT,
         STRUCT/test/a",
```

```
        RANGES = "min-'100':svg1,'101'-'200':svg1_2,min-'100':svg1_3,*:svg1_4"
```

If the configuration file is configured like the previous example, in the previous version, a server in svg1 is first inspected to process data within the range min-100. If a node containing svg1 is not running normally, its backup is examined, and if backup is not running or does not exist, an error is sent to the client. However, in the new version, Tmax will operate in the following way by specifying MATCH in the ROUTING element.

◦ When using MATCH

```
*ROUTING
rout1    FIELD = "STRING, CARRAY, FIELD/INPUT,
         STRUCT/test/a",
         RANGES ="min-'100':svg1,'101'-'200':svg1_2, min-'100':svg1_3,*:svg1_4" ,
         MATCH=MULTIPLE
```

If the configuration file is configured as in the previous example, a server in svg1 is first inspected to process data within the range min-100. If a node containing svg1 is not running normally, its backup is examined. If the backup is not running or does not exist, Tmax searches for a next matching server group within the RANGES, and svg1_3 will be selected. If a node containing svg1_3 is alive, but if the server group svg1_3 is also not running, its backup is examined. If the backup does not exist or is not running normally, a next matching server group, svg1_4, will be selected.

Set "MATCH=FIRST" to use the function of the previous Tmax version.

Example

The following are examples of DDR.

- When using an integer

  In the following example, for the 'number' member of the 'student' struct, service requests with data values between a negative number and 3 will be routed to SVG1, between 4~6 will be routed to SVG2, and over 7 will be routed to SVG3.

```
ROUTE_1        FIELD = "STRUCT/student/number",
               RANGES = "MIN - 3 : SVG1,  4 - 6 : SVG2,  7 - MAX : 1SVG3"
```

  In the following example, even numbers will be processed by SVG1 and odd numbers will be processed by SVG2. Since the range is specified by the MOD operator, this can produce a much wider range.

```
ROUTE_1        FIELD = "STRUCT/student/number",
               RANGES = "MOD2 = 0 : SVG1,  MOD2 = 1 : SVG2"
```

- When using a buffer field

  In the following example, if the INPUT field value is 'STRING', the service request will be processed by SVG1, otherwise it will be processed by SVG2. When service requests are routed across domains, the gateway name assigned to the domain needs to be specified in place of the server group name. If DDR is performed using a string value, it must be enclosed in single quotations.

```
ROUTE_1        FIELD = "FIELD/INPUT",
               RANGES = "'STRING' : SVG1,  *: SVG2"
```

- When using a STRING or CARRAY (Matching DDR)

  In the following example, if the value of STRING or CARRAY is 'aaa', the request will be processed by SVG1, otherwise it will be processed by SVG2. When using the matching type DDR, it is important to remember that the data will only be considered as a match when the length of the input value is same as the defined range. In the following example, the defined range for SVG1 is 'aaa', so any other strings such as 'aaab' and 'aaac' will be processed by SVG2.

```
ROUTE_1        FIELD = "STRING",
               RANGES = "'aaa' : SVG1,  *: SVG2"
ROUTE_2        FIELD = "CARRAY",
               RANGES = "'aaa' : SVG1,  *: SVG2"
```

- When using STRING, CARRAY (Range type)

  In the following example, if the value of STRING or CARRAY is between 'aaa' and 'ccc', the service request will be processed by SVG1, otherwise it will be processed by SVG2. Range type DDR will check the entire string regardless of the data length for cases like the following example.

```
ROUTE_1        FIELD = "STRING",
               RANGES = "'aaa'-'ccc' : SVG1, *: SVG2"
ROUTE_2        FIELD = "CARRAY",
               RANGES = "'aaa'-'ccc' : SVG1, *: SVG2"
```

- When using STRING, CARRAY (Range type, specify the length for DDR comparison)

  In the following example, only the string between the offset (0) and the length (4) will be used as the comparison criteria for DDR ("aaaa"). If the input data is "aaaaaaa", only the initial element "aaaa" will be used as the comparison criteria. In the following example, since "aaaa" is smaller than the range specified for SVG1, the service request will be processed by SVG2. If the input data is "bbbbbbb", the request will be processed by SVG1, since it falls within the specified range.

```
ROUTE_1        FIELD = "STRING/0/4",
               RANGES = "'aaaaaaa'-'ccccccc':SVG1,*:SVG2",
```

- When simultaneously using various data types

  In the following example, STRING, CARRAY, and FIELD are simultaneously used as comparison criteria for DDR. If the input data is less than or equal to 1000, the service request will be processed by SVG1; if the input data is between 1001 and 2000, it will be processed by SVG2; otherwise it will be processed by SVG3.

```
ROUTE_1        FIELD = "STRING/0/4, CARRAY, FIELD/INPUT",
               RANGES = "min-'1000' : svg1, '1001'-'2000':
               svg2, *:svg3"
```

3.6. Reliable Queue Settings

A common problem in systems that do not use a TP Monitor is that if the system shuts down due to system failure, all queued data is lost. Tmax's reliable queue feature resolves this problem by loading service data into the disk file instead of memory, enabling requests to be processed after a system reboot.

Basic RQ settings are defined within the DOMAIN, NODE, SVRGROUP, SERVER, and SERVICE elements, and a separate RQ element must also be added to the file.

RQ can help preserve data during a system failure, but may slow down the performance.

3.6.1. SVRGROUP

To use the RQ system, define the server groups that contain RQ. To do this, specify the SVGTYPE item to 'RQMGR' in addition to the default items of the SVRGROUP section. The efficiency of system operations can be increased by configuring the CPC.

Set the SVRGROUP section in the environment configuration as follows:

```
*SVRGROUP
SVRGROUP Name     NODENAME = node-name,
                  CPC = number,
                  SVGTYPE = [RQMGR|TXRQMGR],
                  TMSNAME= "POD"
```

- *SVRGROUP Name* = string
  - Size: Less than 16 characters
  - Server group name. Must be unique in the SVRGROUP section.
- NODENAME = string
  - Node name where a server group belongs to. The node name must be the same as the one defined in the NODE section.

- CPC = numeric
  - Range: 4~128
  - If the RQ item, which sets the number of parallel communication channels between the RQS process and the CLH process, is not used, then this will be ignored. If RQ is used frequently, its process speed can be improved by maintaining multiple channels.
- SVGTYPE = [RQMGR|TXRQMGR]
  - Server group type.

Value	Description
RQMGR	Set to use RQ.
TXRQMGR	Set to use RQ in a transaction environment. Note the following: • Set SVGTYPE (RQ server group type) to TXRQMGR, not RQMGR. • Transactions are not supported for existing tpenq() and tpdeq(). • tpenq_ctl cannot be combined with tpdeq_ctl in a transaction. • When binding tpenq_ctl into a transaction, up to the process of storing in RQ can be handled as a transaction. Transactions are not supported for the svc call. • When setting a deq_time, svc call will be performed after the specified time. The format is "Current time + delay_time". The following example sets delay_time to 10. `time(&t); ctl->deq_time = t + 10;` • deq_time cannot be used along with a transaction.

- TMSNAME = "POD"
  - RQ operates like a POD server. Set this item to "POD" to process tpenq requests.

Examples

```
*SVRGROUP
svg1    NODENAME = tmax1,
        CPC = 4,
        SVGTYPE=RQMGR
```

The following is the sample configuration to use RQ in a transaction environment.

```
*SVRGROUP
```

```
Txrqsvg    NODENAME = tmaxh4,
           CPC = 4 ,
           SVGTYPE = TXRQMGR
```

3.6.2. RQ

Each server group can use one or more RQs. In the RQ section, set the management method of the RQ. The RQ section has the QSIZE, FILEPATH, BOOT, FSYNC, and BUFFERING fields. Among them, SVGNAME is mandatory.

Set the RQ section in the environment configuration as follows:

```
*RQ
RQ Name   SVGNAME = server-group-name,
          BOOT = COLD | WARM,
          QSIZE = number,
          BUFFERING = Y | N ,
          FSYNC = Y | N ,
          FILEPATH = file-path,
MAX_MEMQCOUNT = number
```

- *RQ Name* = string
  - Can be up to 14 characters.
  - Logical name of the RQ section. Multiple RQs can be defined in the RQ section. Each RQ name must be unique. In a single server group, multiple RQs can be defined.
- SVGNAME = string
  - Server group name that uses RQ.
  - The server group name must be defined in the SVRGROUP section.
- BOOT = COLD | WARM
  - Default value: COLD
  - The BOOT field adjusts the data stored in RQ when the Tmax system reboots.
  - The following describes configurable values:

Value	Description
WARM	Enables data recovery in a queue after recovery from a system error.
COLD	Data stored in a disk will be deleted when the Tmax system restarts.

- QSIZE = numeric
  - Range: 1~2047 MB
  - Default: 16 MB
  - RQ size can be specified. RQ is created in the TMAXDIR path by default. If the queue entries

exceed the specified queue size as a result of continuous services, TPEQFULL will occur when performing tpenq(). The RQ size will not be increased automatically so the size must be set properly. If RQ data will not be stored even after a RQ file is written, the file will be deleted automatically when Tmax is terminated.

- BUFFERING = Y | N
  - Default: Y
  - Option to cache the RQ file content to memory.
  - The following describes a configurable value;

Value	Description
N	The RQ process will get relatively slower but the required memory will get smaller.

- FSYNC = Y | N
  - Default value: Y
  - Even BUFFERING is set to 'N', OS does not store data to a disk directly but stores the data in a memory at most 30 seconds. This field eliminates such waiting time and stores the data in a memory to a disk.
  - The following describes configurable values:

Value	Description
N	Data might be lost when an error occurs in the system but the RQ process speed will get faster.
Y	RQ data does not use memory and is always stored in a disk safely but the process speed is slower than that when the value is set to 'N'.

- FILEPATH = literal
  - Default value: $(TMAXDIR)/path/RQ name.data
  - Path to the file used by RQ. This field creates the RQ data files and must be defined with the entire file name and the directory name.

TMAXDIR is a home directory of Tmax installation. Hereafter the Tmax home directory will be referred to as "TMAXDIR."

- MAX_MEMQCOUNT = numeric
  - Default: 1024
  - Maximum number of messages to save in memory when RQ stores data in memory and does not write the data to a file (TPRQS_NON_PERSISTENT).

Example

```
*RQ
rqtest  SVGNAME = svg1,
        BOOT = COLD,
        QSIZE = 1024,
        BUFFERING = Y,
        FSYNC = Y,
        FILEPATH = "/user1/tmax/myrq.data"
```

3.7. HMS Settings

A Messaging System is a communication mediator that enables a loose coupling between the sender and the receiver. Through the messaging system, they can communicate with each other by using only the destination information (a virtual channel) without information about each other. The times when the message is sent and received are recorded separately to allow for delayed processing, and reliability is guaranteed for sending/receiving messages.

To use HMS(Hybrid Messaging System), the DOMAIN, NODE, SVRGROUP, and HMS elements must be configured. For details about HMS, refer to *Tmax HMS User Guide*.

3.7.1. DOMAIN

HMS is configured in the DOMAIN section as follows:

```
*DOMAIN
Domain1     [MAXSESSION = numeric]
```

Optional Items

- MAXSESSION = numeric
  - Range : 1 ~ 65535
  - Default value: 1024
  - Maximum number of sessions that can be created in HMS within the domain.

3.7.2. NODE

HMS is configured in the NODE section as follows:

```
*NODE
Node1       [MAXSESSION = numeric]
```

Optional Items

- MAXSESSION = numeric
  - Range: 1 ~ 65535
  - Default value: 1024
  - Maximum number of sessions that can be created in HMS within the node.

3.7.3. SVRGROUP

HMS is configured in the SVRGROUP section as follows:

```
*SVRGROUP
ServergroupName      NODENAME = node-name,
                    SVGTYPE = HMS,
                    HMSNAME = string,
                    OPENINFO = literal,
                    HMSINDEX = numeric,
                    HMSMAXTHR = numeric,
                    HMSMAXDBTHR = numeric,
                    [HMSMAXBULKTHR = numeric,]
                    [HMSMAXBULKSIZE = numeric,]
                    [HMSOPT = literal,]
                    [HMSSUBSCFG = string,]
                    [HMSMSGLIVE = numeric,]
                    [HMSPORT = numeric,]
                    [HMSHEARTBEAT = numeric,]
                    [HMSGQINT = numeric]
```

Required Items

- *ServergroupName* = string
  - Can be up to 63 characters.
  - Logical name of the server group. Must be unique in the SVRGROUP element.
- NODENAME = string
  - Can be up to 63 characters.
  - Node where the server group exists. Must be a node that is defined in the NODE element.
- SVGTYPE = string
  - Server group type. To use HMS, this must be set to HMS.
- HMSNAME = string
  - Can be up to 63 characters.
  - Name of a user-built HMS process.
- OPENINFO = literal
  - By default, HMS uses a database for storage. Specify OPENINFO of the DBMS to which HMS

will connect.

- Initialize the database connection, and define OPENINFO using the syntax required by the database.

- HMSINDEX = numeric
  - Range: 0 ~ 65535
  - Used for processing an HMS message. Must be unique within the domain.
- HMSMAXTHR = numeric
  - Range: 0 ~ 65535
  - Number of threads that will not be used for storage processing.
  - Must be set to a large value when sending and receiving only nonpersistent messages. For basic use of HMS, this item must be set above a certain value.
- HMSMAXDBTHR = numeric
  - Range: 0 ~ 65535
  - Number of threads used for storage processing.
  - Must be set to a large value when there are many persistent messages. For basic use of HMS, this item must be set above a certain value.

Optional Items

- HMSMAXBULKTHR = numeric
  - Range: 1 ~ 65535
  - Number of threads that perform batch storage processing for HMS messages.
- HMSMAXBULKSIZE = numeric
  - Range: 2 ~ 65535
  - Maximum number of HMS messages that can be simultaneously processed for batch storage processing.
- HMSOPT = literal
  - Can be up to 255 characters.
  - Command options transmitted to an HMS process when HMS starts up.

Option	Description
-e file_name	Writes standard errors that occur during HMS operation to the specified file.
-o file_name	Writes standard outputs that occur during HMS operation to the specified file.

- HMSSUBSCFG = string
  - Can be up to 255 characters.

    - Path and name of a configuration file where Durable Subscriber is defined. In the file, each label is set with [TopicName] and ClientName:Listener:Selector is specified.
    - The durable subscriber is automatically registered when HMS starts up.
- HMSMSGLIVE = numeric
    - Range: 0 ~ 65535 (Unit: hours)
    - An interval to delete persistent messages that HMS accumulates in a storage.
    - Topic messages that all subscribers receive and queue messages that are consumed are deleted from the storage if they are expired. Messages that are not sent to all receivers are not deleted even if they are expired.
- HMSPORT = numeric
    - Port number used by HMS to communicate with other HMSs in a cluster.
    - Do not use a port number that is already used because HMS listens by using the port if there is a destination with the GLOBAL property.
- HMSHEARTBEAT = numeric
    - Range: 0 ~ 65535 (Unit: seconds)
    - Heartbeat check interval. Clustered HMSs regularly exchange heartbeat messages to detect any network failure among them.
    - If an HMS does not send a heartbeat message within the interval, the connection to the HMS is ended.
- HMSGQINT = numeric
    - Range: 0 ~ 65535 (Unit: milliseconds)
    - An interval to send and receive status information between queues with the GLOBAL property.
    - Must be set to use a clustered queue. Data is more evenly distributed in the queue if the interval is shorter, but network load is heavier. Set a value accordingly.

3.7.4. HMS

HMS is configured in the HMS section as follows:

```
*HMS
DestinationName      SVGNAME = string,
                     TYPE = TOPIC | QUEUE,
                     [BOOT = WARM | (COLD),]
                     [GLOBAL = Y | (N),]
                     [GQTHR = numeric,]
                     [GQMAXREQ = numeric,]
                     [GQFULL = numeric]
```

Required Items

- *DestinationName* = string
  - Can be up to 63 characters.
  - Destination name. A producer and consumer are created using this name.
- SVGNAME = string
  - Can be up to 63 characters.
  - HMS server group name of the destination.
  - The SVGNAME must be an HMS server group name specified in the SVRGROUP element.
- TYPE = TOPIC | QUEUE
  - Destination type. Currently, only TOPIC and QUEUE are supported.

Optional Items

- BOOT = WARM | COLD
  - Default value: COLD
  - How to treat service requests stored in the RQ after a reboot.
  - Like in RQ, set either to WARM or COLD.

Value	Description
WARM	Services accumulated in the queue can be recovered after the system reboots.
COLD	Service requests saved on the disk will be ignored after the system reboots. (Default)

- GLOBAL = Y | N
  - Default value: N
  - Option to use destinations with the same name in clustered HMSs logically as a single destination.
  - For GLOBAL setting, HMSPORT must be set in HMS of the SVRGROUP element.
  - For GLOBAL queue setting, the GQTHR, GQMAXREQ, and GQFULL items are also required.
- GQTHR = numeric
  - Range: 0 ~ 65535
  - Coefficient to determine the buffer size maintained by a clustered queue.

```
Buffer size = (the number of consumers) X (GQTHR value)
```

  - If the length of the current queue is shorter than the buffer size, the queue takes messages from another clustered queue. If the queue contains less messages than the buffer size, it does not distribute its messages upon a distribution request.

- GQMAXREQ = numeric
  - Range: 1 ~ 65535
  - Number of messages that can be requested at once to another queue. If the specified value is large, the queue will be filled with messages very quickly. If the value is too large, there may be unnecessary message transmissions between nodes. Thus, an appropriate value must be set.
- GQFULL = numeric
  - Range: 0 ~ 65535
  - Number of messages that can be accumulated in a queue. If more than the specified number of messages is accumulated in a queue, any subsequent messages are evenly distributed to the clustered queues on another node.

3.8. Fault Tolerance Settings

Tmax provides the following fault tolerance features.

- Protection against hardware failures (node or network failures): fault tolerance using backup services
- Protection against software failures (process shut down): fault tolerance using server process restart

3.8.1. Hardware Failure

When this is properly configured, a backup service will continue to process services when they cannot be processed due to a system failure. This is implemented by specifying the BACKUP item within the SVRGROUP element.

SVRGROUP

```
* SVRGROUP
SVRGROUP Name          [BACKUP = group-name]
```

- BACKUP = literal
  - Size: Up to 255 characters
  - Backup server group.
  - Multiple backup server groups can be specified by using a comma (,) as a separator. If the original server group fails, the first specified backup will be activated, and if this backup server group also fails, the next server group will be activated, and so on.
  - All of the specified backup server groups must be registered within the SVGROUP element. The backup server groups must be configured to comply with the settings of the original

server group. To change the configuration options, relevant items can be redefined for each server group or server. However, service settings must be the same for the original and backup server groups. These rules are the same as those for duplicate server groups created for load balancing.

Unlike the duplicate server groups created for load balancing, backup nodes are not initially active when Tmax is launched. Instead, they are activated to replace the original server group when the original server group fails.

3.8.2. Software Failure

When this feature is properly configured, server processes, such as CLH and CAS, will be automatically restarted when they shut down unexpectedly.

NODE

```
* NODE
NODE Name           [RESTART = (Y) | N,]
                   [MAXRSTART = numeric,]
                   [GPERIOD = time-value]
```

- RESTART = Y | N
  - Default: Y
  - Option to restart CLH, CAS, CLL, and TLM upon abnormal termination.
  - If set to Y, the server processes will be restarted when they shut down unexpectedly due to a system failure.
- MAXRSTART = numeric
  - Range: -1 ~ MAX_INT
  - Default: -1
  - Maximum number of times a CHL or CAS process can be restarted.
  - MAXRSTART is used in combination with the RESTART and GPERIOD items. If RESTART is set to 'Y', a process will be restarted MAXRSTART times during the GPERIOD. If set to 'N', Tmax will ignore this setting. If set to '-1', there will be no limit to the number of times that a process can be restarted.
- GPERIOD = numeric
  - Range: 1 ~ MAX_INT
  - Default: 86400 (24 hours) (Unit: second)
  - Time period for which MAXRSTART is valid.
  - If a server process is unexpectedly terminated, it will be restarted MAXRSTART times during the GPERIOD. Once the GPERIOD expires, this procedure will be repeated.

SVRGROUP

```
* SVRGROUP
SVRGROUP Name       [RESTART = (Y) | N,]
                    [MAXRSTART = numeric,]
                    [GPERIOD = time-value]
```

- RESTART, MAXRSTART, GPERIOD
  - If the SVGTYPE is RQMGR, these items configure the restart setting of the RQ. If the SVGTYPE is XA svg, they configure the restart setting of the TMS.
  - If a server process is abnormally terminated, the process automatically restarts to provide services.
  - For more information of each item, refer to the previous "3.8.2. NODE" section.

SERVER

```
* SERVER
SERVER Name         [RESTART = (Y) | N,]
                    [MAXRSTART = numeric,]
                    [GPERIOD = time-value]
```

- RESTART, MAXRSTART, GPERIOD
  - These items configure the restart setting of the server processes. If a gateway process is abnormally terminated, the process automatically restarts to provide services.
  - The default value of MAXRSTART is 5.
  - For more information of each item, refer to the previous "3.8.2. NODE" section.

GATEWAY

```
* GATEWAY
GATEWAY Name        [RESTART = (Y) | N,]
                    [MAXRSTART = numeric,]
                    [GPERIOD = time-value]
```

- RESTART, MAXRSTART, GPERIOD
  - These items configure the restart setting of the gateway.
  - If the gateway process is abnormally terminated, the process automatically restarts to provide services.
  - For more information of each item, refer to the previous "3.8.2. NODE" section.

3.9. Security Settings

3.9.1. Security Levels

There are three security levels in Tmax system.

- Level 1: System access control

  Controls access to Tmax.

- Level 2: User authentication

  Limits service access to the authenticated users.

- Level 3: Service access control

  User access control can be configured per service.

Each level includes the previous level(s). If level 2 security is configured, level 1 security settings will also be applied. Level 3 security includes security levels 2 and 1. To properly configure Tmax security settings, the SECURITY item must be set within the DOMAIN element.

DOMAIN

```
* DOMAIN
DOMAIN Name     [SECURITY = ("NO_SECURITY")| "DOMAIN_SEC"|"USER_AUTH"|"OTHER_AUTH",]
                [OWNER = user-name]
```

- SECURITY = "NO_SECURITY" | "DOMAIN_SEC" | "USER_AUTH" | "ACL" | "MANDATORY" | "OTHER_AUTH"
  - Default: "NO_SECURITY"
  - The following are available values.

Value	Description
NO_SECURITY	No security features are used.
DOMAIN_SEC	Option to use level 1 security (system access control only). "DOMAIN_SEC" must be specified in the SECURITY item within the DOMAIN element. Use mkpw to set a single password for Tmax system, and users who know the password can access the system. To use this level, the OWNER item must be defined in the DOMAIN element.

Value	Description
USER_AUTH	Option to use level 2 security (user authentication). USER_AUTH must be specified in the SECURITY item within the DOMAIN element. Tmax manages user accounts and passwords and allows service access to only authorized users. User IDs and passwords are created with the mkpw utility. A user must register a valid ID and password in the usrname and usrpwd items of TPSTART_T. Users who fail to be authenticated cannot access the system.
ACL, MANDATORY	Option to use level 3 security (service access control). Users will only be able to use those specific services that they have been granted access to. Service access is controlled for each user group. A service can be accessed by users of a group, which is allowed to access the service. For this feature to be properly implemented, a group file as well as files for the users that belong to the group must be created. There must also be an acl file that assigns the services accessible to the user groups.
OTHER_AUTH	Option to use when the third-party authentication library is used instead of the default authentication library provided by Tmax. SECURITY_LIB in the NODE section must be set to the third-party library path.

Utilities such as mkgrp, mkpw and mkacl must be used to generate each of these files. Both ACL and MANDATORY are set to level 3 security. The following describes their differences.

Value	Description
ACL	Only the users that are registered in the group file have access to the services in the acl file. Services not in the acl file may be accessed by any user with access to the Tmax system.
MANDATORY	This is identical to ACL in that only registered users will be able to access the services. However, services not in the acl file cannot be accessed by any user with access to the Tmax system.

- OWNER = string
  - Size: Up to 63 characters
  - Password for the dompwd of TPSTART_T. This password will be used to log into the Tmax system. If no password or an invalid one is provided, users will not be able to access the Tmax system. The user and password defined here can be created with the mkpw utility.

    After configuring the security, a password file must be created with the mkpw utility. This file includes passwords of each user, and is encrypted. For Tmax security system to work

properly, the password file must be created before configuring the Tmax system.

- For DOMAIN_SEC, if the specified password is different from the dompwd value of the TPSTART_T structure in tpstart(), the client program cannot make a service request. The tperrno is set to TPESECURITY(25). For USER_AUTH, the usrpwd and usrname fields in the TPSTART_T struct must be the same as the account and password registered in the password file created with the mkpw utility.
- Since level 2 (user authentication) security includes level 1 (system access control), the specified dompwd must be the same as the set domain password.

Note that a password file must be created or reset before starting the Tmax system.

- Client setting
  - To use authentication-related functions, set **TMAX_SEMANTICS="AUTH_SW:Y"** in a client environment variable.

3.9.2. Other Security Settings

A Tmax user can allow or not allow a client to access the system based on IP by creating 'tmax.allow' and 'tmax.deny' configuration files for allowing and denying access. In CLL of each file, select 'allowing access' or 'denying access' of TCP/IP clients.

Access Control based on IP

- Client allowed access

  Create a file to allow access (tmax.allow) in the "$TMAXDIR/path" directory, and specify IP of the clients to be allowed to access the system in the file.

  - IPv4

```
192.168.1.43
192.168.1.48
```

  - IPv6

```
fe80::213:77ff:fe4d:c57d
fe80::213:77ff:fe4d:c570
```

- Client denied access

  Create a file to deny access (tmax.deny) in the "$TMAXDIR/path" directory, and specify IP of

clients to be denied from accessing the system in the file.

- IPv4

```
192.168.1.35
192.168.1.45
```

- IPv6

```
fe80::213:77ff:fe4d:c50d
fe80::213:77ff:fe4d:c500
```

- Applying ACL(Access Control List)
  - Search the tmax.allow file first, and if there are clients that are in the ACL, allow them to access the system.
  - Search the tmax.deny file, and if there are clients that are in the ACL, deny them from accessing the system. If they attempt to access the system (TPSTART), the TPECLOSE error occurs.
  - Allow access to clients who exist in neither tmax.allow nor tmax.deny files.
- Syntax
  - If the first character is '#', it is processed as a comment.
  - Allow only IP or NETWORK/NETMASK type.

    Example)

```
(IPv4)(IP_V4)192.168.1.1 or 192.168.1.0/24,
(IPv6)fe80::213:77ff:fe4d:c57d or fe80::213:77ff:fe4d:c57d/64
```

  - One ACCESS CONTROL LIST is allowed per line, and a blank or tab character is not allowed.
  - ALL is a reserved word meaning all IPs.

The tmax.allow and tmax.deny files are applied when CLL starts, but any changes to the files cannot be applied dynamically.

3.9.3. Third-party Settings

Tmax supports a third-party security system by loading the library. To use the third-party security system, you must implement API related to authentication and encryption provided by Tmax. For more information about security-related functions, refer to *Tmax Reference Guide*.

Authentication and encryption must be set separately in the DOMAIN and NODE sections as follows:

DOMAIN

```
* DOMAIN
DOMAIN Name     [SECURITY = "OTHER_AUTH",]
                [CRYPT = Y]
```

- SECURITY = "OTHER_AUTH"
  - To use a third-party authentication system, set this item to "OTHER_AUTH", and set SECURITY_LIB in the NODE section.
- CRYPT = Y
  - To use a third-party encryption system, set this item to Y, and set CRYPT_LIB in the NODE section.

NODE

```
* NODE
NODE Name       [SECURITY_LIB = literal,]
                [CRYPT_LIB = literal]
```

- SECURITY_LIB = literal
  - To use a third-party authentication system, set this item to the library path and file name. Unless SECURITY is set to "OTHER_AUTH" in the DOMAIN section, compiling the configuration file by using the cfl command fails.
- CRYPT_LIB = literal
  - To use a third-party encryption system, set this items to the library path and file name. Unless CRYPT is set to Y in the DOMAIN section, compiling the configuration file by using the cfl command fails.

3.10. Multi Domain Settings

In systems comprised of multiple domains, Tmax can enable the processing of inter-domain transactions, providing users with extended global transactions and supporting inter-domain data dependent routing to reduce development burdens.

To properly enable a multi-domain system, gateway information must be configured in the GATEWAY item of the DOMAIN element. Additional routing information needs to be specified in the ROUTING and SERVICE elements. For details about each element, refer to Basic Configuration.

3.10.1. DOMAIN

Basic gateway information, including Shared Memory Key information, port number, MINCLH, and

MAXUSER, must be registered within the DOMAIN element. For distributed transactions, 2PC (Two Phase Commit) availability and transaction time-out settings can also be configured.

Refer to the following example of the DOMAIN element for multi domain configuration.

