NonStop™ Systems
NonStop 1+™ System

Transaction Monitoring Facility (TMF™) Reference Manual

Data Management Library

82541

NOTICE

Effective with the B00/E08 software release, Tandem introduced a more formal nomenclature for its software and systems.

The term "NonStop 1+™ system" refers to the combination of NonStop 1+ processors with all software that runs on them.

The term "NonStop™ systems" refers to the combination of NonStop II™ processors, NonStop TXP™ processors, or a mixture of the two, with all software that runs on them.

Some software manuals pertain to the NonStop 1+ system only, others pertain to the NonStop systems only, and still others pertain both to the NonStop 1+ system and to the NonStop systems.

The cover and title page of each manual clearly indicate the system (or systems) to which the contents of the manual pertain.

NonStop™ Systems
NonStop 1+™ System

Transaction Monitoring Facility (TMF™) Reference Manual

Abstract
This manual describes the syntax of all TMFCOM commands and TMF media-handling commands. It also describes certain related commands in PATHCOM, the Peripheral Utility Program (PUP), the File Utility Program (FUP), and the system procedures involved in using TMF.

Product Version
TMF B00
TMF E08

Operating System Version
GUARDIAN B00 (NonStop Systems)
GUARDIAN E08 (NonStop 1+ System)

Part No. 82541 A00

March 1985

Tandem Computers Incorporated
19333 Vallco Parkway
Cupertino, CA 95014-2599

DOCUMENT HISTORY

Edition	Part Number	Operating System Version	Date
First Edition	82341 A00	GUARDIAN E05/A04	October 1982
Second Edition	82341 B00	GUARDIAN E06/A05	April 1983
Update 1	82193	GUARDIAN E07/A06	December 1983
Third Edition	82541 A00	GUARDIAN E08/B00	March 1985

New editions incorporate all updates issued since the previous edition. Update packages, which are issued between editions, contain additional and replacement pages that you should merge into the most recent edition of the manual.

NEW AND CHANGED INFORMATION

This revised *Transaction Monitoring Facility (TMF) Reference Manual* introduces five new TMFCOM commands and several new options for two existing commands. It also incorporates other product changes for the March 1985 release and corrects various minor technical and typographical errors.

Most of the product changes introduced in this revision are for the new autorollback feature. In most failure situations, this feature makes rollforward unnecessary. To help you control autorollback, the TMFCOM command START TMF has two new parameters and there are three new commands: EXCLUDE VOLUME, ENABLE VOLUMES, and STATUS VOLUMES.

Two other new commands, REPORT DUMPS and ALTER AUDITTRAIL, are not directly related to autorollback.

Also, the ADD AUDITTRAIL command is now slightly different for users of DP2 disc processes.

CONTENTS

TABLES

PREFACE

This reference manual describes the syntax of all TMFCOM commands and TMF media-handling commands for all Tandem computer systems. It also dêscribes certain related commands in PATHCOM, the Peripheral Utility Program (PUP), and the File Utility Program (FUP). The tasks involved in configuration and operation of TMF for your system are described in the *Transaction Monitoring Facility (TMF) System Management and Operations Guide* for your system. Programming considerations are discussed in the *PATHWAY SCREEN COBOL Reference Manual* and the *GUARDIAN Operating System Programmer's Guide*.

All TMF users should first read the *Introduction to the Transaction Monitoring Facility (TMF)*, which presents the basic concepts related to TMF.

For detailed reference information about FUP or PUP, see the *GUARDIAN Operating System Utilities Reference Manual* for your system. For detailed operations information about FUP, see the *GUARDIAN Operating System User's Guide* for your system. For detailed operations information about PUP, see the *System Operator's Guide* for your system.

For detailed reference information about system generation (SYSGEN), see the *System Management Manual* for your system.

For detailed information about PATHWAY, see the *Introduction to PATHWAY*, the *PATHAID Reference Manual*, the *PATHWAY System Management Reference Manual*, and the *PATHWAY SCREEN COBOL Reference Manual*.

In this manual, Section 1 outlines the categories of tasks involved in using TMF.

Section 2 describes all the TMFCOM commands, arranged in alphabetical order.

Section 3 describes the TMF media-handling commands.

Section 4 describes the FUP commands that are related to TMF.

Section 5 describes the PUP commands that are related to TMF.

Section 6 describes the PATHCOM commands that are related to TMF.

Section 7 describes the SCREEN COBOL verbs and special registers that are related to TMF.

Section 8 describes the GUARDIAN procedures that are related to TMF.

Appendix A discusses error messages.

Appendix B describes file identifiers and the basic commands used in TMFCOM. These commands, such as FC, HELP, and OBEY, are similar to those used in most other Tandem software.

SYNTAX CONVENTIONS USED IN THIS MANUAL

This table describes the characters and symbols used in this manual's syntax notation. For distinction, syntactical elements appear in a typeface different from that of ordinary text.

Notation	Meaning
UPPERCASE CHARACTERS	All keywords and reserved words appear in capital letters. If a keyword can be abbreviated, the part that can be omitted is enclosed in square brackets.
lowercase characters	All variable entries supplied by the user are shown in lowercase characters.
Brackets	Square brackets ([ ]) enclose all optional syntactic elements. A vertically aligned group of items enclosed in brackets represents a list of selections from which to choose one or none.
Braces	A vertically aligned group of items enclosed in braces ({ }) represents a list of selections from which exactly one must be chosen.
Vertical Bars	A vertical bar (|) between syntax elements represents "or" in a situation in which one item is to be chosen. This usually occurs when a small number of simple elements are involved.
Ellipses	When an ellipsis (...) immediately follows a pair of brackets or a pair of braces, the enclosed syntax can be repeated any number of times.
Punctuation	Parentheses, commas, and other punctuation or symbols not described above must be entered precisely as shown. If any of the punctuation above appears enclosed in quotation marks, that character is not a syntax descriptor but a required character, and must actually be entered.
(I) or (I only)	This notation, alongside a command syntax diagram or option description, indicates an option that is valid only on a NonStop 1+ system (not on a NonStop system).
(II/DP1)	This notation, alongside a command syntax diagram, indicates an option that is valid only on a NonStop system with DP1 disc processes. Similarly, the notation (II/DP2) indicates an option that is valid only on a NonStop system with DP2 disc processes.

SECTION 1

INTRODUCTION

The Transaction Monitoring Facility (TMF) is one of the products in the ENCOMPASS Distributed Data Management System. The purpose of TMF is to simplify the task of maintaining data consistency for a distributed data base that is being updated by concurrent transactions.

TMF's features include concurrency control, transaction backout, audit trails, online dumps, and the rollforward facility.

To support a system with TMF, someone must:

- program the requester and server processes that use TMF
- configure the objects required by TMF for its operations
- control the TMF operations.

Performance of these tasks is described in detail in other publications. This reference manual presents the syntax and some considerations for the TMFCOM commands and other commands which facilitate the use of TMF.

TMFCOM COMMANDS

TMFCOM is a utility provided by TMF to let you configure and control TMF. There are several types of TMFCOM commands:

- configuration commands that let you define and display the attributes of objects that are required by TMF for its operations
- operational commands that let you operate TMF after it has been configured
- catalog management commands that let you change information in the catalog
- dumping and recovery commands that let you recover from multiple-component failures
- basic commands that help you use the TMFCOM utility.

The TMFCOM commands are grouped by function in the next subsection and explained in detail in Section 2.

Configuration Commands

The TMFCOM commands listed below let you configure the objects that TMF requires for its operations and display their attributes.

Initialization:

- INITIALIZE TMF purges all configuration information defined by the ALTER BACKOUT, ALTER CATALOG, ALTER TMF, ADD AUDITDUMP, and ADD AUDITTRAIL commands and deletes all dumps information from the catalog. INITIALIZE TMF is required before you issue any command that changes an existing configuration of the audit trails.

- ALTER TMF specifies an operator terminal or process for tape and disc requests and permits improvement of transaction time by disabling preparation for rollforward recovery.

Configuring the backout process:

- ALTER BACKOUT changes the execution priority of the backout process and specifies the primary and backup processors where it is to be run.

- INFO BACKOUT displays the current attributes of the backout process, as configured by previous ALTER BACKOUT commands.

Configuring the catalog:

- ALTER CATALOG changes attributes of the catalog process.

- INFO CATALOG displays the current attributes of the catalog, as configured by previous INITIALIZE TMF and ALTER CATALOG commands.

Configuring the audit trails:

- SET AUDITTRAIL establishes the creation values of attributes for audit trails.

- RESET AUDITTRAIL restores default values to any audit-trail attributes that you have established previously by the SET AUDITTRAIL command.

- SHOW AUDITTRAIL displays the current audit-trail attributes, as established by the SET AUDITTRAIL and RESET AUDITTRAIL commands.

- ADD AUDITTRAIL defines and names all audit trails. This command also defines which audit processes or disc processes will write to each audit trail and which audit-dump process (if any) will dump each audit trail.

- ALTER AUDITTRAIL changes some audit-trail attributes.

- INFO AUDITTRAIL displays the current attributes of selected audit trails, as configured by previous ADD AUDITTRAIL commands.

Configuring one or more audit-dump processes:

- SET AUDITDUMP establishes the creation values of attributes for audit-dump processes.

- RESET AUDITDUMP restores default values to any audit-dump attributes that you have established previously by the SET AUDITDUMP command.
- SHOW AUDITDUMP displays the current audit-dump attributes, as established by the SET AUDITDUMP and RESET AUDITDUMP commands.
- ADD AUDITDUMP defines and names processes for the automatic dumping of audit-trail files.
- INFO AUDITDUMP displays the current attributes of selected audit-dump processes, as configured by previous ADD AUDITDUMP or ALTER AUDITDUMP commands.
- ALTER AUDITDUMP changes attributes of one or more audit-dump processes.
- DELETE AUDITDUMP removes audit-dump process descriptions previously specified by the ADD AUDITDUMP or ALTER AUDITDUMP commands.

Displaying configured attributes:

- INFO TMF displays the current attributes of the backout, catalog, and audit-dump processes, and of the audit trails, as configured by previous ADD AUDITTRAIL, ADD AUDITDUMP, ALTER CATALOG, and ALTER BACKOUT commands.

Operational Commands

After you configure TMF, you can use these TMFCOM commands to operate TMF:

Allowing and disallowing transaction processing:

- START TMF activates TMF after TMF has been configured or shut down.
- STOP TMF shuts down TMF.
- START TRANSACTION allows transaction processing after a START TMF command with the *TRANSACTION OFF* option or a STOP TRANSACTION command.
- STOP TRANSACTION disallows new transaction processing without stopping TMF.
- On a NonStop system, ENABLE VOLUMES attempts autorollback, if necessary, and makes one or more disc volumes available for TMF transaction processing.
- On a NonStop system, EXCLUDE VOLUMES lets TMF purge audit-trail files that may be required for autorollback of the specified disc volume. This purging may make autorollback on that volume impossible; if so, some files on the volume may require rollforward recovery.
- On a NonStop system, STATUS VOLUMES tells which disc volumes are enabled for TMF transaction processing.

Reconfiguring audit-dump processes:

- CONTROL AUDITDUMP changes some attributes for audit-dump processes while TMF is running (i.e., after START TMF).
- STATUS AUDITDUMP displays the current status of one or more audit-dump processes.

Controlling audit trails:

- NEXT AUDITTRAIL begins creation of the next audit file in the specified audit-file sequence.

- STATUS AUDITTRAIL displays the current state of one or more audit trails.

Displaying TMF system status:

- STATUS BACKOUT displays the current state of the backout process.

- STATUS CATALOG displays the current state of the catalog process.

- STATUS TMF displays the current state of TMF on the system. It also displays the current attributes of the audit trails and the backout, audit-dump, and catalog processes.

Controlling specific transactions:

- ABORT TRANSACTION aborts a transaction. TMF backs out the transaction and releases any locks held for it.

- On a NonStop 1+ system only, DELETE TRANSACTION is an emergency measure which releases any locks held for the specified transaction, regardless of its state of completion.

- END TRANSACTION forces a transaction to commit its changes and releases any locks held for it.

- STATUS TRANSACTION displays the transaction identifier and status of selected transactions in the system in which TMFCOM is executing.

The DELETE TRANSACTION command is an emergency measure only; you should not use it unless you have already tried ABORT TRANSACTION or END TRANSACTION (and even these should only be needed under exceptional circumstances). Beware that DELETE TRANSACTION may allow a data base inconsistency. The DELETE TRANSACTION command is not valid on NonStop systems.

Catalog Management Commands

While TMF is running (i.e., after START TMF), you can use the commands described below to inspect and modify the contents of the catalog. These commands affect only those catalog entries which belong to the user issuing the command.

Purging catalogued data:

- INITIALIZE CATALOG deletes all tape, disc, and dump records from the catalog data base.

Controlling dump references:

- On a NonStop (not NonStop 1+) system, the REPORT DUMPS command uses precompiled ENFORM reports to allow a thorough inspection of the TMF catalog.

- INFO DUMPS lists all or some of the catalog, online or audit dumps currently recorded in the catalog.

- SET DUMPS establishes new creation values for catalog entries of dumps, for use by subsequent ADD DUMPS commands.

- RESET DUMPS restores default values to any catalogable attributes that you have established previously by the SET DUMPS command.

- SHOW DUMPS displays the current creation values for catalog entries of dumps, as established by the SET DUMPS command.

- ADD DUMPS inserts new dump references into the catalog. Its only use should be for manual recovery of catalog data if something damages the catalog.

- ALTER DUMPS changes one or more dump references in the catalog.

- DELETE DUMPS removes one or more dump references from the catalog and can, thereby, change a tape reel's status to scratch or released.

Controlling tape or disc references:

- INFO MEDIA lists all or some of the tapes and discs currently recorded in the catalog's media directory.

- SET MEDIA establishes new creation values for catalog entries of tapes and discs, for use by subsequent ADD MEDIA commands.

- RESET MEDIA restores default values to any catalogable attributes that you have established previously by the SET MEDIA command.

- SHOW MEDIA displays the current creation values for catalog entries of tapes and discs, as established by the SET MEDIA command.

- ADD MEDIA inserts new tape-reel and disc-pack references into the catalog's media directory and can, at the same time, (re)write tape labels.

- ALTER MEDIA changes tape-reel and disc-pack references in the catalog's media directory and can, at the same time, rewrite tape labels.

- DELETE MEDIA deletes tape-reel and disc-pack references from the catalog's media directory and also deletes dump-directory records of files on those tapes and discs.

Dumping and Rollforward Commands

While TMF is running (that is, after START TMF), you should use the TMFCOM and PUP commands described below to create online dumps of audited files on tapes or discs and to use those dumps to recover files.

- The DUMP FILES command initiates an online dump, copying all of the specified audited files to tape for possible use later by the rollforward process.

- The PUP commands REMOVEAUDITED and REVIVE let you remove one member of a mirrored disc pair for use as an online dump, then replace it with a "scratch" disc to restore mirrored operation.

- The RECOVER FILES command initiates rollforward recovery, using previously created online dumps, then uses audit trails to roll the reloaded data-base files forward to their latest possible consistent state.

Basic Commands

The basic commands are similar to those incorporated in other Tandem software products. These commands, and their functions, are summarized below.

- CMDSYS sets the default system for expansion of any file names, with the exception of OBEY file names.
- CMDVOL sets the default volume and subvolume for expansion of any file names, with the exception of OBEY file names.
- ENV displays the current settings of various parameters of the GUARDIAN Command Interpreter.
- EXIT terminates execution of TMFCOM.
- FC fixes a command, using facilities of the FIXSTRING procedure.
- HELP displays information about TMFCOM commands.
- OBEY introduces commands from a command file.
- OBEYSYS sets the default system for expansion of OBEY file names.
- OBEYVOL sets the default volume and subvolume for expansion of OBEY file names.
- OUT redirects listing output.
- SYSTEM sets the default system for expansion of all file names.
- VOLUME sets the default volume and subvolume for expansion of all file names.

TAPE MANAGEMENT

TMF sometimes asks you to load a tape or a disc dump, or to supply a tape-drive identifier or other information. TMF provides a special set of commands to help you perform these media-handling tasks.

Terminology

TMF's media catalog has four possibilities for the status of each tape:

- An *assigned* tape is one which contains valid, catalogued data. When a dump process writes to a tape, it assigns a dump serial number to that tape.
- A *bad* tape is one which is physically defective.
- A *scratch* tape is one which is free for reassignment.

- A *released* tape is one which has been freed from assignment, but has not yet been declared scratch. Tapes can be released only if an ALTER CATALOG command with the *RELEASED ON* option has been given, and the tape has subsequently been freed from assignment. To re-use a released tape, you must first use the ALTER MEDIA command. For instance, you may want to keep scratch tapes in a physically separate location. You would periodically use INFO MEDIA to determine which tapes should be moved to that location, then use ALTER MEDIA to scratch them in the media directory and, if you like, overwrite their labels.

A *reel identifier* names a physical reel of tape. The identifier can comprise any combination of one to six letters and digits, such as MARKET, X, JER12, or 123456, except that no reel identifier which contains letters can begin with a digit. Note that the reel identifier names the physical tape, not the data recorded on it. In the identifier, a lowercase letter is not equivalent to the corresponding uppercase letter; "abc" and "ABC" identify two different reels.

A *tape label* is a magnetically encoded area at the beginning of a reel of tape, containing the reel identifier and other information. You can label tapes with the ADD MEDIA or ALTER MEDIA command.

A *tape reel* is a physical reel of tape, sometimes called a *tape volume.* Its reel identifier is written in a label at the beginning of the tape and should also be written visibly on the physical reel.

A *tape file* is a sequence of one or more *file sections*, each of which resides on a separate reel; thus, a file which spans three reels has three file sections.

A new online-dump *generation* is created each time a file is dumped. Each distinct generation of the file on tape or disc has a different dump serial number. All copies of all sections of a given file have the same serial number.

Media-Handling Commands

TMF can require operator intervention (such as mounting a tape) before automatic dumping of an audit trail and when you use the TMFCOM commands DUMP FILES, RECOVER FILES, ADD MEDIA, and ALTER MEDIA. Whenever such intervention is needed, the media-handling facility uses an at-sign (@) to prompt you for a response.

Depending on the situation, your response may include one or more of the media-handling commands described in Section 3. These commands are DISABLE, DRIVE, ENABLE, EXIT, FC, HELP, NEXT, REEL, RELEASE, STATUS, STOP, and WAIT.

RELATED ASPECTS OF OTHER PRODUCTS

Proper use of TMF can involve not only the TMFCOM commands, but the use of GUARDIAN procedures, SCREEN COBOL verbs and registers, and certain FUP, PUP, and PATHCOM commands.

FUP Commands

Associated with every file is a flag indicating whether TMF is to audit that file. You can designate files as audited or change the status of a file between audited and non-audited by using the CREATE procedure or one or more of the File Utility Program (FUP) commands ALTER, CREATE, RESET, and SET.

Section 4 discusses these FUP commands and explains the special options which let you designate audited files. For detailed reference information about FUP, see the *GUARDIAN Operating System Utilities Reference Manual* for your system. For detailed operations information about FUP, see the *GUARDIAN Operating System User's Guide* for your system.

PUP Commands

Two commands of the Peripherals Utility Program (PUP), REMOVEAUDITED and REVIVE, together allow you to create online dumps on removable discs. Section 5 shows the syntax of these PUP commands. See the *TMF System Management and Operations Guide* for discussion of the use of certain other PUP commands, including LISTBAD, LISTDEV, LISTFREE, and SPARE, in preparation for rollforward recovery.

For detailed reference information about PUP, see the *GUARDIAN Operating System Utilities Reference Manual* for your system. For detailed operations information about PUP, see the *System Operator's Guide* for your system.

PATHCOM Commands

PATHCOM, the command utility for PATHWAY, includes several features to help you configure PATHWAY systems to use TMF. You can:

- use the SET PATHWAY command to specify a global *transaction restart limit* that defines the maximum number of times a failed transaction can be restarted automatically
- use the ADD SERVER, ALTER SERVER, RESET SERVER, and SET SERVER commands to define server classes that can update audited files
- in configuring terminal program units, use the ADD TERM, ALTER TERM, RESET TERM, and SET TERM commands to specify which program units (if any) are not configured to operate in TMF transaction mode, and the STATUS TERM command to report on how each terminal is configured.

Section 6 shows the syntax of these PATHCOM commands. For more complete details on the commands, see the *PATHWAY System Management Reference Manual.*

SCREEN COBOL Verbs and Registers

SCREEN COBOL includes four verbs which enable programming of PATHWAY applications for TMF:

- ABORT-TRANSACTION aborts and backs out a transaction.
- BEGIN-TRANSACTION starts a transaction.
- END-TRANSACTION ends a transaction.
- RESTART-TRANSACTION aborts, then restarts a transaction from the BEGIN-TRANSACTION point.

Also, the SCREEN COBOL compiler automatically defines three special registers for TMF users: TRANSACTION-ID, TERMINATION-STATUS, and RESTART-COUNTER. Their uses and implicit declarations, together with the TMF-oriented SCREEN COBOL verbs, are discussed in Section 7 and are described in detail in the *PATHWAY Programming Manual.*

GUARDIAN Procedures

Several callable procedures exist to help programmers write or adapt applications for handling TMF transactions. These procedures, ABORTTRANSACTION, ACTIVATERECEIVETRANSID, BEGINTRANSACTION, ENDTRANSACTION, GETTMPNAME, GETTRANSID, and RESUME-TRANSACTION, are described in detail in Section 8.

RUNNING TMFCOM

TMFCOM is designed to run either non-interactively, by reading commands from a command file, or interactively.

Starting TMFCOM Operations

TMFCOM resides in the file $SYSTEM.SYSTEM.TMFCOM; the command to run it from the GUARDIAN Command Interpreter has this form:

```
TMFCOM [ / [ IN file ] [ , OUT [file] / ] [command] [; command ]...]
```

where

`IN file`

specifies a file from which commands are to be read. If you do not specify a command *file*, the GUARDIAN Command Interpreter automatically supplies the name of its own input source—usually the home terminal.

`OUT file`

specifies a file to which all output, other than prompts, is to be written. If you omit this option, the GUARDIAN Command Interpreter automatically supplies the name of its own output *file*—usually the home terminal. If you specify *OUT* without *file*, output is suppressed.

`command`

is a TMFCOM command or other optional specification. If any commands are present, TMF-COM executes the commands and terminates without reading the input source file. If no commands are present, TMFCOM awaits interactive commands.

In the TMFCOM interactive command cycle, TMFCOM prints a tilde (~) to prompt you for a command, you enter the command, TMFCOM executes the command, and the cycle repeats.

To run TMFCOM from the GUARDIAN Command Interpreter, using a command file, enter a command like

```
TMFCOM /IN TMFFILE,OUT $LP/
```

In this example, TMFCOM would execute the commands in TMFFILE and send all their output to process $LP. When an error, a control-Y, or an EXIT command is encountered, TMFCOM terminates and the GUARDIAN Command Interpreter is activated.

Terminating TMFCOM Operations

The EXIT command or control-Y terminates TMFCOM execution. The form of the EXIT command is:

```
EXIT
```

with no options.

Using the HELP Command

The HELP command displays brief descriptions of all TMFCOM commands, including HELP itself. Some of these descriptions are in successively more detailed stages, as in the examples shown below. The HELP displays are intended to be reminders, not substitutes for this publication. These examples demonstrate the information you can display using HELP:

The command

```
HELP ADD
```

displays this list of the ADD commands in TMFCOM:

```
ADD AUDITDUMP <local process name>
ADD AUDITTRAIL <generic filespec>
ADD DUMPS <file-set list>
ADD MEDIA <media list>
```

The command

```
HELP ADD DUMPS
```

displays the syntax of the ADD DUMPS command:

```
ADD DUMPS <file-set list> [ , <add dumps option> ]...
```

The command

```
HELP <ADD DUMPS OPTION>
```

displays the syntax of the options available for the ADD DUMPS command:

```
<add dumps option> is:
          { BRIEF | DETAIL                |
            MEDIUM <medium specification>                  |
            SERIAL <integer>                               |
            TYPE { AUDITDUMP                            |
                   ONLINEDUMP (<integer1>, <integer2>) }  }
```

The command

```
HELP <MEDIUM SPECIFICATION>
```

displays the syntax of one specific option within the ADD DUMPS command:

```
{ DISC <disc label>           |
  TAPE <tapes specification> }
```

See Appendix A for a detailed description of the HELP command.

TMFCOM Security

Any member of the SUPER group can execute any TMFCOM command except INITIALIZE TMF. Any member of the SUPER group can execute the first INITIALIZE TMF after SYSGEN. Subsequently, however, only that SUPER member can initialize TMF, because that member is the owner of the TMF configuration files in $SYSTEM.TMF.*. (Of course, SUPER.SUPER can always execute any command.) If $SYSTEM.TMF.* is purged, then any SUPER-group member can initialize TMF; if ownership of $SYSTEM.TMF.* is given to another SUPER-group member, then the new owner alone can initialize TMF.

The ADD MEDIA command, as well as the various REPORT, INFO, STATUS, SET, SHOW, and basic commands, are available to any user. To use DELETE DUMPS, ALTER DUMPS, or ALTER MEDIA, however, your user name must be either SUPER.SUPER or the owner of the object to be deleted or altered. All other TMFCOM commands require that you be a member of the SUPER group.

The owner of any dump or storage medium is the SUPER-group member who owned the configuration files when the dump was made.

SECTION 2

TMFCOM COMMANDS

Table 2-1. TMFCOM Command Keywords

		DUMPS	MEDIA	AUDITDUMP	AUDITTRAIL	BACKOUT	CATALOG	TMF	TRANSACTION	VOLUMES	FILES
Transaction Processing	START							▨	▨		
	STOP							▨	▨		
	END								▨		
	ABORT								▨		
Configuration & Catalog	DELETE	▨	▨	▨					■		
	RESET	▨	▨	▨	▨						
	SET	▨	▨	▨	▨						
	SHOW	▨	▨	▨	▨						
	ADD	▨	▨	▨	▨						
	REPORT	▩									
Active Processes	ALTER	▨	▨	▨	▩	▨	▨	▨			
	INFO	▨	▨	▨	▨	▨	▨	▨			
	STATUS			▨	▨	▨	▨	▨	▨	▩	
	INITIALIZE						▨	▨			
Autorollback	ENABLE									▩	
	EXCLUDE									▩	
	NEXT				▨						
Rollforward	CONTROL			▨							
	DUMP										▨
	RECOVER										▨

Note: ■ Indicates valid combinations for NonStop 1+ System only.

▩ Indicates valid combinations for NonStop Systems only.

▨ Indicates valid combinations for NonStop 1+ or NonStop Systems.

S5036-001

Most of your tasks in configuring and operating TMF are performed through TMF-COM commands. Each of these commands is identified by two keywords, the first specifying an action to be taken and the second specifying the target of that action.

Table 2-1 shows which action keywords are associated with each object. For example, the backout process is the target of ALTER, STATUS, and INFO commands.

Including HELP, TMFCOM has five kinds of information commands. SHOW commands display creation values, as established by SET and RESET commands. INFO commands display currently configured attributes or catalog references. STATUS commands display the current states of specified processes, audit processes, or transactions. On a NonStop (not NonStop 1+) system, the REPORT DUMPS command allows a thorough inspection of the TMF catalog. The remaining commands perform specific configuration or operation tasks.

TRANSACTION IDENTIFIERS

In a Tandem system with TMF, each transaction is a uniquely identified entity. Each transaction is distinguished from other transactions by a transaction identifier. The general TMFCOM syntax for a transaction identifier is:

```
[ \system-name   ]( crash-count ). cpu . sequence-number
[ \system-number ]
```

where

`\system-name or \system-number`

specifies the name or number of the home node (the system where the transaction originated). If you do not specify the *system name* or the *system number*, this defaults to the system on which TMFCOM is running.

`crash-count`

is the number of times that the home node for the transaction has had a total system failure since the INITIALIZE TMF command was last issued on that node.

`cpu`

is the number of the processor where BEGIN-TRANSACTION was issued for the transaction being specified.

`sequence-number`

is the sequence number of the transaction identifier in the CPU where it originated.

With the STATUS TRANSACTION command only, you can substitute an asterisk (*) for any element of the transaction identifier. The asterisk then serves as a "wild card," matching any value for that element. For example, the command

```
STATUS TRANSACTION 3.*
```

applies to any transaction whose home node is the local system and which originated in CPU 3, regardless of its transaction sequence number.

The transaction identifier's internal form is described in Section 8, "Related GUARDIAN Procedures."

GENERIC FILE-SET IDENTIFIERS

An audit-trail *sequence* is a named group of files which TMF creates as it builds the audit trail. In several of the TMFCOM commands, you identify one or more such sequences by a *generic file-set* identifier in this form:

```
[ [ [ \system. ] $volume. ] subvolume. ] generic-file-name
```

To specify a *generic file name*, you use only two alphabetic letters. In the course of building its audit trails, TMF appends six-digit numbers to create actual file names. With *generic file name* XY, for example, TMF would write audit trails to files XY000001, XY000002, etc.

The disc specified by *volume* must be a mirrored disc and must be on the local system; that is, the system on which (1) the command is issued and (2) the affected audit process or disc process exists. If *volume* is not specified, these restrictions must be satisfied by the current default volume.

With some commands, you can substitute an asterisk (*) for one or more elements of the *generic file-set* identifier, as with standard GUARDIAN file identifiers. Certain other TMFCOM commands, however, do not allow this use of asterisks.

ABORT TRANSACTION

ABORT TRANSACTION backs out a transaction and releases any locks it holds. The command syntax is:

```
ABORT TRANSACTION transaction-identifier [ ! ]
```

where

`transaction-identifier`

identifies the transaction to be aborted. Its form is defined in the "Transaction Identifiers" subsection near the beginning of this section.

`!`

indicates that the operation should be performed without waiting for confirmation. If this is omitted, TMF asks you for confirmation; a response of YES or Y allows the operation to proceed, but a response of NO, N, or STOP aborts the operation. With any other response, TMF displays an error message and reprompts you.

Considerations

If a distributed transaction is aborting on some network node, but a communications failure prevents notifying all the other nodes to abort it, the transaction will be stuck in the aborting state. If this happens, you can use the ABORT TRANSACTION command to abort the transaction on each local system.

Before using this command, use the STATUS TRANSACTION command to determine the status of the transaction on its home node. (If the EXPAND lines are down, this may require a telephone call or other external form of communication with someone on the other system.) If the transaction is ending on its home node, you can safely use END TRANSACTION on your local system. Otherwise, you should use ABORT TRANSACTION.

The ABORT TRANSACTION command initiates abortion as a "no-wait" operation which may take a considerable amount of time to complete.

Examples

The command

```
ABORT TRANSACTION (1).3.6668!
```

attempts to abort the transaction with transaction identifier (1).3.6688; that is, the transaction begun in CPU 3 of the local system, after one system failure, with sequence number 6668. It does not prompt for confirmation.

In the next example:

```
ABORT TRANSACTION \JUNEAU(1).4.8899977
\JUNEAU(1).4.8899977, PROCESS $DELREC(4,11), STATE ACTIVE
CONFIRM? YES
```

TMF accepts the ABORT TRANSACTION command, displays the status of the specified transaction, prompts for confirmation, then attempts to abort the transaction that has transaction identifier \JUNEAU(1).4.8899977; that is, a transaction that originated in CPU 4 of system \JUNEAU, after one system failure there, with a sequence number of 8899977.

ADD AUDITDUMP

ADD AUDITDUMP defines the names and initial attributes for an audit-dump process which the transaction monitor process (TMP) will create when TMF is started. An audit-dump process controls the automatic dumping of one or more audit-trail files. The names and attributes you can define with ADD AUDITDUMP are:

- the name of the audit-dump process
- the execution priority of the audit-dump process
- the primary and backup processors where the audit-dump process is to run
- the tape drive or drives to be used
- how many copies are to be made of each dump
- the output file for audit-dump reports.

The ADD AUDITDUMP command syntax is:

```
                          [ , COPIES quantity                          ]
                          [         ( cpu1          )                  ]
                          [ , CPUS  { (cpu1)        }                  ]
                          [         { [cpu1]:cpu2   }                  ]
ADD AUDITDUMP [process]   [         ( ([cpu1]:cpu2) )                  ] ...
                          [ , DRIVE { device                       }   ]
                          [         {   ( device [ , device ]...)  }   ]
                          [ , OUT file                                 ]
                          [ , PRI integer                              ]
```

where

`process`

names a process to be added to the local system. This name cannot include a subdevice, subvolume, or file-name component. If you do not name the process, TMF makes up a name.

`COPIES quantity`

specifies the number of copies to be made of each dump. The default *quantity* is 1. The maximum *quantity* is 99. Multiple copies are written simultaneously and, therefore, require multiple tape drives (see the DRIVE option below).

`CPUS`

specifies the primary and backup processors to be used for the audit-dump process. If either processor is not specified, the TMP selects it from among the available processors.

`cpu1`

is the number of the primary processor. Its range is 0 to 15.

`cpu2`

is the number of the backup processor. Its range is 0 to 15 and it must be different from *cpu1.*

→

`DRIVE`

specifies one or more tape drives to which the audit-dump process can dump audit-trail files. If you use this option, the audit-dump process will "know" the specified drives until you explicitly release them. If you do not use this option, the audit-dump process will prompt the operator for a device name each time it is ready to dump a file.

`device`

identifies a tape drive. This is a standard GUARDIAN device name, such as $TAPE or \KEOKUK.$TAPE2.

`OUT file`

specifies where the audit-dump reports should be written. If *file* names an existing disc file, the reports will be appended to the file. If it specifies a disc file which does not exist, an EDIT-type file will be created. If it specifies a process, such as the spooler, that process must exist when TMF is started and must remain available as long as TMF is running; writing to a nonexistent OUT process may cause TMF to crash.

Until OUT is specified with an ADD AUDITDUMP, ALTER AUDITDUMP, or CONTROL AUDITDUMP command, no reports are produced.

`PRI integer`

specifies the priority to be given to the audit-dump process. The range of valid priorities is 1 to 199. If no priority is specified, the process assumes TMFCOM's priority.

Considerations

While TMF is running (that is, after START TMF), you cannot use the ADD AUDITDUMP command; you can, however, use the TMFCOM command CONTROL AUDITDUMP to change some attributes of an audit-dump process while TMF is running.

On a NonStop system only, an audit-dump process is not a NonStop process pair. Therefore, the primary and secondary CPU designations are only starting points. If both CPUs fail, a console message is printed and the TMP tries to restart the process in the specified primary CPU. If that fails, it tries the specified secondary CPU, then CPU 0, CPU 1, ... until it finds a working CPU. (At least one CPU must be working, or the TMP could not be active.)

Examples

Assuming that none of the omitted options have had values set by SET AUDITDUMP, the command

```
ADD AUDITDUMP $AUD1, PRI 142, CPUS 9:12, COPIES 2
```

defines an audit-dump process named $AUD1 with

- an execution priority of 142
- processors 9 and 12 to be the primary and backup processors, respectively
- two copies to be made of each dump
- the operator to be prompted whenever $AUD1 needs a tape (because DRIVE is not specified)
- no reports.

The command

```
ADD AUDITDUMP $AUD2,DRIVE $TAPE1,OUT $DATA.TMF.DUMPREPT
```

defines audit-dump process $AUD2 with

- $TAPE1 as the tape drive to which audit-trail files will be dumped (one copy)
- reports to be written to disc file $DATA.TMF.DUMPREPT.

Whenever $AUD2 dumps an audit trail, it expects a scratch tape to be mounted on drive $TAPE1. If one is mounted, $AUD2 proceeds with no need for operator interaction; otherwise, it prompts the operator to mount a scratch tape. See Section 3, "Media-Handling Responses," for valid responses to system prompts.

For each audit-dump, the report is similar to this:

```
AUDITDUMP - T9066B00 - (28JAN85)   System \SAB   20-Feb-1985 19:47:18
[Dump serial number is 375.]
  File Name                      Error
[Tape JER234 used for copy 1 of reel 1 on drive $TAPE.]
$AUDIT.AP.AA000132
[Total files dumped = 1.]
```

This example shows that audit-trail file $AUDIT.AP.AA000132 was dumped to the tape labeled JER234, using drive $TAPE, at 7:47 p.m. on April 20, and the serial number of that dump is 375.

ADD AUDITTRAIL

ADD AUDITTRAIL defines and names all audit trails. This command also defines which audit processes or disc processes will write to each audit trail and which audit-dump process will dump the audit trail. Attributes defined by this command are:

- master audit trails (for NonStop systems with the DP2 disc process) and monitor audit trails (for all other systems)
- data audit trails
  —audit-process audit trails (for NonStop 1+ systems)
  —disc-process audit trails (for NonStop systems with the DP1 disc process)
  —optional auxiliary audit trails (for NonStop systems with the DP2 disc process)
- the generic file-set identifier for the sequence of files in the audit trail
- whether the audit trail is to be dumped to tape periodically and, if so, which audit-dump process is to dump it
- the names or logical device numbers of the audit processes (for NonStop 1+ systems) or disc processes (for NonStop systems) that will write to the audit trails defined by this command
- the primary and secondary extent size for the audit-trail files
- how the audit trails are to be used
- the maximum number of audit-trail files that a single active transaction can span before becoming eligible for automatic abortion
- the maximum number of files that can be maintained concurrently on disc for the audit-trail sequence before the system suspends processing of new transactions. On a NonStop 1+ system only, reaching this limit also suspends creation of new files in the sequence.

The ADD AUDITTRAIL command syntax differs slightly between NonStop 1+ systems and NonStop systems. For the latter, the syntax also differs between those with DP1 disc processes and those with DP2 disc processes. The syntax is:

```
ADD AUDITTRAIL generic-file-set

[ , AUDITDUMP { AUTOMATIC [ process ] }                              ]
[             { NONE                  }                              ]
[ , EXT{ primary-extent                                   }          ]
[      { ( primary-extent [ , secondary-extent ] )        }          ]  ...
[ , MAXFILES integer                                                 ]
[ , MINFILES integer                                                 ]
[         { MONITOR                                            }     ]
[         {                  { process                        }}     ]
[ , USAGE {{ AUDITPROCESS }  { ( process [ , process ]... )   }}     ]  (I)
[         {{ DISCPROCESS  }  { *                              }}     ]  (II/DP1)
[         {         [ process                       ]           }    ]
[         { MASTER  [ ( process [ , process ]... )  ]           }    ]
[ , USAGE {         [ *                             ] ]         }    ]  (II/DP2)
[         {            { process                       }        }    ]
[         { AUXILIARY  { ( process [ , process ]... )  }        }    ]
[         {            { *                             }        }    ]
                                                                      →
```

where

`generic-file-set`

identifies the sequence of files that form an audit trail. The identifier is defined in the "Generic File-Set Identifiers" subsection near the beginning of this section. No "wild-card" asterisks are allowed with this command.

`AUDITDUMP AUTOMATIC [ process ]`

specifies that the indicated audit-dump process will automatically dump this audit trail's files to tape as they fill up. The *process* identifier is a standard GUARDIAN process identifier. If you do not name an audit-dump process, TMF selects one. If no audit-dump process has been defined (by the ADD AUDITDUMP command) when TMF is started, TMF defines and names one, using any defaults previously established by the SET AUDITDUMP command. AUDITDUMP AUTOMATIC is the default setting.

After being dumped, each audit-trail file will be purged from the disc as soon as all transactions that began before the file was filled have either ended or aborted.

If any audit trail is configured for automatic dumping, then START TMF will fail if (1) you have also used ALTER TMF, RECOVERY OFF or (2) you have not designated an operator process. Both of these factors are discussed with the ALTER TMF command.

`AUDITDUMP NONE`

specifies that this audit trail's files will not be dumped to tape, but each will nevertheless be purged from the disc as soon as all transactions that began before the file was closed have either ended or aborted. AUDITDUMP AUTOMATIC is the default. If you use AUDITDUMP NONE on any audit trail, then the rollforward process cannot function for the corresponding audited files. If you use AUDITDUMP NONE on the monitor audit trail, then you should also use the ALTER TMF command's RECOVERY OFF option because the rollforward process will not be able to recover any file.

`EXT`

sets the extent size for the files in the sequence.

`primary-extent`

is the size of the primary extent in 2048-byte units (pages). The default is 50. On a NonStop system only, every extent must be at least four pages long.

`secondary-extent`

is the size of the secondary extent in 2048-byte units (pages). The default is the size of the primary extent.

`MAXFILES integer`

On a NonStop 1+ system only, MAXFILES sets the maximum number of files that can be maintained concurrently on disc for the audit-trail sequence. On any Tandem system, when

→

the number of existing files in an audit trail has grown to MAXFILES, a console message is printed and new transaction processing (BEGINTRANSACTION) is suspended. For example, if *integer* is set to 5 and audit-trail files AA003501 through AA003504 are all on the disc, suspension will occur when AA003505 is opened. This suspension will cease when AA003501, at least, has been purged from the disc.

The range of values for *integer* is 3 to 255. The default is either 1 plus the value of *integer* specified for MINFILES, or 4, whichever is greater. The largest value of MAXFILES for any one audit trail, multiplied by the number of audit trails, must be no greater than 1024.

`MINFILES integer`

sets the number of audit-trail files that a single active transaction can span (see the considerations below). Abortion may be initiated automatically on any transaction that lasts long enough to have before- and after-images spanning more than *integer* files. The range of values for *integer* is 2 to 254. The default is 2.

`USAGE AUDITPROCESS` or `USAGE DISCPROCESS` (not for use with DP2)

specifies one or more audit processes or disc processes that are to write to the audit trail defined by this command. In doing so, it declares that this is to be an audit-process or disc-process audit trail, not a monitor audit trail. USAGE AUDITPROCESS is for use with NonStop 1+ systems only. USAGE DISCPROCESS is only for use with NonStop systems with DP1 disc processes.

`process`

identifies an audit process or disc process. If this identifier is in network form, its system component must indicate the system on which this copy of TMFCOM is running. The *process* identifier can be either the logical device number (in *$integer* format) or the name of an audit process or disc process. If you use the asterisk (*), then any audit processes or disc processes that have not been assigned (by ADD AUDITTRAIL) will write to the audit trail defined in this command.

`USAGE MONITOR` (not for use with DP2)

declares that the audit trail defined by this command is to be the monitor audit trail, not an audit-process audit trail or a disc-process audit trail. USAGE MONITOR is only for use with NonStop 1+ systems and with NonStop systems with DP1 disc processes.

`USAGE MASTER` (DP2 only)

declares that the audit trail defined by this command is to be the master audit trail in a NonStop system with DP2 disc processes. This audit trail will contain all of the TMF system's monitor information (records of what actions TMF has performed). It can also contain audit images (records of audited changes to the data base).

Each TMF system must have exactly one master audit trail and zero or more auxiliary audit trails. If one or more disc process names are specified with the USAGE MASTER option, those processes will write their before- and after-images to the audit trail defined by this command.

USAGE MASTER is only for use with NonStop systems with DP2 disc processes.

→

`USAGE AUXILIARY` (DP2 only)

specifies one or more DP2 disc processes that are to write to the auxiliary audit trail defined by this command. Although each TMF system must have a master audit trail, it need not have any auxiliary audit trails. USAGE AUXILIARY is only for use with NonStop systems with DP2 disc processes.

Considerations

Before TMF auditing can start on a NonStop 1+ system, each audit process configured during system generation must also be configured (by ADD AUDITTRAIL) to write to an audit trail. On a NonStop system, each disc process must be configured to write to an audit trail, regardless of whether any files on its disc will be audited. Using the "wild-card" asterisk (USAGE ... *) in the process list of your last ADD AUDITTRAIL command will satisfy this requirement.