```
*DOMAIN
site1          SHMKEY = 79990, MAXUSER = 100, MINCLH = 1,
               MAXCLH = 3, TPORTNO = 8888,
               CMTRET = Y, BLOCKTIME = 60,
               DOMAINID = 1
```

3.10.2. NODE

Information about each node within the domain must be registered in the NODE element.

Refer to the following example of the NODE element for multi domain configuration.

```
*NODE
tmax1          TMAXDIR = "/user3/tmax",
               APPDIR = "/user3/tmax/appbin",
               PATHDIR = "/user3/tmax/path",
               TLOGDIR = "/user3/tmax/log/tlog",
               ULOGDIR = "/user3/tmax/log/ulog",
               SLOGDIR = "/user3/tmax/log/slog"
```

3.10.3. SVRGROUP

In a multi-domain environment, server groups are classified and configured based on the database that they use. In order to implement cross-domain global transactions (with 2 Phase Commit), servers and services must be grouped into an XA group.

Refer to the following example of the SVRGROUP element for multi domain configuration.

```
*SVRGROUP
NXAGRP    NODENAME = tmax1
XAGRP     NODENAME = tmax1, DBNAME = ORACLE,
          OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm=80+logdir=.",
          TMSNAME = XAGRP_tms, MINTMS = 2, MAXTMS = 5
```

3.10.4. SERVER

In a multi-domain environment, basic settings are configured in each SERVER element.

Refer to the following example of the SERVER element for multi domain configuration.

```
*SERVER
```

```
SVG_X     SVGNAME = XAGRP, MIN = 1, MAX = 2,
          CLOPT = "-o $(SVR).$(DATE).log -- -u scott -p tiger"
          SVG_NX     SVGNAME = NXAGRP, MIN = 1, MAX = 2,
          CLOPT = "-o $(SVR).$(DATE).log"
```

3.10.5. SERVICE

In a multi-domain environment, basic settings are configured in each SERVICE element. When routing is used, a routing name must be specified in the ROUTING element.

Refer to the following example of the SERVICE element for multi domain configuration.

```
*SERVICE
SVC_X           SVRNAME = SVG_X, ROUTING = XRID
SVC_NX          SVRNAME = SVG_NX, ROUTING = NXRID
```

3.10.6. GATEWAY

When routing is used in a multi-domain environment, the required information must be configured in each GATEWAY element. The IP address and port number of a server in the destination domain must be specified in RGWADDR and RGWPORTNO within each GATEWAY element. The name of the node where the gateway process will be executed must be specified in the NODENAME.

Refer to the following example of the GATEWAY element for multi domain configuration.

```
*GATEWAY
DOMAIN_GW1     GWTYPE = TMAX, PORTNO = 5000,
               RGWADDR = "192.168.63.133",
               RGWPORTNO = 5000,
               NODENAME = tmax1
```

3.10.7. ROUTING

This element must be specified when routing is used in a multi-domain environment. The routing name, field name, SUBTYPE (BUFFER type), and routing range (RANGES) must all be specified.

Example Configuration File with Multi-Domain Routing

<Example configuration file: Domain Site>

```
*DOMAIN
site1          SHMKEY = 79990, MAXUSER = 100, MINCLH = 1, MAXCLH = 3,
               TPORTNO = 8888, CMTRET = Y, BLOCKTIME = 60,
               DOMAINID = 1
```

```
*NODE
tmax1          TMAXDIR = "/user3/tmax",
               APPDIR = "/user3/tmax/appbin",
               PATHDIR = "/user3/tmax/path",
               TLOGDIR = "/user3/tmax/log/tlog",
               ULOGDIR = "/user3/tmax/log/ulog",
               SLOGDIR = "/user3/tmax/log/slog"

*SVRGROUP
NXAGRP         NODENAME = tmax1
XAGRP          NODENAME = tmax1, DBNAME = ORACLE,
               OPENINFO = "Oracle_XA+Acc=P/scott/tiger + SesTm = 80 + logdir = . " ,
               TMSNAME = XAGRP_tms, MINTMS = 2, MAXTMS = 5

*SERVER
SVG_X          SVGNAME = XAGRP, MIN = 1, MAX = 2,
               CLOPT = "-o $(SVR).$(DATE).log -- -u scott –p tiger"

SVG_NX         SVGNAME = NXAGRP, MIN = 1, MAX = 2,
               CLOPT = "-o $(SVR).$(DATE).log"

*SERVICE
SVC_X          SVRNAME = SVG_X, ROUTING = XRID
SVC_NX         SVRNAME = SVG_NX, ROUTING = NXRID

*GATEWAY
DOMAIN_GW1     GWTYPE = TMAX, PORTNO = 5000,RGWADDR ="192.168.63.133"
               RGWPORTNO = 5000,NODENAME = tmax1

*ROUTING
XRID          FIELD = FIELD/ROUTING_ID,
              RANGES = "'6400':XAGRP, '6471':DOMAIN_GW1"

NXRID         FIELD = FIELD/ROUTING_ID,
              RANGES = "'6400':NXAGRP, '6471':DOMAIN_GW1"

*DOMAIN
site2         SHMKEY = 79990, MAXUSER = 100, MINCLH = 1, MAXCLH = 3,
              TPORTNO = 8888, CMTRET = Y, BLOCKTIME = 60,
              DOMAINID = 2
*NODE
tmax2         TMAXDIR = "/user3/tmax",
              APPDIR = "/user3/tmax/appbin",
              PATHDIR = "/user3/tmax/path",
              TLOGDIR = "/user3/tmax/log/tlog",
              ULOGDIR = "/user3/tmax/log/ulog",
              SLOGDIR = "/user3/tmax/log/slog"
*SVRGROUP
NXAGRP        NODENAME = tmax2

XAGRP         NODENAME = tmax2, DBNAME = ORACLE,
              OPENINFO = "Oracle_XA+Acc=P/scott/tiger + SesTm = 80 + logdir = . " ,
              TMSNAME = XAGRP_tms, MINTMS = 2, MAXTMS = 5

*SERVER
SVG_X          SVGNAME = XAGRP, MIN = 1, MAX = 2,
               CLOPT = "-o $(SVR).$(DATE).log -- -u scott –p tiger"

SVG_NX         SVGNAME = NXAGRP, MIN = 1, MAX = 2,
```

```
              CLOPT = "-o $(SVR).$(DATE).log"

*SERVICE
SVC_X         SVRNAME = SVG_X, ROUTING = XRID
SVC_NX        SVRNAME = SVG_NX, ROUTING = NXRID

*GATEWAY
DOMAIN_GW2    GWTYPE = TMAX, PORTNO = 5000,RGWADDR = 192.168.63.132",
              RGWPORTNO = 5000,NODENAME = tmax2

*ROUTING
XRID          FIELD = FIELD/ROUTING_ID,
              RANGES = "'6471':XAGRP, '6400':DOMAIN_GW2"
NXRID         FIELD = FIELD/ROUTING_ID,
              RANGES = "'6471':NXAGRP, '6400':DOMAIN_GW2"
```

< Example configuration file for using various Tmax functions>

```
*DOMAIN
site1        SHMKEY = 79990, MAXUSER = 100, MINCLH = 1, MAXCLH = 5,
             TPORTNO = 8888, CMTRET = Y, BLOCKTIME = 30

*NODE
tmax1        TMAXDIR = "/tmax/tmax",
             APPDIR = "/tmax/tmax/appbin",
             PATHDIR = "/tmax/tmax/path",
             TLOGDIR = "/tmax/tmax/log/tlog",
             ULOGDIR = "/tmax/tmax/log/ulog",
             SLOGDIR = "/tmax/tmax/log/slog"

tmax2        TMAXDIR = "/tmax/tmax",
             APPDIR = "/tmax/tmax/appbin",
             PATHDIR = "/tmax/tmax/path",
             TLOGDIR = "/tmax/tmax/log/tlog",
             ULOGDIR = "/tmax/tmax/log/ulog",
             SLOGDIR = "/tmax/tmax/log/slog"

*SVRGROUP
SVGtmax1     NODENAME = tmax2, DBNAME = ORACLE,
             COUSIN = "SVGtmax1",
             OPENINFO = "Oracle_XA+Acc=P/scott/tiger + SesTm = 60+DbgFl=0x01",
             TMSNAME = svg1_tms, MINTMS = 2, MAXTMS = 5

SVGtmax2     NODENAME = tmax1, DBNAME = ORACLE,
             OPENINFO = "Oracle_XA+Acc=P/scott/tiger + SesTm = 60+DbgFl=0x01",
            TMSNAME = svg1_tms, MINTMS = 2, MAXTMS = 5

SVGtmax2NX  NODENAME = tmax1
            SVGRQ1     NODENAME = tmax1,
            SVGTYPE = RQMGR, CPC = 8,
            COUSIN = "SVGRQ1", LOAD = -1

*RQ
rq1         SVGNAME = SVGRQ, BOOT = COLD, FILEPATH = "/tmp/rq1"
            QSIZE = 24, FSYNC = Y, BUFFERING = N

*SERVER
```

```
svr1        SVGNAME = SVGtmax1, MIN = 1, MAX = 5

svr2        SVGNAME = SVGtmax2, MIN = 1, MAX = 5
svr3        SVGNAME = SVGtmax2NX, MIN = 1, MAX = 5

*SERVICE
svc1        SVRNAME = svr1, ROUTING = rout1
svc2        SVRNAME = svr2
svc3        SVRNAME = svr3
svc4        SVRNAME = GW2
            #Gateway is directly defined on the server name.

*GATEWAY
GW1         GWTYPE = TMAX, PORTNO = 5001,NODENAME = "tmax1",
            RGWADDR = "GW1_MAIN", RGWPORTNO = 5001
GW2         GWTYPE = TMAX, PORTNO = 5002,NODENAME = "tmax2",
            RGWADDR = "GW2_MAIN", RGWPORTNO = 5001
GW3         GWTYPE = TMAX, PORTNO = 5003,NODENAME = "tmax2",
            RGWADDR = "GW3_MAIN", RGWPORTNO = 5001

*ROUTING
rout        FIELD = FIELD/PLACE_CD,
            RANGES = "'00000':SVGtmax1, 'A0001'-'A0006':GW2,
           'A0007'-'A4000':GW3"
```

3.11. Compiling Tmax Configuration File

Generally, there are several phases in creating application programs. First, the source code is created and then compiled. The compilation step is used to detect and correct any errors in the source code. After a successful compilation, a valid executable file is created. The Tmax configuration file is also compiled in the same way to ensure that a valid configuration file is created.

Since the configuration file is used to provide the foundations of a Tmax system, Tmax usually cannot be started if the file contains errors. However, if the system is started and any errors do exist, the system will generate error during startup. Hence, the compilation step is very important in ensuring the proper operation of Tmax system.

Compilation is executed by using the cfl command. For more information, refer to *Tmax Reference Guide*.

```
$ cfl [-i text Tmax configuration file name] [-o binary Tmax configuration file name]
      [-h] [-V] [-n node_name] [-A] [-v num] [-I] [-r] [-a Tmax configuration file name] [-Z]
```

Option	Description
[-i *text Tmax configuration file name*]	Name of the configuration file to compile. A user can specify the directory. If not specified, the default configuration directory is **$TMAXDIR/config**. If the original configuration file is not found, a warning message is displayed.
[-o *binary Tmax configuration file name*]	Name of the created binary configuration file. It can be specified with a path. If the path is not specified, the binary configuration file is created in the **$TMAXDIR/config** directory. The default file name is **tmconfig**.
[-h]	Online Help
[-V]	Checks the version of the execution file.
[-n *node_name*]	Used to selectively compile the configuration file of the specified node in a multi-node environment. To manage other nodes in a multi-node environment, racd must be configured.
[-A]	Available only when service access control (3rd security level) is used. Since the access permission for ACL service within a domain must be applied evenly across all nodes, this option is used to distribute the $TMAXDIR/config/group, acl, and user files of the current node to $TMAXDIR/config on other nodes in the same domain. When environment files are compiled using this option, **group**, **acl**, and **user** files must be created in advance using **mkgrp**, **mkpw**, and **mkacl.** Since these files are used by all nodes in the domain, all users of nodes in the domain must be considered when creating the files. The group_name, group_id, user_name, and user_id must be unique within the domain. Since the passwd file is not automatically copied to each node, one must be created on or copied to each node.
[-v *num*]	Version information. This can be set to either '0' or '1'. The default value is 1. • 0: If the nodes of a multi-node domain are all of the same type (for example, all use SUN OS 2.6) and the administrator wants to manage the configuration file from a single node, set this option to '0'. Once compiled, the binary configuration file will be sent to all nodes in the domain. • 1: The configuration file will be automatically compiled through racd on each machine.
[-I]	If the value configured in the SHMKEY item of the DOMAIN element is being used when cfl is executed, compare UIDs of the value. If it is different, an error message is displayed. This option is used to not check whether shared memory and the same UID are in use.

Option	Description
[-r]	Used to check in advance the maximum number of FDs available in the current system that is displayed when 'ulimit -n' option is used during cfl execution. Calculates and checks the maximum number of FDs per CLH in advance. If the FD value that Tmax uses is configured greater than the FD value available on the system, the following error message is displayed. `(E) CFL9990 Current Tmax configuration contains more servers or nodes than current system can support[CFL5056]`
[-a *Tmax configuration file name*]	Name of Tmax configuration file to dynamically add. To add a server dynamically, the [-a] option must be used when executing cfl to create the binary configuration file, otherwise the (E) ADM2048 error occurs.
[-Z]	Currently in cfl, MAXSACALL/MAXCACALL is limited to 1024 or less. Use this option to override this limit. Example) `- cfl -Z -i sample.m`

3.12. Creating Service Table

Services that are not registered within the SERVICE element will not be processed by Tmax. Each server may process a variety of services, so the administrator must generate a service table to identify which services will be handled by which servers. The service table must be compiled together with the server program.

The service table is a file that lists each server and the services that they provide. This file is created using the configuration file. It is created by referencing the servers and services registered in the Tmax configuration file. When server application programs are compiled, they must be linked with the service table to enable service routines to be located within the system.

The service table is created using the gst command, which references the binary Tmax configuration file. For more information, refer to *Tmax Reference Guide*.

```
$ gst [-f binary Tmax configuration file name | -v server_name | -h | -n node_name | -V]
```

Option	Description
[-f *binary Tmax configuration file name*]	Binary Tmax configuration file name, which Name of the binary Tmax configuration file, which is the result of the **cfl** command. The file is referred to when executing **tmboot** and **tmdown**. The file can be set along with its path. If not set, the default value is **tmconfig** in the config directory under the directory set in **$TMAXDIR/config**.
[-v *server_name*]	Name of the server for which a service table is created.
[-h]	Command Help.
[-n *node_name*]	Node name. Creates the service table of another node's server process in the $TMAXDIR/svct directory of the current node.
[-V]	Checks the version of the execution file.

3.13. Service Timeout Monitoring

If there are services that are expired but not deleted, Tmax forcibly deletes them by monitoring services periodically.

3.13.1. tmapm

Service timeout operates by using signal alarms in Tmax systems. tmapm is a server that redefines signal alarms and provides service timeout for services that cannot use signal alarms.

tmapm, which is a USC server, is configured in the SERVER section of a configuration file as follows:

- Usage

```
*SERVER
tmapm    CLOPT = [ -i sec ]
                 [-r sec ]
                 [-c command ]
                 [-l [ 0 | 1 | 2 ]]
```

Option	Description
[-i *sec*]	The interval (in seconds) for checking service time. Shorter intervals increase system load. A decimal number is allowed.
[-r *sec*]	A command execution timeout (in seconds) after a service time is exceeded. Only an integer is allowed.
[-c *command*]	A command that will be executed when a corresponding service is still in a RUNNING state after an execution time (SCVTIME + command execution timeout) has expired. Use a shell command or a script file.

Option	Description
[-l [0 | 1 | 2]]	A user log level. The default value is 0. A larger value outputs more detailed information.

The following parameters are passed on when a specified command is configured with a -c option.

User Command	PID	SVRNAME	SVCNAME[Elapse Time]
kill.sh	9659	svr2	TOUPPER[5]

- Example

  The following is an example.

```
*SERVER
tmapm    SVGNAME = svg,
         SVRTYPE = UCS,
         CLOPT = "-o ulog  --  -i 5 -r 1 -c kill.sh -l 1"
```

4. Starting Up and Shutting Down

This chapter describes how to start up and shut down Tmax system in various environments.

4.1. Starting Up Tmax

After configuring the configuration file, the administrator can start up the Tmax system. Once Tmax is running, it is rarely shut down except for when an external fatal error, such as a hardware or OS failure, occurs. However, the administrator will need to shut down and restart the system when the configuration file must be modified or recreated, or when the system was unexpectedly terminated due to external reasons.

The basic configuration of the Tmax system must be complete before attempting to start up the system. Run through the following checklist prior to starting up the system.

- Does the Tmax configuration binary file exist?
- Do Tmax executables (such as TMM, CLL, CLH, and TMS) exist in the directory specified in the TMAXDIR item of the NODE section?
- Can the server programs, registered within the SERVER section of the configuration file, be found in the directory specified in the APPDIR item of the NODE section?
- Is racd running on each node to enable centralized management?

In general, a Tmax system is made up of multiple nodes in a single domain. This type of system usually requires a centralized management system. If racd (remote access control demon) is installed on each node in advance, it is possible to manage the entire Tmax system from a single node. A single command can be issued through racd to compile the configuration file, start or shut down the Tmax system, or dynamically modify the configuration file.

In order to load racd, TMAX_RAC_PORT must be defined through an environment variable. The system can start up without referencing the environment variables in the Tmax configuration file by specifying the port number of the racd, which is used for centralized management, and starting up racd by executing "racd – k".

4.1.1. racd

The racd command centralizes the management of nodes in a distributed multi-node environment. racd is a daemon process that starts in each node to allow a single node to manage a Tmax system built with multiple nodes in a domain. The node that manages a Tmax system is not required to execute a racd.

racd allows a node in a domain to manage all nodes with a **tmadmin** or a configuration file, which is applied to all nodes in a domain by a **cfl**.

If the IP version is IPv6 or InfiniBand's Socket Direct Protocol (SDP), set the following in a configuration file.

```
SYSTEM_PROTOCOL="IPV6"
SYSTEM_PROTOCOL="SDP"
```

To use racd along with the -k option, set the following environment variables.

```
TMAX_RAC_IPV6="IPV6"
TMAX_RAC_IPV6="SDP"
```

The following is the usage.

- Usage

```
$ racd [-d] [-f binary Tmax configuration file name] [-h] [-k]
       [-i filename] [-l Label] [-P umask] [-V]
```

Option	Description
[-d]	Works in the debug mode.
[-f *binary Tmax configuration file name*]	A binary Tmax configuration file to be used. The binary configuration file is the result of executing a cfl command and is referenced by **tmboot** and **tmdown**. The path to the file can also be specified. If a path is not specified, <**tmconfig**> in the **$TMAXDIR/config** directory will be used.
[-h]	Shows command Help.
[-k]	Option to use a binary Tmax configuration file. If this option is specified, the configuration file will not be used. A racd is usually executed with this option (passive listen mode). If the IP version is IPv6 or InfiniBand's Socket Direct Protocol (SDP), set the following in a configuration file: 'TMAX_RAC_IPV6=IPV6' or 'TMAX_RAC_IPV6=SDP' because the configuration file is not be used.
[-i *filename*]	Used to define multiple logical nodes in a single physical machine. If the NODETYPE of a logical node is SHM_RACD, a racd must be executed for each logical node and each node must have a unique RACPORT. A TMAX_RAC_PORT environment variable must be set before executing a racd. When there are too many logical nodes, the name of the file that defines TMAXHOME, TMAXDIR, and TMAX_RAC_PORT can be specified, because it is difficult to set the variable for each node. For more information, see the examples in the following sections.
[-l *Label*]	A delimiter used for information registered in a configuration file. When information of two or more systems is registered in a single file, the delimiter separates the information of each system.

Option	Description
[-P *umask*]	Allows a user to create a file with desired permissions for a process started by a racd.
[-V]	Shows the version of an executable file.

- Environment

  This command can be used in a system with a Tmax system installed.

- Examples
  - The following sets a desired umask by specifying a -P option to a racd.

    <tmax.racd>

```
[tmaxs1]
TMAXHOME = /user2/starbj81/tmax32
TMAXDIR = /user2/starbj81/tmax32
TMAX_RAC_PORT = 3333

[NODE1]
TMAXHOME = /user2/starbj81/tmax32
TMAXDIR = /user2/starbj81/proj1
TMAX_RAC_PORT = 4335

[NODE2]
TMAXHOME = /user2/starbj81/tmax32
TMAXDIR = /user2/starbj81/proj2
TMAX_RAC_PORT = 4337
```

    1. Create a configuration file.

```
*DOMAIN
tmax1      SHMKEY = @SHMEMKY@, MINCLH = 1, MAXCLH = 3,
           TPORTNO = @TPORTNO@, BLOCKTIME = 30, RACPORT = 3255

*NODE
@HOSTNAME@ TMAXDIR = "@TMAXDIR@",
           APPDIR = "@TMAXDIR@/appbin",
           PATHDIR = "@TMAXDIR@/path",

@RMTNAME@  TMAXDIR = "@RMTDIR@",
           APPDIR = "@RMTDIR@/appbin",
           PATHDIR = "@RMTDIR@/path",

*SVRGROUP
svg1       NODENAME = "@HOSTNAME@", COUSIN = "svg2"
svg2       NODENAME = "@RMTNAME@"

*SERVER
svr2       SVGNAME = svg1, CLOPT = "-o $(SVR).out -e $(SVR).err"

*SERVICE
```

```
TOUPPER    SVRNAME = svr2
```

2. Start a racd in a remote node.

```
$ export TMAX_RAC_PORT = 3255
$ racd –k –P 055
```

3. Start all Tmax instances in a HOST node (tmboot).
4. Check the permission for the <svr2.out> file in the ULOGDIR of the RMT node.

◦ The following executes a racd in NODE 1.

```
$ racd -k -i tmax.racd -l NODE1
```

◦ The following uses only information used by another command, such as tmboot, without referring to a configuration file.