On a NonStop (not NonStop 1+) system, another reason to use the "wild-card" asterisk is so you can add a new volume to the system without reinitializing and reconfiguring TMF. When TMF is started, TMFCOM automatically adds any new volumes to the audit trail that was configured with the asterisk.

A particular *generic file name* can be used to define only one audit-trail sequence. Subsequent attempts to define an audit trail with the same name will be rejected with an "audit trail already defined" message.

Any given audit process or disc process can write to only one audit trail. If the same process is named in a second ADD AUDITTRAIL command, it will be rejected with an "audit trail already defined" message.

The audit-trail files that a completed transaction *spans* are the sequence beginning with the first file holding a before- or after-image for that transaction and ending with the last file holding an image for that transaction. For instance, if the first image for a given transaction is on audit-trail file $AUDIT.AP.AA000045 and the last is on $AUDIT.AP.AA000047, that transaction spans three files—regardless of whether the file in between, $AUDIT.AP.AA000046, contains any images for that transaction. If the transaction is still active (not completed), it spans all audit-trail files from its first before- or after-image through the file currently open for that transaction. This concept of spanning is what the MINFILES parameter regulates.

Examples

The first example is for a NonStop system with DP2 disc processes. Assuming that none of the omitted options have had values set by SET AUDITTRAIL, the command sequence

```
ADD AUDITTRAIL $AUDIT.MASTER.MA, USAGE MASTER ($ACCT, $SALES), EXT (70,70)
ADD AUDITTRAIL $AUDIT.AUX.AA, USAGE AUXILIARY *, EXT (70,70)
```

configures

- a master audit trail with a generic file name of $AUDIT.MASTER.MA, to contain both monitor information (for the whole system) and auditing images for disc volumes $ACCT and $SALES
- an auxiliary audit trail with a generic file name of $AUDIT.AUX.AA, to contain auditing images for all disc volumes other than $ACCT and $SALES

- a total file size (for each file in both audit-trail sequences) of 2240 kilobytes (16 extents, each having 70 pages of 2048 bytes each)
- a MINFILES value of 2 and a MAXFILES value of 4 (the defaults)
- a TMF-generated assignment to an audit-dump process (by default).

The remaining examples are for a NonStop system with DP1 disc processes, but they apply equally well (1) to a NonStop 1+ system if you substitute AUDITPROCESS for DISCPROCESS or (2) to a NonStop system with DP2 disc processes if you substitute MASTER for MONITOR and AUXILIARY for DISCPROCESS.

Assuming that none of the omitted options have had values set by SET AUDITTRAIL, the command

```
ADD AUDITTRAIL $AUDIT.AP.AA,USAGE MONITOR,EXT (10,10)
```

configures

- a monitor audit trail with a generic file name of $AUDIT.AP.AA
- primary and secondary extents of 20 kilobytes (10 pages of 2048 bytes each) for each file in the audit-trail sequence
- a MINFILES value of 2 and a MAXFILES value of 4 (the defaults)
- a TMF-generated assignment to an audit-dump process.

The command

```
ADD AUDITTRAIL $AUDIT.AP.AA, AUDITDUMP AUTOMATIC $ADMPO, &
  USAGE DISCPROCESS ($VOLO,$VOL1), EXT 20, MINFILES 6
```

configures

- a disc-process audit trail whose generic file name is $AUDIT.AP.AA
- assignment to audit-dump process $ADMP0 for automatic dumping of the audit-trail file when it becomes full
- disc processes $VOL0 and $VOL1 to write to audit trail $AUDIT.AP.AA
- primary and secondary extent sizes of 40 kilobytes (20 pages of 2048 bytes each)
- a MINFILES value of 6 and, by default, a MAXFILES value of 7.

The command sequence

```
ADD AUDITTRAIL $AUDIT.AP.AA, USAGE DISCPROCESS $VOL3
ADD AUDITTRAIL $AUDIT.AP.AB, USAGE DISCPROCESS *
```

configures two disc-process audit trails with generic file identifiers of $AUDIT. AP.AA and $AUDIT. AP.AB, respectively. Disc process $VOL3 will write to $AUDIT.AP.AA and all other disc processes will write to $AUDIT.AP.AB.

ADD DUMPS

The ADD DUMPS command inserts new dump references into the catalog. Its only use should be for manual recovery of catalog data if:

- the catalog portion (only) of the TMF subvolume is lost or otherwise logically damaged, or
- dump information has been deleted by a DELETE DUMPS command.

The ADD DUMPS command syntax is:

```
ADD DUMPS {file                    }
          {( file [ , file ]... )  }

  [ ,{BRIEF | DETAIL}                                                    ]
  [            {DISC pack                                             }  ]
  [ , MEDIUM{TAPE {reel [:part:copy]                              }}  ] ...
  [            {     {(reel[:part:copy][,reel [:part:copy]]... ) }}  ]
  [ , SERIAL number                                                      ]
  [ , TYPE {AUDITDUMP                          }                         ]
  [        {ONLINEDUMP ( audit , monitor )     }                         ]
```

where

`file`

specifies a dump (of an audited file or audit-trail file) to be catalogued. If you specify two or more online-dump files, you must list them in the same order in which they appear on the tape or disc.

`BRIEF` or `DETAIL`

determines how much information is to be displayed. With BRIEF, no information is displayed when dumps are added. With DETAIL, TMF displays the dump serial number to acknowledge the creation of each of four kinds of catalog records (see the example below). The default is BRIEF.

`MEDIUM DISC`

indicates that the specified dump is on a disc and identifies which disc pack it is on. When cataloguing a disc pack, the TYPE must be ONLINEDUMP.

`pack`

is the six-character (or shorter) alphanumeric identifier of a physical, removable disc pack. This identifier is encoded on the disc pack when the PUP command REMOVEAUDITED creates an online dump; it should also be written visibly on the disc pack's cover. Note that the *pack* identifier names the physical disc, not the data recorded on it or the volume name of the disc drive.

→

`MEDIUM TAPE`

indicates that the specified dump is on tape and identifies which tape reel (or reels) it is on.

`reel`

is a reel identifier, as defined in the "Tape Terminology" subsection in Section 1 of this manual.

`part`

specifies which part of a multireel dump is represented on this tape. Thus if each copy of the dump spans three reels, *part* would be 1, 2, or 3. The default for the first-specified reel is 1, for the second is 2, etc.

`copy`

specifies the number of the dump copy represented on this tape. The default *copy* is 1. The maximum *copy* is 99.

`SERIAL number`

specifies the dump serial number for the dump to be catalogued.

`TYPE`

specifies the type of files being catalogued. When cataloguing a disc pack, the TYPE must be ONLINEDUMP.

`audit`

is an integer specifying the audit subsystem audit-trail sequence number.

`monitor`

is an integer specifying the monitor audit-trail sequence number.

Considerations

The ADD DUMPS command is insufficient to bring the state of the catalog forward in time, because this command changes only the catalog data base, without updating critical state information in other system files. ADD DUMPS is useful, however, if the catalog portion (only) of the TMF subvolume is lost or otherwise logically damaged, or if dump information has been deleted by a DELETE DUMPS command.

The *file* identifiers cannot include any "wild-card" asterisks (*). When using the TYPE AUDITDUMP option, you can specify only one *file* identifier per ADD DUMPS command. With TYPE ONLINEDUMP, a list of files is allowed.

The SERIAL and TYPE options are required.

If the SERIAL number is not that of a dump already in the catalog, the MEDIUM option is required. If the catalog already contains data under the specified SERIAL number, then MEDIUM should not be used; in this case, TMF uses the already-catalogued tape or disc information, appending it to the new catalog entry.

The serial numbers issued during normal cataloguing of dumps are never less than 64. If, in ADD DUMPS, you specify a serial number less than 64, an error message will be returned.

See Section 3, "Media-Handling Responses," for valid responses to system prompts.

Examples

The command

```
ADD DUMPS ($DATA.ACCT.JAN13, $DATA.ACCT.JAN14), &
  MEDIUM TAPE (JER001:1:1, JER002:1:2, JER003:2:1, JER004:2:2), &
  SERIAL 286, TYPE ONLINEDUMP(493,68), DETAIL
```

creates catalog records saying that files $DATA.ACCT.JAN13 and $DATA.ACCT.JAN14 are both part of the online dump whose serial number is 286. The inserted records show:

- When the dump was made, the data audit trail was at sequence number 493 and the monitor audit trail was at sequence number 68.
- At least two copies were made of this dump.
- The dump spans two reels for each copy. Copy 1 is on the tape reels labelled JER001 and JER003 and copy 2 is on the tape reels labelled JER002 and JER004.

The DETAIL option causes this output to be displayed:

```
SERIAL 286, VOLUME RECORD ADDED
SERIAL 286, MEDIA RECORD ADDED: JER001
SERIAL 286, MEDIA RECORD ADDED: JER002
SERIAL 286, MEDIA RECORD ADDED: JER003
SERIAL 286, MEDIA RECORD ADDED: JER004
SERIAL 286, DUMP-MEDIA RECORD ADDED: JER001
SERIAL 286, DUMP-MEDIA RECORD ADDED: JER002
SERIAL 286, DUMP-MEDIA RECORD ADDED: JER003
SERIAL 286, DUMP-MEDIA RECORD ADDED: JER004
SERIAL 286, FILE RECORD ADDED: $DATA.ACCT.JAN13
SERIAL 286, FILE RECORD ADDED: $DATA.ACCT.JAN14
```

The first line of output merely acknowledges that information about dump 286 is being added to the catalog. Because the media directory must know about a tape before assigning that tape to a dump, the next four lines acknowledge those reels being entered into the media directory. The following four lines acknowledge that those four reels have been assigned. The last two lines acknowledge entering the two file identifiers into the dump directory.

Subsequently, the command

```
ADD DUMPS ($DATA.ACCT.JAN15, $DATA.ACCT.JAN16), &
  SERIAL 286, TYPE ONLINEDUMP(493,68), DETAIL
```

adds two more files to the existing dump reference having serial number 286.

ADD MEDIA

The ADD MEDIA command inserts new tape-reel and disc-pack references into the catalog's media directory and can, at the same time, rewrite tape labels. It is also useful if you have to manually rebuild a catalog under certain circumstances (see also ADD DUMPS). This command replaces the former ADD TAPES command, but TMFCOM still recognizes ADD TAPES and interprets it as ADD MEDIA with TYPE TAPE. The ADD MEDIA command syntax is:

```
            ( pack                     )
            ( ( pack [ , pack ]... )   )
ADD MEDIA   { reel                     }
            ( ( reel [ , reel ]... )   )

   [ , {BRIEF | DETAIL}                                    ]
   [ , DRIVE { device                                 }    ]
   [         { ( device [ , device ]... )             }    ]
   [ , LABEL {OFF | ON}                                    ]
   [ , OPERATOR {OFF | ON}                                 ]
   [ , RELABEL {OFF | ON}                                  ] ...
   [          ( ASSIGNED )                                 ]
   [ , STATUS { BAD      }                                 ]
   [          { RELEASED }                                 ]
   [          ( SCRATCH  )                                 ]
   [ , TYPE {DISC | TAPE}                                  ]
```

where

`pack`

is a physical, removable disc pack, identified by a six-character alphanumeric identifier. This identifier is encoded on the disc pack when the PUP command REMOVEAUDITED creates an online dump; it should also be written visibly on the disc pack's cover. Note that the *pack* identifier names the physical disc, not the data recorded on it or the volume name of the disc drive.

`reel`

is a reel identifier, as defined in the "Tape Terminology" subsection in Section 1 of this manual.

`BRIEF` or `DETAIL`

determines how much information is to be displayed. With BRIEF, TMF tells how many tapes or discs are added to the media directory. With DETAIL, it also displays each reel or pack identifier that is added. If you do not specify either BRIEF or DETAIL, TMF displays the brief report.

`DRIVE`

used with the LABEL option, specifies the tape drives on which the tapes are to be mounted.

`device`

identifies a tape drive. This is a standard GUARDIAN device name, such as $TAPE or \KEOKUK.$TAPE2.

→

`LABEL OFF`

disables rewriting of tape labels as "scratch" (described below). This is the default setting.

`LABEL ON`

relabels as scratch each tape mounted on a drive specified in the DRIVE option. If RELABEL ON is not also specified, then each tape must be either unlabelled, or labelled with a reel identifier given in this command. After each label is written, the tape is rewound and unloaded. When a tape is relabelled, it automatically becomes a scratch tape; therefore, you cannot use the STATUS option in the same command as LABEL ON. The default setting is LABEL OFF.

`OPERATOR OFF`

when used with the LABEL ON option, specifies that any requests for operator intervention will go to the terminal from which the command was given. If no TMF operator has been designated, this is the default setting. Otherwise, the default setting is OPERATOR ON.

`OPERATOR ON`

when used with the LABEL ON option, specifies that any requests for operator intervention will go to the TMF operator previously designated by the ALTER TMF command. If a TMF operator has been designated, this is the default setting. Otherwise, the default setting is OPERATOR OFF.

`RELABEL OFF`

used with LABEL ON, causes TMF to read the label (if any) on the mounted tape and refuse to rewrite the reel identifier if it differs from the one specified in this command. The default setting of this option is RELABEL OFF, unless it has been changed by the RELABEL option of the ALTER CATALOG configuration command.

`RELABEL ON`

used with LABEL ON, allows TMF to rewrite any existing tape label, even if it has a different reel identifier. When using RELABEL ON, you must specify exactly one tape reel and no more than one drive. The default setting of this option is RELABEL OFF, unless it has been changed by the RELABEL option of the ALTER CATALOG configuration command.

`STATUS ASSIGNED`

tells the catalog that each specified tape or disc is assigned to a dump. For each such tape or disc, TMF tries to find the dump serial number in the dump directory and include it in the media directory entry; if TMF cannot find any reference to the tape or disc, it flags the media directory to show that the tape's or disc's contents are unknown. This is the only STATUS option that can be used with discs.

`STATUS BAD`

tells the catalog that each specified tape is defective and unavailable for use. This option cannot be used with discs.

→

`STATUS RELEASED`

tells the catalog that each specified tape is released from assignment but not yet available for reassignment. This option cannot be used with discs.

`STATUS SCRATCH`

tells the catalog that each specified tape is available for reassignment. This option cannot be used with discs.

`TYPE`

specifies whether you are cataloguing a disc pack or one or more tape reels. The default is TYPE TAPE.

Considerations

When adding a disc entry, the only effective options are BRIEF, DETAIL, and STATUS ASSIGNED. The other options apply only to tapes.

If the catalog already contains an entry for a tape reel or disc drive specified in this command, an error message is displayed. ADD MEDIA does not affect previously existing catalog entries.

To reuse "released" tapes, you must first use the ALTER MEDIA command to scratch them in the media directory and, if you like, overwrite their labels.

When you use the RELABEL ON option, you must specify no more than one drive. You can, however, specify a list of tape reels to be relabelled sequentially on that drive.

The DRIVE option is not required with LABEL ON, but it is ignored if LABEL ON does not accompany it.

See Section 3, "Media-Handling Responses," for valid responses to system prompts.

Examples

The command

```
ADD MEDIA (JER123, ABC456), TYPE TAPE, STATUS ASSIGNED
```

searches the catalog's media directory for references to tape reels JER123 and ABC456 and, if none are found, creates such references. It then displays

```
2 TAPES ADDED.
```

A subsequent INFO MEDIA command would show these tapes' status as ASSIGNED (CONTENTS UNKNOWN).

The command

```
ADD MEDIA JER456, LABEL ON, RELABEL ON, DRIVE $TAPE1, OPERATOR OFF, DETAIL
```

- if no tape reel is loaded on drive $TAPE1, prompts you to mount one there
- labels the reel on $TAPE1 as a scratch tape having reel identifier JER456, regardless of any label already on the tape.

- adds or replaces a record in the catalog's media directory for that tape
- directs output to the terminal from which TMFCOM was started, instead of the operator terminal
- produces this report:

```
TAPE JER369 ADDED.
1 TAPE ADDED.
```

ALTER AUDITDUMP

ALTER AUDITDUMP changes attributes for the audit-dump processes which the transaction monitor process (TMP) will create when TMF is started. These audit-dump processes control the automatic dumping of audit-trail files. ALTER AUDITDUMP affects the same attributes as ADD AUDITDUMP. The ALTER AUDITDUMP command syntax is:

```
                               ┌ , COPIES quantity                                   ┐
                               │          ┌ cpu1            ┐                        │
                               │ , CPUS  <  (cpu1)           >                       │
                               │          │ [cpu1]:cpu2     │                        │
                               │          └ ([cpu1]:cpu2)   ┘                        │
                               │ , DRIVE ┌ device                        ┐           │
ALTER AUDITDUMP [ process ]    │         └ ( device [ , device ]... )    ┘           │ ...
                               │ , OUT file                                          │
                               │ , PRI integer                                       │
                               │          ┌ CPUS  ┐                                  │
                               │ , RESET <  DRIVE  >                                 │
                               │          │ OUT   │                                  │
                               └          └ PRI   ┘                                  ┘
```

where

`process`

names an audit-dump process to be changed on the local system. This name cannot include a subdevice, subvolume, or file-identifier component. If you do not name the process, TMF assumes that only one process exists, and alters that process.

`COPIES quantity`

specifies the number of copies to be made of each dump. The *quantity* can be from 1 to 99, inclusive. Multiple copies are written simultaneously and, therefore, require multiple tape drives (see the DRIVE option below).

`CPUS`

specifies the primary and backup processors to be used for the audit-dump process.

`cpu1`

is the number of the primary processor. Its range is 0 to 15.

`cpu2`

is the number of the backup processor. Its range is 0 to 15 and it must be different from *cpu1*.

`DRIVE`

specifies one or more tape drives to which the audit-dump process can dump audit-trail files. If other drives have been specified previously, any newly specified drives are added to the list. Respecifying a previously named drive has no effect.

These assignments are not exclusive, so you can assign the same list of drives to each process if you want.

→

`device`

identifies a tape drive. This is a standard GUARDIAN device name, such as $TAPE or \KEOKUK.$TAPE2.

`OUT file`

specifies where the audit-dump reports should be written. If *file* specifies an existing disc file, the reports will be appended to that file. If it specifies a disc file that does not exist, an EDIT-type file will be created. If it specifies a process, such as the spooler, that process must exist when TMF is started and must remain available as long as TMF is running; writing to a nonexistent OUT process may cause TMF to crash.

Until OUT is specified with an ADD AUDITDUMP, ALTER AUDITDUMP, or CONTROL AUDITDUMP command, no reports are produced.

`PRI integer`

changes the priority to be given to the audit-dump process. The range of valid priorities is 1 to 199.

`RESET`

cancels any previous setting for the specified attribute (PRI, CPUS, DRIVE, or OUT).

Considerations

While TMF is running (that is, after START TMF), you cannot use the ALTER AUDITDUMP command; you can, however, use the TMFCOM command CONTROL AUDITDUMP to change some attributes of an audit-dump process while TMF is running.

Examples

The command

```
ALTER AUDITDUMP $AUD1, PRI 188, CPUS 9:12, COPIES 2
```

- changes the execution priority of audit-dump process $AUD1 to 188
- selects processors 9 and 12 to be the primary and backup processors, respectively
- provides for two copies to be made of each dump (requiring two tape drives).

The command

```
ALTER AUDITDUMP $AUD1, DRIVE $TAPE1, RESET CPUS
```

- selects $TAPE1 as the tape drive to receive all audit-trail files dumped by process $AUD1
- cancels any previous specification of default primary and backup processors.

ALTER AUDITTRAIL

The ALTER AUDITTRAIL command is valid on NonStop systems only, not on NonStop 1+ systems.

ALTER AUDITTRAIL changes certain attributes of the specified audit trail, after TMF has been started and stopped at least once. The ALTER AUDITTRAIL command syntax is:

```
ALTER AUDITTRAIL generic-file-set                          (II only)

┌, EXT {primary-extent                               }┐
│      {( primary-extent [ , secondary-extent ] )    }│...
│, MAXFILES integer                                   │
└, MINFILES integer                                   ┘
```

where

`generic-file-set`

names a previously configured audit trail to be changed on the local system. The identifier is defined in the "Generic File-Set Identifiers" subsection near the beginning of this section. No "wild-card" asterisks are allowed with this command.

`EXT`

changes the configured size, in 2048-byte units (pages), of the primary or secondary extents for each file in the audit-trail sequence. On a NonStop system only, every extent must be at least four pages long. The default size of the secondary extent is the size of the primary extent.

Under the DP1 disc process, ALTER AUDITTRAIL cannot decrease either extent size. Under the DP2 disc process, it can decrease or increase either extent size.

`MAXFILES integer`

changes the audit trail's MAXFILES limit, as defined for the ADD AUDIT TRAILS command. The range of values for *integer* is 3 to 255. The largest value of MAXFILES for any one audit trail, multiplied by the number of audit trails, must be no greater than 1024.

`MINFILES integer`

changes the number of audit-trail files that a single active transaction can span (see the ADD AUDITTRAIL "Considerations" subsection). Abortion may be initiated automatically on any transaction that lasts long enough to have before- and after-images spanning more than *integer* files. The range of values for *integer* is 2 to 254.

Considerations

You can use the ALTER AUDITTRAIL command only after TMF has been started and stopped at least once. You cannot use this command between INITIALIZE TMF and START TMF or between START TMF and STOP TMF.

Examples

The command

```
ALTER AUDITTRAIL $AUDIT.AP.AA, EXT 20, MINFILES 6
```

changes the MINFILES limit for the $AUDIT.AP.AA audit-trail sequence and also respecifies the primary and secondary extent sizes of all files to be created in that sequence.

ALTER BACKOUT

ALTER BACKOUT sets the attributes of the backout process, specifically the name, the execution priority, and the processor where the backout process is to be run. The ALTER BACKOUT command syntax is:

```
              ┌                                        ┐
              │          ┌ cpu1                   ┐    │
              │  , CPUS  │ [ cpu1 ] : cpu2        │    │
              │          │ ( cpu1 )               │    │
ALTER BACKOUT │          └ ( [ cpu1 ] : cpu2 )    ┘    │ ...
              │  , NAME generic-process                │ (II only)
              │  , PRI integer                         │
              │  , RESET  { CPUS }                     │
              │           { PRI  }                     │
              └                                        ┘
```

where

CPUS

sets the primary and backup processors of the backout process.

cpu1

is the number of the primary processor. Its range is 0 to 15.

cpu2

is the number of the backup processor. Its range is 0 to 15 and it must be different from *cpu1*.

NAME generic-process (II only)

names the backout process. *generic-process* name must be a dollar sign ($) followed by an alphabetic letter, followed by two additional alphanumeric characters. TMF appends a two-digit number to complete the name. Currently, the number is always "01" but this may change in a future release. For example, specifying NAME $BAK would name the process $BAK01.

Although it is optional, you should always name the backout process. If this option is omitted, TMF names the process $Z*nnn*, where *nnn* is a three-digit number, and, in a NonStop system, it is possible that the system-chosen name can be identical to the system-chosen name of some other process.

PRI integer

specifies the priority of the backout process. The maximum possible value for *integer* is 199.

RESET

cancels any previous setting for the specified attribute (backout priority or processor designations).

Considerations

If any attribute for the backout process is not specified when TMF is started, TMF selects a value for it. After you have issued the START TMF command, you can use the STATUS BACKOUT command to display the configured attributes.

You cannot use this command when the START TMF command has been issued (that is, while TMF is running).

On a NonStop system only, the backout process is not a NonStop process pair. Therefore, the primary and secondary CPU designations are only starting points. If both CPUs fail, a console message is printed and the TMP tries to restart the process in the specified primary CPU. If that fails, it tries the specified secondary CPU, then CPU 0, CPU 1, ... until it finds a working CPU. (At least one CPU must be working, or the TMP could not be active.)

On a NonStop system, you should always name the backout process. Otherwise, it is possible that the system-chosen name can be identical to the system-chosen name of some other process.

The backout process starts when the START TMF command is issued and stops as the STOP TMF command completes.

Examples

The command

```
ALTER BACKOUT, PRI 194,CPUS 1:2
```

configures the execution priority of the backout process to 194 and the primary and backup processors as 1 and 2, respectively. It allows TMF to pick a $Z*nnn* name for the process; on a NonStop system, this can sometimes result in a name conflict with some other system-named process.

The command

```
ALTER BACKOUT,RESET PRI,CPUS 1:2,NAME $BAK
```

names the process $BAK01 (on a NonStop system only), configures the primary and backup processors as 1 and 2, respectively, and removes the current value for the execution priority. TMF will select a default value for the priority when the START TMF command is issued.

ALTER CATALOG

ALTER CATALOG changes attributes for the catalog. It cannot be used on a system while TMF is running (that is, after START TMF). The ALTER CATALOG command syntax is:

```
              ┌        ⎧ cpu1              ⎫ ┐
              │ , CPUS ⎨ (cpu1)            ⎬ │
              │        ⎪ [cpu1]:cpu2       ⎪ │
              │        ⎩ ([cpu1]:cpu2)     ⎭ │
              │ , MEDIALOG {OFF | ON}        │
              │ , NAME generic-process       │ (II only)
              │ , OUT file                   │
ALTER CATALOG │ , PRI integer                │ ...
              │ , RELABEL {OFF | ON}         │
              │ , RELEASED {OFF | ON}        │
              │         ⎧ CPUS         ⎫     │
              │ , RESET ⎨ OUT          ⎬     │
              │         ⎪ PRI          ⎪     │
              │         ⎩ RETAINDEPTH  ⎭     │
              └ , RETAINDEPTH limit          ┘
```

where

`CPUS`

specifies the primary and secondary processors to be used for the catalog process. If either processor is not specified, the TMP selects it from among the available processors.

`cpu1`

is the number of the primary processor. Its range is 0 to 15.

`cpu2`

is the number of the secondary processor. Its range is 0 to 15 and it must be different from *cpu1*.

`MEDIALOG OFF`

disables the listing of scratch tapes in the catalog's media directory. This destroys any such information currently stored in the media directory, but does not affect the cataloguing of valid dump tapes. The default setting is MEDIALOG ON.

`MEDIALOG ON`

enables the listing of scratch tapes in the catalog's media directory. This is the default setting.

`NAME generic-process` (II only)

names the backout process. The *generic-process* name must be a dollar sign ($) followed by an alphabetic letter, followed by two additional alphanumeric characters. TMF appends a

→

two-digit number to complete the name. Currently, the number is always "01" but this may change in a future release. For example, specifying NAME $CAT would name the process $CAT01.

Although it is optional, you should always name the catalog process. If this option is omitted, TMF names the process $Z*nnn*, where *nnn* is a three-digit number, and, in a NonStop system, it is possible that the system-chosen name can be identical to the system-chosen name of some other process.

`OUT file`

specifies a file to collect notices of tape status changes. When a tape is released or scratched, its new status will be written to the specified file or device. If OUT is not specified, or if RESET OUT is specified subsequently, no such log is maintained.

If *file* specifies an existing, entry-sequenced disc file, the notice will be appended to that file. If it specifies a disc file that does not exist, the subsequent START TMF command will fail. If it specifies a process, such as the spooler, that process must already exist when TMF is started and must remain available as long as TMF is running; writing to a nonexistent OUT process may cause TMF to crash.

`PRI integer`

specifies the priority to be given to the catalog process. The range of valid priorities is 1 to 199. If no priority is specified, the process assumes TMFCOM's priority.

`RELABEL OFF`

specifies that no ADD MEDIA or ALTER MEDIA command can change an existing, valid reel identifier unless that command specifically includes the RELABEL ON option. With RELABEL OFF, TMF will read the label on the mounted tape (if any) but will refuse to rewrite its reel identifier if it differs from the one specified in the ADD MEDIA or ALTER MEDIA command. RELABEL OFF is the default for ALTER CATALOG.

`RELABEL ON`

specifies that an ADD MEDIA or ALTER MEDIA command can always change an existing, valid reel identifier unless that command specifically includes the RELABEL OFF option. RELABEL OFF is the default for ALTER CATALOG.

`RELEASED OFF`

enables immediate designation of tapes as scratched when they have been marked for deletion (see the RELEASED ON option below). This is the default setting.

`RELEASED ON`

specifies that, when all dumps on a tape have been marked for deletion (see RETAINDEPTH option below), the tape will be marked as released rather than scratched. Released tapes must be explicitly scratched by the TMFCOM command ALTER MEDIA. This gives the operator the option of explicitly designating scratched tapes, rather than depending on the information in the catalog's media directory. The default setting is RELEASED OFF.

→

`RESET`

cancels any previous setting for the specified attribute (CPU, OUT, PRI, or RETAINDEPTH). RESET RETAINDEPTH disables automatic deletion of old online dumps, by disabling the RETAINDEPTH limit; note that this does *not* return *limit* to its default value of three generations.

`RETAINDEPTH limit`

specifies a limit of online-dump generations, of any single file, to be kept in the catalog. The valid range is 1 to 32767. When a new online dump raises the number of catalogued generations to more than *limit*, the oldest unneeded one is marked for deletion. When all files on a tape have been so marked, that tape is released or scratched (see the RELEASED option above). The default *limit* is three generations. Note, however, that the RESET RETAINDEPTH option does *not* return *limit* to its default value.

Considerations

You can use the RELABEL OFF and RELEASED ON options together to help enforce a requirement that scratch tapes be explicitly labelled as such, by the LABEL option of the ALTER MEDIA command.

The catalog process is a NonStop process pair. It starts when the START TMF command is issued and stops as the STOP TMF command completes; there is no other way to stop it.

Examples

The command

```
ALTER CATALOG, NAME $CAT, OUT $DATA.TAPES.LOG1, RETAINDEPTH 4, RELEASED ON, RELABEL OFF
```

- names the process $CAT01 (on a NonStop system only)
- specifies that tape status changes should be noted in disc file $DATA.TAPES.LOG1
- requires the catalog to keep no more than four generations of online dumps for each data file
- disables automatic scratching of released tapes
- specifies that tapes can be rewritten only if they have been given scratch status.

The command

```
ALTER CATALOG, MEDIALOG OFF, RESET OUT, RESET RETAINDEPTH
```

- allows TMF to pick a $Z*nnn* name for the process; on a NonStop system, this can sometimes result in a name conflict with some other system-named process.
- disables use of the media directory
- disables recording of tape status changes
- disables automatic deletion of outdated online dumps.

ALTER DUMPS

The ALTER DUMPS command changes the INVALID flag for one or more dump references in the catalog. On a NonStop system only, it can also change the RELEASED flag. The ALTER DUMPS command syntax is:

```
ALTER DUMPS { file-set                                }
            { ( file-set [ , file-set ]... )          }

  [ , {BRIEF | DETAIL}                                ]
  [ , INVALID {OFF | ON}                              ]
  [           { DISC pack                         }   ]  ...
  [ , MEDIUM  { TAPE { reel                     } }   ]
  [           {      { (reel [ , reel ]...)     } }   ]
  [ , RELEASED {OFF | ON}                             ]  (II only)
  [ , SERIAL number                                   ]
```

where

`file-set`

specifies one or more dumps (of audited files or audit trails) whose catalog references are to be changed. File-set identifiers are discussed in Appendix B. TMF displays a nonfatal error message for any explicitly specified dump that is not listed in the catalog.

`BRIEF` or `DETAIL`

determines how much information is to be displayed. With BRIEF, TMF tells how many dump references are altered in the media directory. With DETAIL, it also displays each file identifier whose reference is altered. If you do not specify either BRIEF or DETAIL, TMF displays the brief report.

`INVALID OFF`

flags the catalog entry for each specified file to indicate that the file is available for use by the rollforward process. Normally, an entry is flagged "valid" (available) when it is first made. When a file becomes unaudited or purged, the entries for it are flagged "invalid" automatically. The INVALID option permits manual control of that flag. If all files for a dump are marked invalid, then all catalog records for that dump are purged and its media are released or scratched.

`INVALID ON`

flags the catalog entry for each specified file to indicate that the file is unreadable and therefore not available for use by the rollforward process.

`MEDIUM`

limits the dump directory entries to be changed by this command. When you use the MEDIUM option, ALTER DUMPS affects only those files that are specified in the command and also reside on one of the specified *reels* or *pack*.

→

`DISC pack`

identifies the specific disc pack whose catalog entries are to be displayed. The *pack* identifier, comprising six or fewer alphanumeric characters, is encoded on the disc pack when the PUP command REMOVEAUDITED creates an online dump; it should also be written visibly on the disc pack's cover.

`TAPE`

identifies the tape reels of the dump entries to be changed.

`reel`

is a reel identifier, as defined in the "Tape Terminology" subsection in Section 1 of this manual.

`RELEASED OFF` (II only)

changes the specified dump directory entries to indicate that a dump can be used. If TMF has discarded an old dump and flagged it "released," and you still have the physical dump tape or disc, the RELEASED OFF option changes the catalog entry to allow use of the dump.

`RELEASED ON` (II only)

changes the specified dump directory entries to indicate that a dump cannot be used. RELEASED ON can be useful if a dump is new enough to remain in the catalog but the physical dump tape or disc has been damaged or removed. If the catalog shows only one valid dump in the catalog for a given serial number, and you release that dump, the entire dump serial number will be removed from the catalog.

`SERIAL number`

limits the dump directory entries to be changed by this command. When you use the SERIAL option, ALTER DUMPS affects only those files that are in a specified file set and also share the specified dump serial number. You can obtain these numbers by using the INFO DUMPS command or by reading the report from an audit dump or online dump.

Considerations

The MEDIUM and SERIAL options let you select the exact entries you want to change.

INVALID ON means the recorded file is unusable, perhaps because it has been partially overwritten, or for some other reason that does not preclude reuse of the tape or disc. If the tape or disc itself is bad and should not be reused, you should use the ALTER MEDIA command to set the BAD ON attribute in the media log. If all files on a tape or disc are flagged as invalid, TMF automatically frees it as a scratch or released tape or disc.

Example

The command

```
ALTER DUMPS $DATA.ACCT.*, MEDIUM TAPE JER123, INVALID ON, DETAIL
```

- scans the catalog entries of tape JER123 for any dump files from disc volume $DATA, subvolume ACCT
- flags those files' catalog entries to indicate that the files are unreadable
- produces this report:

```
SERIAL 569, FILE $DATA.ACCT.DAVID 1 DUMP ALTERED.
```

ALTER MEDIA

The ALTER MEDIA command changes tape-reel and disc-pack references in the catalog's media directory and can, at the same time, rewrite tape labels. This command replaces the former ALTER TAPES command, but TMFCOM still recognizes ALTER TAPES and interprets it as ALTER MEDIA with TYPE TAPE. The ALTER MEDIA command syntax is:

```
             ( pack                          )
             | ( pack [, pack ]... )         |
ALTER MEDIA  | reel                          |
             ( ( reel [, reel ]... )         )

  [ , {BRIEF | DETAIL}                             ]
  [ , DRIVE { device                          }    ]
  [         { ( device [ , device ]... )      }    ]
  [ , LABEL {OFF | ON}                             ]
  [ , OPERATOR {OFF | ON}                          ]
  [ , RELABEL {OFF | ON}                           ] ...
  [           ( ASSIGNED )                         ]
  [ , STATUS  | BAD      |                         ]
  [           | RELEASED |                         ]
  [           ( SCRATCH  )                         ]
  [ , TYPE {DISC | TAPE}                           ]
```

where

`pack`

is the six-character (or shorter) alphanumeric identifier of a physical, removable disc pack. This identifier is encoded on the disc pack when the PUP command REMOVEAUDITED creates an online dump; it should also be written visibly on the disc pack's cover. Note that the *pack* identifier names the physical disc, not the data recorded on it or the volume name of the disc drive.

`reel`

is a reel identifier, as defined in the "Tape Terminology" subsection in Section 1 of this manual. When you use the RELABEL ON option, you can specify only one reel with this command.

`BRIEF` or `DETAIL`

determines how much information is to be displayed. With BRIEF, TMF tells how many tape or disc references are altered in the media directory. With DETAIL, it also displays each reel or pack identifier whose reference is altered. If you do not specify either BRIEF or DETAIL, TMF displays the brief report.

`DRIVE`

used with the LABEL option, specifies the tape drives on which the tapes are to be mounted. The DRIVE option is not required with LABEL ON but it is ignored if LABEL ON does not accompany it. When you use the RELABEL ON option, the DRIVE option can specify no more than one drive.

→

`device`

identifies a tape drive. This is a standard GUARDIAN device name, such as $TAPE or \KEOKUK.$TAPE2.

`LABEL OFF`

disables rewriting of tape labels as "scratch" (described below). This is the default setting.

`LABEL ON`

relabels as scratch each tape mounted on a drive specified in the DRIVE option. If RELABEL ON is not also specified, then each tape must be either unlabelled, or labelled with a reel identifier given in this command. After each label is written, the tape is rewound and unloaded. When a tape is relabelled, it automatically becomes a scratch tape; therefore, you cannot use LABEL ON in the same command as STATUS. The default setting is LABEL OFF.

`OPERATOR OFF`

when used with the LABEL ON option, specifies that any requests for operator intervention will go to the terminal from which the command was given. If no TMF operator has been designated, OPERATOR OFF is the default setting. Otherwise, the default setting is OPERATOR ON.

`OPERATOR ON`

when used with the LABEL ON option, specifies that any requests for operator intervention will go to the TMF operator previously designated by the ALTER TMF command. If a TMF operator has been designated, OPERATOR ON is the default setting. Otherwise, the default setting is OPERATOR OFF.

`RELABEL OFF`

used with LABEL ON, causes TMF to read the label (if any) on the mounted tape and refuse to rewrite the reel identifier if it differs from the one specified in this command. The default setting of this option is RELABEL OFF, unless it has been changed by the RELABEL option of the ALTER CATALOG configuration command.

`RELABEL ON`

used with LABEL ON, allows TMF to rewrite any existing tape label, even if it has a different reel identifier. When using RELABEL ON, you must specify exactly one tape reel and no more than one drive. The default setting of this option is RELABEL OFF, unless it has been changed by the RELABEL option of the ALTER CATALOG configuration command.

`STATUS ASSIGNED`

tells the catalog that each specified tape or disc is assigned to a dump. For each tape or disc, TMF tries to find the dump serial number in the dump directory and include it in the media directory entry; if TMF cannot find any reference to the tape or disc, it flags the media directory to show that the tape's or disc's contents are unknown. This is the only STATUS option that can be used with discs.

→

STATUS BAD

tells the catalog that each specified tape is defective and unavailable for use. If ALTER CATALOG, RELEASED OFF is used, then for each tape, TMF deletes from the dump directory any entries involving that tape. This option cannot be used with discs.

STATUS RELEASED

tells the catalog that each specified tape is released from assignment but not yet available for reassignment. For each tape, TMF deletes from the dump directory any entries involving that tape. This option cannot be used with discs.

STATUS SCRATCH

tells the catalog that each specified tape is available for reassignment. For each tape, TMF deletes from the dump directory any entries involving that tape. This option cannot be used with discs.

TYPE

specifies whether you are changing a disc-pack entry or one or more tape-reel entries. The default is TYPE TAPE.

Considerations

If the catalog does not already contain a reference for a tape reel or disc pack specified in this command, no action is taken.

For a disc reference, the only effective options are BRIEF, DETAIL, and STATUS ASSIGNED. The other options apply only to tapes.

See Section 3, "Media-Handling Responses," for valid responses to system prompts.

Examples

The command

```
ALTER MEDIA (JER123, ABC456), STATUS ASSIGNED, BRIEF
```

- searches the catalog's dump directory for references to tape reels JER123 and ABC456
- changes the catalog's media directory record for each tape to reflect either (1) the dump serial number associated with that tape or (2) a flag indicating that the tape's contents are unknown
- produces this report:

```
2 TAPES ALTERED.
```

The command

```
ALTER MEDIA JER456, TYPE TAPE, LABEL ON, RELABEL ON, DRIVE $TAPE1
```

- if no tape reel is loaded on drive $TAPE1, prompts you to mount one
- labels the reel as a scratch tape having reel identifier JER456, regardless of any label already on the tape
- changes the catalog's media directory record for that tape
- produces this report:

```
1 TAPE ALTERED.
```

The command

```
ALTER MEDIA JER789, STATUS RELEASED, DETAIL
```

- changes the catalog's media directory record for tape reel JER789 to give the reel "released" status, but does not rewrite the label (if any) on the physical reel.
- produces this report:

```
TAPE JER369 ALTERED. 1 TAPE ALTERED.
```

ALTER TMF

ALTER TMF specifies an operator terminal or process for tape and disc requests and permits improvement of transaction time by disabling preparation for recovery. The ALTER TMF command syntax is:

```
ALTER TMF [ , OPERATOR operator       ]...
          [ , RECOVERY {OFF | ON}     ] (DP1 only)
```

where

`OPERATOR operator`

specifies a terminal or process to be the operator interaction unit; this is identified by a standard GUARDIAN file specification. If a process is specified, it should be specifically designed for this purpose; do not specify a system process. If any audit trail is configured for automatic dumping, and no operator has been configured, then START TMF will fail. Otherwise, if no operator is specified, messages are directed to TMFCOM's home terminal.

`RECOVERY OFF` (DP1 only)

disables the writing of monitor audit trail records, thereby improving transaction time but making recovery impossible. After TMF has been started, you cannot change the RECOVERY specification until INITIALIZE TMF has been executed. The default is RECOVERY ON.

When you use RECOVERY OFF, you should also configure your audit trails with AUDIT-DUMP NONE and not configure any audit-dump processes.

RECOVERY OFF is only for use with NonStop 1+ systems or NonStop systems with DP1 disc processes.

`RECOVERY ON`

enables the writing of monitor audit trail records; these records are necessary to the rollforward process. After TMF has been started, you cannot change the RECOVERY specification until INITIALIZE TMF has been executed. The default is RECOVERY ON.

The RECOVERY option is only for use with NonStop 1+ systems or NonStop systems with DP1 disc processes. RECOVERY is always ON for DP2 systems.

Considerations

You cannot use the ALTER TMF command while the TMF auditing subsystem is running.

Examples

The command

```
ALTER TMF, OPERATOR $OPTERM
```

specifies $OPTERM as the TMF operator.

The command

```
ALTER TMF, RECOVERY OFF
```

makes it impossible to perform recovery, but effects some improvement in transaction time by eliminating all I/O to the monitor audit trail.

CONTROL AUDITDUMP

CONTROL AUDITDUMP, like ALTER AUDITDUMP, changes attributes for audit-dump processes. Unlike ALTER AUDITDUMP, however, CONTROL AUDITDUMP is used while TMF is running (that is, after START TMF) and its effects are less permanent. The CONTROL AUDITDUMP command syntax is:

```
                              ┌, COPIES quantity                             ┐
                              │, DISABLE { copy-number }                     │
                              │          { *           }                     │
                              │, DRIVE { device                         }    │
                              │        { ( device [ , device ]... )     }    │
                              │, ENABLE { copy-number }                      │
                              │         { *           }                      │
                              │, OPERATOR operator                           │ (II)
CONTROL AUDITDUMP process     │, OUT file                                    │ ...
                              │, REEL { reel                  }              │
                              │       { ( reel [ , reel ]... )}              │
                              │, RELEASE { device                       }    │
                              │          { ( device [, device ]... )    }    │
                              └, WAIT {OFF | ON}                             ┘
```

where

`process`

names an audit-dump process, on the local system, which is to be changed.

`COPIES quantity`

specifies the number of copies to be made of each audit dump. The quantity can be 1 to 99, inclusive. Multiple copies are written simultaneously and, therefore, require multiple tape drives (see the DRIVE option below).

`DISABLE copy-number`

disables creation of the specified audit-dump copy. The copy number must be between 1 and the quantity currently configured for the audit-dump process. If no other copies remain enabled, TMF prompts you to reenable at least one copy (by another CONTROL AUDITDUMP command with the ENABLE option) so dumping can be performed.