```
$ racd -k
```

1. For more information about tmboot and tmdown, refer to tmboot and tmdown respectively.

2. For more information about logical node settings, refer to HOSTNAME in the "3.2.2. NODE Section" of *Tmax Administrator's Guide*.

4.1.2. tmboot

The tmboot command starts a Tmax system (or a portion of it) using a Tmax configuration file. When this command is executed without an option or only with a [-f] option, all Tmax management processes and all server processes registered in the SERVER section of a Tmax configuration file are executed.

Tmax management processes are executed in all nodes registered in a NODE section. The execution order is **TMM**, **CLL**, and finally **CLH**. If an OPENINFO is registered in the SVRGROUP section in a server group, TMS processes are executed by referring to **TMSNANE** and **MINTMS** for each server group. Tmax management processes are executed in the bin directory under a TMAXDIR directory defined by each node.

After Tmax management processes are created, all application server processes in a SERVER section are executed. The application server processes are executed in the order they were registered in a section. The tmboot command initializes the server processes with tpsvrinit() and then executes them after they are initialized. The number of application server processes executed by a tmboot command is defined by a MIN value. If MIN is not specified, the default value will be 1.

The tmboot command uses **CLOPT**, **MIN**, and **MAX** values for servers in a SERVER section. These values are the boot parameters used by a tmboot when starting a server process. Other items are runtime parameters used by a system after a server starts. For more information about how to set parameters, refer to the SERVER section in a source configuration file.

All application server processes are executed from a APPDIR directory defined for a running node:

- Usage

```
$tmboot [-A] [-b] [-c] [-f binary Tmax configuration file name]
        [-g servergroup_name [-a]] [-h] [-i] [-V]
        [-n node_name] [ -o clopt_string ] [-q RQ svg_name]
        [-s server_name [-k count] [-a]]
        [-S server_name [-a]] [-t TMS_name[-k all]]
        [-B TRB_node] [-T] [-w] [-d boot_time] [-D]
        [-e clh | cas | tlm ] [-k] [-R]
```

Option	Description
[-A]	Executes all application server processes defined in the **SERVER** section of a Tmax configuration file.
[-b]	Used to arbitrarily start a server process specified as a backup.
[-c]	Executes an additional CLH process. The number of CLH processes cannot exceed a **MAXCLH** value, which is defined in a Tmax configuration file.
[-f *binary Tmax configuration file name*]	A binary Tmax configuration file (the result of compiling a source file with a cfl), which is specified with a path. If a [-f] option is not specified, <**tmconfig**> in a config directory under a TMAXDIR is used by default.
[-g *servergroup_name* [-a]]	Executes a server process in a specified server group. The server group name is registered in the **SVRGROUP** section of a Tmax configuration file. If a [-a] option is also used, a TMM process, not a TMBOOT process, will start a server.
[-h]	Shows command Help.
[-i]	Boots unstarted servers even if a server reaches the MAX value in the SERVER section.
[-V]	Shows the version of an executable file.
[-n *node_name*]	Executes a server process in a specified node. The node name must be previously registered in the **NODE** section of a Tmax configuration file.
[-o *clopt_string*]	Adds a CLOPT string.
[-q *RQ svg_name*]	Starts a RQS.

Option	Description
[-s *server_name* [-k *count*] [-a]]	Executes a specified server processes. The server processes must be previously registered in the **SVRGROUP** section of a Tmax configuration file. To specify the number of server processes, use a [-k] option. The number of server processes, which includes currently running server processes, cannot exceed the value of MAX in the SERVER section. If a [-k] option is omitted, only a specified server process will be executed. If an -a option is used, a TMM process, not a TMBOOT process, will start a server.
[-S *server_name*]	Executes a MIN number of server processes. If an [-a] option used, a TMM process, not a TMBOOT process, will start a server.
[-t *TMS_name* [-k all]]	Executes an additional specified TMS process. The number of TMS processes cannot exceed a **MAXTMX** value defined in a Tmax configuration file. [-k all] enables transaction recovery. Because recovery is executed when all groups are terminated and then restarted, recovery is performed by each TMS group. To start and terminate all TMSs with a specific name, use this option.
[-B *TRB node*]	Starts a TRB node.
[-T]	Executes only TMM, CLL, CLH, and TMS processes.
[-w]	Option to start processes sequentially. If no option is used for a tmboot command, all registered server processes will start at the same time. Some server processes may not start normally because certain OSs cannot create sufficient resources. • LOCK usage when server processes connect to a TMM (LOCK | NOLOCK) • WAIT condition when server processes start (NO-WAIT | FINITE-WAIT) ◦ Has the same effect as "-d -1000000 (1sec)". ◦ When a [–d] option is used, this option will be ignored.

Option	Description
[-d *boot_time*]	Starting multiple server processes simultaneously can cause a problem because a CLH can overload with registration requests. To solve this problem, this option allows a registration interval to be adjusted by specifying the period spent to start server processes. (Default value: LOCK and NO-WAIT, unit: usec) • LOCK usage condition when server processes connect to a TMM (LOCK | NOLOCK) • WAIT condition when server processes start (NO-WAIT | FINITE-WAIT) ◦ -d val < 0: LOCK, |VAL| FINITE-WAIT * ◦ -d val = 0: NO-LOCK, NO-WAIT ◦ -d val > 0: NO-LOCK, |VAL| FINITE-WAIT ◦ When the val of a [–d] option is not 0, (|VAL|) will be used by default. Unit: usec. • For a FINITE-WAIT, |VAL| is the maximum WAIT time for each process. It is not the total wait time for all processes. • When a server process sends a signal, a WAIT will be released. This means that a next process will be started when a signal is sent even if |VAL| is not reached. If VAL is negative, a LOCK will be used. If this option is used, [-w] options will be ignored.
[-D]	Similar to a [-d] option, but for a FINITE-WAIT, a server will unconditionally WAIT until a time specified in a |VAL|, even if a signal is received from a server process.
[-e clh | cas | tlm]	Starts CLH, CAS, and/or TLM Tmax engine processes. This option is used when an error occurred while using a tmboot command or when an engine process is terminated by a kill command. The tmdown command does not support this option. • clh: Starts a CLH. • cas: Starts a CAS. • tlm: Starts a TLM.
[-R]	Starts a Tmax system in a remote node using a remote shell (rsh or remsh).

- Environment

  This command can be used in a system with a Tmax system installed.

- Examples
  - The following executes all Tmax processes and application server processes by referring to a <tmconfig> file in a config directory under a TMAXDIR directory:

```
$ tmboot
```

- The following starts all TMS created with tms_name:

```
$ tmboot -t tms_name -k all
```

- In the following, a configuration option of a server group is used. When several server groups use the same tms_name, the name of a server group that includes a corresponding TMS can be specified with a [-g] option. If a name is not specified, the TMS of a first server group with a specified name is started.

```
$ tmboot -t tms_name -k all -g svgname
```

- The following executes all application server processes defined in a SERVER section by referring to a <tmconfig> file:

```
$ tmboot -A
```

- The following executes application server processes in a cosmo node registered in a NODE section by referring to a <exconfig> configuration file in a /user1/tmax/con directory:

```
$ tmboot -n cosmo -f /user1/tmax/con/exconfig
```

- The following executes five svr1 processes by referring to a <tmconfig2> configuration file:

```
$ tmboot -s svr1 -k 5 -f tmconfig2
```

For more information about the cfl and tmdown commands, refer to cfl in *Tmax Reference Guide* and tmdown.

tmconfig path reference in tmboot

When a tmboot command is used to boot Tmax, a <tmconfig> binary configuration file in ${TMAXDIR}/config/ is copied to ${TMAXDIR}/path/ and a configuration file under a ${TMAXDIR}/path is used. However, if ${TMAXDIR}/config/tmconfig is referenced to start a specific server with -S or -s options and a Tmax engine is already started, a problem can arise in the environment.

- Configuration File

  <Original Configuration File>

```
*SVRGROUP
svg1       NODENAME = "tmaxh4"
*SERVER
svr1       SVGNAME = svg1
svr2       SVGNAME = svg1
*SERVICE
TOUPPER1   SVRNAME = svr1
TOUPPER2   SVRNAME = svr2
```

<Changed Configuration File during System Operation>

```
*SVRGROUP
svg1       NODENAME = "tmaxh4"
*SERVER
svr1       SVGNAME = svg1
svr3       SVGNAME = svg1
svr2       SVGNAME = svg1
*SERVICE
TOUPPER1   SVRNAME = svr1
TOUPPER3   SVRNAME = svr3
TOUPPER2   SVRNAME = svr2
```

Use CFL to recompile a configuration file that was changed during system operation.

```
$ cfl –i node1.m
```

Start a newly added server.

```
$ tmboot –S svr3
```

If tmboot -S is used after changing a configuration during runtime, the following error can occur.

```
(E) BOOT3007 maxsvr (1) is over for svr(svr3:svr2): nodeno = 0, svri = 5, cur = 1, ksvr = 1
[BOOT0015]
```

When CFL is used in an operating environment, changes are reflected in ${TMAXDIR}/config/tmconfig but actual shared memory is specified as defined in ${TMAXDIR}/path/tmconfig. If a newly added server process using tmboot -S is booted, a server process that is already running can be restarted, which can result in a problem. The CFL command is not allowed when a Tmax engine has been started. If CFL is used, the resulting error can cause a critical problem in the system operation and hinder debugging.

The path to a <tmconfig> file referenced when booting each server with a tmboot command with [–S], [-s], [–g], [-q], [-t], and [-A] options during the operation of a Tmax engine does not use ${TMAXDIR}/config/tmconfig but instead uses ${TMAXDIR}/path/tmconfig. However, ${TMAXDIR}/path/tmconfig is referenced when starting the engine.

```
$ cfl –i new_config.m –o tmchg
$ tmadmin : cfgadd –I tmchg
$ tmboot –S new_svr –f tmchg
```

When executing a tmboot, if a specific binary configuration file is designated using a [-f] option, ${TMAXDIR}/config/tmconfig will be used to start a server. To dynamically add a server, a specific configuration file must be designated by using a [–f] option. The server is then started by using the changed binary configuration file placed in ${TMAXDIR}/config/.

4.2. Shutting Down Tmax

The Tmax binary configuration file is used to shut down Tmax system. During the shutdown process, shared memory, which was used by the system, is cleared, and all running Tmax processes (TMM, TMS, CLL, CLH, RQS) and application programs are terminated.

4.2.1. tmdown

The tmdown command shuts down a Tmax system (or a portion of it). tmdown shuts down a Tmax system by referring to a Tmax configuration file, while a binary Tmax configuration file (a result of compiling 'srcconfig' with 'cfl ': See cfl) must be specified with a path by using a [-f] option.

If a [-f] option is not specified, the **tmconfig** file in a TMAXDIR/config directory will be used by default. If only a [-f] option is used, tmdown will end all Tmax management processes and all server processes registered in the SERVER section of a Tmax configuration file and will remove IPC resources related to a Tmax system. The following is the termination order:

1. Application server processes registered in the SERVER section are ended.
2. Any active TMS process in each server group is terminated.
3. Tmax management processes are terminated: first **CLH**, second **CLL**, and then **TMM**. However, if the MIN value of CLH is not 1, CLL may be terminated before CLH.

It is assumed that a service is dynamically registered in a server and the service is removed from the shared memory because the server is shut down. In this case, if fault tolerance is activated, a naming service cannot be provided for clients that accessed a backup node. (All backup servers are terminated and servers running on normal nodes are restarted.) Therefore, dynamic services of backup servers should not be deleted from shared memory even after a server is shut down.

- Usage

```
$tmdown [-A] [-f binary Tmax configuration file name] [-g servergroup_name] [-h] [-V][-i]
        [-n node_name] [-p server_num] [-q RQ svg_name]
        [-s server_name [-k count]] [-S server_name] [-t TMS_name[-k all]]
```

```
        [-w wait_time] [-B Nodename][-R] [-y]
```

Option	Description
[-A]	Terminates all application server processes.
[-f *_binary Tmax configuration file name_*]	A binary Tmax configuration file referred to when shutting down a system. If a file name is not specified, the **tmconfig** file in the $TMAXDIR/config directory will be used by default.
[-g *servergroup_name*]	Terminates all server processes of a specified server group.
[-h]	Shows command line help.
[-V]	Shows the version of an executable file.
[-i]	Immediately executes the tmdown command. Without an [-i] option, the tmdown command is executed after all current operations are complete. With this option, all processes are unconditionally terminated and tmdown is executed. Use this option carefully.
[-n *node_name*]	Ends server processes of a specific node. The node name must be pre-registered in the **NODE** section of a Tmax configuration file.
[-p *server_num*]	Ends a specified server process. Unlike a [-s] option, this option ends a specific process using a process number (spr_no), which can be displayed with a "st -p" command through tmadmin.
[-q *RQ svg_name*]	Terminates a RQS.
[-s *server_name* [-k *count*]]	Ends only specified server processes. The server process names must be pre-registered in the SERVER section of a Tmax configuration file.
[-S *server_name*]	Terminates all specified server processes.
[-t *TMS_name* [-k all]]	Terminates only a specified TMS process. [-k all] activates the Transaction Recovery feature. Recovery is performed in a TMS Group unit. This option is used to boot/terminate all TMS that have a specific name.
[-w *wait_time*]	Begins tmdown after wait_time seconds.
[-B *Nodename*]	Terminates a TRB node.
[-R]	Used for rolling down.
[-y]	Terminates a Tmax system without asking whether or not to terminate (y | n).

- Examples
  - The following is an example of shutting down an entire Tmax system by referring to a <tmconfig> file in the TMAXDIR/config directory. Tmax management processes and application processes are all terminated.

```
$ tmdown
```

- The following is an example of shutting down all TMSs named tsm_name.

```
$ tmdown -t tms_name -k all
```

- The following is an example of using a server group configuration option. If multiple server groups have the same tms_name, use a [–g] option to specify the name of the server group to which the TMS belongs.

  If a server group name is not specified, the first server group with a matching TSM name will be terminated.

```
$ tmdown -t tms_name -k all -g svgname
```

- The following is an example of shutting down an entire Tmax system by referring to a <tmconfig2> file.

```
$ tmdown -f tmconfig2
```

- The following is an example of terminating all application server processes in a svr1 by referring to a <tmconfig> file.

```
$ tmdown -S svr1
```

- The following is an example of forcibly terminating application server processes in a svr1 by referring to a <tmconfig> file. Any server process not terminated immediately can be terminated by using a [-i] option.

```
$ tmdown -S svr1 -i
```

- The following is an example of forcibly terminating a server process using a <spr_no> by referring to a <tmconfig> file. When there are multiple server processes in a server, only the server process of a server that is in a loop can be forcibly terminated.

```
$ tmdown -k <spr_no> -i
```

- The following is an example of terminating all application server processes in a 'cosmo' node registered in a NODE section by referring to a <exconfig> file in a '/user1/tmax/con' directory.

```
$ tmdown -n cosmo -f /user1/tmax/con/exconfig
```

- The following is an example of terminating only one active svr1 process by referring to a <tmconfig2> file.

```
$ tmdown -s svr1 -f tmconfig2
```

Rolling Down

In previous versions of Tmax, when a Tmax system was abnormally terminated while processing client requests, requests being processed were completed but enqueued requests would fail, and an error message was reported. However, Tmax 5 provides the Rolling Down function, which allows a server to reply to all requests (including enqueued requests) before the server terminates.

- Usage

```
$ tmdown –R –n node_name
```

Option	Description
–n node_name	A node to be terminated.

- Environment

  This command can be used in a system with a Tmax system installed.

- Example

  Assume that NODE A and NODE B are part of a multi-node (or multi-domain) system and a total of 100 clients are currently accessing NODE A.

  - When terminating the Tmax system of NODE A:

```
$ tmdown –R –n NODEA
```

1. A CLL in NODE A blocks a listening port for a client.
2. A Tmax system in NODE A finishes processing requests currently being handled by a server and then sends the results to the client.
3. The Tmax system in NODE A sends enqueued requests to NODE B, which is set as TMAX_BACKUP_ADDR.
4. The Tmax system in NODE A is shut down.
5. A Tmax system in NODE B handles the requests received from NODE A and directly replies to the client.

6. All clients connected to NODE A get the normal reply.

◦ When terminating Tmax system of NODE B:

```
$ tmdown –R –n NODEB
```

1. A CLH in NODE A distributes 100 clients requests to NODE A and NODE B equally.
2. Enter tmdown -R –n NODEB to terminate the Tmax system in NODE B.
3. A CLL in NODE B blocks a listening port for a client.
4. The Tmax system in NODE B finishes processing requests currently being handled.
5. Because the client is connected to NODE A, the Tmax system in NODE B sends the processing results to CLH in NODE A. CLH in NODE A replies to each client.
6. The Tmax system in NODE B sends enqueued requests to NODE A, which is specified as TMAX_BACKUP_ADDR.
7. The Tmax system in NODE B is shut down.
8. The Tmax system in NODE A processes the requests and replies to each client.
9. All clients connected to NODE A get a normal reply. All 100 clients must get a normal reply.

For NODE B to process client requests instead of NODE A, TMAX_BACKUP_ADDR and TMAX_BACKUP_PORT of the client connected to NODE A must be specified to that of NODE B. Otherwise, when the Tmax system in NODE A is shut down, the enqueued client requests fail and the TPESYSTEM error is reported.

5. Tmax Management

This chapter introduces Tmax management tools provided for efficient management of Tmax.

5.1. Overview

After the Tmax system has been fully configured and is running, the administrator will need access to the management functions to dynamically modify the configuration of the system or add servers/services. The Tmax management application also enables the administrator to monitor services, including their processing state, processing count, average processing time, queuing count, and expected waiting time. Using this information, the administrator will be able to decide when to load additional server processes or terminate excess processes.

The Tmax management program, **tmadmin**, is used to dynamically manage the Tmax system while it is running. Administrators can dynamically manage the system by using tmadmin and command interpreter.

5.2. tmadmin

The Tmax management program, tmadmin, is a command interpreter similar to the UNIX command line. The program waits for an input command and interprets and executes the command. In a system comprised of multiple nodes in a domain, tmadmin enables centralized management of the entire domain from a single node, or localized management from each node.

- Usage

```
$ tmadmin [-l] [-s|m] [-h] [-f [Config File]] [-n [Node Name]]
          [-v] [-V] [-p] [-t]
```

Option	Description
[-l]	Option to allow each node to manage its own system in a multi-node system that is managed centrally through racd.
[-s]	Option to use read only mode. This option enables users to load up to 10 tmadmin tools, but users will not be able to dynamically modify the system. (Default mode)
[-m]	Option to dynamically modify the system in master mode. It is recommended that only one administrator use the master mode. If more than one user modifies the system configuration, fatal system errors can occur. Do not modify the configuration file, instead follow the preset manual.
[-h]	Online Help.

Option	Description
[-f [*Config File*]]	Option to specify the name of the configuration binary file. Only needs to be used when the configuration binary does not use the default name (tmconfig).
[-n [*Node Name*]]	Option to monitor the specified node. When used in a multi-node environment, tmadmin only monitors the information on the specified node for convenient system management.
[-v]	Option to check the Tmax version number. This option can be used to check the version from any location within the system.
[-V]	Option to check the version of the execution file.
[-p]	Option to print out the result screen when performing st –p, st –s, si, ci, or cfg command. (more function).
[-t]	Option to output the start and end times to execute the command. Refer to the following descriptions for detailed information about the option.

Execute tmadmin with the -t option to output the time in the following format.

```
[TIME][Type   ] : hh:MM:ss:millisec
```

Type	Description
START	Output the start time when a command is executed on the console.
END	Output the end time when command execution is complete (after executing commands on the remote node).
R_END	When a command is executed on the console and the execution on the remote node ends, the end time (not the time on the remote node but the local time where tmadmin was executed) is output.
RP_START	If the repeat command of tmadmin is executed, time before starting each execution is output.
RP_END	If the repeat command of tmadmin is executed, the end time of each execution is output (after executing commands on the remote node).

Command Execution

In tmadmin, the Tmax system can be started up or shut down through a command, and the administrator can search and modify the configuration of the running system. It can also be used to check the server process state and service processing state.

The tmadmin program is executed with the following command.

```
$tmadmin
```

Once executed, a prompt will appear to indicate that tmadmin has been properly started.

```
--- Welcome to Tmax Admin (Type "quit" to leave) ---
$$1 tmax1 (tmadm):
```

To terminate tmadmin, use the 'quit' or 'q' command.

```
$$1 tmax1 (tmadm):quit
```

The following are commands available in tmadmin.

- Configuration information commands

Command	Description
config (cfg)	Displays environment configuration information.
configopt (cfgopt)	Displays current settings that can be modified dynamically among the configured options such as TMMOPT.
history (hist)	Displays command history (commands used previously).
tmaxinfo (ti)	Displays Tmax system information.

- Status information commands

Command	Description
stat (st)	Provides statistics on process and service states.
gwinfo	Checks the channel states of the gateways defined in the GATEWAY section.
txgwinfo (txgwi)	Displays Tmax gateway information.
nontxgwinfo	Displays Tmax non-transaction gateway information.
jgwinfo	Displays JEUS gateway information.
ajgwinfo	Displays JEUS async gateway information.
wsgwinfo	Displays Web service gateway information.
smtrc	Checks the service status using GID.
clhsinfo	Checks the connection status between CLHs in a multi node environment.
tmmsinfo	Checks the connection status between TMMs in a multi node environment.
repeat (r)	Repeats the command.
clientinfo (ci)	Displays information about clients connected to the system.
svrinfo (si)	Displays server information.

Command	Description
txquery (txq)	Displays information about transaction processing.
rqstat (rqs)	Displays RQ status, or processes services accumulated in the disk queue.

- Administrative commands

Command	Description
suspend (sp)	Suspends a running server process.
resume (rs)	Resumes a suspended server process.
advertise/unadvertise	Advertises/unadvertises a name of a particular service.
restat	Resets the statistics on a specific server process or all processes.
rebootsvr (rbs)	Restarts a server process.
cfgadd (ca)	Dynamically adds a service.
set	Modifies configuration options dynamically.
setopt	Dynamically modifies settings among options defined in the configuration, such as TMMOPT.
qpurge (qp)	Deletes tasks that are waiting in the queue.
discon (ds)	All clients currently connected are forcibly disconnected from the system.
logstart / logend	Starts/ends logging.
chtrc	Manages trace activity.
chlog	Dynamically changes the log level of TMM, CLH, or a specific server at runtime.
chlog2	Dynamically changes the log level at runtime.
txcommit / txrollback	Reissues Commit or Rollback to terminate the transaction, if a fault occurs during transaction processing.
wsgwreload	Applies or modifies service information of the web service gateway.
restart	Restarts the server process.
notify_reconnect_clh (nrc)	Connects a specific server process to a specific CLH.
admnoti (an)	Sends events to TCS, UCS, and RDP servers.

- Other commands

Command	Description
!	Executes the previous command.
quit (q)	Ends tmadmin.
help (h)	Displays a list of available options.

Command	Description
nodeset (ns)	Sets to get information only about a specific node.
nodeunset (nus)	Releases the setting of getting information only about a specific node.
tmd	Starts a virtual client emulator used to check service validity without creating a client program.

5.3. Configuration Information Commands

5.3.1. tmaxinfo (ti)

Displays Tmax system configuration information.

- Usage

```
$$1 tmax1 (tmadm): ti
```

- Example

  Displays information about the Tmax system, including the system version and the maximum allowed number of concurrent users (maxuser).

```
$$1 tmax8 (tmadm): ti

Tmax System Info: REAL version 6.0:

         maxuser = UNLIMITED,
         Supported maximum user per node = 16076,
         Supported maximum user per handler = 16077,
         domaincount = 1,
         nodecount = 1,
         svgrpcount = 1,
         svrcount = 1, svccount = 1
         rout_groupcount = 0, rout_elemcount = 0
         cousin_groupcount = 0, cousin_elemcount = 0
         backup_groupcount = 0, backup_elemcount = 0

Tmax All Node Info: nodecount = 1:
-----------------------------------------------------------------
  no   name     portno  racport  shmkey  shmsize  minclh  maxclh
-----------------------------------------------------------------
   0   tmax8      7789   3378    87789   755872       1      3
```

5.3.2. history (hist)

Displays a list of the previously executed commands.

- Usage

```
$$1 tmax1 (tmadm): history
```

- Example

  The following is an example of using the command.

```
$$15 tmax1 (tmadm): history
5 :  si
4 :  txq
3 :  ci
2 :  st  -p
1 :  st  -s
0 :  ci
```

  Enter "!" at the command line to repeat the previous command. Any of the previously entered commands can also be repeated by entering “!” and the number of the command in the history list.

```
$$1 tmax1 (tmadm): !5
si
-----------------------------------------------------------------
clh    svri     status    count    qcount     qpcount    emcount
-----------------------------------------------------------------
0        0        RDY         2         0           0          0
0        1        RDY         0         0           0          0
0        2        RDY         0         0           0          0
0        3        RDY        45         0           0          0
```

5.3.3. config (cfg)

Displays system configuration information. It enables users to check the configuration information, including the default value, of the domain, node, server group, server, and service. For details about the information accessed through this command, refer to Environment Configuration.

- Usage

```
$$1 tmax1 (tmadm): config (cfg) [-d] [-n] [-g[server_group_name]]
                                 [-v[service_name]] [-s [service_name] [-x]]
                                 [-w [gateway_name]] [-r] [-b] [-f] [-pr]
```

Option	Description
[-d]	Option to display information about domain configuration. Displays all items in the DOMAIN section and all information on the domain.
[-n]	Option to display node settings. Displays all items in the NODE section and node settings defined on the system.
[-g [*server_group_name*]]	Option to display server group settings. If the server group name is specified, only information about the server group is displayed, otherwise information about all server groups is displayed. All items related to the SVRGROUP section can be displayed for each server group.
[-v [*service_name*]]	Option to display server settings. If a server name is specified, only information about the server is displayed, otherwise information about all servers is displayed. For each server, all items related to the SERVER section, server numbers(svr_no), and data dependent routing information (DDRI) are displayed.
[-s [*service_name*] [-x]]	Option to display service settings. If a service name is specified, only information about the service is displayed, otherwise information about all services is displayed. For each service, all items related to the SERVICE section is displayed. Using the [-x] option displays txtime information.
[-w [*gateway_name*]]	Option to display gateway settings. If a gateway name is specified, only information about the gateway is displayed, otherwise information about all gateways is displayed.
[-r]	Option to display routing settings.
[-b]	Option to display backup settings.
[-f]	Option to display TopEnd function information.
[-pr]	Option to display TopEnd product information.

- Example
  - Domain settings (-d)

    The following is an example of checking the domain settings.

```
$$1 tmax8 (tmadm): cfg -d
    domain_name = tmax1,
         shmkey = 79190,
         minclh = 1,
         maxclh = 3,
         maxuser = UNLIMITED,
         portno(portno) = 7789,
         racport = 3333,
         cmtret = YES,
         blocktime(bt) = 30.000,
```

```
        txtime(tt) = 0.000,
        nliveinq(ni) = 30,
        security = NONE,
        cpc = 1,
        maxfunc = 0,
        clichkint(clichkint) = 0,
        idletime(idletime) = 0,
        node_count = 1,
        svg_count = 6,
        svr_count = 14,
        cousin_count = 0,
        cousin_gcount = 0,
        backup_count = 0,
        backup_gcount = 0,
        rout_count = 0,
        rout_gcount = 0,
        maxsacall = 8,
        maxcacall = 16,
        nclhchktime(nclhchktime) = -1,
        txpendingtime(txpendingtime) = 0,
        maxconv_node = 16,
        maxconv_server = 8,
        maxnode = 32,
        maxsvg = 32,
        maxspr = 64,
        maxsvr = 64,
        maxsvc = 512,
        maxcpc = 150,
        maxtms = 32,
        maxrout = 16,
        maxroutsvg = 32,
        maxrq = 2,
        maxgw = 2,
        maxcousin = 16,
        maxcousinsvg = 32,
        maxbackup = 16,
        maxbackupsvg = 32,
        maxtotalsvg = 64,
        maxprod = 0,
        tipsvc = ,
        crypt = NO,
        maxthread = 128,
        tmmloglvl = DETAIL,
        clhloglvl = DETAIL,
        tlmloglvl = DETAIL,
        casloglvl = DETAIL,
        rsloglvl = DETAIL,
        sqloglvl = DETAIL,
        tgloglvl = DETAIL,
        tmsloglvl = DETAIL,
        hmsloglvl = DETAIL,
        rqloglvl = DETAIL,
        gwloglvl = DETAIL,
        cllloglvl = DETAIL,
        loglvl = DETAIL,
        cllblock = NO,
        tdl = NO,
        maxconvn = 16,
        maxconvs = 8,
```

```
        domainid = 0,
        fdlversion = 1,
        tgshmsize = 0
        tgmax = -1
        tgmax_child = -1
        tgmax_watcher = -1
        tgheartbeat = 0
        tgtimeout = 0
        tg_mcast_ip = 224.0.0.100,
        tg_portno = 9999,
        tghistorycount = 50
        tgstandbycount = 50
        tgmaxbuffersize = 65000
        tgdownwaittime = 30
```

After the command, cfg -d, is executed, the following items are displayed.

Item	Description
nodecount	Number of nodes in the domain
svgcount	Number of server groups
cousin_gcount	Number of groups with COUSIN defined (number of replicated server groups for load balancing)
cousin_count	Number of all replicated server groups
rout_count	Number of routing objects defined in the ROUTING section
relem_count	Number of routing units for each range, which is defined in the range item of each routing object

◦ All nodes settings (-n)

The following is an example of checking all node settings.