You can also use the media-handling command DISABLE to abort an audit-dump copy, but that affects only the current dump. CONTROL AUDITDUMP affects all subsequent audit dumps until TMF is stopped.

`DISABLE *`

disables creation of any copies of the audit dump. This causes TMF to prompt you to reenable at least one copy (by another CONTROL AUDITDUMP command with the ENABLE option) so dumping can be performed.

→

`DRIVE`

specifies one or more tape drives to which the audit-dump process can dump audit-trail files. Even if a previously specified drive is not included in the new list, it remains known to the process. These assignments are not exclusive, so you can assign the same list of drives to each process if you want.

`device`

identifies a tape drive. This is a standard GUARDIAN device name, such as $TAPE or \KEOKUK.$TAPE2.

`ENABLE copy-number`

reenables the specified, disabled copy. If a dump is in progress, the audit-dump process does not start writing the newly enabled copy until the next dump begins.

`ENABLE *`

reenables all disabled copies that were configured when TMF was started. If a dump is in progress, the audit-dump process does not start writing the newly enabled copies until the next dump begins.

`OPERATOR operator` (II only)

specifies a terminal or process to become the operator interaction unit for the secified audit-dump process; if you specify a process, it should be specifically designed for this purpose; do not specify a system process. After you use this option, the next prompt from the audit-dump process will go to the newly specified operator process.

`OUT file`

specifies where the audit-dump reports should be written. If file specifies an existing disc file, the reports will be appended to that file. If it specifies a disc file that does not exist, an EDIT-type file will be created. If it specifies a process, such as the spooler, that process must already exist when TMF is started and must remain available as long as TMF is running; writing to a nonexistent OUT process may cause TMF to crash.

Until OUT is specified with an ADD AUDITDUMP, ALTER AUDITDUMP, or CONTROL AUDITDUMP command, no reports are produced.

`REEL`

specifies one or more physical tape reels on which dumps are to be written. When the audit-dump process needs a scratch reel, it selects the next reel in this list and tries to find that reel on one of the currently assigned tape drives. If it cannot find the reel, or if the reel is not catalogued as a scratch tape, TMF prompts the operator to either supply another reel or mount the missing reel. When this list is expended, TMF selects scratch tapes from the catalog.

`reel`

is a reel identifier, as defined in the "Tape Terminology" subsection in Section 1 of this manual.

→

```
RELEASE
```

releases all specified tape drives that are currently assigned (by a previously issued DRIVE option) to the audit-dump process. If such a drive is in the middle of a dump, TMF will display a console message and refuse to release the drive; another CONTROL AUDITDUMP command with a RELEASE option, given after the copy being written on that drive is disabled or completed, will release the drive.

```
WAIT OFF
```

awakens the specified audit-dump process from the dormant state initiated through either the media-handling command WAIT (see Section 3) or a CONTROL AUDITDUMP...WAIT ON command.

```
WAIT ON
```

puts the specified audit-dump process in a dormant state. You can also induce this state through the media-handling command WAIT. To reawaken the process, issue another CONTROL AUDITDUMP command, with the WAIT OFF option.

Considerations

CONTROL AUDITDUMP changes only the specified, currently active process, not the underlying configuration. Only the ADD AUDITDUMP, ALTER AUDITDUMP, and DELETE AUDITDUMP commands can change the configuration. CONTROL AUDITDUMP's changes remain in effect as long as the process remains (that is, until TMF is shut down).

New lists of drives temporarily replace previously configured drive lists. New *reel* lists or *drive* lists temporarily replace those specified in tape-mounting responses or in CONTROL AUDITDUMP commands.

No CONTROL AUDITDUMP changes can happen while WAIT ON is in effect. If WAIT is OFF and a dump is in progress, then REEL and DRIVE changes are delayed until the end of the current reel or dump, whichever comes first. Otherwise, all CONTROL AUDITDUMP changes take effect immediately.

If you use the DISABLE and RELEASE options together, DISABLE will be applied first. If you use the DRIVE and RELEASE options together, DRIVE will be applied first.

Examples

The command

```
CONTROL AUDITDUMP $AUD1, COPIES 3, DRIVE ($TAPE1, $TAPE2, $TAPE3),&
  OUT $DATA.TMF.DUMPRPT
```

causes audit-dump process $AUD1 to generate three copies of each dump; use tape drives $TAPE1, $TAPE2, and $TAPE3 for those dumps; and produce a report in disc file $DATA.TMF.DUMPRPT.

For each audit-dump, the report is similar to this:

```
AUDITDUMP - T9066B00 - (28JAN85) System \SAB 20-Feb-1985 19:47:18
[Dump serial number is 375.]
  File Name                Error
[Tape JER234 used for copy 1 of reel 1 on drive $TAPE.]
$AUDIT.AP.AA000132
[Total files dumped = 1.]
```

This example shows that audit-trail file $AUDIT.AP.AA000132 was dumped to the tape labeled JER234, using drive $TAPE, at 7:47 p.m. on April 20, and the serial number of that dump is 375.

The command

```
CONTROL AUDITDUMP $AUD2, RELEASE $TAPE2, DISABLE 2, &
  DRIVE ($TAPE1, $TAPE2, $TAPE3, $TAPE4)
```

causes audit-dump process $AUD2 first to disable dump copy 2 and add $TAPE4 to the list of enabled tape drives, then to release tape drive $TAPE2.

If $TAPE2 were not specified in the DRIVE option, it would not have to be released; use of the DRIVE option replaces the previous tape drive list.

The command

```
CONTROL AUDITDUMP $AUD3, REEL (ABC123, ABC456, ABC789), &
  DRIVE ($TAPE5, $TAPE6, $TAPE7)
```

causes audit-dump process $AUD3 to write its dumps automatically to scratch tape ABC123, then ABC456, then ABC789, so long as the desired tapes are mounted on one of the specified tape drives when needed. If no scratch tapes are mounted on one of those drives, TMF prompts the operator to mount one.

DELETE AUDITDUMP

DELETE AUDITDUMP removes audit-dump process descriptions previously specified by ADD AUDITDUMP or ALTER AUDITDUMP commands. The DELETE AUDITDUMP command syntax is:

```
DELETE AUDITDUMP { process }  [ ! ]
                 { *       }
```

where

process

names a process to be deleted from the local system. If you use the asterisk (*), all audit-dump processes will be deleted.

!

indicates that the operation should be performed without waiting for confirmation. If this is omitted, TMF asks you for confirmation; a response of YES or Y allows the operation to proceed, but a response of NO, N, or STOP aborts the operation. With any other response, TMF displays an error message and reprompts you.

Considerations

You cannot use the DELETE AUDITDUMP command while the TMF auditing subsystem is running. Also, you cannot delete an audit-dump process after an audit trail has been linked to it (by the AUDITDUMP option of an ADD AUDITTRAIL command).

Example

The command

```
DELETE AUDITDUMP $AUD1
```

removes the specifications for audit-dump process $AUD1.

DELETE DUMPS

The DELETE DUMPS command removes one or more dump references from the catalog. The DELETE DUMPS command syntax is:

```
DELETE DUMPS { file-set                           }  [ , {BRIEF | DETAIL}    ]
             { ( file-set [ , file-set ]... )     }  [ , RETAINDEPTH limit   ]...
                                                     [ , SERIAL number       ]
```

where

`file-set`

specifies one or more dumps (of audited files or audit trails) whose catalog references are to be deleted. File-set identifiers are discussed in Appendix B. TMF displays a nonfatal error message for any explicitly specified dump that is not listed in the catalog.

`BRIEF or DETAIL`

determines how much information is to be displayed. If you do not specify either BRIEF or DETAIL, TMF displays the brief report.

`RETAINDEPTH limit`

specifies a limit of online-dump generations, of any single file, to be kept in the catalog. The valid range is 1 to 32767. This option limits the deletable dumps to those older than the *limit* newest ones.

`SERIAL number`

limits the dump directory entries to be changed by this command. When you use the SERIAL option, TMF deletes only those files that are specified in the command and also have the specified dump serial number. You can obtain these numbers by using the INFO DUMPS command or by reading the report from an audit dump or online dump.

Considerations

If several files were dumped together (with a common serial number), and you only specify some of them in the DELETE DUMPS command, only the specified files are deleted from the dump directory; the other files with that serial number remain catalogued.

If all dumps on a tape are deleted from the catalog, that tape will be scratched or released in the catalog. If all dumps on a disc are deleted, that disc will be removed from the catalog.

Example

The command

```
DELETE DUMPS DATA.ACCT.*, SERIAL 276, RETAINDEPTH 4, DETAIL
```

deletes from the dump directory any record of any file dump meeting *all* of these conditions:

- the dump is of a file in subvolume $DATA.ACCT.
- the dump has serial number 276.
- the directory shows at least four newer dumps of this file.

It also displays a report similar to this:

```
SERIAL 381, FILE $DATA.ACCT.ARTHUR DELETED.
SERIAL 381, FILE $DATA.ACCT.FORD DELETED.
SERIAL 381, FILE $DATA.ACCT.ZAPHOD DELETED.
3 DUMPS DELETED.
```

If the DETAIL option is omitted, only the

```
3 DUMPS DELETED
```

line is to be displayed.

DELETE MEDIA

The DELETE MEDIA command removes tape-reel and disc-pack references from the catalog's media directory and also removes dump-directory records of files on those tapes or discs. This command replaces the former DELETE TAPES command, but TMFCOM still recognizes DELETE TAPES and interprets it as DELETE MEDIA with TYPE TAPE. The DELETE MEDIA command syntax is:

```
              { pack                              }
DELETE MEDIA  { reel                              }  [ , {BRIEF | DETAIL}  ]...
              { ( { pack } [ , pack ] ... )       }  [ , TYPE {DISC | TAPE} ]
              {   { reel } [ , reel ]             }
```

where

`pack`

is the six-character (or shorter) alphanumeric identifier of a physical, removable disc pack. This identifier is encoded on the disc pack when the PUP command REMOVEAUDITED creates an online dump; it should also be written visibly on the disc pack's cover. Note that the *pack* identifier names the physical disc, not the data recorded on it or the volume name of the disc drive.

`reel`

is a reel identifier, as defined in the "Tape Terminology" subsection in Section 1 of this manual.

`BRIEF or DETAIL`

determines how much information is to be displayed. With BRIEF, TMF tells how many tape or disc references are deleted from the media directory. With DETAIL, it also displays each reel or pack identifier whose reference is deleted. If you do not specify either BRIEF or DETAIL, TMF displays the brief report.

`TYPE`

specifies whether you are deleting disc packs or tape reels.

Example

The command

```
DELETE MEDIA (ABC678, JER123), DETAIL
```

deletes from the media and dump directories any record of any file on tape reel ABC678 or JER123. It also produces this report:

```
TAPE ABC678 DELETED.
TAPE JER123 DELETED.
1 TAPE DELETED.
```

DELETE TRANSACTION

The DELETE TRANSACTION command is valid on NonStop 1+ systems only, not on NonStop systems.

DELETE TRANSACTION is an emergency measure which releases any locks held for the specified transaction, regardless of its state of completion. The DELETE TRANSACTION command syntax is:

```
DELETE TRANSACTION transaction-identifier [ ! ]                    (I only)
```

where

`transaction-identifier`

identifies the transaction to be deleted. The identifier is defined in the "Transaction Identifiers" subsection near the beginning of this section. No "wild-card" asterisks are allowed with this command.

`!`

indicates that the operation should be performed without waiting for confirmation. If this is omitted, TMF asks you for confirmation; a response of YES or Y allows the operation to proceed, but a response of NO, N, or STOP aborts the operation. With any other response, TMF displays an error message and reprompts you.

Considerations

Beware that this command is an emergency measure only, and its use may allow a data base inconsistency. After using DELETE TRANSACTION, you should trace the troublesome transaction and take whatever steps are necessary to correct the data affected by it, since these data may now be inconsistent.

Before using this command, use the STATUS TRANSACTION command to determine the status of the transaction on its home node. For all transactions that are currently active on system \A, the TMFCOM command

```
STATUS TRANSACTION *.*
```

issued on system \A displays the disposition of those transactions on system \A, regardless of where each transaction originated. If a transaction is "hung" on your system and network communication with its home system is interrupted, you should contact someone at that system (by telephone or other means) and run STATUS TRANSACTION there, to determine whether you should end or abort the "hung" transaction.

Before using DELETE TRANSACTION, you should try to terminate the transaction with END TRANSACTION or ABORT TRANSACTION.

Example

Consider a transaction being backed out when a failure has occurred in the disc drive containing the audit trail's mirror volume, and the backout process cannot read the needed before-image for some reason. In this situation, the backout attempt fails and the transaction monitor process (TMP) does not release the locks that the transaction holds, so you may want to intervene and release the locks. The command

```
   DELETE TRANSACTION \JUNEAU(0).4.8899977
```

releases any locks held for the transaction whose transaction identifier is \JUNEAU(0).4.8899977; that is, a transaction that originated in CPU 4 of system \JUNEAU with a sequence number of 8899977 after no crashes.

DUMP FILES

The DUMP FILES command initiates an online dump, copying all of the specified audited data-base files to tape for possible use later by the rollforward process. DUMP FILES does not create disc dumps; these are created by the PUP commands REMOVEAUDITED and REVIVE. The DUMP FILES command syntax is:

```
DUMP FILES { file-set                          }
           { ( file-set [ , file-set ]... )    }

  [ , ALTFILES {OFF | ON}                             ]
  [ , COPIES quantity                                 ]
  [ , DRIVE { device                       }          ]
  [         { ( device [ , device ]... )   }          ] ...
  [ , NOT  { file-set                          }      ]
  [        { ( file-set [ , file-set ]... )    }      ]
  [ , OPERATOR {OFF | ON}                             ]
  [ , PARTONLY {OFF | ON}                             ]
  [ , REEL { reel                   }                 ]
  [        { (reel [ , reel ]... )  }                 ]
```

where

`file-set`

specifies one or more audited disc files that are to be dumped to tape. File-set identifiers are discussed in Appendix B. Any unaudited file is ignored, but TMF displays a warning message.

`ALTFILES OFF`

specifies that an alternate-key file will be dumped only if its *file-set* identifier is specified explicitly. The default setting is ALTFILES ON.

`ALTFILES ON`

specifies that, for each file dumped, TMF will also dump any alternate-key files that are automatically updated with that file. An alternate-key file that is not automatically updated will be ignored, but TMF will display a warning message. This is the default setting.

`COPIES quantity`

specifies the number of copies to be made of each dump. The default *quantity* is 1. The maximum *quantity* is 99. Multiple copies are written simultaneously and, therefore, require multiple tape drives (see the DRIVE option below).

`DRIVE`

specifies the tape drives to which the files are to be dumped. If you do not specify any drives in the command, TMF prompts you for them.

`device`

identifies a tape drive. This is a standard GUARDIAN device name, such as $TAPE or \KEOKUK.$TAPE2.

→

`NOT`

specifies one or more files that are not to be dumped, regardless of whether they are specified elsewhere in the command.

`OPERATOR OFF`

specifies that any dump requests for operator intervention will go to the terminal from which the command was given. If no TMF operator has been designated, this is the default setting. Otherwise, the default setting is OPERATOR ON.

`OPERATOR ON`

specifies that any dump requests for operator intervention will go to the TMF operator previously designated by the ALTER TMF command. If a TMF operator has been designated, this is the default setting. Otherwise, the default setting is OPERATOR OFF.

`PARTONLY OFF`

specifies that, for each partitioned file in the *file-set* list, the file is to be dumped entirely or not at all. If a file's primary partition is specified in the *file-set* list, then all of its secondary partitions will be dumped automatically. Any secondary partition specified in the *file-set* list will be dumped only if its primary partition is also in the list. PARTONLY affects only partitioned files; its default setting is OFF.

`PARTONLY ON`

specifies that no files outside the specified *file-set* list are to be dumped. If a file partition is specified in the *file-set* list, it will be treated independently; any other partitions will be dumped only if specified individually in the *file-set* list. PARTONLY affects only partitioned files; its default setting is OFF.

This option is useful if you want to dump all files (and portions of files) residing on a particular disc volume, without dumping the related files on other volumes.

`REEL`

specifies one or more physical tape reels on which dumps are to be written. When the process needs a scratch reel, it selects the next *reel* in this list and tries to find that reel on one of the currently assigned tape drives. If it cannot find the reel on a drive, or if the reel is not marked as a scratch tape, TMF prompts the operator to either supply another reel or mount the missing reel.

`reel`

is a reel identifier, as defined in the "Tape Terminology" subsection in Section 1 of this manual.

Considerations

This command, with ALTFILES ON and PARTONLY OFF (the default situation), dumps all audited alternate-key files and partitions whose primary files are specified in the command, regardless of where these other files reside. Each *implicitly* specified file thus dumped becomes a separate entry in the catalog.

With both ALTFILES ON and PARTONLY ON together, PARTONLY takes precedence. Thus, any secondary partitions for alternate-key files would not be dumped unless individually included in the *file-set* list.

Note that any files specified in the *file-set* list will be dumped, regardless of whether they are alternate-key files and regardless of the ALTFILES setting.

See Section 3, "Media-Handling Responses," for valid responses to system prompts.

An online dump would be useless if TMF is configured with an ALTER TMF, RECOVERY OFF command. Therefore, if recovery is disabled, TMF rejects the DUMP FILES command and prints an error message.

Examples

The command

```
DUMP FILES $VOLA.*.*, DRIVE $TAPE1
```

- expects a scratch tape to be mounted on tape drive $TAPE1
- writes to that tape one copy of each audited data file on disc volume $VOLA, including all qualified alternate-key files and all secondary partitions of files whose primary partitions are on $VOLA
- directs any operator prompts to the TMF operator, if one has been designated, or otherwise to TMFCOM's home terminal
- provides a report similar to this one:

```
TMFCOM - T9066B00 - (28JAN85)     System \SAB     20-Apr-1985 19:47:18
[Dump serial number is 375.]
  File Name                       MATSN    FATSN   Error
[Tape JER234 used for copy 1 of reel 1 on drive $Tape.]
$VOLA.ADMIN.GEORGE                367      724
$VOLA.ADMIN.JOHN                  367      724
$VOLA.ADMIN.SALLY                 367      724
[Total files dumped = 3.]
```

In the report, the MATSN column shows the sequence number of the monitor audit trail file in use at the time of the dump. Similarly, the FATSN column shows the sequence number of the data audit trail in use at the time of the dump.

The command

```
DUMP FILES $VOLA.ADMIN.*, NOT $VOLA.ADMIN.JOHN, REEL (ABC001, ABC003), DRIVE $TAPE1, &
  ALTFILES OFF, OPERATOR OFF
```

- expects either volume ABC001 or ABC003 to be mounted on tape drive $TAPE1
- writes to that tape one copy of each audited data file on disc volume $VOLA, subvolume AUDIT, except the one named JOHN

- does not dump any alternate-key files not otherwise specified
- dumps all secondary partitions of any primary-partition files (because PARTONLY ON was not specified)
- directs any operator prompts to TMFCOM's home terminal.

If $VOLA.ADMIN.JOHN is the primary partition of an audited, partitioned file, the command

```
DUMP FILES $VOLA.ADMIN.JOHN, COPIES 2, PARTONLY ON
```

dumps two copies of $VOLA.ADMIN.JOHN, including any alternate-key files (because ALTFILES OFF was not specified), but does not dump any of its secondary partitions.

ENABLE VOLUMES

The ENABLE VOLUMES command is valid on NonStop systems only, not on NonStop 1+ systems.

If a disc volume is "up" (according to PUP) but unavailable for TMF transaction processing, you can use this command to make it available. Some possible reasons for a volume's unavailability are:

- use of the NOT *volume* option with START TMF
- unresolved network transactions involving audited files on this volume
- previous use of the PUP command DOWN on this volume during the current TMF session, even if the UP command has been used subsequently
- a disc failure (for example, loss of contact with both copies of the disc process or loss of power to the drive or controller)
- unavailability of the related audit-trail volume.

ENABLE VOLUMES automatically performs autorollback for any specified volume requiring it, before enabling that volume.

The ENABLE VOLUMES command syntax is:

```
                 { volume                          }
ENABLE VOLUMES   { ( volume [ , volume ]... )      }(II only)
                 { *                               }

  [ , NOT  { volume                     } ]
  [        { ( volume [ , volume ]... ) } ]
```

where

`volume`

is disc volume name, such as $DATA. If no volume list is given, TMF tries to enable all volumes that are not in the NOT list.

`*`

specifies all disc volumes in the system.

`NOT`

specifies one or more disc volumes that are to be excluded from the volume list.

Considerations

The actions of ENABLE VOLUMES are a subset of the actions of START TMF. After START TMF has created various TMF processes, it tries to enable all (if its VOLUMES option is not used) or some volumes for transaction processing. If any volume is not enabled at that time, or is subsequently disabled by PUP DOWN or other means, the only way to enable or reenable that volume (while TMF is still running) is to use ENABLE VOLUMES. If autorollback is required, ENABLE VOLUMES automatically recovers the volume before enabling it.

If any network transactions are unresolved at the beginning of autorollback, the volumes that are affected by those transactions will not be recovered. If such a transaction is subsequently aborted or ended, and the volumes are physically available, the ENABLE VOLUMES command can then make those volumes available for TMF processing.

If a disc failure of some kind has made a volume unavailable, ENABLE VOLUMES can reenable the volume (after the underlying failure is corrected).

If an audit-trail volume is unavailable when START TMF or ENABLE VOLUMES is issued, no data volume that is audited to it can be enabled for TMF transaction processing until the audit-trail volume becomes available.

Examples

The command

```
ENABLE VOLUMES ($DATA, $ACCT, $SALES)
```

begins the autorollback process (if necessary) for audited files on all three of the specified volumes, but no others. If any other disc volume in the system has a audited file whose crash-open flag is on, any transactions affecting that file volume will return file-system error 82.

In the command sequence

```
START TMF, NOT ($DATA, $ACCT)
...
ENABLE VOLUMES *, NOT $DATA
...
ENABLE VOLUMES $DATA
```

the START TMF command enables transaction processing on all volumes in the system except $DATA and $ACCT. At this point, all transactions affecting any audited file on $DATA or $ACCT will return file-system error 82.