```
$$1 tmax8 (tmadm): cfg -n
    node_name = tmax8, hostname = tmax8, node_no = 0
         load = 0,
         shmkey = 79190,
         minclh = 1,
         maxclh = 3,
         maxuser = UNLIMITED,
         Supported maximum user per node = 15931,
         Supported maximum user per handler = 15932,
         clhqtimeout(cqt) = 0,
         portno(portno) = 7789,
         racport = 3333,
         tmaxhome = /data/tmaxha/tmax/,
         tmaxdir = /data/tmaxha/tmax/,
         appdir = /data/tmaxha/tmax/appbin/,
         pathdir = /data/tmaxha/tmax/path/,
         tlogdir = /data/tmaxha/tmax/log/tlog/,
         slogdir = /data/tmaxha/tmax/log/slog/,
         ulogdir = /data/tmaxha/tmax/log/ulog/,
         envfile = ,
```

```
        svgcount = 6,
        svrcount = 14,
        svccount = 24,
        curclh = 1,
        sprcount = 14,
        tmscount = 6,
        cpccount = 10,
        cmprsize(cmprsize) = -1,
        ipcperm = 1c0,
        clichkint(clichkint) = 0,
        idletime(idletime) = 0,
        tmmopt = ,
        clhopt = ,
        tlmopt = ,
        realsvr = ,
        rscpc = 4,
        maxsvg = 32,
        maxsprs = 64,
        maxsvr = 64,
        maxtmss = 32,
        maxcpc = 150,
        maxgwsvr = 2,
        maxrqsvr = 2,
        maxgwcpc = 8,
        restart(rs) = YES,
        maxrstart(mr) = -1,
        gperiod(gp) = 86400,
        autobackup(ab) = YES,
        extport = 0,
        maxthread = 128,
        cllblock(cb) = NO,
        cllconnlb(cllconnlb) = RR,
        cllbindip = YES,
        tdl = NO,
        sqkey = 78550,
        sqsize = 8388608,
        sqmax = 1024,
        sqkeymax = 64,
        sqtimeout = 30,
        smsupport = NO,
        msgsizewarn(msw) = 1073741824,
        msgsizemax(msx) = 1073741824,
        tgshmkey = -1,
        tgid = -1,
        tgmcast_ttl = 1,
        tgmcast_if = ,
        crypt_algorithm = ""
        svclog_format = ""
        casthread = -1
        cas = ""
        security_lib = ""
        crypt_lib = ""
```

After the command, cfg -n, is executed, the following items are displayed.

Item	Description
node_no	Node number defined in the system

Item	Description
svgcount	Number of server groups on the node
svrcount	Number of servers
svccount	Number of services
curclh	Number of CLH processes active on the node
maxsprs	Maximum allowed number of server processes
maxtms	Maximum allowed number of TMS processes
autobackup	Option to automatically start backup servers included in svg. Set to either Y or N. When a main node is downed by tmdown -i, set this option to N to prevent that backup servers start. Execute tmdown, and then set to Y. If a node where tmadmin is executed or a connected node is not a node for which the set command is executed, the modification is not applied and the following error message is displayed. `no such name(nodename) for the section defiend in config.` To change settings of another node, racd must be executed for the node.

◦ Server group settings (-g)

The following is an example of checking the server group settings.

```
$$2 tmax8 (tmadm): cfg -g svg2
    svg_name = svg2, svg_no = 3
         xaoption = ,
         openinfo =
Oracle_XA+Acc=P/scott/tiger+SesTm=60+DbgFl=7+LogDir=/data/tmaxha/tmax/log/xalog,
         closeinfo = ,
         appdir = ,
         ulogdir = ,
         svgtype = TMAX,
         envfile = ,
         tmsname = tms_ora,
         mintms = 2,
         maxtms = 3,
         curtms = 2,
         tmstype = STD,
         tmsthreads = 0,
         tmsopt =  -o svg2 -e svg2,
         tmsrecovery = YES,
         tmsloglvl = DEBUG4,
         tmsrange = DOMAIN,
         tmsxatime(tmsxatime) = 0,
```

```
         load(ld) = -9,
         tms starti = 0,
         tms endi = 2,
         restart(rs) = YES,
         maxrstart(mr) = -1,
         gperiod(gp) = 86400,
         autobackup = YES,
         dummy = NO,
         dbname = ORACLE
         rmid = 0
         svclog_format = ""
```

◦ Server settings (-v)

The following is an example of checking the server settings.

```
$$4 tmax8 (tmadm): cfg -v svr2
svr_name = svr2, svr_no = 5
         svgno = 2,
         cursvr = 1,
         clopt = ,
         seq = -1,
         minsvr = 1,
         maxsvr(max) = 1,
         ulogdir = ,
         maxqcount(mq) = -1,
         asqcount(aq) = -1,
         conv = NO,
         ddri = DDR_NO_ROUT,
         lifespan(ls) = -1,
         restart(rs) = YES,
         maxrstart(mr) = 5,
         gperiod(gp) = 86400,
         svrtype = TMAX_STD,
         schedule(schedule) = FA,
         cpc = 1,
         dummy = NO,
         aus = NO,
         mac(mac) = NO,
         roc(roc) = NO,
         multiclh = YES,
         ctx_ereply = NO,
         svrqtimeout(sqt) = -1,
         inbound_cpc = 1,
         svclog_format = ""
```

◦ Service settings (-s)

The following is an example of checking the service settings.

```
$$5 tmax8 (tmadm): cfg -s

-------------------------------------------------------------------------------------------
   svc_name          funcname      prio(pr)  autotran  svctime(st)  routno  svrname
svgno
```

```
-------------------------------------------------------------------------------------
   _hms01                          50          NO          0.000      -1    _hms01
4
   TOLOWER                         50          NO          0.000      -1    svr2
2
   TOUPPER                         50          NO          0.000      -1    svr2
2
   HMS                             50          NO          0.000      -1    svr_hms
2
   FDLTOLOWER                      50          NO          0.000      -1    svr3
2
   FDLTOUPPER                      50          NO          0.000      -1    svr3
2
   gw1                             50          NO          0.000      -1    gw1
0
   gw2                             50          NO          0.000      -1    gw2
1
   SDLTOLOWER                      50          NO          0.000      -1    svr1
2
   SDLTOUPPER                      50          NO          0.000      -1    svr1
2
   _rq1                            50          NO          0.000      -1    _rqsvg
5
   LOGIN                           50          NO          0.000      -1    svr_ucs
2
   TPDEQ                           50          NO          0.000      -1    svr_rq
2
   TPENQ                           50          NO          0.000      -1    svr_rq
2
   FDLDEL                          50          NO          0.000      -1    fdltest
3
   SDLDEL                          50          YES         0.000      -1    sdltest
3
   FDLSEL                          50          NO          0.000      -1    fdltest
3
   FDLINS                          50          NO          0.000      -1    fdltest
3
   TOUPPER_CONV                    50          NO          0.000      -1    svr_conv
2
   SDLSEL                          50          YES         0.000      -1    sdltest
3
   SDLINS                          50          YES         0.000      -1    sdltest
3
   FDLUPT                          50          NO          0.000      -1    fdltest
3
   SDLUPT                          50          YES         0.000      -1    sdltest
3
   GET_SQ                          50          NO          0.000      -1    svr_sq
2
```

◦ Gateway settings (-w)

The following is an example of checking the gateway settings.

```
$$7 tmax8 (tmadm): cfg -w gw1
    gw_name = gw1, node_no = 0, gw_no = 0
```

```
    gw_type = TMAXNONTX,
    portno = 4021,
    rgwaddr = 192.168.1.86,
    rgwportno = 4021,
    backup_rgwaddr = ,
    backup_rgwportno = -1,
    backup_rgwaddr2 = ,
    backup_rgwportno2 = 0,
    backup_rgwaddr3 = ,
    backup_rgwportno3 = 0,
    cpc = 1,
    timeout(timeout) = 11000,
    direction(direction) = BIDIR,
    maxinrgw = 32,
    clopt = -i,
    ptimeout(ptimeout) = -1,
    ptimeint(ptimeint) = -1
    gwchkint(gwchkint) = -1
    gwconnect_timeout = -1
    nliveinq(nliveinq) = -1
```

5.3.4. configopt (cfgopt)

Displays current settings that can be modified dynamically among the defined options such as TMMOPT.

- Usage

```
$$1 tmax1 (tmadm): configopt (cfgopt) [-tmm]
```

Option	Description
[-tmm]	Option to check the current settings that are dynamically configurable among options defined in the TMMOT element.

- Example
  - TMMOPT setting (-d)

    The following is an example of checking the TMMOPT setting.

```
$$1 tmax1 (tmadm): cfgopt -tmm
------------------------------------------------------------------------
        -tmm configurable value
------------------------------------------------------------------------
         accept retry count(-A) = 100
         max forked threshold(-F) = 765
         booting timeout(-t) = 10
------------------------------------------------------------------------
```

The following describes items that are displayed when the command is executed.

Item	Description
accept retry count(-A)	Number of server process access requests that can be accepted immediately. Refer to the -A option of TMMOPT.
max forked threshold(-F)	The maximum number of concurrent requests that can be processed when creating a new process due to TMM starting an additional server process or restarting a process. Refer to the -F option of TMMOPT.
booting timeout(-t)	Access timeout for a server process that was started via ASQCOUNT. Refer to the -t option of TMMOPT.

5.4. Status Information Commands

5.4.1. stat (st)

Displays the status of various elements in the system, including server processes. Information that can be displayed through this command includes the current state of server processes, the names of services being processed, the number of processed services, the state of services, and the number of service requests in the service queue. The stat command is abbreviated as **st**.

- Usage

```
$$1 tmax1 (tmadm): stat (st) [-p [server_process_name] [-b]] [-s [service_name]]
                             [-t [TMS_name]] [-v [server_process_name] [-pod]
                             [-b]] [-o [sort condition]] [-n [output message line]]
                             [-x] [-X] [-q [destination_name [-c]] ] [-d]
                             [-j] [-tg]
```

Option	Description
[-p [*server_process_name*]]	Option to display information about server processes. Servers registered in the Tmax configuration file can use MIN and MAX to start multiple processes, and this option is used to check the status of each server. If a server process name is specified, only information about the server process is displayed, otherwise information about all server processes is displayed. The overall system processing state is displayed on the last line of the 'st –p' command execution result. Statistical information including total processing count, average processing time, and the total number of active services are displayed.

Option	Description
[-s [*service_name*]]	Option to display information about services. If a service name is specified, only information about the service is displayed, otherwise information about all services is displayed.
[-t [*TMS_name*]]	Option to display dynamic information about TMS provided by the system. • If a TMS name is specified, only information about the TMS is displayed. • If a TMS name is not specified, information about all TMSs is displayed.
[-v [*server_process_name*] [-pod]]	Option to display information about server processes. (same as si) If the [-pod] option is used, "(POD)" is always displayed with the POD server status. If this option is not used, RDY or NRDY is displayed when a POD server is running or not running respectively. POD server's status is not important because it automatically starts when there is a service request even when the status is NRDY. Use this option to avoid mistaken a POD server in the NRDY status as a server error.
[-o [*sort condition*]]	Option to sort output results by the specified condition. Dynamic information about server processes and services provided by the Tmax system can be sorted by the specified condition. This option must be used with the [–p] or [–s] options. Use one of the following options to sort either in descending or ascending order. • [-o ca]: descending order • [-o ca-]: ascending order
[-n [*output message line*]]	Option to set the number of lines of sorted messages to display (number of server processes or services). This option must be used with the [–o] option.
[-x]	Option to display the following details when using 'st –p' or 'st –s' command to search for the state of server processes or services. Must be used with either [–s] or [–p] option. For more details, refer to "Detailed Info (-x)". • Fail count, Error count, and Process ID(PID) provided by OS • Minimum/maximum service processing times (min_time, max_time) • xid and xa_status used to search for TMS(-t) information

Option	Description

[-X]	• When using with the [-s] option
	
	Searches for the local status of COUSIN service in the form of status1(status2). (Example: RDY(NRDY))
	
	(Status2) is the actual local status. This option prevents the problem of a local status always being displayed as RDY in a COUSIN environment. Must be used with the [–s] option.
	
	• When using with the [-p] option
	
	Searches for suspended server processes. A suspended server process is displayed as follows. (Example: RDY(BLK:Pp))
	
	Only some of the server processes can be suspended. In such a case, the server status is displayed as RDY when using 'stat -v', and each suspended process is displayed with '(BLK:Xx)' appended to the end of the actual status string when using 'stat -p -X'. Xx depends on the -n and -N options used when running the sp command.
	
	◦ Pp
	
	`sp -n P -N P`
	
	◦ Tp
	
	`sp -n T -N P`
	
	◦ Tt
	
	`sp -n T -N T`
	
	◦ Pt
	
	`sp -n P -N T`

[-q [*destination_name*]]	Option to display information about HMS destinations. Displays a list of destinations defined in the configuration file, and the messages and clients being processed at each destination.

Option	Description
[-d]	Option to display the elapsed time when a server process is queried (st -p) and its status is RUN. '-' is displayed for any other statuses. Must be used with 'st -p'. (Example: st -p -d, st -p -x -d)
[-b]	Option to display last server boot time when retrieving servers (st -v) or server processes (st -p).

[-j]	Option to display items that span in multiple lines as a single line when using st -p -x and st -p -d.
	
	Used along with 'st -p'.
	
	`st -p -j`
	`st -p -x -j`
	`st -p -d -j`
	`st -p -d -x -j`

[-tg]	Option to display information about TmaxGrid.

- Example
  - Server process information (–p)

    The following is an example of checking server process information.

```
$$24 tmax8 (tmadm): st -p

CLH 0:
   ------------------------------------------------------------------------
   svr_name     svgname     spr_no     status      count        avg        svc
   ------------------------------------------------------------------------
   gw1          gw1             32        RDY          0      0.000         -1
   gw2          gw2             33        RDY          0      0.000         -1
   _hms01       hms01           34        RDY          2      0.000         -1
   _hms01       hms01          105        RDY          0      0.000         -1
   _hms01       hms01          106        RDY          0      0.000         -1
   _hms01       hms01          107        RDY          0      0.000         -1
   _rqsvg       rqsvg           35        RDY          0      0.000         -1
   _rqsvg       rqsvg          109        RDY          0      0.000         -1
   _rqsvg       rqsvg          110        RDY          0      0.000         -1
   _rqsvg       rqsvg          111        RDY          0      0.000         -1
   svr1         svg1            36        RDY          0      0.000         -1
   svr2         svg1            37        RDY          0      0.000         -1
   svr3         svg1            38        RDY          0      0.000         -1
   svr_ucs      svg1            39        RDY          0      0.000         -1
   svr_conv     svg1            40        RDY          0      0.000         -1
   svr_rq       svg1            41        RDY          0      0.000         -1
   svr_sq       svg1            42        RDY          0      0.000         -1
   svr_hms      svg1            43        RDY          0      0.000         -1
   fdltest      svg2            44        RDY          0      0.000         -1
   sdltest      svg2            45        RDY          0      0.000         -1
   ------------------------------------------------------------------------
```

```
   TOTAL COUNT = 2
   TOTAL AVG = 0.000
   TOTAL RUNNING COUNT = 0
```

After the command 'st -p' is executed, the following items are displayed.

Item	Description
svr_name	Server process name.
svgname	Name of the server group that the server process belongs to.
spr_no	ID of the server process.
status	Current status of the server process. The following describes each system status. • RDY: Ready. • NRDY: Not ready. The tpstart has been executed and client is connected to a socket, but data cannot be transmitted from the server. This status may occur due to a network problem. • RUN: Running. • Unregisted: Unregistered. Shown to the client immediately after tpend execution before the status is changed. • BLK: All processes of the service are suspended. • PBLK: Only some server processes of the service are suspended. • UNADV: All server processes of the service are in the unadvertise status. • PUNADV: Only some server processes of the service are in the unadvertise status.
count	Number of services processed.
avg	Average processing time.
svc	Name of the service being processed. If no service is currently being processed, -1 is displayed.

Wild cards can be used. For example, the command 'st –p s*' will display information about all server processes with names that begin with 's'.

```
$$25 tmax8 (tmadm): st -p s*

CLH 0:
   ------------------------------------------------------------------------
   svr_name     svgname     spr_no     status      count        avg        svc
   ------------------------------------------------------------------------
   svr1         svg1            36        RDY          0      0.000         -1
   svr2         svg1            37        RDY          0      0.000         -1
```

```
   svr3         svg1            38        RDY          0      0.000         -1
   svr_ucs      svg1            39        RDY          0      0.000         -1
   svr_conv     svg1            40        RDY          0      0.000         -1
   svr_rq       svg1            41        RDY          0      0.000         -1
   svr_sq       svg1            42        RDY          0      0.000         -1
   svr_hms      svg1            43        RDY          0      0.000         -1
   sdltest      svg2            45        RDY          0      0.000         -1
   ------------------------------------------------------------------------------
   TOTAL COUNT = 0
   TOTAL RUNNING COUNT = 0
```

◦ Service information (–s)

The following is an example of checking the service information.

```
$$13 tmax8 (tmadm): st -s

CLH 0:

------------------------------------------------------------------------------------------------
---
   svc_name                          svr_name    count    q_cnt   aq_cnt    q_avg      avg
status

------------------------------------------------------------------------------------------------
---
   _hms01                            _hms01          2        0        1    0.000    0.000
RDY
   TOLOWER                           svr2            0        0        0    0.000    0.000
RDY
   TOUPPER                           svr2            0        0        0    0.000    0.000
RDY
   HMS                               svr_hms         0        0        0    0.000    0.000
RDY
   FDLTOLOWER                        svr3            0        0        0    0.000    0.000
RDY
   FDLTOUPPER                        svr3            0        0        0    0.000    0.000
RDY
   gw1                               gw1             0        0        0    0.000    0.000
RDY
   gw2                               gw2             0        0        0    0.000    0.000
RDY
   SDLTOLOWER                        svr1            0        0        0    0.000    0.000
RDY
   SDLTOUPPER                        svr1            0        0        0    0.000    0.000
RDY
   _rq1                              _rqsvg          0        0        0    0.000    0.000
RDY
   LOGIN                             svr_ucs         0        0        0    0.000    0.000
RDY
   TPDEQ                             svr_rq          0        0        0    0.000    0.000
RDY
   TPENQ                             svr_rq          0        0        0    0.000    0.000
RDY
   FDLDEL                            fdltest         0        0        0    0.000    0.000
RDY
   SDLDEL                            sdltest         0        0        0    0.000    0.000
```

```
RDY
   FDLSEL                          fdltest         0        0        0    0.000    0.000
RDY
   FDLINS                          fdltest         0        0        0    0.000    0.000
RDY
   TOUPPER_CONV                    svr_conv        0        0        0    0.000    0.000
RDY
   SDLSEL                          sdltest         0        0        0    0.000    0.000
RDY
   SDLINS                          sdltest         0        0        0    0.000    0.000
RDY
   FDLUPT                          fdltest         0        0        0    0.000    0.000
RDY
   SDLUPT                          sdltest         0        0        0    0.000    0.000
RDY
   GET_SQ                          svr_sq          0        0        0    0.000    0.000
RDY
```

After the command 'st -s' is executed, the following items are displayed.

Item	Description
svc_name	Service name.
svr_name	The name of the server that the service belongs to.
count	Number of services processed.
avg	Average processing time.
q_count	Number of service requests currently in the queue.
aq_count	Number of service requests that spent time in the queue.
q_avg	Average service waiting time.
status	Current service status. For more information, refer to the previous description of "system status".

It is possible to use a wild card for the service name option. For example, the command 'st -s *W' will display information about all services ending with W.

```
$$27 tmax8 (tmadm): st -s *R

CLH 0:

------------------------------------------------------------------------------------------
---
   svc_name                        svr_name    count    q_cnt   aq_cnt    q_avg      avg
status

------------------------------------------------------------------------------------------
---
   TOLOWER                         svr2            0        0        0    0.000    0.000
RDY
   TOUPPER                         svr2            0        0        0    0.000    0.000
RDY
   FDLTOLOWER                      svr3            0        0        0    0.000    0.000
```

```
RDY
   FDLTOUPPER                     svr3           0       0       0    0.000    0.000
RDY
   SDLTOLOWER                     svr1           0       0       0    0.000    0.000
RDY
   SDLTOUPPER                     svr1           0       0       0    0.000    0.000
RDY
```

◦ TMS information (-t)

The following is an example of checking the TMS information.

```
$$29 tmax8 (tmadm): st -t

CLH 0:
   ------------------------------------------------------------------------------
   tms_name     svgname    spr_no      status   count    avg   cqcount aqcount qavg
   ------------------------------------------------------------------------------
   tms_ora      svg2            0       RDY        0    0.00      0      0     0.00
   tms_ora      svg2            1       RDY        0    0.00   (  0)    (  0)     (0.00)
```

After the command, st -t, is executed, the following items are displayed.

Item	Description
tms_name	Name of the TMS.
svgname	Name of the server group that the TMS belongs to.
spr_no	Process ID of the TMS.
status	Current status of the TMS. For more information, refer to the previous description of "system status".
count	Number of TMS requests processed.
avg	Average TMS processing time.
cqcount	Number of service requests currently in the queue.
aqcount	Number of service requests that spent time in the queue.
qavg	Average service waiting time.

For a multi-threaded TMS, the status of each thread can be displayed.

```
$$1 tmax1 (tmadm): st -t -x
CLH 0:
-----------------------------------------------------------------
 tms_name    svgname    spr_no(tid) status   count    avg   cqcount
            XID           xastate
-----------------------------------------------------------------
   tms_ora_mt   svgora1         0       RUN        0    0.00      0
        000:000:13505         commit
   tms_ora_mt   svgora1           0(  1)  RDY        0%   0.00   (  0)
        000:000:00000        -
```

```
   tms_ora_mt   svgora1         0(  2)  RDY        0%   0.00   (  0)
        000:000:00000        -
-----------------------------------------------------------------
   tms_ora_mt   svgora1         1       RDY        0    0.00   (  0)
        000:000:00000        -
   tms_ora_mt   svgora1         1(  1)  RDY        0%   0.00   (  0)
        000:000:00000        -
   tms_ora_mt   svgora1         1(  2)  RDY        0%   0.00   (  0)
        000:000:00000        -
-----------------------------------------------------------------
   tms_ora_mt   svgora2        10       RDY        0    0.00      0
        000:000:00000        -
   tms_ora_mt   svgora2        10(  1)  RDY        0%   0.00   (  0)
        000:000:00000        -
   tms_ora_mt   svgora2        10(  2)  RDY        0%   0.00   (  0)
        000:000:00000        -
-----------------------------------------------------------------
   tms_ora      svgora3        20       RDY        0    0.00      0
        000:000:00000        -
-----------------------------------------------------------------
   tms_ora      svgora3        21       RDY        0    0.00   (  0)
        000:000:00000        -
-----------------------------------------------------------------
```

◦ Detailed Info (-x)

When using with the 'st –p' command:

```
$$1 tmax8 (tmadm): st -p -x

CLH 0:
   ------------------------------------------------------------------------------
   svr_name     svgname     spr_no     status      count        avg        svc
                PID        RBS_TAG   fail_cnt    err_cnt   min_time   max_time
                utime    umin_time  umax_time      stime  smin_time  smax_time
   ------------------------------------------------------------------------------
   gw1          gw1             32        RDY          0      0.000         -1
                10702     00000000          0          0      0.000      0.000
                     0.000      0.000      0.000      0.000      0.000      0.000
   svr2         svg1            37        RDY          0      0.000         -1
                10700     00000000          0          0      0.000      0.000
                     0.000      0.000      0.000      0.000      0.000      0.000
   ------------------------------------------------------------------------------
   TOTAL COUNT = 0
   TOTAL SVCFAIL COUNT = 0
   TOTAL ERROR COUNT = 0
   TOTAL RUNNING COUNT = 0
```

When using with the 'st –s' command:

```
$$2 tmax8 (tmadm): st -s -x

CLH 0:

-------------------------------------------------------------------------------------------
```

```
---
   svc_name                         svr_name    count    q_cnt   aq_cnt    q_avg      avg
status
                                                      fail_cnt  err_cnt           mintime
maxtime
                                               utime umintime umaxtime    stime smintime
smaxtime

-------------------------------------------------------------------------------------------
---
   TOLOWER                          svr2            0        0        0    0.000    0.000
RDY
                                                         0        0            0.000
0.000
                                               0.000    0.000    0.000    0.000    0.000
0.000
   TOUPPER                          svr2            0        0        0    0.000    0.000
RDY
                                                         0        0            0.000
0.000
                                               0.000    0.000    0.000    0.000    0.000
0.000
```

After the command, st -s -x, is executed, the following items are displayed.

Item	Description
PID	Server process PID.
fail_cnt	Failure count of the server process/service.
err_cnt	Error count of the server process/service.
mintime	Minimum time used to handle a service.
maxtime	Maximum time used to handle a service.
utime	User time consumed whenever a server process executes a service.
umin_time	Minimum consumed user time.
umax_time	Maximum consumed user time.
stime	System time consumed whenever a server process executes a service.
smin_time	Minimum consumed system time.
smax_time	Maximum consumed system time.

◦ Search result; sort(–o))

The following is an example of using the [-o] option with the 'st –p' command.

```
$$1 tmax1 (tmadm): st –p –o ca
CLH 0 :
- -- - - - - - - - - - - -- - - - - - - - - - - - - - - - - -
```

```
svr_name   svgname         spr_no     status      count    avg    svc
- -- - - - - - - - - - - - -- - - - - - - - - - - - - - - - - - - -
svr2          svg1           37     RDY          10     0.000      -1
svr1          svg1           36     RDY          6     0.000      -1
svr3          svg1           38     RDY          2     0.000      -1
TOTAL COUNT = 18
TOTAL AVG = 0.000
TOTAL RUNNING COUNT = 0
```

The following is an example of using the [-o] option with the 'st –s' command.

```
$$1 tmax1 (tmadm): st –s –o ca
CLH 0:
- -- - - - - - - - - - - -- - - - - - - - - - - - - - - - - - - -
svc_name svr_name count  avg   cq_count  aq_count q_avg   status
- -- - - - - - - - - - - -- - - - - - - - - - - - - - - - - - - -
TOUPPER     svr2     10    0.000      0          0      0.000    RDY
SDLTOUPPER  svr1     5     0.000      0          0      0.000    RDY
FDLTOUPPER  svr3     2     0.000      0          0      0.000    RDY
SDLTOLOWER  svr1     1     0.000      0          0      0.000    RDY
FDLTOLOWER  svr3     0     0.000      0          0      0.000    RDY
TOLOWER     svr2     0     0.000      0          0      0.000    RDY
```

- Descending order sort conditions

Condition	Description
ca	Number of processed requests (count)
aa	Average processing time (avg)
cq	Number of requests currently in the queue (cq_count)
aq	Average number of requests in the queue (aq_count)
qa	Average queuing time (q_avg)
ce	Sorts by server based on the number of requests processed by each server.
ae	Sorts by server based on the average processing time of each server.

- Ascending order sort conditions

Condition	Description
ca-	Number of processed requests (count)
aa-	Average processing time (avg)
cq-	Number of requests currently in the queue (cq_count)
aq-	Average number of requests in the queue (aq_count)
qa-	Average queuing time (q_avg)

Condition	Description
ce-	Sorts by server based on the number of requests processed by each server.
ae-	Sorts by server based on the average processing time of each server.

◦ Specifying the number of lines of an output message (-n)

The following is an example of specifying the number of lines of an output message to 2.