Later, the first ENABLE VOLUMES command begins the autorollback process (if necessary) for $ACCT and any other disc volumes with active crash-open flags except $DATA. Now transactions can be processed for audited files on $ACCT but not $DATA.

Finally, the second ENABLE VOLUMES command is applied to $DATA. Now transactions can be processed for audited files on any volume in the system.

END TRANSACTION

END TRANSACTION forces a distributed transaction, which was started on some other network node and stuck in ending state, to commit its changes on this system. It also releases any locks held for that transaction. The END TRANSACTION command syntax is:

```
END TRANSACTION transaction-identifier [ ! ]
```

where

`transaction-identifier`

identifies the transaction to be ended. Its form is defined in the "Transaction Identifiers" subsection near the beginning of this section.

`!`

indicates that the operation should be performed without waiting for confirmation. If this is omitted, TMF asks you for confirmation; a response of YES or Y allows the operation to proceed, but a response of NO, N, or STOP aborts the operation. With any other response, TMF displays an error message and reprompts you.

Considerations

If a distributed transaction has actually completed successfully on each of its network nodes, but a communications failure prevents its home node from notifying all the other nodes to release their locks, the transaction will be stuck in the ending state. If this happens, you can use the END TRANSACTION command to complete the transaction on each local system.

Before using this command, use the STATUS TRANSACTION command to determine the status of the transaction on its home node. (If the EXPAND lines are down, this may require a telephone call or other external form of communication with someone on the other system.) If the transaction is ending on its home node, you can safely use END TRANSACTION on your local system. Otherwise, you should use ABORT TRANSACTION; releasing locks for an uncommitted transaction or a transaction that has been aborted on its home system could result in an inconsistent data base.

For all transactions that are currently active on system \A, the TMFCOM command

```
STATUS TRANSACTION *.*
```

issued on system \A displays the disposition of those transactions on system \A, regardless of where each transaction originated. If a transaction is "hung" on your system and network communication with its home system is interrupted, you should contact someone at that system (by telephone or other means) and run STATUS TRANSACTION there, to determine whether you should end or abort the "hung" transaction.

Example

In the next example:

```
END TRANSACTION \JUNEAU(1).4.8899977
\JUNEAU(1).4.8899977, PROCESS $DELREC(4,11), STATE ACTIVE
Confirm? YES
```

TMF accepts the END TRANSACTION command, displays the status of the specified transaction, prompts for confirmation, then attempts to end the transaction that has transaction identifier \JUNEAU(1).4.8899977; that is, a transaction that originated in CPU 4 of system \JUNEAU, after one system failure there, with a sequence number of 8899977.

EXCLUDE VOLUMES

The EXCLUDE VOLUMES command is valid on NonStop systems only, not on NonStop 1+ systems.

EXCLUDE VOLUMES lets TMF purge audit-trail files that might be required for autorollback of the specified disc volume. This can make autorollback on that volume impossible; if so, some files on the volume may require rollforward recovery.

The EXCLUDE VOLUMES command syntax is:

```
EXCLUDE VOLUMES { volume                       } (II only)
                { (volume [ , volume ]... )    }
```

where

`volume`

is a disc volume name, such as $DATA, or a logical device number, such as $65.

Considerations

The EXCLUDE VOLUMES command can be used only while TMF is running (that is, after START TMF), on volumes that failed to be enabled by START TMF or ENABLE VOLUMES. It has no effect on a volume that is enabled for TMF processing.

For instance, if any network transactions are unresolved when TMF is started, autorollback cannot proceed on the volumes that are affected by those transactions and those volumes will not be enabled for TMF transaction processing. If such a transaction is subsequently ended or aborted, the ENABLE VOLUMES command can then initiate autorollback (where necessary) and make those volumes available for TMF processing. Meanwhile, TMF keeps all of those volumes' related audit-trail files on disc for the potential autorollback (if an incomplete transaction must be rolled back, the autorollback process might need the old auditing images). If other processing continues while the old audit-trail files are kept on disc, TMF could reach the MAXFILES limit and suspend transaction processing or the audit-trail volume could run out of space.

EXCLUDE VOLUMES can relieve this problem by letting TMF purge any audit files that are being kept solely for the future autorollback of the specified volume. Note, however, that elimination of those audit files can make it impossible subsequently to use autorollback on the excluded volume. If recovery were needed, it would then have to be performed by the rollforward process.

Once EXCLUDE VOLUMES is given, it remains in effect until the specified volumes are successfully reenabled—either by the ENABLE option of START TMF (explicitly or implicitly) or by ENABLE VOLUMES.

Examples

The command

```
EXCLUDE VOLUMES $DATA
```

lets TMF eliminate audit-trail files that might be required for autorollback on $DATA.

INFO AUDITDUMP

INFO AUDITDUMP displays the current attributes of the specified audit-dump processes, as established by previous ADD AUDITDUMP and ALTER AUDITDUMP commands. The INFO AUDITDUMP command syntax is:

```
                 { process                        }
INFO AUDITDUMP   { ( process [, process ]... )    }  [, {BRIEF | DETAIL} ]
                 { *                              }
```

where

`process`

names a process, on the local system, whose configuration you want displayed.

`*`

provides information on all audit-dump processes on the local system.

`BRIEF` or `DETAIL`

determines how much information is to be displayed. With BRIEF, TMF always displays the identifier and the number of copies; if the output file or drives were specified, they are also displayed in the BRIEF report. With DETAIL, if the CPUs or priority were specified, they are also displayed. If you do not specify either BRIEF or DETAIL, TMF displays the detailed report.

Considerations

Unlike the ADD AUDITDUMP, ALTER AUDITDUMP, and DELETE AUDITDUMP commands, INFO AUDITDUMP can be used at any time, regardless of whether the auditing subsystem has been started. Note, however, that it only displays the defined parameters of the indicated audit-dump processes (cf. STATUS AUDITDUMP), regardless of when it is issued.

The information displayed by the INFO AUDITDUMP command can only reflect those audit-dump process attributes which have been specifically defined by previous ALTER AUDITDUMP and ADD AUDITDUMP commands, including SET AUDITDUMP parameters effected by subsequent ADD AUDITDUMP commands. With the exception of COPIES, the display does *not* include any options that have been completely left to their default values. For instance, if you have issued the command

```
ADD AUDITDUMP $AUD1, COPIES 2
```

with no previous SET AUDITDUMP command and no subsequent ALTER AUDITDUMP command, then INFO AUDITDUMP will show only the COPIES value—not CPUS, DRIVE, OUT, or PRI—because these other options were never specified explicitly.

The information displayed by INFO AUDITDUMP is a subset of that displayed by the INFO TMF command.

Examples

If $AUD0 is configured with no options and $AUD1 is configured with all options used, then the command

```
INFO AUDITDUMP *, BRIEF
```

provides a display similar to this:

```
AUDITDUMP PROCESSES:
    AUDITDUMP $ADP0, COPIES 1
    AUDITDUMP $ADP1, COPIES 2, DRIVE $TAPE, OUT $SYSTEM.AUDIT.ADP1
```

Under the same configuration, the command

```
INFO AUDITDUMP *, DETAIL
```

provides a display similar to this:

```
AUDITDUMP PROCESSES:
    AUDITDUMP $ADP0, COPIES 1
    AUDITDUMP $ADP1, CPUS (0:1), PRI 160, COPIES 2,
         OUT $SYSTEM.AUDIT.ADP1, DRIVE ($TAPE)
```

INFO AUDITTRAIL

INFO AUDITTRAIL displays the current attributes of selected audit trails, as configured by previous ADD AUDITTRAIL commands. These attributes include:

- the generic file identifiers of each selected audit-trail sequence
- the size of the primary and secondary extents for each selected audit trail
- the names of the processes that write to each audit trail selected in this command
- the maximum depth of files in the audit-trail sequence that can contain before- and after-images for a single active transaction (if a transaction's processing lasts long enough to span more than the maximum number of files, the transaction will be aborted)
- the maximum number of files that will be maintained concurrently on disc for each of the audit trails.

The INFO AUDITTRAIL command syntax is:

```
INFO AUDITTRAIL { generic-file-set                                 }
                { ( generic-file-set [ , generic-file-set ]... )   }

  [ , {BRIEF | DETAIL}                                                  ]
  [          { {AUDITPROCESS}  {process                          } }    ] (I only)
  [          { {DISCPROCESS }  {( process [ , process ]... )     } }    ] (II/DP1)
  [ , USAGE  { {AUXILIARY   }  { *                               } }    ] (II/DP2)...
  [          { MASTER                                            }      ] (II/DP2)
  [          { MONITOR                                           }      ] (DP1)
```

where

`generic-file-set`

identifies the sequence of files that form an audit trail. The identifier is defined in the "Generic File-Set Identifiers" subsection near the beginning of this section.

With this command, you can substitute an asterisk (*) for one or more elements of the *generic file-set* identifier, as with standard GUARDIAN file identifiers.

Each *generic file-set* identifier must have been specified in a previous ADD AUDITTRAIL command.

`BRIEF` or `DETAIL`

determines how much information is to be displayed. With BRIEF, TMF always displays the file identifier only. With DETAIL, all attributes except the extent size are always displayed; if the extent size was specified, it is also displayed. If you do not specify either BRIEF or DETAIL, TMF displays the detailed report.

→

`USAGE`

restricts the audit trails for which information is to be displayed. If you use this option, then the display includes only those audit trails whose audit processes or disc processes are specified in the option. On a NonStop 1+ system, you can use USAGE MONITOR or USAGE AUDITPROCESS. On a NonStop system with DP1 disc processes, you can use USAGE MONITOR or USAGE DISCPROCESS. On a NonStop system with DP2 disc processes, you can use USAGE MASTER or USAGE AUXILIARY.

`process`

identifies an audit process or disc process whose audit trail's information is to be displayed. If this identifier is in network form, its system component must indicate the system on which this copy of TMFCOM is running. The *process* identifier can be either the logical device number (in *$integer* format) or the name of the process. If you use the asterisk (*), information on all audit-process or disc-process audit trails will be displayed.

`USAGE MONITOR`

restricts the audit trail for which information is to be displayed. If you use this option, then the display includes only the monitor audit trail.

Considerations

Only those audit trails that match all of the options will be selected for information display. For instance, if you specify USAGE DISCPROCESS $AP1 with audit trail $AUDIT.AD.*, information will be displayed for only those audit trails on subvolume $AUDIT.AD that get records from disc process $AP1.

You can use INFO AUDITTRAIL at any time, regardless of whether TMF has been started. Note, however, that it only displays the defined parameters of the indicated audit trails (cf. STATUS AUDITTRAIL), regardless of when it is issued.

The information displayed by INFO AUDITTRAIL is a subset of that displayed by the INFO TMF command.

Examples

These examples are for a NonStop system, but they apply equally well to a NonStop 1+ system if you substitute AUDITPROCESS for DISCPROCESS.

The command

```
INFO AUDITTRAIL $AUDIT.AP.AA, USAGE MONITOR, BRIEF
```

provides information on audit trail $AUDIT.AP.AA only if it is also the monitor audit trail. The brief information display is similar to this:

```
AUDITTRAILS: AUDITTRAIL $AUDIT.AP.AA
```

The command

```
INFO AUDITTRAIL *.*.*, USAGE DISCPROCESS $VOL1, DETAIL
```

provides information on whatever audit trail is configured for disc process $VOL1. The detailed information display is similar to this:

```
AUDITTRAILS:
    AUDITTRAIL $AUDIT.AP.AA, EXT (500,300), MINFILES 3, MAXFILES 5,
        AUDITDUMP AUTOMATIC $ADP1, USAGE DISCPROCESS *
```

INFO BACKOUT

INFO BACKOUT displays the attributes of the backout process configured by the ALTER BACKOUT command. The INFO BACKOUT command syntax is:

```
INFO BACKOUT [ , {BRIEF | DETAIL} ]
```

where

`BRIEF` or `DETAIL`

determines how much information is to be displayed. If you do not specify either BRIEF or DETAIL, TMF displays the brief report.

Considerations

INFO BACKOUT displays the configured execution priority of the backout process and identifies the primary and backup processors, if you specified these options in a previous ALTER BACKOUT command. The STATUS BACKOUT command displays all configuration attributes of the backout process.

You can use INFO BACKOUT at any time, regardless of whether TMF has been started.

The information displayed by the INFO BACKOUT command reflects only those backout process attributes that have been specifically defined by previous ALTER BACKOUT commands. The display does *not* include any other options. For instance, if you have issued the command

```
ALTER BACKOUT, PRI 185
```

with no other ALTER BACKOUT command, then INFO BACKOUT will show only the assigned execution priority—not the CPUS option—because the CPUS option was never specified explicitly in TMFCOM.

The information displayed by INFO BACKOUT is a subset of that displayed by the INFO TMF command.

INFO CATALOG

INFO CATALOG displays the current configuration of the TMF catalog, as established by previous ALTER CATALOG commands. The INFO CATALOG command syntax is:

```
INFO CATALOG [ , {BRIEF | DETAIL} ]
```

where

`BRIEF` or `DETAIL`

determines how much information is to be displayed. If you do not specify either BRIEF or DETAIL, TMF displays the brief report.

Considerations

You can use the INFO CATALOG command at any time, regardless of whether the auditing subsystem has been started. Note, however, that it only displays the defined parameters of the catalog (cf. STATUS CATALOG), regardless of when it is issued.

The information displayed by the INFO CATALOG command is available only if one or more catalog attributes has been specifically defined by a prior ALTER CATALOG command. The display does not include the OUT, CPUS, or PRI options if they have been reset or left to their default values; also, it does not include the RETAINDEPTH option if it has been reset. However, INFO CATALOG will show the RELABEL, MEDIALOG, and RELEASED options at all times after any catalog attribute has been specifically defined; if the RETAINDEPTH option has not been reset, it is also among those displayed at all such times.

For instance, if you have issued no ALTER CATALOG command, then INFO CATALOG will produce no output. If you then alter the catalog priority, INFO CATALOG will show the PRI, RETAINDEPTH, RELABEL, MEDIALOG, and RELEASED attributes. Even if PRI is reset, subsequent INFO CATALOG commands will show the RETAINDEPTH, RELABEL, MEDIALOG, and RELEASED attributes until TMF is reinitialized.

The information displayed by INFO CATALOG is a subset of that displayed by the INFO TMF command.

Example

After the catalog priority is altered, the command

```
INFO CATALOG, DETAIL
```

provides a display similar to this:

```
TMF CATALOG: PRI 150, RETAINDEPTH 3, RELABEL OFF, MEDIALOG ON, RELEASED OFF
```

INFO DUMPS

INFO DUMPS lists all or some of the online or audit dumps currently recorded in the catalog. The INFO DUMPS command syntax is:

```
INFO DUMPS { file-set                          }
           { ( file-set [ , file-set ]... )    }

 [ , {BRIEF | DETAIL}                            ]
 [            {DISC pack                  }      ]
 [ , MEDIUM   {TAPE { reel               }}      ] ...
 [            {     { (reel [,reel]...)  }}      ]
 [ , SERIAL number                               ]
 [ , TYPE { AUDITDUMP | ONLINEDUMP}              ]
```

where

`file-set`

specifies one or more dumps (of audited files or audit trails) whose catalogued information you want displayed. File-set identifiers are discussed in Appendix B. TMF displays a non-fatal error message for any explicitly specified dump that is not listed in the catalog.

`BRIEF` or `DETAIL`

determines how much information is to be displayed. BRIEF displays only the serial number of each specified dump. DETAIL displays all information, including reel identifiers for all reels used and audit-file sequence numbers for online dumps. If neither BRIEF nor DETAIL is specified, the display has one line per file, with the dump serial number, date, time, and identifier of the first reel of the first copy of each file.

`MEDIUM`

limits the catalog entries to be listed by this command. When you use the MEDIUM option, INFO DUMPS lists only those files that are specified in the command and also reside (at least in part) on one of the specified *reel*s or *pack*s.

`DISC pack`

identifies the specific disc pack whose catalog entries are to be displayed. The *pack* identifier comprises six or fewer alphanumeric characters.

`TAPE`

identifies the tape reels whose catalog entries are to be displayed.

`reel`

is a reel identifier, as defined in the "Tape Terminology" subsection in Section 1 of this manual.

`SERIAL number`

limits the directory entries to be listed by this command. When you use the SERIAL option, INFO DUMPS lists only those files that are specified in the command and also have the specified dump serial number. You can get this number either from a previous INFO DUMPS command or by reading the report from an audit dump or online dump.

→

TYPE

limits the directory entries to be listed by this command. When you use the TYPE option, INFO DUMPS lists only those files that are specified in the command and also are the specified type of dump: audit or online. If you do not use the TYPE option, all types of dumps are listed, subject to the SERIAL and REEL options.

Considerations

The dump serial numbers shown in INFO DUMPS displays are useful as parameters for DELETE DUMPS or any other command that has a SERIAL option.

If a dump is marked INVALID, the recorded file is unusable, perhaps because it has been partially overwritten or for some other reason that does not preclude reuse of the tape. BAD, however, generally means that the tape itself is defective and should not be reused.

One file on a tape can be bad or invalid while other files on the same tape remain usable. If the tape gets creased, for example, only the file recorded at that point will probably be bad; nevertheless, you should discard that tape when it becomes a scratch tape.

Examples

The command

```
INFO DUMPS ($DATA.JAN14.*, $DATA.JAN15.*), BRIEF
```

lists only the serial numbers, for all dumps of files from disc volume $DATA, subvolume JAN14 or JAN15. On a NonStop 1+ system, the display looks similar to this:

```
TMF CATALOG DUMP SERIAL NUMBERS:

    4000, 1501, 1001, 1002, 1004, 1005, 1006, 1007, 1008, 1009,
    1052, 1053, 1054, 1055, 65, 67, 64, 66, 71, 1500
```

On a NonStop system, however, the BRIEF display looks like this:

```
TMF CATALOG DUMP GENERATION LIST FOR SYSTEM \KIMCHI AT 28-Feb-1985 13:27:29

             FILENAME                DUMP SERIAL NUMBERS

    $DATA.JAN14.DAVID            200, 263
    $DATA.JAN14.MARY             250
    $DATA.JAN14.RENNIE           220, 283
    $DATA.JAN15.CHARLES          240
    $DATA.JAN15.MARIE            255
```

The command

```
INFO DUMPS ($DATA.RMW.*, $DATA.JER.*), SERIAL 4235
```

lists a moderate amount of information for all dumps of files from disc volume $DATA, subvolume RMW or JER, having serial number 4235. On a NonStop 1+ system, the moderate information display looks similar to this:

```
TMF CATALOG DUMP INFORMATION AT 25-Feb-83 14:23:37
    DUMP SERIAL NUMBER      DATE       TIME     REEL ID    MEDIUM
          4235          21-Apr-1982 12:13:06    JKH036     TAPE
          4235          21-Apr-1982 12:51:50    JKH037     TAPE
```

On a NonStop system, however, the moderate display looks like this:

```
TMF CATALOG DUMP INFO FOR SYSTEM \KIMCHI AT 28-Feb-1985 13:27:29

             FILENAME                 DATE       TIME    MEDIA ID   SERIAL

    $DATA.JER.DEC11            (R) 28-Feb-1985 12:21:51   JKH036  4235
    $DATA.JER.DEC29                28-Feb-1985 12:20:50   pack19      4235
    $DATA.RMW.DEC21                28-Feb-1985 12:24:36   JKH037 *    4235
```

The letter R for file JKH036 indicates that this dump has been released but not yet scratched. The asterisk (*) beside media identifier JKH037 indicates that this is only the first reel of a multipart dump.

The command

```
INFO DUMPS $DATA.ABC.*, DETAIL
```

lists detailed information for all dumps of files from disc volume $DATA, subvolume ABC. The detailed information display looks similar to this:

```
TMF CATALOG DETAILED DUMP INFORMATION AT 25-Feb-83 14:23:37

    $DATA.ABC.ADAMS , SERIAL 365
         TIME: 24-Jan-1982 11:34:53
         TYPE: ONLINEDUMP (34,6)
         MEDIUM: TAPE
         BAD OFF
         INVALID OFF
             PART  COPY  MEDIA ID  STATUS
               1     1    ABC004   READABLE
               1     2    ABC035   READABLE

    $DATA.ABC.DONNE , SERIAL 366
         TIME: 1-Feb-1982 21:47:49
         TYPE: ONLINEDUMP (39,7)
         MEDIUM: TAPE
         BAD OFF
         INVALID OFF
             PART  COPY  MEDIA ID  STATUS
               1     1    ABC876   READABLE
               1     2    ABC542   READABLE
               2     1    ABC568   READABLE
               2     2    ABC183   READABLE

    $DATA.ABC.DONNE , SERIAL 387
         TIME: 5-Feb-1982 17:26:29
         TYPE: ONLINEDUMP (42,8)
         MEDIUM: DISC
         BAD OFF
         INVALID OFF
         MIRROR DUMP
             PART  COPY  MEDIA ID  STATUS
               1     1    PACK12   READABLE
```

The detailed display shows that $DATA.ABC.ADAMS is part of the dump whose serial number is 365, and that this dump was made when the data audit trail was at sequence number 34 and the monitor audit trail was at sequence number 6. $DATA.ABC.DONNE is part of dump 366, with audit-trail sequence numbers 39 and 7, respectively.

The dump of $DATA.ABC.DONNE has two parts (two reels of tape) for each of two copies. The dump of $DATA.ABC.ADAMS also has two copies, but only one part (one reel) for each copy.

On a NonStop system, the display is similar except that "RELEASED ON" or "RELEASED OFF" is displayed on the line after the "INVALID" status and "BAD ON" or "BAD OFF" is not displayed.

INFO MEDIA

INFO MEDIA lists all or some of the tape reels and disc packs currently recorded in the catalog's media directory. This command replaces the former INFO TAPES command, but TMFCOM still recognizes INFO TAPES and interprets it as INFO MEDIA. The INFO MEDIA command syntax is:

```
           ( pack                   )  [ , {BRIEF | DETAIL}      ]
INFO MEDIA { reel                   }  [          (ASSIGNED)     ] ...
           { ( reel [, reel] ... )  }  [ , STATUS {BAD      }    ]
           ( *                      )  [          {RELEASED }    ]
                                       [          (SCRATCH )     ]
```

where

`pack`

is the six-character (or shorter) alphanumeric identifier of a physical, removable disc pack. This identifier is encoded on the disc pack when the PUP command REMOVEAUDITED creates an online dump; it should also be written visibly on the disc pack's cover. Note that the *pack* identifier names the physical disc, not the data recorded on it or the volume name of the disc drive.

`reel`

is a reel identifier, as defined in the "Tape Terminology" subsection in Section 1 of this manual.

`*`

specifies all entries in the directory.

`BRIEF` or `DETAIL`

determines how much information is to be displayed. BRIEF displays only the matching tape-reel or disc-pack identifiers. DETAIL displays all information in media directory about the specified tapes or discs, including its status (assigned, bad, released, or scratch), owner, type (disc or tape), and current activity (if any). If you do not specify either BRIEF or DETAIL, the display has one line per tape or disc, showing its media identifier, owner, and status or current activity.

`STATUS`

limits the media directory entries to be listed by this command. INFO MEDIA with the STATUS option lists only those tape reels or disc packs that are specified in the command and also have the specified status.

Considerations

If the media directory has not been configured, INFO MEDIA produces an error message.

To display more information, including the contents (if known) of assigned tapes, you can use INFO DUMPS with the DETAIL option.

Examples

The command

```
INFO MEDIA *, STATUS RELEASED, BRIEF
```

lists the reel identifiers of all released tapes in the directory (discs in the catalog can only have "assigned" status). The brief display looks similar to this:

```
TMF CATALOG MEDIUM LABEL IDENTIFIERS:

    254315, 335458, 346103, ABC001, ABC002, ABC003, ABC004, ABC005,
    JKH004, JKH005, JKH00T, WJE001, WJE002, WJE003, WJE004, WJE005, a,
    gsk001, x, y, z
```

The identifiers are listed in ASCII-code order, in which a numerical digit comes before any alphabetic letter and an uppercase letter comes before any lowercase letter.

The command

```
INFO MEDIA (ABC001, ABC002, ABC003, ABC004, ABC005, PACK12)
```

lists the reel identifiers, owner, and status of each of the six specified volumes. The moderate information display looks similar to this:

```
TMF CATALOG MEDIUM INFORMATION AT 25-Feb-83 14:23:37
    MEDIUM LABEL ID        OWNER          STATUS
        ABC001         SUPER.SUPER     ASSIGNED (CONTENTS RECORDED)
        ABC002         SUPER.SMITH     ASSIGNED (CONTENTS UNKNOWN)
        ABC003                         SCRATCH
        ABC004         SUPER.JONES     CURRENTLY BEING WRITTEN
        ABC005         SUPER.ROGERS    BAD
        PACK12 (DISC)  SUPER.SUPER     ASSIGNED (CONTENTS RECORDED)
```

The command

```
INFO MEDIA *, DETAIL
```

lists all information in the media directory about tapes or discs. The detailed information display is in a format similar to this:

```
TMF CATALOG MEDIUM INFORMATION AT 25-Feb-83 14:23:37

    MEDIUM LABEL ID: JKH004
         STATUS: ASSIGNED (CONTENTS RECORDED)
         OWNER: SUPER.SUPER
         TYPE: TAPE

    MEDIUM LABEL ID: JKH005
         STATUS: SCRATCH
         TYPE: TAPE

    MEDIUM LABEL ID: JKH006
         STATUS: ASSIGNED (CONTENTS RECORDED)
         OWNER: SUPER.SMITH
         TYPE: TAPE
         CURRENTLY ACTIVE: PROCESS $AUD1

    MEDIUM LABEL ID: PACK12
         STATUS: ASSIGNED (CONTENTS RECORDED)
         OWNER: SUPER.SMITH
         TYPE: DISC
```

INFO TMF

INFO TMF displays the current attributes of the backout and catalog processes and of the audit trails, as configured by previous ADD AUDITTRAIL, ADD AUDITDUMP, ALTER AUDITDUMP, ALTER CATALOG, and ALTER BACKOUT commands. Defaults for any attributes not defined by these commands are also displayed. The INFO TMF command syntax is:

```
INFO TMF   , {BRIEF | DETAIL}
```

where

`BRIEF` or `DETAIL`

determines how much information is to be displayed. BRIEF provides only the minimum information display. DETAIL provides the maximum information display. If neither BRIEF nor DETAIL is specified, only a moderate amount of information is displayed.

Considerations

You can use the INFO TMF command at any time, regardless of whether TMF has been started. Note, however, that it only displays the defined parameters of the various processes (cf. the STATUS commands), regardless of when it is issued.

The data displayed by INFO AUDITDUMP, INFO AUDITTRAIL, INFO BACKOUT, and INFO CATALOG are subsets of those displayed by the INFO TMF command.

Examples

The command

```
INFO TMF, BRIEF
```

provides a display similar to this:

```
TMF:
    OPERATOR $OPTERM, RECOVERY ON
BACKOUT:
    PRI 149
AUDITTRAILS:
    AUDITTRAIL $AUDIT.AP.MN
    AUDITTRAIL $AUDIT.AP.AA
AUDITDUMP PROCESSES:
    AUDITDUMP $ADP0, COPIES 2
TMF CATALOG:
    RETAINDEPTH 3, RELABEL OFF, MEDIALOG ON, RELEASED OFF
```

The command

```
INFO TMF, DETAIL
```

provides a display similar to this:

```
THE TMF AUDITTRAIL CONFIGURATION IS COMPLETE.
TMF:
    OPERATOR $OPTERM, RECOVERY ON
BACKOUT:
    PRI 149
AUDITTRAILS:
    AUDITTRAIL $AUDIT.AP.MN, EXT (20,20), MINFILES 2, MAXFILES 4,
         AUDITDUMP $ADPO, USAGE MONITOR
    AUDITTRAIL $AUDIT.AP.AA, EXT (90,90), MINFILES 2, MAXFILES 4,
         AUDITDUMP $ADPO, USAGE DISCPROCESS *
AUDITDUMP PROCESSES:
    AUDITDUMP $ADPO, COPIES 2
TMF CATALOG:
    RETAINDEPTH 3, RELABEL OFF, MEDIALOG ON, RELEASED OFF
```

The report says "AUDITTRAILS MAY BE ADDED TO THE TMF CONFIGURATION" if you have not used the START TMF command since the last time TMF was initialized. When the report says "THE TMF AUDITTRAIL CONFIGURATION IS COMPLETE," no audit trails can be added or changed without first reinitializing TMF.

This example is for a NonStop system, but it applies equally well to a NonStop 1+ system with AUDITPROCESS substituted for DISCPROCESS in the audit-trail usage.

INITIALIZE CATALOG

INITIALIZE CATALOG deletes all tape, disc, and dump records from the catalog data base. It also resets to 64 the counter from which dump serial numbers are assigned. The INITIALIZE CATALOG command syntax is:

```
INITIALIZE CATALOG    !
```

where

!

indicates that the operation should be performed without waiting for confirmation. If this is omitted, TMF asks you for confirmation; a response of YES or Y allows the operation to proceed, but a response of NO, N, or STOP aborts the operation. With any other response, TMF displays an error message and reprompts you.

Considerations

The functions performed by INITIALIZE CATALOG are not related to those performed by the INITIALIZE TMF command. INITIALIZE CATALOG has no effect on configuration attributes. To change any characteristics of the catalog, use the ALTER CATALOG command.

Because INITIALIZE CATALOG clears the catalog of all records of prior online dumps and audit-trail dumps, those dumps become unusable. This command also resets the serial number counter, so a dump made after catalog initialization can have the same number as an earlier, now-useless dump. The shutdown serial number counter is simultaneously reset, so a TMF shutdown after catalog initialization can have the same number as a pre-initialization shutdown (see STOP TMF).

INITIALIZE TMF

The INITIALIZE TMF command purges all configuration information defined by the ADD AUDITTRAIL, ALTER AUDITTRAIL, ALTER BACKOUT, ADD AUDITDUMP, ALTER AUDITDUMP, ALTER TMF, and ALTER CATALOG commands. It also removes all dumps information from the catalog, as if you had used DELETE DUMPS *.*.*.

On a NonStop 1+ system, INITIALIZE TMF is required before you issue any commands that change an existing configuration of the audit trails. For example, any change that deletes an audit process requires reinitialization.

The INITIALIZE TMF command syntax is:

```
INITIALIZE TMF [ ! ]
```

where

!

indicates that the operation should be performed without waiting for confirmation. If this is omitted, TMF asks you for confirmation; a response of YES or Y allows the operation to proceed, but a response of NO, N, or STOP aborts the operation. With any other response, TMF displays an error message and reprompts you.

Considerations

This command resets TMF to an initial state. You must not use INITIALIZE TMF

- when any transaction has begun but not completed
- when the data base does not meet the application's definition of consistency
- when the data base is damaged
- on one node of a network, when other nodes may have uncompleted transactions or undelivered messages for the node you are initializing.

The functions performed by the INITIALIZE CATALOG command are not a subset of those performed by INITIALIZE TMF. INITIALIZE CATALOG removes all catalog entries, but INITIALIZE TMF removes only the dump entries and retains the media entries.

Example

This example shows a typical sequence of events that would require use of the INITIALIZE TMF command on a NonStop system but it applies equally well to a NonStop 1+ system if you substitute AUDITPROCESS for DISCPROCESS.

1. `INITIALIZE TMF`

2.
```
ADD AUDITTRAIL $AUDIT.AP.AA,USAGE DISCPROCESS $VOL3
ADD AUDITTRAIL $AUDIT.AP.AB,USAGE DISCPROCESS *
ADD AUDITTRAIL $AUDIT.AP.MN,USAGE MONITOR
ALTER TMF, OPERATOR $OPTERM
```

3. `START TMF`

   .
   .
   .
   (transaction processing proceeds)
   .
   .
   .

4. `STOP TMF`

5. Define a new disc process (on a NonStop system) or audit process (on a NonStop 1+ system) named $VOL4.

6. `INITIALIZE TMF`

7.
```
ADD AUDITTRAIL $AUDIT.AP.AA,USAGE DISCPROCESS ($VOL3,$VOL4)
ADD AUDITTRAIL $AUDIT.AP.AB,USAGE DISCPROCESS *
ADD AUDITTRAIL $AUDIT.AP.MN,USAGE MONITOR
ALTER TMF, OPERATOR $OPTERM
```

8. `START TMF`

9. `DUMP FILES *.*.*`

In this example, TMF is initialized, configured, and started, then it is stopped to allow the addition of a new disc process, named $VOL4. This requires reinitialization, after which the TMFCOM configuration is redone, TMF is started, and an online dump is made.

Even if the second set of ADD AUDITTRAIL commands were identical to the first set, so $VOL4 would write to audit trail AB, reinitialization would still be required because the overall audit-trail configuration would be changed.

NEXT AUDITTRAIL

For the specified *generic file-set* sequence, the NEXT AUDITTRAIL command causes each TMF system process that has the current audit-trail file open to close that file and prepare to use the next one. The NEXT AUDITTRAIL command syntax is:

```
NEXT AUDITTRAIL generic-file-set
```

where

```
  generic-file-set
```

identifies the audit trail whose next file is to be created. The identifier is defined in the "Generic File-Set Identifiers" subsection near the beginning of this section. No "wild-card" asterisks are allowed with this command.

The *generic file-set* identifier must have been specified in an ADD AUDITTRAIL command before TMF was started.

Considerations

If any TMF process has the current file of the specified sequence open, then NEXT AUDITTRAIL closes the open file. The next process writing to that audit trail will create and open a new one in the sequence. With *generic file name* XY, for example, any TMF process that currently has file XY000567 open would close it and write its next audit record to file XY000568. If the audit trail is configured for automatic dumping (see "ADD AUDITTRAIL"), the closed file will be dumped (but not necessarily purged) as soon as the required tape drive and reel are available.

Examples

The command

```
NEXT AUDITTRAIL $JER.A.BB
```

switches to the next file of audit trail $JER.A.BB.

RECOVER FILES

The RECOVER FILES command initiates data-base recovery, using online dumps previously created on tapes or discs, then uses audit-trail dumps to roll the reloaded data base forward to its latest possible consistent state or to a specified shutdown point. The RECOVER FILES command syntax is:

```
RECOVER FILES { file-set                            }
              { ( file-set  [, file-set] ... )      }

  [ , {BRIEF | DETAIL}                                ]
  [ , ALTFILES {OFF | ON}                             ]
  [ , COPYDISC {OFF | ON}                             ]
  [ , CRASHOPEN {OFF | ON}                            ]
  [ , DRIVE  {device                          }       ]
  [          {( device [ , device ]... )      }       ]
  [ , NOT  {file-set                            }     ] ...
  [        {( file-set [ , file-set ]... )      }     ]
  [ , OPERATOR {OFF | ON}                             ]
  [ , PARTOF { $disc-volume | *}                      ]
  [ , PARTONLY {OFF | ON}                             ]
  [ , PREFER {DISC | TAPE}                            ]
  [ , SHUTDOWN SERIAL number                          ]
```

where

`file-set`

specifies which online-dump files are to be restored to disc. File-set identifiers are discussed in Appendix B. TMF displays a nonfatal error message for any explicitly specified file that is not listed in the catalog; no unlisted file is rolled forward.

Note that the nature of a disc online dump means recovery necessarily involves restoring *all* files on the disc to their online-dump state, then rolling forward only the audited files that are specified in the command.

`ALTFILES OFF`

specifies that an alternate-key file will be recovered only if its *file-set* identifier is specified explicitly. The default setting is ALTFILES ON.

`ALTFILES ON`

specifies that, for each file, recovery will attempt to include any automatically updated alternate-key files associated with that file. This is the default setting.

`COPYDISC OFF`

specifies that each disc pack is to be restored directly, without first making a copy of it. The default setting is COPYDISC ON. This option is only applicable when you use a disc online dump.

`COPYDISC ON`

for each disc pack to be restored, copies that pack to a scratch pack and rolls forward on the copy. This lets you keep the original disc pack for use in any future rollforward. This is the default setting. This option is only applicable when you use a disc online dump.

→

`CRASHOPEN OFF`

specifies that TMF should recover and roll forward all specified files, not just those which were open when the system failure occurred. This option is for *media-failure recovery*— a disc-drive head crash, for example. The default is CRASHOPEN ON. If you use the SHUTDOWN option, you must also use CRASHOPEN OFF in the same command.

`CRASHOPEN ON`

specifies that TMF should recover and roll forward only those specified files which were open when the system failure occurred. This is the default setting. You cannot use CRASHOPEN ON and the SHUTDOWN option in the same command.

`DRIVE`

specifies the tape drives or disc drives from which the files are to be recovered. If you do not specify any drives in the command, TMF prompts you for them.

`device`

identifies a tape drive or a disc drive. For a tape drive, *device* is a standard GUARDIAN device name, such as $TAPE or \KEOKUK.$TAPE2. For a disc drive, it is the logical device number (LDEV) with which that drive was configured.

`NOT`

specifies one or more files that are not to be recovered, regardless of whether they are specified in the *file-set* list and regardless of their CRASHOPEN status.

`OPERATOR OFF`

specifies that any requests for operator intervention will go to the terminal from which the command was given. The default setting is OPERATOR ON.

`OPERATOR ON`

specifies that any requests for operator intervention will go to the TMF operator previously designated by the ALTER TMF command. This is the default setting.

`PARTOF disc-volume`

limits recovery to partitions that are defined in the file identifier list and whose primary partitions reside on the specified *disc volume*. When you use PARTOF, no nonpartitioned files are recovered. Also, when you use PARTOF, all specified files must reside on the same disc volume. If you use the asterisk (*), all partitions specified in the *file-set* list are restored.

`PARTONLY OFF`

specifies, for each partitioned file in the *file-set* list, the file is to be recovered entirely or not at all. If a file's primary partition is specified in the *file-set* list, then all of its secondary partitions will be recovered automatically. Any secondary partition specified in the *file-set* list will be recovered only if its primary partition is also in the list. PARTONLY affects only partitioned files; its default setting is OFF.

→

`PARTONLY ON`

specifies that no files outside the specified *file-set* list are to be recovered. If a file partition is specified in the *file-set* list, it will be treated independently; any other partitions will be recovered only if specified individually in the *file-set* list. PARTONLY affects only partitioned files; its default setting is OFF.

This option is useful if you want to recover all files (and portions of files) residing on a particular disc volume, without recovering the related files on other volumes.

`PREFER`

specifies whether the rollforward process should, where it has a choice, use the tape copy or the disc copy of an online dump. The default strategy is to use the most recent valid copy, regardless of its medium; if a mirrored-volume backup exists, it will be mounted and, if any newer file versions exist on tape, these will be copied over the disc's files. If the disc is old enough to have very few most-recent files, you may want to specify PREFER TAPE and not mount the disc at all. On the other hand, you may have very few tape dumps more recent than their disc counterparts, in which case you may want to specify PREFER DISC and not mount the tape.

`SHUTDOWN SERIAL number`

specifies that recovery should proceed to the shutdown (STOP TMF) point identified by the specified SERIAL number, rather than to the most recent consistent state. When the STOP TMF command completes, the shutdown serial number is printed in a console message like this one:

```
12:30 02JUN82 FROM 000,00,21 LDEV 0026 CU %410 TMP: SHUTDOWN SERIAL 00000576
```

To recover to the shutdown point in this example, you would use *SERIAL number 576* with the RECOVER FILES command.

If you specify a *number* for which no shutdown record exists, the command fails and an error message is displayed.

The SHUTDOWN option is useful if you have stopped TMF before performing a batch operation and a failure occurs during the batch operation; you may prefer to recover to that shutdown point and repeat the batch operation.

If you use the SHUTDOWN option, you must also specify CRASHOPEN OFF in the same command.

Immediately after a recovery using the SHUTDOWN option, you should make a new online dump of all recovered files, because all of their previous audit trails will have become invalid.

Considerations

If your TMF installation is configured with an ALTER TMF, RECOVERY OFF command, then RECOVER FILES will be rejected with an error message.

If TMF does not find a required tape already mounted, it prompts the TMF operator to mount the tape.

Selection of files to be recovered differs substantially according to the CRASHOPEN option. To recover from a media failure (such as a head crash), you should use CRASHOPEN OFF. With this option, the system first searches the catalog for the specified *file-set* identifiers; it ignores the catalogued files that are specified in the NOT option and restores the rest.

Under the default setting, CRASHOPEN ON, the system

1. searches the actual disc
2. makes a list of audited files whose crash-open flags are set
3. removes from this list any files that were not named in the *file-set* list
4. applies the NOT option
5. recovers the files that remain.

If a file in the remaining *file-set* list has multiple partitions or alternate-key files associated with it, rollforward does not restore these associated files unless they are included in the remaining *file-set* list.

Selection of files to be recovered can also differ according to whether the online dump is on tape or disc. When recovering from discs, only the specified *file-sets* are recovered. When recovering from tapes, you have the ALTFILES and PARTONLY features whose defaults are to recover the partitions and alternate-key files associated with the specified *file sets*.

Each generation of an online dump provides a point from which the rollforward process can start. If the most recent dump is unusable for any reason, RECOVER FILES notes this in the catalog entry; if you then give another RECOVER FILES command, it will skip the defective dump and try the next previous one.

Remember that whenever you roll forward files on a disc online dump (unlike tape dumps), you are *modifying the actual dump.* If a system failure occurs during such a rollforward, the information on that disc will be damaged. The COPYDISC ON feature is a valuable protection against this damage, because it lets you keep a safe copy of the disc dump. If you specify COPYDISC OFF, the catalog entries for that disc will be invalidated because the rollforward process will be changing the data in the disc's files.

The COPYDISC feature requires that the new disc be labelled and that it not contain any valid files before the rollforward activity. If you want to insert a used disc, you should first use the PUP command LABEL to label the disc and to erase its directory. LABEL is described in the *GUARDIAN Operating System Utilities Manual* for your system.

See Section 3, "Media-Handling Responses," for valid responses to system prompts.

Examples

The command

```
RECOVER FILES $VOLA.*.*
```

uses the default disc-or-tape preference strategy (because PREFER was not used). If a disc online dump is used, TMF first copies the dump of $VOLA to a scratch pack (because COPYDISC OFF was not specified), then uses that copy to recover each audited file (on that volume) that was open when the crash occurred.

The recovery includes any partitioned-file segments on $VOLA that were open when the crash occurred, as well as any dumped alternate-key files on $VOLA that were open when the crash occurred. TMF prompts the operator to specify a tape drive and mount the appropriate tape on it.

The command

```
RECOVER FILES $VOLA.*.*, PARTOF *, NOT $VOLA.JOHN.OCT11, &
  CRASHOPEN OFF, OPERATOR OFF, DRIVE $TAPE1, &
  PREFER TAPE, SHUTDOWN SERIAL 576,
```

- recovers all partitioned files on disc volume $VOLA, except $VOLA.JOHN.OCT11, regardless of their crash-open flags
- excludes from the recovery any nonpartitioned files
- directs operator prompts to TMFCOM's home terminal
- expects the appropriate tape to be mounted on drive $TAPE1
- uses tape dumps, rather than disc dumps, wherever possible
- recovers only up to shutdown number 576.

REPORT DUMPS

The REPORT DUMPS command uses precompiled ENFORM queries to inspect the TMF catalog more thoroughly than the INFO commands permit. You can use this command regardless of whether TMF is active. The REPORT DUMPS command syntax is:

```
                                          , {BRIEF | DETAIL}
                                          , DUMPTYPE {ONLINE | AUDIT}
                                          , MEDIUM {DISC | TAPE}
REPORT [/OUT device/] DUMPS [ file ]    { , SERIAL number                    }
                                          , SORT {BYFILE | BYSERIAL}
                                          , STATUS {ASSIGNED | RELEASED}
```

where

`OUT device`

specifies the file or device to which the report will be written. If you specify a nonexistent file, an unstructured file will be created; if no volume and subvolume are specified, the file will be in TMFCOM's volume and subvolume. Output to remote systems or to the system spooler is permitted. The default is to use the currently active output device.