```
$$1 tmax1 (tmadm): st -p -o ca -n 2
CLH 0 :
- -- - - - - - - - - - - -- - - - - - - - - - - - - - - - - - -
svr_name   svgname         spr_no     status      count    avg    svc
- -- - - - - - - - - - - -- - - - - - - - - - - - - - - - - - -
svr2          svg1           37     RDY          10     0.000      -1
svr1          svg1           36     RDY          6     0.000      -1
TOTAL COUNT = 18
TOTAL AVG = 0.000
TOTAL RUNNING COUNT = 0
```

The following is an example of specifying the number of lines of an output message to 3.

```
$$1 tmax1 (tmadm): st -s -o ca -n 3
CLH 0:
- -- - - - - - - - - - - -- - - - - - - - - - - - - - - - - - -
svc_name svr_name count  avg   cq_count  aq_count q_avg   status
- -- - - - - - - - - - - -- - - - - - - - - - - - - - - - - - -
TOUPPER     svr2     10    0.000      0          0      0.000    RDY
SDLTOUPPER  svr1      5    0.000      0          0      0.000    RDY
FDLTOUPPER  svr3      2    0.000      0          0      0.000    RDY
```

◦ HMS destination information (-q)

The following is an example of querying information about HMS Destination.

```
$$1 tmax1 (tmadm): st -q
--------------------------------------------------------------------------------
G  dest    cqcount  type   apqcnt   acqcnt  f_dscrd  t_dscrd cons_cnt prod_cnt
--------------------------------------------------------------------------------
-  queue01      58  QUEUE     169      111        0        0       30        5
-  topic01      32  TOPIC      42      283        0        0       28        3
```

After the command, st -q, is executed, the following items are displayed.

Item	Description
G	If the destination is set to GLOBAL, this is displayed as 'O', otherwise it is displayed as a hyphen (-).

Item	Description
dest	Destination name defined in the HMS section of the configuration file.
cqcount	Number of messages that have not been processed at the destination.
type	Destination type. Displayed as either QUEUE or TOPIC.
apqcnt	Total number of accumulated messages up to now.
acqcnt	The total number of messages that have been processed by a consumer. Since a Topic message is not considered to be complete until all consumers receive the message, acqcnt may become larger than apqcnt.
f_dscrd	Number of messages that could not be processed.
t_dscrd	Number of messages, with TTL set, that have expired.
cons_cnt	Number of consumers connected to the destination.
prod_cnt	Number of producers connected to the destination.

The following shows the details about the clients of a particular destination.

```
$$1 tmax1 (tmadm): st -q queue01 -c
 ------------------------------------------------------------------------------
     sesi     clid     cname      clitype qcnt  cnt  f_dscrd t_dscrd   listener
 ------------------------------------------------------------------------------
        0    0x39d prodasync        ARCV    49    0    0    0         ASYNCSVC
      0x1        0 prod41           SND      0   32    0    0
      0x1        0 prod42           SND      0   11    0    0
      0x1        0 cons51           RCV      3    0    0    0
      0x1        0 cons52           RCV      9    0    0    0

$$2 tmax1 (tmadm): st -q topic01 -c
 ------------------------------------------------------------------------------
     sesi     clid     cname      clitype qcnt  cnt  f_dscrd t_dscrd   listener
 ------------------------------------------------------------------------------
      0x1        0 prod11           PUB      0    2    0    0
      0x1        0 prod12           PUB      0    1    0    0
      0x1        0 cons11           SUB     30    0    0    0
      0x2      0x2 prod21           PUB      0   16    0    0
      0x2      0x2 prod22           PUB      0    8    0    0
      0x2      0x2 cons21           SUB     23    0    0    0
      0x3        0 prod31           PUB      0    3    0    0
      0x4      0x4 cons41           ADSUB   32    0    0    0          ASYNCSVC2
```

After the previous commands are executed, the following items are displayed.

Item	Description
sesi	Number of the session that created the client.
clid	Client ID.

Item	Description
cname	Client name.
clitype	Type of the client connected to the destination • SND: Queue type sender (created by hms_create_sender()) • PUB: Topic type publisher (created by hms_create_publisher()) • RCV: Queue type receiver (created by hms_create_receiver()) • SUB: Topic type subscriber (created by hms_create_subscriber()) • DSUB: Topic type durable subscriber (created by hms_create_durable_subscriber()) • ARCV: Queue type receiver created in an async session • ASUB: Topic type subscriber created in an async session • ADSUB: Topic type durable subscriber created in an async session
qcnt	Number of messages that each subscriber actually possesses.
cnt	Number of messages sent or received by each client.
f_dscrd	Number of messages that could not be processed.
t_dscrd	Number of messages, with TTL set, that have expired.
listener	Name of the service that will receive a message for a consumer created by an ASYNC session.

Executing the following command displays TmaxGrid information in detail.

```
$$1 tmax1 (tmadm): st -tg
TG Info:
    Preferences:
        shmkey               : 38852 (0x97c4)
        shmsize              : 40960 Kbyte
        max data             : 10000
        max watcher          : 10
        id                   : 1
        portno               : 9999 (UDP)
        heartbeat            : 1000 ms
        timeout              : 5000 ms
        historycount         : 100
        standbycount         : 10
        gqmaxbuffersize      : 1000 byte
        gqdownwaittime       : 30
    Status:
        status               : INIT
        suspend status       : NOT SUSPENDED
        created node count   : 0
        deleted node count   : 0
        set data count       : 0
```

```
        get data count       : 0
        current node count   : 0
        temporary node count : 0
        watcher count        : 0
        current data count   : 0
        total data size      : 0 byte
        current shm size     : 4460544 byte (10.64%)
```

5.4.2. gwinfo

Displays the connection status of a remote gateway from tmadmin. It also displays detailed information such as if the gateway is currently connected to the main node or the backup node and which nodes are connected to the gateway.

It can be used to check the connection information of all gateways (except for web service gateways).

- Usage

```
$$1 tmax1 (tmadm): gwinfo [-w gw_name] [-t gw_type]
```

Option	Description
[-w *gw_name*]	Option to search for a specific gateway's information. Use a gateway name defined in the GATEWAY section.
[-t *gw_type*]	Option to search for a specific gateway type. Use a type defined in the GATEWAY section. (TMAX, TMAXNONTX, JEUS, JEUS_ASYNC, TUXEDO, TUXEDO_ASYNC, or XAGW) (Example: TMAX)

- Example

  The gwinfo command results vary depending on whether the connected node is the main node or the backup node.

  - If connected to the main node

```
10/user2/starbj81>tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxs1 (tmadm): gwinfo
------------------------------------------------------------------
  gw_no   channel    type    foreign_address              status
------------------------------------------------------------------
   0   OUTCH      PRIM    tmaxc1(192.168.1.12:9400)     RDY
   1   OUTCH      PRIM    tmaxc1(192.168.1.12:9400)     RDY
   4   INCH          -     tmaxc1(192.168.1.12:1434)     RDY
   5   INCH          -     tmaxc1(192.168.1.12:1436)     RDY
```

```
   6   INCH          -     tmaxh2(192.168.1.48:58094)    RDY
   7   INCH          -     tmaxh2(192.168.1.48:58093)    RDY
```

The following describes the command result.

Item	Description
gw_no	Gateway number.
channel	Displays whether the channel is INBOUND or OUTBOUND.
type	Displays whether the connected node is the main or the backup node.
foreign_address	IP address of the connected remote gateway.
status	Connection status to webt, jtmax, and webtasync. For more information about the status, refer to "system status".

- If connected to the backup node

```
$$1 tmax1 (tmadm): gwinfo
------------------------------------------------------------
  gw_no   channel    type    foreign_address              status
------------------------------------------------------------
      0   OUTCH      BACK    tmaxh2(192.168.1.48:9200)     RDY
      1   OUTCH      BACK    tmaxh2(192.168.1.48:9200)     RDY
      6   INCH          -     tmaxh2(192.168.1.48:58091)    RDY
      7   INCH          -     tmaxh2(192.168.1.48:58092)    RDY
```

5.4.3. txgwinfo (txgwi) / nontxgwinfo

These commands display the connection status to a remote gateway. It can also be used to check whether the gateway is currently connected to a main or a backup node and to which specific node it is connected to.

txgwinfo (txgwi) is used to display Tmax Gateway information, while nontxgwinfo (ntxgwi) is used to display Tmax non-transaction (TMAXNONTX) Gateway information.

- Usage
  - txgwinfo

```
$$1 tmax1 (tmadm): txgwinfo
```

  - nontxgwinfo

```
$$1 tmax1 (tmadm): nontxgwinfo
```

- Example

  The following is an example of using each command.

  - txgwinfo

```
10/user2/starbj81>tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxs1 (tmadm): txgwinfo
------------------------------------------------------------------
  gw_no   channel    type    foreign_address              status
------------------------------------------------------------------
   0   OUTCH      PRIM    tmaxc1(192.168.1.12:9400)     RDY
   1   OUTCH      PRIM    tmaxc1(192.168.1.12:9400)     RDY
   4   INCH          -     tmaxc1(192.168.1.12:1434)     RDY
   5   INCH          -     tmaxc1(192.168.1.12:1436)     RDY
   6   INCH          -     tmaxh2(192.168.1.48:58094)    RDY
   7   INCH          -     tmaxh2(192.168.1.48:58093)    RDY
```

    After the command, txgwinfo, is executed, the following items are displayed.

Item	Description
gw_no	Gateway number.
channel	Displays whether the channel is INBOUND or OUTBOUND.
type	Displays whether the connected node is the main or the backup node.
foreign_address	IP address of the connected remote gateway.
status	Connection status to webt, jtmax, and webtasync. For more information about the status, refer to "system status".

  - nontxgwinfo

    When connected to a backup node

```
$$1 tmax1 (tmadm): nontxgwinfo
------------------------------------------------------------------
  gw_no   channel    type    foreign_address              status
------------------------------------------------------------------
      0   OUTCH      BACK    tmaxh2(192.168.1.48:9200)     RDY
      1   OUTCH      BACK    tmaxh2(192.168.1.48:9200)     RDY
      6   INCH          -     tmaxh2(192.168.1.48:58091)    RDY
      7   INCH          -     tmaxh2(192.168.1.48:58092)    RDY
```

5.4.4. jgwinfo / ajgwinfo

These commands display the connection status to JTmax or WebtAsync. It can also be used to check

whether the gateway is currently connected to a main or a backup node. Jgwinfo is used to display Java(JEUS) gateway information, and ajgwinfo is used to display Async Java(ASYNC_JEUS) gateway information.

- Usage
  - jgwinfo

```
$$1 tmax1 (tmadm): jgwinfo
```

  - ajgwinfo

```
$$1 tmax1 (tmadm): ajgwinfo
```

- Example
  - jgwinfo

    The following is an example of using the command.

```
10/user2/starbj81>tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxs1 (tmadm): jgwinfo
------------------------------------------------------------------------------
 gw_no name            channel type  foreign_address                   status
------------------------------------------------------------------------------
     0 gw1             OUTCH   PRIM  unknown(192.168.35.47:6555)         RDY
```

    After the command, jgwinfo, is executed, the following items are displayed.

Item	Description
gw_no	Gateway number.
channel	Displays whether the channel is INBOUND or OUTBOUND.
type	Displays whether the connected node is the main or the backup node.
foreign_address	IP address of the connected remote gateway.
status	Connection status to webt, jtmax, and webtasync. For more information about the status, refer to "system status".

  - ajgwinfo

    When connected to a backup node

```
10/user2/starbj81>tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---
```

```
$$1 tmax1 (tmadm): ajgwinfo
------------------------------------------------------------------------------
 gw_no name            channel type  foreign_address                   status
------------------------------------------------------------------------------
     1 gw2             OUTCH   BACK  unknown(192.168.35.47:6555)         RDY
```

5.4.5. wsgwinfo

Displays web service gateway information.

- Usage

```
$$1 tmax1 (tmadm): wsgwinfo [-i svrid[ svrid]]
```

Option	Description
[-i *svrid* [*svrid*]]	Svrid list to check.

- Example

  The following is an example of using the command.

```
10/user2/starbj81>tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxs1 (tmadm): wsgwinfo
------------------------------------------------------------------------
svrid    load_time                    current_client    max_attach_time
------------------------------------------------------------------------
2        Wed Jan 13 12:43:39 2010      0                 0.0sec
------------------------------------------------------------------------
```

  After the command, wsgwinfo, is executed, the following items are displayed.

Item	Description
svrid	Gateway's svr index.
load_time	Startup time of the web service gateway.
current_client	Number of clients that are currently connected.
max_attach_time	Maximum time spent for processing a single service.

5.4.6. smtrc

When the SMSUPPORT element in the NODE section of the configuration file is set to Y, executing 'st ?p ?x' displays GID of the running services.

- Usage

```
$$1 tmax1 (tmadm): smtrc [-a] GID0 GID1
```

Option	Description
[-a]	Option to display additional information including server process index(spri), user CPU time, system CPU time, and return information.
GID0	First four bytes of SysMaster GID in hexadecimal.
GID1	Last four bytes of SysMaster GID in hexadecimal.

- Example

The following is an example of executing smtrc to output the GID after executing 'st ?p ?x'.

```
$$1 tmax1 (tmadm): st ?p -x

CLH 0:
------------------------------------------------------------
svr_name    svgname    spr_no  status    count    avg     svc
              PID     fail_cnt  err_cnt  min_time  max_time
              SysMaster_GID
------------------------------------------------------------
   evtsvr     svg1       36      RDY         0      0.000     -1
               17980      0       0        0.000    0.000
               00000000-00000000-00000000
   svr1         svg1      37     RUN         0      0.000    SDLTOUPPER
               17981      0       0       0.000    0.000
               00000000-00000101-00000000
   svr2         svg1      38    RUN          0      0.000   SDLTOUPPER2
               17982      0      0       0.000      0.000
               00000000-00080101-00000000
   svr3         svg1      39    RUN          0      0.000   SDLTOUPPER3
               17983      0      0       0.000      0.000
               00000000-00100101-00000000
   svr_sys      svg1     40     RDY         3      0.000     -1
               17984      0      0      0.000      0.000
               00000000-00000000-00000000
------------------------------------------------------------
   TOTAL COUNT = 3
   TOTAL SVCFAIL COUNT = 0
   TOTAL ERROR COUNT = 0
   TOTAL AVG = 0.000
   TOTAL RUNNING COUNT = 3

$$1 tmax1 (tmadm): smtrc 0 0101
CLH 0:
------------------------------------------------------------
   sysmaster_global_id              status             svc_name
------------------------------------------------------------
00000000:00000101:00000000        SVC_RUNNING         SDLTOUPPER2
```

```
45670701 tmaxi1 (tmadm): smtrc -a 0 0101
CLH 0:
-----------------------------------------------------------------
   sysmaster_global_id              status             svc_name
   ctime          svctime    spri    ucpu    scpu
-----------------------------------------------------------------
00000000:00000101:00000000        SVC_RUNNING         SDLTOUPPER2
14:05:32:159       0.000     38     0.000   0.000
00000000:00000101:00010000        SVC_RUNNING         SMTRACE
14:05:32:159       0.000     40     0.000   0.000
00000000:00000101:00010000        SVC_DONE            SMTRACE
14:05:32:159       0.000     40     0.000   0.000
```

After the command, smtrc, is executed, the following items are displayed.

Item	Description
sysmaster_global_id	SysMaster GID.
status	Status of the service.
svc_name	Service name.
ctime	Log creation time.
svctime	Log service time.
spri	Index of the server processes that executed the service.
ucpu	User CPU time.
scpu	System CPU time.

5.4.7. clhsinfo

The nodes in a multi node environment must be connected each other. If node connections are unstable due to using a firewall or obsolete equipment, connections between nodes may be lost interrupting service processing. Such connection issues can be checked by using clhsinfo and tmmsinfo to check the connection status between nodes.

- Usage

```
$$1 tmax1 (tmadm): clhsinfo
```

- Example

  The following is an example of using clhsinfo when minclh and maxclh are set to 1 and 2, respectively.

```
tmaxh4@starbj81:/EMC01/starbj81/tmax/config>tmadmin
TMADMIN for rnode (tmaxh2): starting to connect to RAC
--- Welcome to Tmax Admin (Type "quit" to leave) ---
```

```
$$1 tmaxh4 (tmadm): clhsinfo

CLH 0:
--------------------------------------
  nodename    clhno     cpc   status
--------------------------------------
  tmaxh2          0       2    RDY
  tmaxh2          1       0    NRDY
--------------------------------------

CLH 1 is not available
Msg from rnode(tmaxh2):
CLH 0:
--------------------------------------
  nodename    clhno     cpc   status
--------------------------------------
  tmaxh4          0       2    RDY
  tmaxh4          1       0    NRDY
--------------------------------------

CLH 1 is not available
```

After the command, clhsinfo, is executed, the following items are displayed.

Item	Description
nodename	Node name.
clhno	CLH number.
cpc	CPC number.
status	Node status. The following describes each node status. • NRDY(NOT_READY): Not connected or disconnected with the target CLH. • RDY(READY): Connected. • REG(REGISTERED): Nodes are currently being connected. • UNR(UNREGISTERED): Node has been terminated via tmdown. • NSTRT(NCLHSTARTED): Nodes are currently being connected, and CLH of the target node has started. • CTNG(TRYINGTOCONNECT): Only TCP/IP socket is connected between nodes, but a connection message has not yet been transmitted to the target node. • NPING(ALIVECHECK): Node is connected, and is currently performing alive check. • NPING2(ALIVECHECK2): No response has been issued from the node for the alive check request.

5.4.8. tmmsinfo

Displays TMM connection information in a multi node environment.

- Usage

```
$$1 tmax1 (tmadm): tmmsinfo
```

- Example

  The following is an example of using the command.

```
tmaxh4@starbj81:/EMC01/starbj81/tmax/config>tmadmin
TMADMIN for rnode (tmaxh2): starting to connect to RAC
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh4 (tmadm): tmmsinfo
-----------------------------------------
  no   nodename     livectime    status
-----------------------------------------
   1   tmaxh2        13:53:53    RDY
Msg from rnode(tmaxh2):
-----------------------------------------
  no   nodename     livectime    status
-----------------------------------------
   0   tmaxh4        13:53:08    RDY
```

  After the command, tmmsinfo, is executed, the following items are displayed.

Item	Description
no	TMM number.
nodename	Node name.
livectime	Latest channel use time.
status	Current node status. For more information, refer to "node status".

5.4.9. repeat (r)

Repeatedly displays status information. It can be used as in the following. Repeating a command is useful for monitoring status information and debugging task processing.

- Usage

```
$$1 tmax1 (tmadm): r -k n -i ms command_to_repeat
```

Item	Description
-k n	Option to repeat n times.
-i ms	Option to repeat at ms-second intervals.
command_to_repeat	Target command to repeatedly execute.

- Example

  In the following example, st -s is to be executed at 5-second intervals 30 times.

```
$$1 tmax1 (tmadm): r -k 30 -i 5000 st -s
```

  In the following example, st -p is executed at 5-second intervals for 30 seconds.

```
$$1 tmax1 (tmadm): r -k 30 -i 5000 st -p
```

5.4.10. clientinfo (ci)

Displays information about the clients currently connected to the Tmax system.

- Usage

```
$$1 tmax1 (tmadm): ci [-s]
```

Option	Description
[-s]	Total number of connected clients.

- Example

  The following is an example of using the command.

```
$$1 tmax1 (tmadm): ci
CLH 0:
----------------------------------------------------------------
   cli_id    status    count   lastin_time   ipaddr      usrname
----------------------------------------------------------------
     0          RDY       0          20       61.77.153.1
     1          RDY       2          10       61.77.153.1     tmax
     2          RDY       4          123      61.77.153.1
----------------------------------------------------------------
Total Connected Clients = 1
----------------------------------------------------------------
```

  After the command, clientinfo, is executed, the following items are displayed.

tem	Description
cli_id	Client ID.
status	Current status of the client. The following describes each client status. • RDY: Client is normally connected to CLH. • NRDY: Only TCP/IP socket is currently connected, but TPSTART message has not yet been sent. • QED: Client service requests are accumulated in the server queue. • CTING: Client is trying to connect to CLH. • UNR: Client is executing tpend(), or NCLH is down. • RUN: Service requested by the client is executing.
count	Number of processed requests.
lastin_time	Time the client spent waiting after calling a service.
ipaddr	Client connection IP address.
usrname	User name defined in the usrname field of the TPSTART_T struct.

Too many abnormal client connections to CLH that are hanging can prevent other clients from connecting to CLH. Clients that do not send a TPSTART connection message within 60 seconds after the socket is connected will automatically be disconnected. When they are disconnected, the following error will occur.

```
(I) CLH0209 internal error : disconnect client because client didn't send tpstart
 msg for 60 sec.(192.168.1.43) [CLH0058]
```

5.4.11. svrinfo (si)

Displays information about each currently active server process.

- Usage

```
$$1 tmax1 (tmadm): si [-b] [-pod]
```

Option	Description
[-b]	Displays boottime, which means that the last boot time of the server.
[-pod]	Displays "(POD)" after the status field of POD servers.

- Example

  The following is an example of using the command.

```
$$5 tmaxs1 (tmadm): si
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
clh   svrname   (svri)   status   count   qcount  qpcount  emcount
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
0    scoresdl  (  0)     NRDY         0        0        0       0
0    bank2     (  1)     RDY          0        0        0       0
0    svr1      (  2)     RDY          0        0        0       0
0    svr2      (  2)     RDY          0        0        0       0
0    svr3      (  2)     NRDY         0        0        0       0
0    api       (  2)     RDY          0        0        0       0
0    syncrtn   (  2)     RDY          0        0        0       0
0    ucs       (  2)     NRDY         0        0        0       0
0    ucssvr    (  2)     RDY          0        0        0       0
0    broad     (  2)     NRDY         0        0        0       0
0    selins    (  2)     NRDY         0        0        0       0
```

  After the command, svrinfo, is executed, the following items are displayed.

Item	Description
clh	CLH number.
svrname	Server name.
(svri)	Server ID.
status	Current status of the server. For more information, refer to "system status".
count	Number of processed requests.
qcount	Amount of enqueued requests.
qpcount	Number of requests deleted from the queue.
emcount	Number of requests returned due to exceeding the maximum number of enqueued requests.
boottime	Last boot time of the server. Displayed if the -b option is used.

5.4.12. txquery (txq)

Displays information about transactions currently being processed.

- Usage

```
$$1 tmax1 (tmadm): txquery(txq) [-x] [-g svgname] [-w [-r] [-G gwname]]
                                 [<upper-global-xid><lower-global-xid>]
```

Option	Description
[-x]	Option to search for information about each transaction branch.
[-g svgname]	Option to search for transaction branch of the group with the svgname.
[-w]	Option to search for a domain subtransaction tree.
[-r]	Option to search with the XID of the remote domain.
[-G gwname]	Option to search for the domain gateway of gwname.
upper-global-xid	First 4 bytes of gtid of the xid to search for.
lower-global-xid	Last 4 bytes of gtid of the xid to search for.

- Example
  - Transaction search

    The following is an example of querying for transactions.

```
$$14 tmax1 (tmadm): txq
CLH0:
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
cli_id   txstime  txqcount txrcount   XID       txstate   xastate
- - - - - - - - - - - - - - - - - - - - - - - - -- - - - - - - -
c0023   8:33:34      1         0   000:000:00022 TXBEGIN    TX_OK
c0024  18:33:34      1         0   000:000:00023 TXBEGIN    TX_OK
c0025  18:33:34      1         0   000:000:00024 TXBEGIN    TX_OK
c0026  18:33:34      1         0   000:000:00025 TXBEGIN    TX_OK
c0027  18:33:34      1         0   000:000:00026 TXBEGIN    TX_OK
c0028  18:33:34      1         0   000:000:00027 TXBEGIN    TX_OK
c0029  18:33:34      1         0   000:000:00028 TXBEGIN    TX_OK
c0030  18:33:34      1         0   000:000:00029 TXBEGIN    TX_OK
c0031  18:33:34      1         0   000:000:00030 TXBEGIN    TX_OK
```

    After the command, txquery, is executed, the following items are displayed.

Item	Description
cli_id	ID of the currently connected client.
txstime	Time when the transaction was executed.
txqcount	Number of transactions waiting in the queue.
txrcount	Number of transactions already processed.
XID	Transaction ID.
txstate	Current transaction status. (TXBEGIN, TXCOMMIT, TXROLLBACK)
xastate	XA status.

  - Transaction information search (-x)

The following is an example of querying for transaction information.

```
$$6 tmaxh4 (tmadm): txquery -x
CLH 0:
------------------------------------------------------------
cli_id  clid   txstime txqcount txrcount   XID             txstate
xastate bqualno   nodename      svgname     txbstate    xabstate
------------------------------------------------------------
  c1526 0x000005f6 15:13:35  4      0    34800000:0008c087 TXBEGIN
 TX_OK   (B)00000000   tmaxh4    svg12301X      BEGIN         XA_OK
         (B)00000001   tmaxh4      gw2301X      BEGIN         XA_OK
         (B)00000002   tmaxh4      gw2302X      BEGIN         XA_OK
         (B)00000003   tmaxh4    svg12302X      BEGIN         XA_OK
  c1527 0x000005f7 15:13:35  4       0    34800000:0008c089 TXBEGIN
  TX_OK   (B)00000000   tmaxh4    svg12301X     BEGIN         XA_OK
          (B)00000001   tmaxh4      gw2301X     BEGIN         XA_OK
          (B)00000002   tmaxh4      gw2302X     BEGIN         XA_OK
          (B)00000003   tmaxh4    svg12302X     BEGIN         XA_OK
```

◦ Searching for transactions of a group (-g)

The following is an example of querying for transactions of a group.

```
$$7 tmaxh4 (tmadm): txq -g svg12301X

CLH 1:
------------------------------------------------------------
  cli_id    clid   txstime txqcount  txrcount     XID     txstate
xastate  bqualno    nodename    svgname     txbstate      xabstate
------------------------------------------------------------
  c1595 0x0000463b 15:14:10   1       0   34800001:0008e953 TXBEGIN
TX_OK   (B)00000000   tmaxh4   svg12301X     BEGIN         XA_OK
  c1596 0x0000463c 15:14:10    1       0    34800001:0008e956 TXBEGIN
TX_OK
   (B)00000000         tmaxh4    svg12301X     BEGIN         XA_OK
  c1597 0x0000463d 15:14:10    1       0  34800001:0008e957 TXBEGIN
TX_OK   (B)00000000   tmaxh4    svg12301X     BEGIN        XA_OK
```

◦ Transaction tree search (-w)

The following is an example of querying a transaction tree.

```
$$9 tmaxh4 (tmadm): txquery -w

CLH 0:
------------------------------------------------------------
  gw_no txstime txqcount txrcount    XID    RXID   txstate xastate
------------------------------------------------------------
  g0004 17:00:00    2     0 34800000:0008c38e 39800000:0003bcf7 PHASE2 TX_OK
  g0004 17:00:00    2     0 34800000:0008c38f 39800000:0003bcf8 PHASE1 TX_OK
  g0004 17:00:00    2     0 34800000:0008c391 39800000:0003bcf9 PHASE2 TX_OK
  g0004 17:00:00    2     0 34800000:0008c39a 39800000:0003bd00 PHASE1 TX_OK
  g0004 17:00:00    2     0 34800000:0008c3a1 39800000:0003bd08 PHASE1 TX_OK
  g0004 17:00:00    1     0 34800000:0008c3a3 39800000:0003bd09 sTXBEGIN TX_OK
```

```
  g0005 17:00:00    2     0 34800000:0008c38b 3e800100:000177f6 PHASE2 TX_OK
  g0005 17:00:00    2     0 34800000:0008c38d 3e800101:00014ff7 PENDING TX_OK
  g0005 17:00:00    2     0 34800000:0008c390 3e800101:00014ff8 PENDING TX_OK
  g0005 17:00:00    2     0 34800000:0008c392 3e800000:0006e050 PHASE1 TX_OK
  g0005 17:00:00    2     0 34800000:0008c396 3e800000:0006e052 PHASE1 TX_OK
  g0005 17:00:00    2     0 34800000:0008c397 3e800100:000177f8 PHASE1 TX_OK
  g0005 17:00:00    2     0 34800000:0008c398 3e800000:0006e055 PHASE1 TX_OK
  g0005 17:00:00    2     0 34800000:0008c399 3e800101:00014ff9 sTXBEGIN TX_OK
  g0005 17:00:00    2     0 34800000:0008c39c 3e800100:000177f9 PENDING TX_OK
  g0005 17:00:00    2     0 34800000:0008c39d 3e800000:0006e05a sTXBEGIN TX_OK
  g0005 17:00:00    2     0 34800000:0008c39e 3e800101:00014ffa PHASE1 TX_OK
  g0005 17:00:00    1     0 34800000:0008c39f 3e800000:0006e05b sTXBEGIN TX_OK
  g0005 17:00:00    1     0 34800000:0008c3a2 3e800100:000177fa sTXBEGIN TX_OK
  g0004 17:00:00    2     0 34800000:0008eb51 39800001:0003e50d PHASE2 TX_OK
  g0004 17:00:00    2     0 34800000:0008eb53 39800001:0003e510 PHASE2 TX_OK
  g0004 17:00:00    2     0 34800000:0008eb58 39800001:0003e516 PHASE1 TX_OK
  g0004 17:00:00    2     0 34800000:0008eb5b 39800001:0003e519 sTXBEGIN TX_OK
  g0005 17:00:00    2     0 34800000:0008eb4d 3e800001:0007084f PHASE2 TX_OK
  g0005 17:00:00    2     0 34800000:0008eb50 3e800001:00070852 PHASE2 TX_OK
  g0005 17:00:00    2     0 34800000:0008eb52 3e800001:00070854 PHASE2 TX_OK
  g0005 17:00:00    1     0 34800000:0008eb56 3e800001:00070858 sTXBEGIN TX_OK
  g0005 17:00:00    1     0 34800000:0008eb5a 3e800101:00014ffb sTXBEGIN TX_OK
  g0005 17:00:00    1     0 34800000:0008eb5c 3e800001:0007085b sTXBEGIN TX_OK
```

The following is an example of using the [-w] and [-r] options.

```
### txquery -w -r ###
$$11 tmaxh4 (tmadm): txquery -w -r

CLH 1:
-----------------------------------------------------------------
  gw_no txstime txqcount txrcount       XID           RXID      txstate xastate
-----------------------------------------------------------------
  g0004 17:00:00    2     0 34800001:0008ed0a 39800001:0003e6bc PENDING TX_OK
  g0004 17:00:00    2     0 34800001:0008ed0b 39800001:0003e6bd PENDING TX_OK
  g0004 17:00:00    2     0 34800001:0008ed0c 39800001:0003e6be PHASE1 TX_OK
  g0004 17:00:00    2     0 34800001:0008ed0f 39800001:0003e6c1 PHASE1 TX_OK
  g0005 17:00:00    2     0 34800001:0008ed00 3e800001:000709aa PHASE2 TX_OK
  g0005 17:00:00    2     0 34800001:0008ed03 3e800101:00015046 PHASE2 TX_OK
  g0005 17:00:00    2     0 34800001:0008ed04 3e800100:00017846 PHASE2 TX_OK
  g0005 17:00:00    2     0 34800001:0008ed05 3e800001:000709b1 PENDING TX_OK
  g0005 17:00:00    2     0 34800001:0008ed06 3e800100:00017847 PENDING TX_OK
  g0005 17:00:00    2     0 34800001:0008ed07 3e800001:000709b2 sTXBEGIN TX_OK
  g0005 17:00:00    2     0 34800001:0008ed08 3e800100:00017848 PENDING TX_OK
  g0005 17:00:00    2     0 34800001:0008ed09 3e800100:00017849 PHASE2 TX_OK
  g0004 17:00:00    2     0 34800001:0008c53f 39800000:0003bea9 PHASE2 TX_OK
  g0004 17:00:00    2     0 34800001:0008c540 39800000:0003beab PHASE2 TX_OK
  g0004 17:00:00    2     0 34800001:0008c545 39800000:0003beaf PHASE1 TX_OK
  g0004 17:00:00    2     0 34800001:0008c546 39800000:0003beb0 PHASE1 TX_OK
  g0004 17:00:00    2     0 34800001:0008c548 39800000:0003beb2 sTXBEGIN TX_OK
  g0005 17:00:00    2     0 34800001:0008c51c 3e800000:0006e192 PHASE2 TX_OK
  g0005 17:00:00    2     0 34800001:0008c536 3e800000:0006e1a5 PHASE2 TX_OK
  g0005 17:00:00    2     0 34800001:0008c537 3e800100:00017845 PHASE2 TX_OK
  g0005 17:00:00    2     0 34800001:0008c539 3e800000:0006e1a7 PHASE2 TX_OK
  g0005 17:00:00    2     0 34800001:0008c53e 3e800000:0006e1ad PENDING TX_OK
  g0005 17:00:00    2     0 34800001:0008c541 3e800101:00015047 PENDING TX_OK
  g0005 17:00:00    2     0 34800001:0008c542 3e800000:0006e1af PENDING TX_OK
```

```
  g0005 17:00:00    2     0 34800001:0008c543 3e800101:00015048 PENDING TX_OK
  g0005 17:00:00    2     0 34800001:0008c544 3e800101:00015049 PHASE1 TX_OK
```

5.4.13. rqstat (rqs)

Displays the status of the currently active RQs. It can also be used by an administrator to manipulate the disk queue.

- Usage

```
$$1 tmax1 (tmadm): rqstat (rqs) [-l] [-s rqname] [-f rqname] [-c rqname] [-a]
```

Option	Description
[-l]	Option to display a list of available RQs.
[-s *rqname*]	Current RQ status.
[-f *rqname*]	Option to process services accumulated on the RQ.
[-c *rqname*]	Option to delete the RQ backlog.
[-a]	Option to display a status list of all RQs.

To process RQ with the [-f] and [-c] options, it must be executed in the master mode by using the tmadmin -m option.

- Example
  - RQ status search (-a)

    Each record represents the following items.

```
$$1 tmaxi9 (tmadm): rqs -a

             rq1 RDY  0 0 0 0 0 5 5
total count : 1
```

    Data is displayed in the following format.

```
RQ_name Status Requests Replies Failures Requests_enqueued Requests_dequeued Replies_enqueued
Replies_dequeued
```

    The last record displays a total count.

  - RQ search (-I)

The following is an example of querying for available RQs.

```
$$1 tmax1 (tmadm): rqs -l
list of available RQs:
rq1 rq2
```

◦ RQ status search (-s)

The following is an example of querying for RQ status.

```
$$1 tmax1 (tmadm): rqs -s rq1
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Number of queue entries
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
request                      0
reply                        0
fail                         0
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Accumulated queue activities
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
request enqueued             0
request dequeued             0
reply enqueued               10
reply dequeued               0
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

◦ RQ service processing (-f)

The following is an example of handling services accumulated in RQ.

```
$$1 tmax1 (tmadm): rqs -f rq2
RQ(rq2) flushed
```

◦ RQ backlog deletion (-c)

The following is an example of clearing a stagnant RQ.

```
$$1 tmax1 (tmadm): rqs -c rq2
RQ(rq2) statics cleared
```

5.5. Administrative Commands

5.5.1. suspend (sp)

Suspends a running server in order to resolve system failures resulting from application program errors. Once this command has been executed, the target server processes will complete processing

current services before being suspended. Services in the queue remain in the queue and any subsequent service requests will be accumulated in the queue. The suspend command is abbreviated as **sp**.

- Usage

```
$$1 tmax1 (tmadm): Usage: suspend {-s svc_name [-p pid | -i spri] [-a]
                      [-n T|P] [-N T|P] | -v svr_name [-k num] [-a]
                      [-q] [-n T|P] [-N T|P]}
```

Option	Description
-s *svc_name*	Service name.
[-p *pid*]	Option to suspend services for the specified server process. Used with the [-s] option.
[-i *spri*]	Option to suspend services for specified spri. Used with the [-s] option.
[-n *T*|*P*]	Server process status when it restarts after ended normally. • T: RDY. • P: Status just before the server process ended.
[-N *T*|*P*]	Server process status when it restarts after ended abnormally. • T: RDY. • P: Status just before the server process ended.
-v *svr_name*	Option to suspend the scheduling of the server process. A requested service will wait in the queue until the process is resumed, or can be removed from the queue according to the MAXQCOUNT and CLHQTIMEOUT settings.

Option	Description
[-k *num*]	Option to suspend the specified number of processes on the target server. Used with the [-v] option. As many server processes, including the already suspended processes, as the specified number are suspended. The specified number must not exceed the MAX value of the server. To check for suspended server processes, execute the [stat -p -X] command. `suspend(sp) -v svr_name [-k 5]` If as many server processes as MAX are all suspended when using the [-k] option, the server will be blocked (BLK). This is same as executing 'sp -v svr_name' without the [-k] option.
[-q]	asqcount starts additional servers. Used with the [-v] option.
[-a]	If a running server receives a suspend request, the service is not suspended until it completes processing current requests. This option is used to suspend a running server immediately without delay. It is used with the -v option.

5.5.2. resume (rs)

Causes the target server processes to resume scheduling. The target server processes will resume processing of the enqueued service requests, and accept any incoming service requests. The resume command is abbreviated as **rs**.

- Usage

```
$$1 tmax1 (tmadm): resume {-s svc_name [-p pid | -i spri] |
                           -v svr_name [-k count ] }
```

Option	Description
-s *svc_name*	Option to resume service scheduling.

Option	Description
[-p *pid*]	Option to resume services for the specified server process. Suspending/resuming the service for a specific server process may take a long time to wait for the processing to complete if the service is running. In this case, the service will be resumed if the tmadmin is forcibly terminated. Any service requests after suspend will be accumulated in the queue. This option is used with the [-s] option. `resume(rs) -s svc_name [-p pid]`
[-i *spri*]	Option to resume services for the specified spri. Suspending/resuming the service for a specific spri may take a long time to wait for the processing to complete if the service is running. In this case, the service will be resumed if the tmadmin is forcibly terminated. Any service requests after suspend will be accumulated in the queue. This option is used with the [-s] option. `resume(rs) -s svc_name [-i spri]`
-v *svr_name*	Option to resume scheduling of the server process.
[-k *count*]	Option to resume the specified number of processes on the target server. Used with the [-v] option. As many suspended server processes as the specified number are resumed. If there are no suspended server processes, the command will be cancelled. The specified value must not exceed the MAX value of the server. To check for suspended server processes, execute the [stat -p -X] command. `resume(rs) -v svr_name [-k 5]` If even one server process is resumed by the -k option, the server status will be changed to RDY. If there are no active processes, the status will be NRDY.

The following describes information related to resume.

- Suspend/resume a server in the COUSIN group of a Node

  If multiple server groups are set as COUSIN of a node, suspend/resume will be executed for all servers in the COUSIN group.

ASQCOUNT can be set in the SVRGROUP section. The [-v] option must be used to set a server process to BLK.

5.5.3. advertise/unadvertise

Advertises/unadvertises a specific service name.

When a service is advertised, the name of the service will be registered in the Service Name Table, which CLH manages for each server. When the service is unadvertised, it will be deleted from the Service Name Table.

In other words, advertise allows the server process to advertise its new service, and unadvertise allows the server process to unadvertise its service. When an unadvertised service is called, a TPENOENT error will occur. However, even though a certain service is unadvertised in one server process, the service routine can be executed if another server process is providing the service.

- Usage
  - Advertise

```
$$1 tmax1 (tmadm): advertise {-s svc_name [-p pid]}
```

Option	Description
-s *svc_name*	Option to advertise the specified service name.
[-p *pid*]	Option to advertise the specified server process PID.

  - Unadvertise

```
$$1 tmax1 (tmadm): unadvertise {-s svc_name [-p pid]}
```

Option	Description
-s *svc_name*	Option to unadvertises the specified service name.
[-p *pid*]	Option to unadvertises the specified server process PID.

- Example
  - Unadvertising a service (unadvertise -s)

    The following is an example of unadvertising a service whose name is TOUPPER and server process ID is 15287. Before executing advertise or unadvertise, check the service status using the st command.

```
$$1 tmax1 (tmadm): st -p -x

CLH 0:
----------------------------------------------------------------
svr_name     svgname    spr_no    status     count     avg      svc
                 PID     fail_cnt  err_cnt  min_time  max_time
                utime  umin_time  umax_time   stime  smin_time  smax_time
----------------------------------------------------------------
svr2         svg1           36       RDY          0       0.000     -1
             15285           0       0      0.000      0.000
             0.000      0.000      0.000      0.000      0.000      0.000
svr2         svg1           37       RDY          0       0.000     -1
             15286           0       0      0.000      0.000
             0.000      0.000      0.000      0.000      0.000      0.000
svr2         svg1           38     RDY           0      0.000     -1
             15287           0       0      0.000      0.000
             0.000      0.000      0.000      0.000      0.000      0.000
----------------------------------------------------------------
   TOTAL COUNT = 0
   TOTAL SVCFAIL COUNT = 0
   TOTAL ERROR COUNT = 0
   TOTAL RUNNING COUNT = 0

$$1 tmax1 (tmadm): unadvertise -s TOUPPER -p 15287
TOUPPER is unadvertise
$$8 tmaxh4 (tmadm): st -s

CLH 0:
----------------------------------------------------------------
svc_name  svr_name   count   cq_cnt   aq_cnt   q_avg    avg   status
----------------------------------------------------------------
TOUPPER    svr2           0       0       0       0.000   0.000  PUNADV
```

◦ Advertising a service (advertise -s)

The following is an example of advertising an unadvertised service. Before executing advertise or unadvertise, check the service status using the st command.

```
$$1 tmax1 (tmadm): st -p -x

CLH 0:
----------------------------------------------------------------
svr_name     svgname    spr_no    status     count     avg      svc
                 PID     fail_cnt  err_cnt  min_time  max_time
                utime  umin_time  umax_time   stime  smin_time  smax_time
----------------------------------------------------------------
svr2         svg1           36       RDY          0       0.000     -1
             15285           0       0      0.000      0.000
             0.000      0.000      0.000      0.000      0.000      0.000
svr2         svg1           37       RDY          0       0.000     -1
             15286           0       0      0.000      0.000
             0.000      0.000      0.000      0.000      0.000      0.000
svr2         svg1           38     RDY           0      0.000     -1
             15287           0       0      0.000      0.000
             0.000      0.000      0.000      0.000      0.000      0.000
----------------------------------------------------------------
```

```
   TOTAL COUNT = 0
   TOTAL SVCFAIL COUNT = 0
   TOTAL ERROR COUNT = 0
   TOTAL RUNNING COUNT = 0

$$1 tmax1 (tmadm): advertise -s TOUPPER -p 15287
TOUPPER is advertise

$$11 tmaxh4 (tmadm): st -s

CLH 0:
----------------------------------------------------------------
svc_name    svr_name    count  cq_cnt  aq_cnt   q_avg    avg  status
----------------------------------------------------------------
TOUPPER      svr2           0       0        0      0.000    0.000  RDY
```

5.5.4. restat

Resets the statistics of the specified server process or all server processes. This command can only be used when tmadmin is running in the master mode.

- Usage

```
$$1 tmax1 (tmadm): restat [ -v [server_process_name] ] [ -a ]
```

Option	Description
[-v [*server_process_name*]]	Resets statistics of the specified server process.
[-a]	Resets statistics of all server processes.

5.5.5. rebootsvr (rbs)

Replaces the specified currently running server processes with new ones.

The directory TMAX_BKAPPDIR must be defined in .profile and the executable file of the new program must be placed under this directory. For example, move the file to $TMAXDIR/bk_appbin, and execute the following command.

- Usage

```
$$1 tmax1 (tmadm): rbs new_file old_file
```

Item	Description
new_file	Name of the new file.
old_file	Name of the existing file.

Procedure for executing rbs

The following are the steps for executing rbs.

1. Add the following variable to .profile.

```
export TMAX_BKAPPDIR=/data2/starbj81/tmax64/bk_appbin
```

2. Create the directory (bk_appbin) in the location specified in step 1.

```
$mkdir bk_appbin
```

3. Copy the execution file from appbin to bk_appbin.

```
$cp appbin/svr2 bk_appbin/
```

4. Execute tmadmin -m, and then run the rbs command.
   - If a server does not start or MIN is set to 1 and MAX is set to 1, suspending and resuming are performed when replacing the server.

     The following example starts the svr2 server process for which MIN is set to 1 and MAX is set to 1.

```
$$1 tmax1 (tmadm): rbs svr2 svr2
>> suspend ok
>> down ok
>> cp ok
>> boot ok
>> resume ok
>> reboot svr /data2/starbj81/tmax64/appbin/svr2 finished
```

   - Unless, server processes are ended and started sequentially for the replacement.

     The following example starts the svr2 server process for which MIN is set to 3.

```
$$1 tmax1 (tmadm): rbs svr2 svr2
TMBOOT for node(tmaxh2) is starting:
        TMBOOT: SVR(svrname: svr2, execname: _rbs00_svr2) is starting:
         Fri Dec 19 11:21:06 2003
TMBOOT for node(tmaxh2) is starting:
        TMBOOT: SVR(svrname: svr2, execname: _rbs00_svr2) is starting:
        Fri Dec 19 11:21:06 2003
TMBOOT for node(tmaxh2) is starting:
        TMBOOT: SVR(svrname: svr2, execname: _rbs00_svr2) is starting:
        Fri Dec 19 11:21:06 2003

>> 3 servers booted using tmp execfile _rbs01_svr2
```

```
>> reboot svr /data2/starbj81/tmax64/appbin/svr2 finished
```

5. Use the 'ps' command to verify that the server processes have been started successfully.
   - When three svr2 server processes are running

```
$ps –ef | grep svr2
starbj81 21710     1  0 11:46:32 pts/ts    0:00 _rbs00_svr2 -b -21709
                  -S svr2 -s svr2 -d -1 -v 21689
starbj81 21713     1  0 11:46:32 pts/ts    0:00 _rbs00_svr2 -b -21712
                  -S svr2 -s svr2 -d -1 -v 21689
starbj81 21716     1  0 11:46:32 pts/ts    0:00 _rbs00_svr2 -b -21715
                 -S svr2 -s svr2 -d -1 -v 21689
```

   - When one svr2 server process is running

```
$ps –ef | grep svr2
starbj81 21607     1  0 11:43:46 pts/ts    0:00 svr2 -s svr2 -g 2
```

If there are two or more server processes with the same name, the new server processes will be started after shutting down the current server processes one by one.

For instance, if the rbs is executed for svr1 whose MIN is set to 2, it will execute in the following order.

1. Terminate the first svr1.
2. Copy the temporary file (_rbs00_svr1) to the bk_appbin directory.
3. Start _rbs00_svr1.
4. Terminate the second svr1.
5. Copy the temporary file (_rbs00_svr1) to the bk_appbin directory.
6. Start _rbs00_svr1.

Version concurrency control when replacing a server

Version concurrency control is preventing processes of different versions from concurrently executing services. Executing rbs allows for concurrency control and dynamic server process update for services that have not been updated.

- Usage

```
$$1 tmax1 (tmadm): rbs [-S | -s svc_name,...] newfile svr_name [svg_name]
```

Item	Description
[-S]	Suspends all services of the server.

Item	Description
[-s *svc_name,...*]	Suspends only one service of the server.
newfile	New file name.
svr_name	Name of the server to suspend.
[*svg_name*]	Server group that the server to be suspended belongs to.

- Example
  - When updating a specific service of the server (-s)

    In the following example, the TOUPPER service of svr2 is temporarily suspended while other services continue to run.

    ```
    $$1 tmax1 (tmadm): rbs -s TOUPPER svr2_new svr2
    ```

    In the following example, TOLOWER and TOUPPER services of svr2 are temporarily suspended while other services continue to run. To specify multiple service names, use a comma(,) without blank spaces as a delimiter.

    ```
    $$1 tmax1 (tmadm): rbs -s TOUPPER,TOLOWER svr2_new svr2
    ```

  - When updating all services of the server (-S)

    In the following example, all services of svr2 are suspended temporarily.

    ```
    $$1 tmax1 (tmadm): rbs -S svr2_new svr2
    ```

    In the following example, all services of svr2 of the svg1 server group are suspended temporarily.

    ```
    $$1 tmax1 (tmadm): rbs -S svr2_new svr2 svg1
    ```

1. If the option(-s or -S option) for rbs is not specified, version concurrency is not guaranteed.
2. If the COUSIN group is set for only one Node, the rbs command cannot be executed for a svgname.

Version concurrency control in a multi-node environment

Controlling version concurrency in a multi node environment means preventing processes of different versions from concurrently executing services in a multi-node environment. Executing rbs in

a multi node environment where load balancing option is enabled allows for concurrency control and dynamic server process update for unmodified services.

- Usage

```
$$1 tmax1 (tmadm): mrbs [-S | -s svc_name,...] newfile svr_name
```

Item	Description
[-S]	Suspends all services of the server.
[-s *svc_name*]	Suspends only one service of the server.
newfile	New file name.
svr_name	Name of the server to suspend.

- Example
  - When updating a specific service of the server (-s)

    In the following example, the TOUPPER service of svr2 is temporarily suspended while other services continue to run.

    ```
    $$1 tmax1 (tmadm): mrbs -s TOUPPER svr2_new svr2
    ```

    In the following example, TOLOWER and TOUPPER services of svr2 are temporarily suspended while other services continue to run. To specify multiple service names, use a comma(,) without blank spaces as a delimiter.

    ```
    $$1 tmax1 (tmadm): mrbs -s TOUPPER,TOLOWER svr2_new svr2
    ```

  - When updating all services of the server (-S)

    In the following example, all services of svr2 are suspended temporarily.

    ```
    $$1 tmax1 (tmadm): mrbs -S svr2_new svr2
    ```

Executing rbs for a server that belongs to the COUSIN group of a node

If two or more server groups are set as COUSIN in a node, rbs (server exchange) can be executed for all servers that belong to the COUSIN server groups.

5.5.6. cfgadd (ca)

Dynamically adds a service to a specific server program while Tmax is running by only shutting down

the related server. A server, server group, or node can also be added dynamically during operation. In Tmax 5 and later, a binary configuration file cannot be added dynamically without using the -a option of CFL. Therefore, a configuration file must be compiled with the -a option.

For restrictions on dynamically adding a service by using cfgadd(ca), refer to Restrictions on dynamic addition using cfgadd(ca).

- Usage

```
$$1 tmax1 (tmadm) :  cfgadd (ca)  - i  cfgfile
```

Item	Description
- i *cfgfile*	Configuration file name of the service to add.

Dynamically adding a service

The following is the process of executing cfgadd(ca) to dynamically add a service.

1. Modify the configuration file.

```
$vi modify.m
```

2. Execute cfl with the -a option to compile the modified file and the -o option to create a binary configuration file with a different name.

```
$cfl -a modify.m -o tmchg
```

3. Execute gst to create a service table.

```
$gst -f tmchg
```

4. Execute "tmadmin –m" and use cfgadd(ca) to add the service.

```
$cfgadd -i tmchg
```

5. Compile the server program.

```
$compile c svr1
```

6. Load the server process.

```
$tmboot –S svr1
```

Dynamically adding a server

The following is the process of executing cfgadd(ca) to dynamically add a server.

1. Add a new server to a configuration file.

   <modify.m>

```
*SERVER
Existing items
........
hello           SVGNAME = svg1
*SERVICE
Existing items
........
HELLO           SVRNAME = hello
```

2. Compile the modified configuration file (modify.m).

```
cfl -a modify.m -o tmchg
```

3. Create a service table.

```
gst -f tmchg
```

4. Use cfgadd –i to add the configuration file.

```
tmaxi1@dhjang ./config > tmadmin -m
$$3 tmaxi1 (tmadm): si
------------------------------------------------------------
clh  svrname (svri) status   count   qcount   qpcount  emcount
------------------------------------------------------------
0   tmaxgw  (  0)    RDY      0        0        0        0
0   toupper ( 18)    RDY      0        0        0        0

$$4 tmaxi1 (tmadm): cfgadd -i tmchg
config is successfully added
$$5 tmaxi1 (tmadm): si
------------------------------------------------------------
clh  svrname (svri) status   count   qcount   qpcount  emcount
------------------------------------------------------------
0   tmaxgw  (  0)    RDY      0        0        0        0
0   toupper ( 18)    RDY      0        0        0        0
0   hello   ( 19)   NRDY      0        0        0       0
```

The hello server is in the NRDY state.

5. Enter the following command: tmboot -S hello -f tmchg.

```
tmaxi1@dhjang ./config > tmboot -S hello -f tmchg
TMBOOT for node(tmaxi1) is starting:
Welcome to Tmax demo system: it will expire 2002/8/31
Today: 2002/8/19
        TMBOOT: SVR(hello) is starting: Mon Aug 19 15:37:44 2002
```

6. Execute the si command to check if the server hello is RDY.

```
$$6 tmaxi1 (tmadm): si
------------------------------------------------------------
clh  svrname (svri) status   count   qcount   qpcount  emcount
------------------------------------------------------------
0   tmaxgw  (  0)    RDY      0        0        0        0
0   toupper ( 18)    RDY      0        0        0        0
0   hello   ( 19)    RDY      0        0        0       0
```

Dynamically adding a server group

The process for dynamically creating a new server group is essentially the same as that for dynamically adding a server. The only difference is the settings in the new configuration file. The configuration file must include settings for the server group being created, as well as the servers and services that will belong to the server group.

<tmchg.m>

```
*SVRGROUP
svg2        NODENAME = "aix5l"

*SERVER
svr3        SVGNAME = svg2, MIN = 1, MAX = 10

*SERVICE
FDLTOUPPER  SVRNAME = svr3
FDLTOLOWER  SVRNAME = svr3
```

Dynamically adding a node

The following is the process of executing cfgadd(ca) to dynamically add a node. It can also be used to add the node that a COUSIN server group belongs to.

The following is the process of executing cfgadd(ca) to dynamically add a node.

1. Modify the configuration file.

   <tmconfig_add.m>

```
*DOMAIN
tmax1           SHMKEY = @SHMEMKY@, MINCLH = 1, MAXCLH = 3,
                TPORTNO = @TPORTNO@, BLOCKTIME = 300, MAXCPC = 100,
                RACPORT = @TRACPORT@
*NODE
@HOSTNAME@      TMAXDIR = "@TMAXDIR@",
                APPDIR  = "@TMAXDIR@/appbin",
@RMTNAME@       TMAXDIR = "@RMTDIR@",
                APPDIR  = "@RMTDIR@/appbin",

*SVRGROUP
#svg1            NODENAME = "@HOSTNAME@",COUSIN = "svg2",LOAD = 2 <- Existing item
svg1            NODENAME = "@HOSTNAME@",COUSIN = "svg2,svg3",LOAD = 2 # <- Modified item
svg2            NODENAME = "@RMTNAME@",LOAD = 1

*SERVER
svr2            SVGNAME = svg1

*SERVICE
TOUPPER         SVRNAME = svr2
TOLOWER         SVRNAME = svr2

#Added items

*NODE
@RMTNAME2@      TMAXDIR = "@RMTDIR2@",
                APPDIR  = "@RMTDIR2@/appbin",

*SVRGROUP
svg3            NODENAME = "@RMTNAME2@",LOAD = 1
```

2. Execute racd on the newly added node.

```
Node3>$ racd -k
```

3. Compile the configuration file. Compile tmconfig_add.m, the configuration file to which the new node has been added. Execute cfl with the –o option to create a binary configuration file with a different name.

```
node1>$cfl -a tmconfig_add.m –o tmchg
CFL is done successfully for node(node1)

CFL: rcfl start for rnode (node2)
CFL is done successfully for node(node2)

CFL: rcfl start for rnode (node3)
CFL is done successfully for node(node3)
```

4. Compile the server. Compile the server of the node to be added.

```
node3>$ gst -f tmchg
```

```
node3>$ compile c svr2
```

5. Add the node. Dynamically add the node with the cfgadd command of tmadmin. Note that the command must be executed on each node as 'tmadmin –l'.

   The following is an example of adding node3 when node1 and node2 already exist.

```
# node1
$ node1>tmadmin –l -m
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$2 node1 (tmadm): cfgadd -i tmchg
(I) TMM0211 General Infomation : CFGADD started [TMM0902]
(I) TMM0211 General Infomation : CFGADD completed [TMM0907]
config is successfully added

# node2
$ node2>tmadmin –l -m
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$2 node2 (tmadm): cfgadd -i tmchg
(I) TMM0211 General Infomation : CFGADD started [TMM0902]
(I) TMM0211 General Infomation : CFGADD completed [TMM0907]
config is successfully added
```

6. Load the newly added node (node3).

```
Node3>tmboot -n tmaxh4 -f tmchg
TMBOOT for node(tmaxh4) is starting:
Welcome to Tmax demo system: it will expire 2008/11/23
Today: 2008/9/24
        TMBOOT: TMM is starting: Wed Sep 24 11:11:59 2008
        TMBOOT: CLL is starting: Wed Sep 24 11:11:59 2008
(I) TMM0211 General Infomation : node register (nodeno = 0(0)) success [TMM0404]
(I) TMM0211 General Infomation : node register (nodeno = 1(1)) success [TMM0404]
        TMBOOT: CLH is starting: Wed Sep 24 11:11:59 2008
(I) CLH9991 Current Tmax Configuration: Number of client handler(MINCLH) = 1
                Supported maximum user per node = 7966
                Supported maximum user per handler = 7966 [CLH0125]
        TMBOOT: TLM(tlm) is starting: Wed Sep 24 11:11:59 2008
        TMBOOT: SVR(svr2) is starting: Wed Sep 24 11:11:59 2008
```

7. Verify that the node has been added successfully.

```
node1>tmadmin
TMADMIN for rnode (node2): starting to connect to RAC
TMADMIN for rnode (node3): starting to connect to RAC
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 node1 (tmadm): ti

Tmax System Info: DEMO version 4.0 SP #3 Fix #8:
```

```
         expiration date = 2008/11/22
         maxuser = UNLIMITED,
         domaincount = 1,
         nodecount = 3,
         svgrpcount = 3,
         svrcount = 9, svccount = 6
         rout_groupcount = 0, rout_elemcount = 0
         cousin_groupcount = 1, cousin_elemcount = 3
         backup_groupcount = 0, backup_elemcount = 0

Tmax All Node Info: nodecount = 3:
-----------------------------------------------------------------
  no   name     portno  racport  shmkey  shmsize  minclh  maxclh
-----------------------------------------------------------------
   0   node1     8350    3155     88350   225760       1      3
   1   node2     8350    3155     88350   225760       1      3
   2   node3     8350    3155     88350   225760       1      3

$$2  (tmadm): st -s

CLH 0:
   ---------------------------------------------------------------
   svc_name    svr_name   count cq_cnt  aq_cnt  q_avg   avg   status
   ---------------------------------------------------------------
   TOLOWER       svr2         0       0       0     0.000   0.000  RDY
   TOUPPER       svr2         0       0       0     0.000   0.000  RDY

Msg from rnode(node2):

CLH 0:
   ---------------------------------------------------------------
   svc_name    svr_name   count cq_cnt  aq_cnt  q_avg   avg   status
   ---------------------------------------------------------------
   TOLOWER       svr2         0       0       0     0.000   0.000  RDY
   TOUPPER       svr2         0       0       0     0.000   0.000  RDY

Msg from rnode(node3):

CLH 0:
   ---------------------------------------------------------------
   svc_name    svr_name   count cq_cnt  aq_cnt  q_avg   avg   status
   ---------------------------------------------------------------
   TOLOWER       svr2         0       0       0     0.000   0.000  RDY
   TOUPPER       svr2         0       0       0     0.000   0.000  RDY
```

Restrictions on dynamic addition using cfgadd(ca)

Dynamically adding a service, server, server group or node by using cfgadd(ca) has the following restrictions:

- An item can only be added, not removed. To remove an item, remove it from the existing environment file and execute cfl before restarting Tmax.
- An item name that exists in the configuration file cannot be duplicated in the newly added configuration file.

- When the contents of a newly added configuration file are applied to the existing configuration file, they must be added to the end of each section. If items are inserted in the middle of the existing configuration or the order is changed, the operating environment information can be modified incorrectly. Attention is required.
- Items can be added to the MAXNODE, MAXSVG, MAXSVR, MAXSPR, MAXSVC, MAXTMS, and MAXCPC settings. Do not modify the items' values.
- When adding an XA server group, specify TMS and check that the program has been created.
- RQ cannot be added dynamically.
- A server group that has a COUSIN/BACKUP cannot be added or removed.
- A gateway that has a COUSIN/BACKUP cannot be added or removed.
- A service that is dynamically added to the server group, which has a COUSIN/BACKUP, cannot be removed.
- When adding the service, server, and server group, cfl must be used to create a binary configuration file for the dynamic addition.

  If a binary configuration file, which is created using cfl without the -a option, is added dynamically, the following error message will be displayed.

```
(E) ADM2048 Engine type mismatch (0): 'a' option must be used [ADM0417]
```

5.5.7. set

Dynamically modifies the settings in the configuration file.

Execute the cfg command to check which items in the configuration file can be modified. The items with an abbreviation inside brackets can be modified by using the set command.

- Usage

```
$$1 tmax1(tmadm): set [-d [domain]| -g [server_group]| -v [service]|
                       -s [server]] Item Value
```

Item	Description
[-d [*domain*]]	Modify DOMAIN section. To modify a specific domain, specify the domain name.
[-g [*server_group*]]	Modify GATEWAY section. To modify a specific server group, specify the server group name.
[-v [*service*]]	Modify SERVICE section. To modify a specific service, specify the service name.
[-s [*server*]]	Modify SERVER section. To modify a specific server, specify the server name.

Item	Description
Item	Item to modify.
Value	New value.

- Example

  The following is an example of using each command.

```
$$1 tmax1 (tmadm): set -d res1 bt 100
$$1 tmax1 (tmadm): set -g svg1 ld 5
$$1 tmax1 (tmadm): set -v kfdl1 mq 1000
$$1 tmax1 (tmadm): set -s SYNC pr 100
$$1 tmax1 (tmadm): set -n tmaxh4 cb n
$$1 tmax1 (tmadm): set -s svc_name pr 99
$$1 tmax1 (tmadm): set -v svr2 maxsvr 10 (If changing, max(svr2) must be smaller than the
previous value)
$$1 tmax1 (tmadm): set -d dom1 portno 9999
$$1 tmax1 (tmadm): set -g svg2 maxrstart 10
$$1 tmax1 (tmadm): set -v svr140_A restart N
$$1 tmax1 (tmadm): set -v gw1 schedule FA
$$1 tmax1 (tmadm):  set -d dom nclhchktime 10
```

5.5.8. setopt

Dynamically changes an environment configuration value among options configured for TMMOPT. Configurable items can be checked using the cfgopt command in tmadmin. The items displayed by executing cfgopt can be modified dynamically.

- Usage

```
$$1 tmax1 (tmadm): setopt [-tmm][-gw] Item Value
```

Item	Description
[-tmm]	Used to change the option value defined in TMMOPT.
[-gw]	Used to change the option value defined in CLOPT under the GATEWAY section.
Item	Item to change. An item displayed by executing cfgopt. (example, -F)
Value	New value.

- Example

  The following is an example of executing the command.

```
$$1 tmax1 (tmadm): setopt -tmm -F 30
new value (30) is set for section = -tmm, name = N/A, fld = -F
$$1 tmax1 (tmadm): setopt -tmm -t 15
new value (15) is set for section = -tmm, name = N/A, fld = -t
```

5.5.9. qpurge (qp)

Deletes the service requests that are waiting in the queue. This is usually used to remove abnormal number of accumulated requests. This command is most useful for banks, government offices and other such institutions that handle hundreds of thousands of tasks per day.

Deleted requests can be processed later when clients reissue the request. Since Tmax manages a queue for each server process, the administrator can empty a queue for a specific server process to enhance the system efficiency.

The queue purge command is abbreviated as **qp**. When a service request is deleted from the queue, a TPEQPURGE message (tperrno = 27) will be sent to the client that issued the request. Thus, the client will be able to reissue the request at a later time. When this command is used, information about the deleted requests and client IDs will be recorded in the slog.

- Usage

```
$$1 tmax1 (tmadm): qpurge {-v svr_name [ -k number ] | -s svc_name}
```

Item	Description
-v *svr_name*	Delete requests accumulated in the queue of the specified server.
[-k *number*]	Can purge some of the requests accumulated in the server Queue. Must be used with the –v option. If N is a number greater than 0, keep the first N number of service requests and purge any subsequent incoming requests. • -1: Keep as many as MAXQCOUNT requests in the queue, and purge any subsequent incoming requests. • 0: Purge all service requests currently in the queue.
-s *svc_name*	Delete requests accumulated in the queue for the specified service.

5.5.10. discon (ds)

Forcibly disconnects inactive clients that are connected but are idle. Before the administrator issues this command, it is recommended to use the **ci** command to check the client information. The discon command is abbreviated as **ds** with the following options.

- Usage

```
$$1 tmax1 (tmadm): ds [-h clhno] [-f] {-a | -n | -i idle_time | -c clid | -A}
```

Option	Description
[-h clhno]	Disconnect clients connected to the CLH.
[-f]	Immediately disconnect clients.
{-a}	Disconnect all clients connected to the CLH. If the [-h] option is not used, the client connected to CLH 0 is disconnected by default.
{-n}	Disconnect clients with inaccurate information. Buffer allocated by tpalloc() is not used.
{-i idle time}	Disconnect clients from sessions that exceed the set amount of time (second).
{-c cliid}	Disconnect the client with the specified ID number. ID number must be specified.
{-A}	Disconnect clients connected to all CLHs.

5.5.11. logstart/logend

The administrator can use these two commands to create a log which will allow them to access various information in real time to take immediate and appropriate actions for each situation. The tmadmin tool logs administrative data for statistics analysis. The log file will be created in the current directory. By viewing the log data, the administrator will be able to analyze the system performance to determine the peak times, unnecessary server processes, and queue status.

Logging will begin when the logstart command is executed, and terminate when the logend command is executed.

- Usage
  - Start logging (logstart)

    ```
    $$1 tmax1 (tmadm): logstart filename
    ```

Item	Description
filename	Log file name.

  - End logging (logend)

    ```
    $$1 tmax1 (tmadm): logend
    ```

5.5.12. chtrc

TMAX_TRACE enables runtime tracing in a Tmax system. To use this function, TMAX_TRACE must be defined in the Tmax configuration file, and the **chtrc** command can be used in tmadmin to dynamically modify the configurations.

- Usage

```
chtrc [-g svgname | -v svrname | -g svgname -v svrname |
       -i spri | -g svgname -i spri | -e gqs] -s spec
```

Option	Description
[-g *svgname*]	Modifies specifications of the server group.
[-v *server*]	Modifies specifications of the server.
[-g *svgname* -v *server*]	Modifies specifications of the server in the server group.
[-i *spri*]	Modifies specifications of the server process.
[-g *svgname* -i *spri*]	Modifies specifications of the server process in the server group.
[-e *gqs*]	Modifies specifications of gqs.
-s *newspec*	New specifications.

- Example
  - Modifying the spec of all servers on a node (-s)

    The following is an example of modifying the spec of all servers on a specific node in a single node environment.

```
$$1 tmaxs1 (tmadm): chtrc -s newspec
```

    The following is an example of applying a new TRACE spec to a specific node in a multi node environment.

```
$$1 tmaxh3 (tmadm): nodeset $HOSTNAME
node is set to $HOSTNAME
$$2 tmaxh3 (tmadm): chtrc –s newspec
```

  - Modifying server group spec (-g)

    The following is an example of modifying the spec of all servers of a specific server group.

```
$$1 tmaxs1 (tmadm): chtrc -g svgname -s newspec
```

◦ Modifying server spec (-v)

The following is an example of modifying the spec of all servers of a specific server group.

```
$$3 tmaxs1 (tmadm): chtrc -v server -s newspec
```

The following is an example of modifying the spec of a specific server of a server group.

```
$$3 tmaxs1 (tmadm): chtrc -g svgname -v server -s newspec
```

◦ Modifying the spec of a specific server process (-i)

The following is an example of modifying the spec of a particular server process.

```
$$4 tmaxs1 (tmadm): chtrc -g svgname -i sprno -s newspec
```

5.5.13. chlog

Dynamically changes the log level in order to rapidly handle specific error conditions. The module must be in debug mode to check the log using chlog. The cfg command can be used to confirm that the change to the log level has been applied successfully.

- Usage

```
$$1 tmaxs1 (tmadm) : chlog [-t | -c | -v [server_name] |
            -g [server_group_name]| -m] -l [loglvl]
```

Option	Description
[-t]	TMM log level.
[-c]	CLH log level.
[-v [*server_name*]]	Log level of the specified server.
[-g [*server_group_name*]]	Log level of the specified server group.
[-m]	TMS log level.
-l [*loglvl*]	Actual log level. Select one of the following options. • COMPACT, BASIC, DETAIL, DEBUG1, DEBUG2, DEBUG3, DEBUG4 COMPACT logs least details, and DEBUG4 logs most details.

- Example
  - Dynamically modifying the TMM log level (-t)

    The following is an example of dynamically changing the log level of TMM.

```
tmaxh2:/data1/starbj81/tmax/config> tmadmin -l -m
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh2 (tmadm): chlog -t -l DEBUG4
log level is updated
$3 tmaxh2 (tmadm): cfg -n
    node_name = tmaxh2, hostname = tmaxh2, node_no = 1
...
    tmmloglvl = DEBUG4,
...
```

  - Dynamically modifying the CLH log level (-c)

    The following is an example of dynamically changing the log level of CLH.

```
tmaxh2:/data1/starbj81/tmax/config> tmadmin -m -l
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh2 (tmadm): chlog -c -l COMPACT
log level is updated
$$2 tmaxh2 (tmadm): cfg -n
    node_name = tmaxh2, hostname = tmaxh2, node_no = 1
    tmmloglvl = DEBUG4,
    clhloglvl = COMPACT,
...
```

  - Dynamically modifying the TMS log level (-m, -g)

    The following is an example of dynamically changing the log level of TMS.

```
$$3 tmaxh4 (tmadm): chlog -m -g svg32306X -l debug3
log level is updated
$$4 tmaxh4 (tmadm): cfg –g
    svg_name = svg32306X, svg_no = d
    tmsloglvl = DEBUG3,
...
```

  - Dynamically modifying the log level of a server group (-g)

    The following is an example of dynamically changing the log level of a specific server group.

```
$$4 tmaxh2 (tmadm): chlog -g svg3 -l DETAIL
log level is updated
$$5 tmaxh2 (tmadm): cfg -g
    svg_name = svg3, svg_no = 10002
```

```
    loglvl = DETAIL
...
```

5.5.14. chlog2

Dynamically changes the log level in order to rapidly handle specific error conditions. This command includes all functions of chlog and can also change log levels for TLM, CAS, CLL, RS, SQ, HMS, RQS, and gateways. The cfg command can be used to confirm that the change to the log level has been applied successfully.

- Usage

```
$$1 tmaxs1 (tmadm) : chlog2 -t [type_name] [-n name] -l [loglvl]
```

Option	Description
-t [*type_name*]	Module whose log level is changed. • TMM: TMM • CLH: CLH • TLM: TLM • CAS: CAS • RS: Real server • SQ: Session queue server • TG: TmaxGrid server • TMS: TMS • HMS: HMS • RQ: Reliable queue server • GW: Gateway • SVG: Server group • SVR: Server • CLL: CLL

Option	Description
-n [*name*]	Configured server group name for TMS, HMS, and RQ, gateway name for GW, and server group name for SVG. No name is necessary for other modules. • TMM: TMM • CLH: CLH • TLM: TLM • CAS: CAS • RS: Real server • SQ: Session queue server • TG: TmaxGrid server • TMS: TMS • HMS: HMS • RQ: Reliable queue server • GW: Gateway • SVG: Server group • SVR: Server • CLL: CLL
-l [*loglvl*]	Actual log level. Select one of the following options. • COMPACT, BASIC, DETAIL, DEBUG1, DEBUG2, DEBUG3, DEBUG4 COMPACT logs least details, and DEBUG4 logs most details.

- Example
  - Dynamically modifying the TMM log level

    The following is an example of dynamically changing the log level of TMM.

```
tmaxh2:/data1/starbj81/tmax/config> tmadmin -l -m
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh2 (tmadm): chlog2 -t TMM -l DEBUG4
log level is updated
$3 tmaxh2 (tmadm): cfg -n
    node_name = tmaxh2, hostname = tmaxh2, node_no = 1
...
    tmmloglvl = DEBUG4,
...
```

  - Dynamically modifying the CLH log level

The following is an example of dynamically changing the log level of CLH.

```
tmaxh2:/data1/starbj81/tmax/config> tmadmin -m -l
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh2 (tmadm): chlog2 -t CLH -l COMPACT
log level is updated
$$2 tmaxh2 (tmadm): cfg -n
    node_name = tmaxh2, hostname = tmaxh2, node_no = 1
    tmmloglvl = DEBUG4,
    clhloglvl = COMPACT,
...
```

5.5.15. txcommit/txrollback

If a transaction processing does not complete for a certain amount of time due to a network or TMS failure, the administrator can reissue the commit / rollback and complete the transaction. Txcommit can only be used when the Decision is COMMIT in the PHASE2 stage. Txrollback can only be used when the Decision is ROLLBACK in the PHASE1 or PHASE stage.

- Usage

```
$$1 tmax1 (tmadm): txrollback(txr) | txcommit(txc) [-w] [-y]
                   <upper-global-xid><lower-global-xid>
```

Item	Description
[-w]	Used for a domain sub-transaction.
[-y]	Does not go through the checking process for txcommit / txrollback.
upper-global-xid	First 4 bytes of gtid of xid to process.
lower-global-xid	Last 4 bytes of gtid of xid to process.

txcommit and txrollback are used to reprocess a transaction that is queried via txquery. After reprocessing, it is important to check the result through txquery.

5.5.16. wsgwreload

Modifies the configuration of service or web service gateway at runtime. Once the command is executed, the web service gateway does not respond to new service requests. It only completes processing services that are in progress, and then applies the changed settings and resumes service processing.

- Usage

```
$$1 tmax1 (tmadm): wsgwreload [-i svrid[svrid]]
```

Option	Description
[-i *svrid* [*svrid*]]	List of svrid's to apply the changes.

5.5.17. restart

Terminates the currently running server processes and restarts as many as MIN number of server processes.

- Usage

```
$$1 tmax1 (tmadm): restart [-v[server_name]] [-g[server_group_name]]
```

Option	Description
[-v[*server_name*]]	Restarts the specified server.
[-g[*server_group_name*]]	Restarts the servers of the specified server group.

- Example
  - The following is an example of using the restart command.

```
$$2 tmaxc1 (tmadm): restart -v svr01021
>> suspend ok
>> down ok
>> boot ok
>> resume ok
>> reboot svr svr01021 finished


$$2 tmaxc1 (tmadm): restart -g svg1
>> suspend ok
>> down ok
>> boot ok
>> resume ok
>> reboot svr svr01021 finished
>> suspend ok
>> down ok
>> boot ok
>> resume ok
>> reboot svr svr01022 finished
```

5.5.18. notify_reconnect_clh (nrc)

Manually establishes a connection to CLH for a server process that cannot automatically make a

connection to the CLH during Tmax operation.

- Usage

```
$$1 tmax1 (tmadm): notify_reconnect_clh clh_index spr_index
```

Option	Description
clh_index	CLH number to connect to.
spr_index	spr number to establish a connection. spr number (spr_no) can be checked by using st -p in tmadim.

5.5.19. admnoti (an)

Sends events to TCS, UCS, and RDP servers. A server that receives an event calls a user-specified callback function to handle the event. If a callback function is not specified, the event is ignored. This command can be used only in master mode (tmadmin -m).

The tpregancb API can be used in TCS, UCS, and RDP to write event handling code.

- Usage

```
$$1 tmax1 (tmadm): admnoti [-a | -g svgname | -p spri| -P pid] [notification message]
```

Option	Description
-a	Sends an event to all server processes.
-g *svgname*	Sends an event to all server processes included in svgname.
-v *svrname*	Sends an event to all server processes with the svrname.
-p *spri*	Sends an event to server process with the spri.
-P *pid*	Sends an event to server process with the pid.
notification message	Message to send to the callback function.

6. IPv6 Configuration

This chapter describes IPv6 provided by Tmax.

6.1. Overview

This section describes the basic concept and features of IPv6 and dual stack, which is an IP conversion technology.

6.1.1. IPv6

IPv6, a network-level protocol, is a next generation Internet protocol established as the Internet Protocol version 6. Due to the limitation of using IPv4 protocol for the Internet, which was used in the past, IETF established IPv6 protocol as the new protocol.

IPv4, expressed in 32 bits, can create 4,294,967,296 addresses. As the number of computers connected to the Internet has grown dramatically, IPv4 addresses have almost been exhausted. IPv6 has been suggested to resolve this issue.

IPv6, expressed in 128 bits, can create 3.4x10^38 addresses (2^128), which is close to infinity. With this many addresses, one IPv6/48 network can be provided per 10㎡ of surface area on the earth.

A big difference between IPv4 and IPv6 is how an address is expressed. An IPv6 address can be represented with a hexadecimal number that includes a colon(:) between every 16 bits (four digits).

```
2001:0db8:85a3:08d3:1319:8a2e:0370:7334
```

6.1.2. IPv6 Conversion Technologies

Until the conversion to IPv6 is complete, IPv4 and IPv6 networks will be used concurrently. To support both IPv4 and IPv6, the following technologies are provided by the basic platform (such as OS, router, and switch) vendors.

- Dual-stack

  IPv4/IPv6 dual stack is the easiest way to maintain compatibility between an IPv6 node and a IPv4 dedicated node. Since an IPv6/IPv4 dual stack node can send or receive both IPv4 and IPv6 packets, it is compatible with both IPv4 node and IPv6 node.

- Tunneling

  Tunneling is used for IPv6/IPv4 host and router to encapsulate IPv6 datagrams in an IPv4 packet, and transmit it through the IPv4 routing topology. Through tunneling, IPv6 packets can be transmitted through the existing IPv4 routing infrastructure.

6.2. Tmax IPv6

IPv6 can be used in Tmax by configuring the necessary settings. IPv4 is the default protocol.

The settings can be applied to the environment variables and Tmax configuration files as needed. (Since configuration files cannot not be used for a client, environment variables must be configured for a client.)

6.2.1. IPv6 Supported Features

In Tmax, client, multi-node, multi-domain, and tcp gateways are implemented using sockets. Each module can choose to use IPv4 or IPv6.

Each module can select to use IPv6 based on the following situations.

- Accepting a connection

  The following describes Tmax components and configuration depending on how a connection is accepted.

Components	Description
cll	Receives a connection from a client. • Configuration file: domain and node sections • Environment variable: CLIENT_IPV6
tmm, clh	Receives an inter-node connection in a multi-node environment. • Configuration file: domain and node sections • Environment variable: SYSTEM_IPV6 Receives a connection from an extern server. • Configuration file: domain and node sections • Environment variable: EXTSVR_IPV6
racd	Receives a utility connection in a multi-node environment. • Configuration file: domain and node sections • Environment variable: TMAX_RAC_IPV6, SYSTEM_IPV6
webagent(twagent)	Receives a WebAdmin connection. • Environment variable: TMAX_WEBADM_ IPV6
gateway([non]tx, [async] tuxedo, [async] java, wsgw, xagw)	Receives a connection. • Configuration file: gateway section • Environment variable: LOCAL_IPV6

Components	Description
hms	Receives a connection from hms cluster. • Configuration file: domain and node sections • Environment variable: SYSTEM_IPV6
tcp g/w	Receives a connection. • Configuration file: CLOPT of SERVER section, or a separate configuration file *Tmax Gateway Guide (TCP/IP)*.) • Environment variable: -X SERVER_ IPV6
tmsnmpd	Receives an SNMP-requested connection. • Arguments for execution: upd6:port or tcp6:port
Java components	Receives a connection from jtmax or webtasync. The java components must be set through environment variables when executing java. (To run them on WAS, they must be set to environment variables at startup.) • java.net.preferIPv4Stack • java.net.preferIPv6Addresses

- Requesting a connection

  The following describes Tmax components and configuration according to how the connection is requested.

Components	Description
c client	Requests a connection to Tmax. • Environment variable: TMAX_HOST_IPV6, TMAX_BACKUP_IPV6
tmm	Connects to another node(tmm) in a multi-node environment. • Configuration file: domain and node sections • Environment variable: SYSTEM_IPV6
clh	Connects to another node(clh) in a multi-node environment. • Configuration file: domain and node sections • Environment variable: SYSTEM_IPV6
tdl	Connects to another node in a multi-node environment when executing the tdl utility. • Configuration file: domain and node sections • Environment variable: SYSTEM_IPV6

Components	Description
hms	Connects to HMS of another node in a clustered HMS environment. • Configuration file: domain and node sections • Environment variable: SYSTEM_IPV6
xa client	Connects to the xa gateway. • Function's argument: tmax_xa_open • Environment variable: ipv6
tcp g/w	Connects to an external server. • Configuration file: CLOPT of SERVER section in configuration file, or a separate configuration file (Refer to *Tmax Gateway Guide (TCP/IP)*.) • Environment variable: -X CLIENT_ IPV6
gateway([non]tx, [async] tuxedo, [async] java)	Connects to another gateway. • Configuration file: gateway section • Environment variable: RGW_ IPV6, RGW_B1_ IPV6, RGW_B2_ IPV6, RGW_B3_ IPV6
gateway(wsgw)	Connects to the Web Service server. • Separate configuration file (xml) • Distinguish ipv6 through uri.
Java components	Requests a connection to jtmax, webtasync, or twadmin. The java components must be set through environment variables when executing java. (To run them on WAS, they must be set to environment variables at startup.) • java.net.preferIPv4Stack • java.net.preferIPv6Addresses

If IPv6 support is configured, Tmax creates a socket for IPv6. If a platform where Tmax is installed provides a conversion technology like dual stack, Tmax can receive a connection request through both IPv4 and IPv6. If IPv6 is not configured for Tmax even though the platform supports both IPv6 and IPv4, Tmax can receive only IPv4 connections.

6.2.2. Additional Features

Tmax supports other IPv6 related features, in addition to the aforementioned connection and transmission related features, for inputting and searching for information, such as ci information, in ACL or tmadmin.

The following are additional features for IPv6.

- ACL

  Limits client connection IP.

  - Supports IPv6 addresses in a separate configuration file.
  - Specified as [2011::xxx].

- tmadmin

Command	Description
ci	Outputs the IP of a connected client.
txgwi, ntxgwi, jgwi, ajgwi	Outputs the IP of a connected gateway.
sqi	Outputs the client IP of Session Q.

- client and server API

API	Description
tpgetsockname	Retrieves the socket address, which is used internally by Tmax, from the server or client. For more information, refer to "3.1.72. tpget sockname" in *Tmax Reference Guide*.
tpgetpeername	Retrieves the socket address of the peer to which a socket is connected from the server or client. Returns the peer socket address (node) after connecting to Tmax. For more information, refer to "3.1.68. tpgetpeername" in *Tmax Reference Guide*.
tpgetpeer_ipaddr	Retrieves the socket address of the peer node from the server. For more information, refer to "3.2.35. tpgetpeer_ipaddr" in *Tmax Reference Guide*.
tpgetcliaddr_ipv6	Retrieves the port number and IP of the client that corresponds to the clid among the clients connected to Tmax. For more information, refer to "3.1.64. tpgetcliaddr_ipv6" in *Tmax Reference Guide*.
tpbroadcast	Sends an unrequested message to all clients registered on Tmax. For more information, refer to "3.1.46. tpbroadcast" in *Tmax Reference Guide*.
tmadmin	Outputs statistical information that can be queried in tmadmin, a Tmax management tool. For more information, refer to "3.2.4. tmadmin" in *Tmax Reference Guide*.

Appendix A: CLOPT Options for Gateway

This appendix describes CLOPT options for Tmax domain gateway, Java gateway, and Tuxedo gateway.

A.1. Tmax

This section describes the CLOPT options for Tmax domain gateway.

A.1.1. Transaction Domain Gateway

CLOPT options for gateways whose GWTYPE is "TMAX".

Option	Description
[-r]	Same as the -r option for CLOPT in GATEWAY.
[-h]	Same as the -h option for CLOPT in GATEWAY.
[-i]	If not set, an attempt will be made to connect to a remote gateway for requests that are received while disconnected. If set, connecting to a remote gateway is attempted at the interval specified in NLIVEINQ. If not connected to a remote gateway, a TPNOREADY message is sent to the client.
[-R DECISION]	Used when the remote gateway version is Tmax 3.x. Gateway determines whether to roll back or commit a transaction that was started from a remote gateway but is pending at the local gateway. • RBK: Rollback. • COM: Commit. • IGN: Do nothing.
[-c TIME]	Attempts to reconnect to CLH at the interval specified in TIME.
[-l "ip-addr,ip-addr2..."]	L4 periodically sends the ping command to a gateway with a connection for alive check and forcibly ends the connection. The gateway records log about the disconnection. This causes an issue of a large volume of unnecessary data. If this option is used, the unnecessary log is not recorded.
[-m]	Terminates tmgw(nt) when TMM is terminated, like MAC = Y in the SERVER section.
[-k]	Same as the -k option for CLOPT in GATEWAY.

A.1.2. Non Transaction Domain Gateway

CLOPT options for gateway whose GWTYPE is "TMAXNONTX".

Option	Description
[-h]	Same as the -h option for CLOPT in GATEWAY.
[-i]	If not set, an attempt will be made to connect to a remote gateway for requests that are received while disconnected. If set, connecting to a remote gateway is attempted at the interval specified in NLIVEINQ. If not connected to a remote gateway, a TPNOREADY message is sent to the client.
[-c TIME]	Attempts to reconnect to CLH at the interval specified in TIME.
[-n]	The number of channels between Tmax gateways is fixed to 2. When message 1 is sent to channel 1, and then message 2 is sent to channel 2 through GW1, message 2 may arrive first and be processed before message 1 at GW2. To prevent this, set the number of channels to one by adding [-n] option to CLOPT of the GATEWAY section in the configuration file. [-n] does not have any arguments.
[-l "ip-addr,ip-addr2..."]	Same as the -l option for Tmax transaction domain gateway.
[-m]	Terminates tmgw(nt) when TMM is terminated, like MAC = Y in the SERVER section.
[-k]	Same as the -k option for CLOPT in GATEWAY.

A.2. Java

This section describes CLOPT options for Java gateway.

A.2.1. JEUS Gateway

CLOPT options for gateway whose GWTYPE is "JEUS".

Option	Description
[-D DEBUG_LEVEL]	Debugging level. • 1: Outputs logs related to request and response. • 2: Outputs XA related logs. • 4: Outputs message DUMP related logs. A pipe(|) operator can be used to determine the log to output.

Option	Description
[-e LOGFILE_PATH]	Log file path to store standard errors.
[-o LOGFILE_PATH]	Log file path to store standard outputs.
[-r]	Must be specified for versions later than WebT 3.14.
[-h VERSION]	• Set to 1 if the header is set as the default value in WebT 3.x and WebT 5.x. • Set to 4 if the header is set to extendedV4 in WebT 5.x.
[-t]	Must be specified to connect multiple domains to one JTmax. If multiple domains attempt to connect to one JTmax without setting this option, a recovery process will fail.

A.2.2. JEUS Async Gateway

CLOPT options for gateway whose GWTYPE is "JEUS_ASYNC".

Option	Description
[-D DEBUG_LEVEL]	Debugging level. • 1: Outputs logs related to request and response. • 2: Outputs XA related logs. • 4: Outputs message DUMP related logs. • 7: Outputs all the message logs that 1,2, and 4 output. A pipe(|) operator can be used to determine the log to output.
[-e LOGFILE_PATH]	Log file path to store standard errors.
[-o LOGFILE_PATH]	Log file path to store standard outputs.
[-r]	Must be specified.
[-h 4]	Must be set to 4.
[-A TIME]	Used to close a connection with an external gateway if no response is received within a specified amount of time or an alive check interval.
[-a FILE_PATH]	Makes a connection using RGWADDR, RGWPORTNO, and "IP:PORT" list from the specified file. Messages are transmitted through the same channel for a single transaction. By default, messages are transmitted through each channel in a round robin fashion. Create the file with one "ip:port" per line.
[-H]	If set, an alive check message is not written to the log file.
[-t]	Must be specified to connect multiple domains to one JTmax. If multiple domains attempt to connect to one JTmax without setting this option, a recovery process will fail.

Option	Description
[-m MAX_COUNT]	Number of requests that the gateway can issue to JTmax. (default value: 500)

A.3. Tuxedo

This section describes CLOPT options for Tuxedo gateway.

A.3.1. Tuxedo Gateway

CLOPT options for gateway whose GWTYPE is "TUXEDO".

Option	Description
[-a LOCAL_DOMAIN_NAME]	Domain name to be used to connect to Tuxedo domain gateway. If no name is specified, the following error message will be displayed. `3005 gateway name (-a domname) not defined`
[-e LOGFILE_PATH]	Log file path to store standard errors.
[-o LOGFILE_PATH]	Log file path to store standard outputs.
[-u UID]	UID is specified to prevent a call request, which a Tmax client has not issued, when using ACL.
[-F]	Used if the type of a message transmitted by Tuxedo is FML16. If no type is specified, this option will be set to FML32.
[-v]	Used if the type of a message transmitted by Tuxedo is VIEW16. If no type is specified, this option will be set to VIEW32.
[-r REMOTE_DOMAIN_NAME]	Used to authenticate Tuxedo when it attempts to connect to Tmax. Only Tuxedo domain gateway whose local domain name is set as REMOTE_DOMAIN_NAME will be allowed to access Tmax. If it fails to be authenticated, then the following error message will be displayed. `0046 incorrect local name(REMOTE_DOMAIN_NAME), remote domain name(local domain name of the opposite end)`

Option	Description
[-h]	If multiple local gateways are set, each gateway tries to connect to each remote gateway. In Tuxgw of Tmax, after a connection has been established, the following error occurs when the second connection is attempted. `(E) GATEWAY3010 connection error from remote gateway [TUXGW0002]` If not needed, use this option to prevent this message from being logged.
[-D]	Among messages that are sent and received through Tuxedo gateway, messages that have not been processed successfully are output as binary data with the following message format. `<time>:discarded [tmax | tuxedo] message(size:<size>)`
[-c time(sec)]	Executes tpcall() for a connected Tuxedo at the interval specified in time Calls the service specified with the -C option. If a service name is not specified, dus$%@tjq is used. If no response is received during the specified time * 2, the connection is ended.
[-C service_name]	Service name used along with the -c option.
[-i]	If Tuxedo gateway is set as COUSIN, normal scheduling is performed even if there is a channel issue. Unlike existing IRT, the first several calls may fail because the Tuxedo gateway is used for transactions. After the first several calls, normal scheduling is performed.

A.3.2. Tuxedo Async Gateway

Same as the CLOPT options for Tuxedo gateway.

Appendix B: Notes on Using Tmax

This appendix describes notes about using Tmax.

B.1. Using Multiple CLHs

This section describes how to do scheduling with multiple CLHs.

B.1.1. ASQCOUNT

ASQCOUNT is specified for each CLH, not for the entire engine. After checking the queue state of each CLH, apply ASQCOUNT to boot additional servers that are needed.

For instance, if ASQCOUNT is set to 4 and 5 requests are currently waiting on a node, then the current queue size is not 5 since each CLH has a server wait queue. There may be 2 waiting on CLH #1 and 3 waiting on CLH #2 which means neither CLH has exceeded its ASQCOUNT and additional server is not booted.

B.1.2. Concurrent Scheduling

In a multi CLH environment, all CLHs can do scheduling for the same spr simultaneously. In general, the states of other CLHs are checked during scheduling to avoid concurrent scheduling, but it may be unavoidable when there are too many concurrent requests. In such cases, requests wait for the socket of the spr, which was issued later, and they do not appear in the queue in tmadmin and are shown in the RUN state when st -p is executed on each CLH. Such requests are considered to be in the queue of CLH, and a TPEQPURGE message is sent if qtimeout has already elapsed before the actual service execution on the server.

To always execute this function, add the [-B] option to the CLOPT parameter of the SERVER section.

B.2. Domain Gateway COUSIN Setting

This section describes how to configure the availability of domain gateway to enable routing between domain gateways.

B.2.1. SVRGROUP

```
*SVRGROUP
ServerGroup Name  [COUSIN = cousin-svg-name,cousin-gw-name,]
```

```
                [LOAD = numeric]
```

- COUSIN = literal
  - Range: Up to 7999 characters
  - Specify a server group/gateway name to allow certain processes to be shared among server groups or gateways, or when routing is required between server groups or gateways.
- LOAD = numeric
  - Refer to "3.2.3. SVRGROUP" LOAD.

B.2.2. GATEWAY

```
*GATEWAY
Gateway Name    [LOAD = numeric]
```

- LOAD = numeric
  - Refer to "3.2.3. SVRGROUP" LOAD.

Example

By configuring the following settings, svg1(tmaxh4), svg2(tmaxh2), gw1(tmaxh4), and gw2(tmaxh2) are bound together in a COUSIN setting. Since LOAD value of each server group or gateway is set to 1, TOUPPER service requests are processed evenly by svg1, svg2, gw1, and gw2 (1:1:1:1).

<domain1.m>

```
*DOMAIN
tmax1       SHMKEY = 78350, MINCLH = 2, MAXCLH = 3,
            TPORTNO = 8350, BLOCKTIME = 10000,
            MAXCPC = 100, MAXGW = 10, RACPORT = 3355

*NODE
tmaxh4      TMAXDIR = "/data1/tmaxqam/tmax",

tmaxh2      TMAXDIR = "/data1/tmaxqam/tmax",

*SVRGROUP
svg1      NODENAME = tmaxh4,
         COUSIN = "svg2, gw1, gw2", LOAD = 1
svg2      NODENAME = tmaxh2, LOAD = 1

*SERVER
svr2      SVGNAME = svg1

*SERVICE
TOUPPER      SVRNAME = svr2

*GATEWAY
```

```
#Gateway for domain 2
gw1      GWTYPE = TMAXNONTX,
         PORTNO = 7510,
         RGWADDR ="192.168.1.43",
         RGWPORTNO = 8510,
         NODENAME = tmaxh4,
         CPC = 3, LOAD = 1

#Gateway for domain 3
gw2      GWTYPE = TMAXNONTX,
         PORTNO = 7520,
         RGWADDR ="192.168.1.48",
         RGWPORTNO = 8520,
         NODENAME = tmaxh2,
         CPC = 3, LOAD = 1
```

<domain2.m>

```
*DOMAIN
tmax1       SHMKEY = 78500, MINCLH=2, MAXCLH=3,
            TPORTNO=8590, BLOCKTIME=10000

*NODE
tmaxh4      TMAXDIR = "/EMC01/starbj81/tmax",

*SVRGROUP
svg1        NODENAME = "tmaxh4"

*SERVER
svr2        SVGNAME = svg1

*SERVICE
TOUPPER     SVRNAME = svr2

*GATEWAY
gw1         GWTYPE = TMAXNONTX,
            PORTNO = 8510,
            RGWADDR="192.168.1.43",
            RGWPORTNO = 7510,
            NODENAME = tmaxh4,
            CPC = 3
```

<domain3.m>

```
*DOMAIN
tmax1       SHMKEY = 78500, MINCLH=2, MAXCLH=3,
            TPORTNO=8590

*NODE
tmaxh2      MAXDIR = "/data1/starbj81/tmax",

*SVRGROUP
svg2        ODENAME = "tmaxh2"

*SERVER
```

```
svr2        SVGNAME = svg2

*SERVICE
TOUPPER    SVRNAME = svr2

*GATEWAY
gw2         GWTYPE = TMAXNONTX,
            PORTNO = 8520,
            RGWADDR="192.168.1.48",
            RGWPORTNO = 7520,
            NODENAME = tmaxh2,
            CPC = 3
```

Note

Up to Tmax 5 SP1, the general server groups (normally dummy servers) of the node, to which the gateway belongs, must also be registered and bound together in the COUSIN option. In the previous example <domain1.m>, gw1 belongs to the node, tmaxh4, and gw2 belongs to tmaxh2. General server groups of the two nodes must be included in the COUSIN server group.

However, starting from Tmax 5 SP2, only the gateway instead of its node can be included in the COUSIN option.

- For versions earlier than Tmax 5 SP2

```
*SVRGROUP
svg1       NODENAME = tmaxh4,
           COUSIN = "svg2, gw1, gw2"
svg2       NODENAME = tmaxh2

*GATEWAY
gw1       NODENAME = tmaxh4
gw2       NODENAME = tmaxh2
```

- For Tmax 5 SP2 and later

  Configure the settings used in Tmax 5 SP2 and previous and the COUSIN setting.

```
Tmax 5 SP2 and later

*GATEWAY
gw1       NODENAME = tmaxh4
          COUSIN = "gw2"
gw2       NODENAME = tmaxh2
```

COUSIN option cannot be directly set in the GATEWAY section, but can be specified for a general server group.

Tmax 5 SP2 and later allow COUSIN configuration in the GATEWAY section in

addition to the existing method.

B.3. Intelligent Routing of Domain Gateway

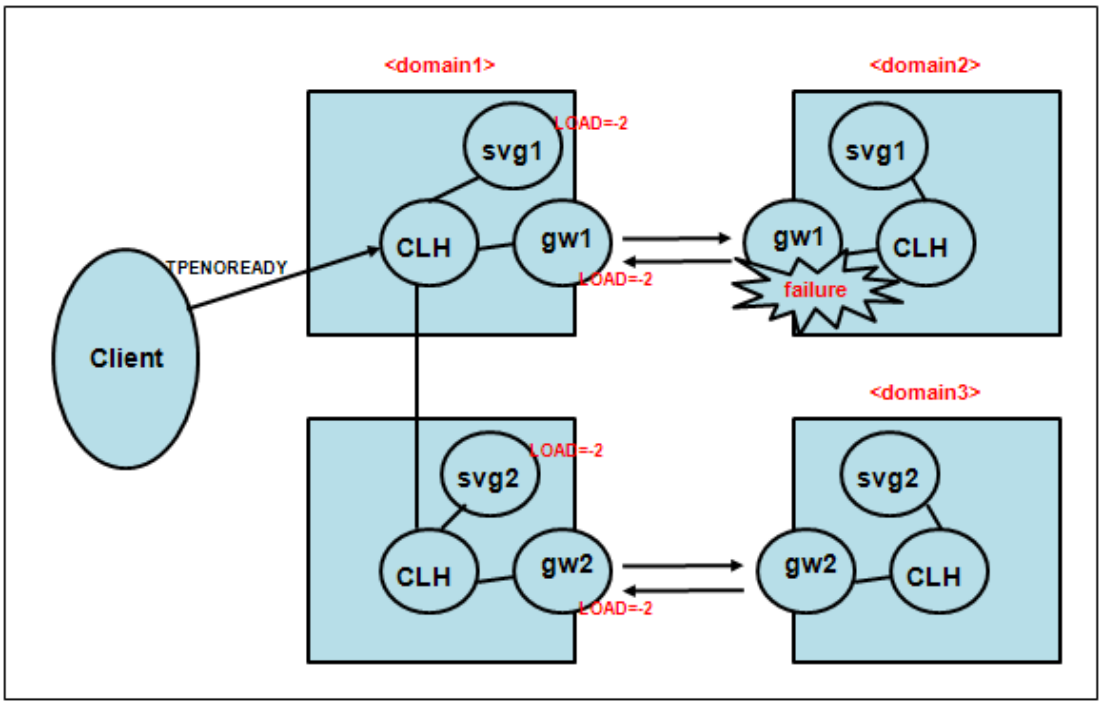
For load balancing in a multi node environment, if a server of a local node is terminated, CLH reschedules the request on a remote node. This way, client requests can be processed without errors. A CLH of a local node can issue a request to the remote node without knowing that a server on the remote node has been downed causing the client to receives a TPENOREADY error.

The intelligent routing feature (IRT) can be used to resolve such issue. It is used to manage the state of remote servers for static load balancing, and to reschedule (IRT – Intelligent Routing) the requests to other available servers. If a process of a remote node is terminated unexpectedly, it will be rescheduled to an available server.

The maximum number of server groups that can be rescheduled to is 10, and rescheduling is only triggered by a TPENOTREADY error.

B.3.1. Existing Domain Gateway Routing

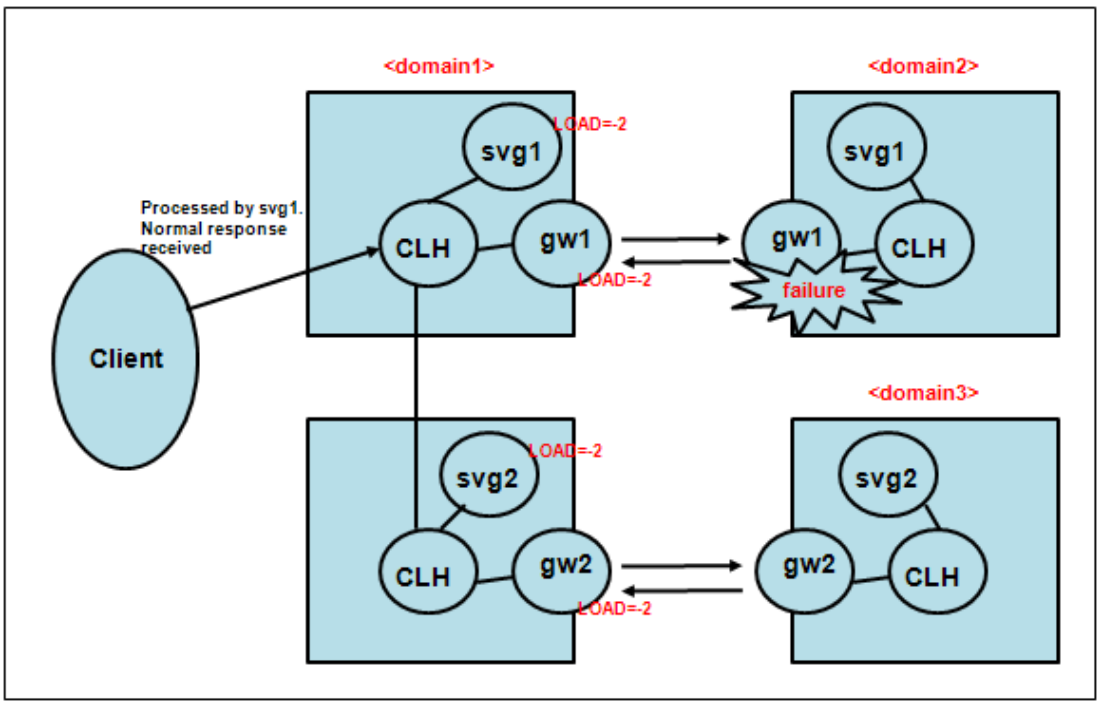
Since the existing domain gateway cannot check the state of a remote node and remote domain, it cannot detect a faulty server (including gateway) on a remote node, and can proceed with scheduling causing the client to receive a TPENOREADY error.



Existing Domain Gateway Routing

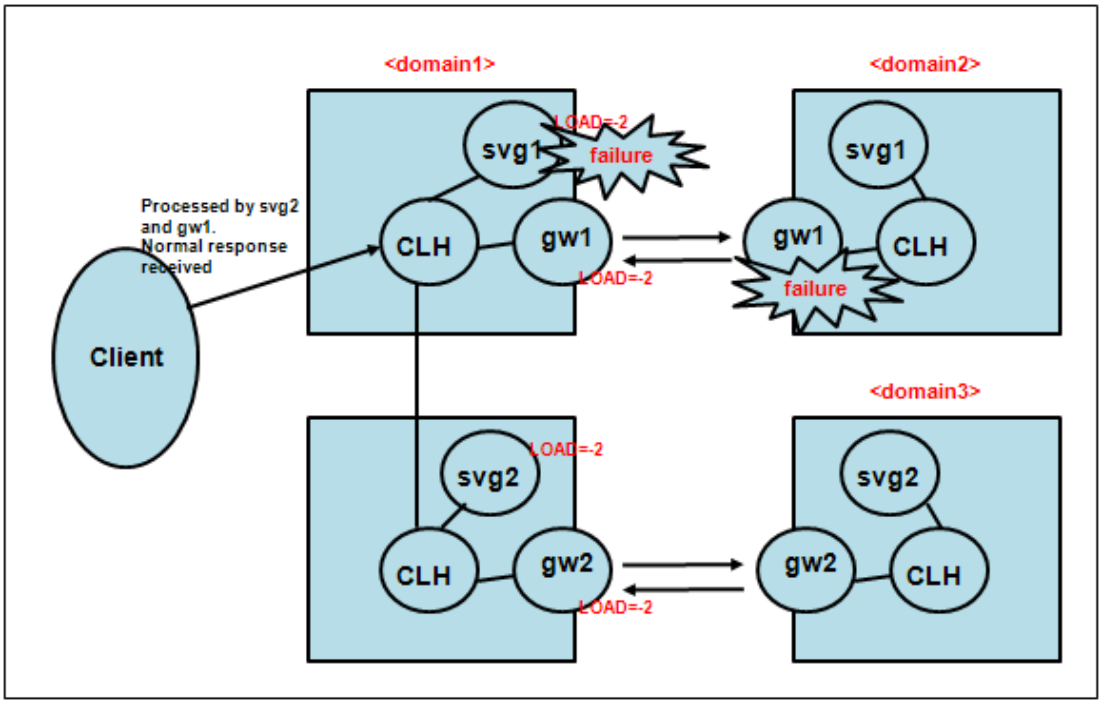
B.3.2. New Domain Gateway Routing

The new domain gateway checks the state of a remote node/domain to detect any remote gateway errors and route the request to another server group (svg1).



New Domain Gateway Routing (1)

If svg1 is a dummy server group or a network failure occurs between gateways, the load is distributed to the remote nodes, svg2 and gw2, at 1:1 ratio.



New Domain Gateway Routing (2)

B.3.3. Gateway that Supports Intelligent Routing Function

IRT was supported only for non-transaction gateways in Tmax 5 SP1 and earlier versions. Starting from Tmax 5 SP2, IRT is supported for TMAXNONTX, TMAX, JEUS, JEUS_ASYNC, TUXEDO, and TUXEDO_ASYNC gateways.

B.4. Version-specific FD Calculation

In Tmax, FD (maximum user) calculation is different for each version. The following are the formulas for calculating the FD value for each version.

- **Tmax 5 - Tmax 5 SP1 Fix#1 b.3.5**

```
maximum user = max_fd
               - maxspr
               - maxcpc
               - maxtms
               - maxclh
               - Tmax internal FD(21)
               - [(nodecount(The number of active nodes) - 1(own node)) * maxclh * domain.cpc]
```

- **Tmax 5 SP1 Fix#1 b.3.7 - Tmax 5 SP2**

```
maximum user = max_fd
               - maxspr
               - maxcpc
               - maxtms
               - maxclh
               - Tmax internal FD(21)
               - [FD connection between CLHs for multi node (maxnode(default:32) * maxclh *
domain.cpc * 2)]
```

As for the **'max_fd'** value, the smaller value between the values specified for the system and for Tmax will be used. The cpc value, **'domain.cpc'**, can be checked using 'cfg -d' command in tmadmin. Each setting can be checked using the cfg command in tmadmin.