`file`

specifies the files whose dumps are to be the subject of this report. If you do not specify a file, information is reported on all dumps in the catalog; this is the default condition. Also, you can substitute an asterisk for any one, two, or three elements of the file identifier to specify more than one file.

`BRIEF`

generates a narrow report (fewer than 80 columns wide), including the name of the dumped file, the tape reel number, and the serial number, date, time, and type of the dump. BRIEF is the default report format and is the more suitable format for listing to a terminal.

`DETAIL`

generates a wide report (fewer than 132 columns wide), including all fields in the BRIEF report plus the monitor (or master) and data (or auxiliary) audit trails' sequence numbers, the medium (disc or tape), and the part number and copy number. This format is the more suitable for listing on a printer.

`DUMPTYPE`

includes in the report only those dumps that are of the specified type. The default is to include both audit and online dumps.

`MEDIUM`

includes in the report only those dumps that reside on the specified medium. The default is to include both tape and disc dumps.

→

`SERIAL number`

includes in the report only those dumps whose serial number is identical to the one specified. The default is to include all dumps, regardless of serial number.

`SORT BYFILE`

specifies that the selected dumps are to be listed alphabetically by the names of the dumped files. This is the default order.

`SORT BYSERIAL`

specifies that the selected dumps are to be listed in descending numerical order of their serial numbers. SORT BYFILE is the default order.

`STATUS`

includes in the report only those dumps that are of the specified status. The default is to include both assigned and released dumps. STATUS refers to the status of the entire dump, not that of the individual reel or disc.

Considerations

Because audit dumps are written to tape only, combining the DUMPTYPE AUDIT and MEDIUM DISC options will always produce a blank report.

REPORT DUMPS may not report the latest catalog information if another TMF process is currently updating the catalog. As soon as that updating is complete, a new REPORT DUMPS command will provide the latest information.

Examples

The command

```
REPORT DUMPS
```

generates an 80-column report showing all catalogued dump information, sorted by file names, as shown here:

```
                      list of audit and online dumps
                    medium: tape and disc    status: any

          file name          serial #     date      time      type    reel
------------------------  ----------  ------------------  ------  ------
$AUDIT  SALES   AT001739        2720  08/27/84  13:08:02   audit  aud162
                                                                  aud163
$DATA   RECEIPT DEC29           2750  08/27/84  14:13:38  online  pack19
$DATA   SALES   DEC11           2700  08/27/84  10:02:29  online  old173
                                                                  old174
$DATA   SALES   DEC21           2740  08/27/84  12:05:20  online  old220
                                                                  old221
                                                                  old222
                                                                  old223
$DATA   SALES   DEC31           2755  08/27/84  11:03:42  online  old236
                                                                  old237
```

The command

```
REPORT /OUT $S.#PRINTER/ DUMPS $DATA.SALES.*, DUMPTYPE ONLINE, DETAIL
```

generates a 132-column report showing catalogued information about online dumps of all files in subvolume $DATA.SALES and spools that report to location #PRINTER. The report will be similar to this:

```
                                    list of audit and online dumps
                                  medium: tape and disc    status: any

         file name          serial #     date       time      type   matsn  fatsn  medium  status     reel   part  copy
--------------------------  ----------  ------------------  ------  -----  -----  ------  ---------  ------  ----  ----
$DATA    SALES    DEC11           2700  08/27/84  10:02:29  online      4      1   tape   assigned   old173     1     1
                                                                                                     old174     2     1
$DATA    SALES    DEC21           2740  08/27/84  12:05:20  online      5      2   tape   assigned   old220     1     1
                                                                                                     old221     1     2
                                                                                                     old222     2     1
                                                                                                     old223     2     2
$DATA    SALES    DEC31           2755  08/27/84  11:03:42  online      6      3   tape   assigned   old236     1     1
                                                                                                     old237     2     1
```

Notice that RECEIPT.DEC29 is omitted because it is not part of the $DATA.SALES.* file set. Also, file $AUDIT.SALES.AT001739 is omitted for two reasons: it is not an online dump and it is not part of the specified file set.

In the DETAIL report, the "matsn" column shows the monitor (or master) audit trail's serial number at the time the dump was made; similarly, the "fatsn" column shows the data (or auxiliary) audit trail's serial number.

The command

```
REPORT /OUT MARY.REPTFILE/ DUMPS ($AUDIT.*.*, $DATA.*.*), DETAIL, SORT BYSERIAL
```

generates a 132-column report showing catalogued information about all dumps of all files on volume $DATA, sorted by dump serial numbers, and writes the report to a disc file MARY.REPTFILE on TMF-COM's volume. The report will be similar to this:

```
                                    list of audit and online dumps
                                  medium: tape and disc    status: any

         file name          serial #     date       time      type   matsn  fatsn  medium  status     reel   part  copy
--------------------------  ----------  ------------------  ------  -----  -----  ------  ---------  ------  ----  ----
$DATA    SALES    DEC31           2755  08/27/84  11:03:42  online      6      3   tape   assigned   old236     1     1
                                                                                                     old237     2     1
$DATA    RECEIPT  DEC29           2750  08/27/84  14:13:38  online      4      3   disc   assigned   pack19     1     1
$DATA    SALES    DEC21           2740  08/27/84  12:05:20  online      5      2   tape   assigned   old220     1     1
                                                                                                     old221     1     2
                                                                                                     old222     2     1
                                                                                                     old223     2     2
$DATA    SALES    DEC11           2700  08/27/84  10:02:29  online      4      1   tape   assigned   old173     1     1
                                                                                                     old174     2     1
```

RESET AUDITDUMP

RESET AUDITDUMP restores the default values of audit-dump attributes that you have established previously by one or more SET AUDITDUMP commands. The RESET AUDITDUMP command syntax is:

```
                 ┌ ┌ COPIES ┐ ┌, COPIES┐     ┐
                 │ │ CPUS   │ │, CPUS  │     │
                 │ │ DRIVE  │ │, DRIVE │ ... │
RESET AUDITDUMP  │ │ OUT    │ │, OUT   │     │
                 │ └ PRI    ┘ └, PRI   ┘     │
                 │                           │
                 └   *                       ┘
```

where

COPIES

resets the number of copies to be made of each dump to the default, which is 1.

CPUS

removes any previous specification of which primary and backup processors are to be used for the audit-dump process. The default is for the TMP to select it from among the available processors.

DRIVE

removes any previous specification of which tape drives the audit-dump process will use for dumping audit-trail files.

OUT

specifies that no audit-dump report will be produced.

PRI

removes any previously specified priority to be given to audit-dump processes. The default is for the process to assume TMFCOM's priority.

*

resets all of the attributes listed above.

Examples

The command

```
RESET AUDITDUMP PRI, OUT
```

specifies that each subsequently defined audit dump, unless established differently by its ADD AUDITDUMP command, will take its priority from TMFCOM and will generate no report.

The command

```
   RESET AUDITDUMP *
```

resets all default audit-dump attributes.

RESET AUDITTRAIL

RESET AUDITTRAIL restores the default values of audit-trail attributes that you have established previously by one or more SET AUDITTRAIL commands. The RESET AUDITTRAIL command syntax is:

```
                   ┌ ┌ AUDITDUMP ┐ ┌ , AUDITDUMP ┐     ┐
                   │ │ EXT       │ │ , EXT       │...  │
                   │ { MAXFILES  } │ , MAXFILES  │     │
RESET AUDITTRAIL   │ │ MINFILES  │ │ , MINFILES  │     │
                   │ └ USAGE     ┘ └ , USAGE     ┘     │
                   │                                   │
                   └   *                               ┘
```

where

AUDITDUMP

removes any audit-dump process assignments or AUDITDUMP NONE specifications that may have been established by previous SET AUDITTRAIL commands.

EXT

restores the default values to the primary and secondary extents of files in an audit-trail sequence.

MAXFILES

restores the maximum number of files that can be maintained concurrently in any audit-trail sequence to its default value, which is either 1 plus the current MINFILES value, or 4, whichever is greater.

MINFILES

restores the number of files that a transaction is allowed to span to its default value, 2.

USAGE

removes any USAGE attribute set by a previous SET AUDITTRAIL command. This attribute, when set, determines whether subsequent applicable TMFCOM commands would apply to monitor or master audit trails or to a specified set of data audit trails.

*

resets all of the attributes listed above.

Examples

The command sequence

```
SET AUDITTRAIL, MAXFILES 10
ADD AUDITTRAIL $AUDIT.AP.MN
.
.
RESET AUDITTRAIL MAXFILES
```

defines an audit trail with a MAXFILES value of 10, then resets the creation value for MAXFILES to its default setting.

The command

```
RESET AUDITTRAIL *
```

resets all audit-trail creation attributes.

RESET DUMPS

The RESET DUMPS command restores default values to any catalogable attributes that you have established previously by one or more SET DUMPS commands. The RESET DUMPS command syntax is:

```
              ┌ ┌MEDIUM┐ ┌, MEDIUM┐ ┐
              │ {SERIAL} │, SERIAL│ │
RESET DUMPS   │ └TYPE  ┘ └, TYPE  ┘ │
              │                     │
              └  *                  ┘
```

where

`MEDIUM`

removes any previous specification of tape-reel or disc-pack identifiers for subsequently catalogued dumps.

`SERIAL`

removes any previous specification of a dump serial number for subsequently catalogued dumps.

`TYPE`

removes any previous specification of the type of files being catalogued (audit dump or online dump).

`*`

resets all of the attributes listed above.

Considerations

The considerations for ADD DUMPS apply to RESET DUMPS.

RESET MEDIA

The RESET MEDIA command restores default values to any catalogable attributes that you have established previously by one or more SET MEDIA commands. This command replaces the former RESET TAPES command, but TMFCOM still recognizes RESET TAPES and interprets it as RESET MEDIA. The RESET MEDIA command syntax is:

```
               [ { DRIVE    } [ , DRIVE    ]     ]
               [ { LABEL    } [ , LABEL    ]     ]
               [ { OPERATOR } [ , OPERATOR ]     ]
RESET MEDIA    [ { RELABEL  } [ , RELABEL  ] ... ]
               [ { STATUS   } [ , STATUS   ]     ]
               [ { TYPE     } [ , TYPE     ]     ]
               [                                 ]
               [   *                             ]
```

where

`DRIVE`

removes any previous specification of which tape drives the tapes are to be mounted on.

`LABEL`

returns the LABEL setting to its default, which is LABEL OFF.

`OPERATOR`

removes any previous OPERATOR OFF/ON specification. If a TMF operator has been designated, the default is OPERATOR ON. Otherwise, the default is OPERATOR OFF.

`RELABEL`

returns the RELABEL setting to its default, which is RELABEL OFF.

`STATUS`

removes any previous STATUS specification.

`TYPE`

removes any previous TYPE specification.

`*`

resets all of the attributes listed above.

Considerations

The considerations for ADD MEDIA apply to RESET MEDIA.

SET AUDITDUMP

SET AUDITDUMP establishes creation values for audit-dump attributes. Once such a value has been set, a subsequent use of ADD AUDITDUMP (within the same invocation of TMFCOM) will add to the system an audit-dump process with those attribute values. The attributes that can be set are:

- the execution priority of the audit-dump process
- the primary and backup processors where the audit-dump process is to run
- how many copies are to be made of each dump
- the output file for the audit-dump report.

The SET AUDITDUMP command syntax is:

```
SET AUDITDUMP [ LIKE { AUDITDUMP  [process] } ]
                     { process              }

  [ , COPIES quantity                               ]
  [          { cpu1             }                   ]
  [ , CPUS   { (cpu1)           }                   ]
  [          { [cpu1]:cpu2      }                   ]  ...
  [          { ([cpu1]:cpu2)    }                   ]
  [ , DRIVE  { device                          }    ]
  [          { ( device [ , device ]... )      }    ]
  [ , OUT file                                      ]
  [ , PRI integer                                   ]
```

where

`LIKE`

establishes, for all attributes, new creation values equal to those of a currently defined audit-dump process. If you use the LIKE clause, it must be the first option in the command. It cannot appear twice in the same SET AUDITDUMP command. If you do not use the LIKE clause, no comma (,) is allowed before the first option.

and all other options are as described for the ADD AUDITDUMP command.

Considerations

The attribute values established through a SET AUDITDUMP command do not persist beyond the execution of TMFCOM in which the SET AUDITDUMP was issued.

Examples

The command sequence

```
SET AUDITDUMP PRI 142, CPUS 9:12, COPIES 2
ADD AUDITDUMP $AUD1
```

defines audit-dump process $AUD1 with

- an execution priority of 142
- processors 9 and 12 to be the primary and backup processors, respectively
- two copies to be made of each dump
- the operator to be prompted when $AUD1 needs a tape name (assuming DRIVE has not been set by a previous SET AUDITDUMP command)
- no report.

The command sequence

```
SET AUDITDUMP DRIVE $TAPE1, OUT $DATA.TMF.DUMPREPT.
ADD AUDITDUMP $AUD2
```

defines audit-dump process $AUD2 with

- $TAPE1 as the tape drive to which audit-trail files will be dumped (one copy)
- a report to be written to disc file $DATA.TMF.DUMPREPT.

For each audit-dump, the report is similar to this:

```
AUDITDUMP - T9066B00 - (28JAN85)   System \SAB    20-Feb-1985 19:47:18
[Dump serial number is 75.]
File Name                          Error
[Tape JER234 used for copy 1 of reel 1 on drive $TAPE.]
$AUDIT.AP.AA000132
[Total files dumped = 1.]
```

This example shows that audit-trail file $AUDIT.AP.AA000132 was dumped to the tape labeled JER234, using drive $TAPE, at 7:47 p.m. on April 20, and the serial number of that dump is 375.

SET AUDITTRAIL

SET AUDITTRAIL establishes creation values for audit-trail attributes. Once an attribute value has been set, any subsequent use of ADD AUDITTRAIL (within the same invocation of TMFCOM) will add to the system an audit trail with those attribute values, except where an attribute has been reset by the RESET AUDITTRAIL command or explicity specified in the ADD AUDITTRAIL command. The attributes that can be set are:

- the name or logical device number of each audit process or disc process that will write to the audit trails defined by a subsequent use of the ADD AUDITTRAIL command
- the primary and secondary extent size of each file in an audit-trail sequence defined by a subsequent use of the ADD AUDITTRAIL command
- the maximum number of audit-trail files that a single active transaction can span before becoming eligible for automatic abortion
- the maximum number of files in the sequence that will be maintained concurrently on disc
- how the audit trails are to be used.

The attributes configured through a SET AUDITTRAIL command do not persist beyond the execution of TMFCOM in which the SET AUDITTRAIL was issued.

The SET AUDITTRAIL command syntax is:

```
SET AUDITTRAIL [ LIKE { AUDITTRAIL [ generic-file-set ] } ]
               [      { generic-file-set               } ]

  [ , AUDITDUMP { AUTOMATIC [ process ] }                           ]
  [             { NONE                  }                           ]
  [ , EXT { primary-extent                               }          ]
  [       { ( primary-extent [ ,secondary-extent ] )     }          ] ...
  [ , MAXFILES integer                                              ]
  [ , MINFILES integer                                              ]
  [           { {AUDITPROCESS} { process                        } } ] (I only)
  [           { {DISCPROCESS } { ( process [ , process ]... )   } } ] (II/DP1)
  [ , USAGE   { {AUXILIARY   } { *                              } } ] (II/DP2)
  [           {  MASTER                                           } ] (II/DP2)
  [           {  MONITOR                                          } ] (DP1)
```

where

`LIKE`

establishes, for all attributes, new creation values equal to those of an existing audit trail. If you use the LIKE clause, it must be the first option in the command. It cannot appear twice in the same SET AUDITTRAIL command. If you do not use the LIKE clause, no comma (,) is allowed before the first option.

and all other options are as described for the ADD AUDITTRAIL command.

Considerations

The attribute values established through a SET AUDITTRAIL command do not persist beyond the execution of TMFCOM in which the SET AUDITTRAIL was issued.

Examples

These examples are for a NonStop system, but they apply equally well to a NonStop 1+ system if you substitute AUDITPROCESS for DISCPROCESS.

The command sequence

```
SET AUDITTRAIL USAGE MONITOR
ADD AUDITTRAIL $AUDIT.AP.MN
```

configures the monitor audit trail, named $AUDIT.AP.MN.

The command sequence

```
SET AUDITTRAIL EXT (20,20), MINFILES 3
ADD AUDITTRAIL $AUDIT.AP.AA, USAGE DISCPROCESS $VOL0, MAXFILES 5
SET AUDITTRAIL LIKE $AUDIT.AP.AA
ADD AUDITTRAIL $AUDIT.AP.AB, USAGE DISCPROCESS *
```

configures

- two disc-process audit trails named $AUDIT.AP.AA and $AUDIT.AP.AB, one assigned to disc process $VOL0 and the other assigned to all other disc processes
- primary and secondary extent sizes of 40 kilobytes (20 pages of 2,048 bytes each) for each file in the audit-trail sequences
- a limit of three files to be spanned on each of the audit-trail sequences by an active transaction before the transaction is eligible for abortion
- a maximum limit of five files in each sequence that can be maintained concurrently on the disc.

If a similar command sequence is given on a NonStop 1+ system, all SYSGEN-configured audit processes, with the exception of $VOL0, will write to $AUDIT.AP.AA and audit process $VOL0 will write to $AUDIT.AP.AB.

SET DUMPS

The SET DUMPS command establishes new creation values for catalog entries of dumps, for use by subsequent ADD DUMPS commands (within the same invocation of TMFCOM). The only use of ADD DUMPS should be for manual recovery following certain catalog failures. The SET DUMPS command syntax is:

```
SET DUMPS

/   {BRIEF | DETAIL}                                  \
|          /DISC pack                                \ |
|   MEDIUM{TAPE{reel                                 }|
<          \    {( reel [ , reel ] ... )             / >
|   SERIAL number                                      |
|   TYPE  {AUDITDUMP                                 } |
\         {ONLINEDUMP ( audit, monitor )             } /

[   , {BRIEF | DETAIL}                                  ]
|             /DISC pack                                |
|   , MEDIUM{TAPE  {reel                             }} |
|             \     {( reel [ , reel ]...)           }} |  ...
|   , SERIAL number                                     |
|   , TYPE  {AUDITDUMP                               }  |
[           {ONLINEDUMP ( audit, monitor )           }  ]
```

where all options are as described for the ADD DUMPS command.

SET MEDIA

The SET MEDIA command establishes new creation values for catalog entries of tapes and discs, for use by subsequent ADD MEDIA commands (within the same invocation of TMFCOM). These commands can be useful if you have to rebuild a catalog manually (see also SET DUMPS and ADD DUMPS). This command replaces the former SET TAPES command, but TMFCOM still recognizes SET TAPES and interprets it as SET MEDIA. The SET MEDIA command syntax is:

```
            ( {BRIEF | DETAIL}                               )
            | DRIVE  {device                       }         |
            |        {( device [ , device ]... )   }         |
            | LABEL {OFF | ON}                               |
            |                                                |
SET MEDIA  <  OPERATOR {OFF | ON}                             >
            | RELABEL {OFF | ON}                             |
            |          {ASSIGNED}                            |
            | STATUS   {BAD     }                            |
            |          {RELEASED}                            |
            |          {SCRATCH }                            |
            ( TYPE {DISC | TAPE}                             )

   [ , {BRIEF | DETAIL}                          ]
   [ , DRIVE  {device                      }     ]
   [          {( device [ , device ]... )  }     ]
   [ , LABEL {OFF | ON}                          ]
   [ , OPERATOR {OFF | ON}                       ]  ...
   [ , RELABEL {OFF | ON}                        ]
   [           { ASSIGNED }                      ]
   [ , STATUS  { BAD      }                      ]
   [           { RELEASED }                      ]
   [           { SCRATCH  }                      ]
   [ , TYPE {DISC | TAPE}                        ]
```

where all options are as described for the ADD MEDIA command.

SHOW AUDITDUMP

SHOW AUDITDUMP displays the creation attributes for audit-dump processes, as established by the SET AUDITDUMP command. The SHOW AUDITDUMP command syntax is:

```
               ┌ ┌ COPIES ┐ ┌, COPIES┐    ┐
               │ │ CPUS   │ │, CPUS  │    │
               │ │ DRIVE  │ │, DRIVE │... │
SHOW AUDITDUMP │ { OUT    } │, OUT   │    │
               │ │ PRI    │ │, PRI   │    │
               └ └ *      ┘ └        ┘    ┘
```

where

COPIES

displays the current creation value for the number of copies to be made of each dump (if a value has been set).

CPUS

shows which primary and backup processors are to be used for the audit-dump process (if any have been set).

DRIVE

shows the current creation value for the tape drives for audit-dump processes to use in dumping audit-trail files (if any have been set).

OUT

shows whether the current creation value is for the audit-dump process to produce a report and, if so, shows the report's file identifier.

PRI

displays the current creation value (if one has been set) for the priority to be given to audit-dump processes.

*

displays all of the attributes listed above.

Considerations

This command displays only those creation values that have been established by SET AUDITDUMP and that have not been reset by RESET AUDITDUMP.

SHOW AUDITTRAIL

SHOW AUDITTRAIL displays the audit-trail creation attributes that you have established by the SET AUDITTRAIL command. The SHOW AUDITTRAIL command syntax is:

```
                  [ { AUDITDUMP }  [, AUDITDUMP]     ]
                  [ { EXT       }  [, EXT      ]     ]
                  [ { MAXFILES  }  [, MAXFILES ] ... ]
SHOW AUDITTRAIL   [ { MINFILES  }  [, MINFILES ]     ]
                  [ { USAGE     }  [, USAGE    ]     ]
                  [   *                              ]
```

where

`AUDITDUMP`

displays any audit-dump process assignment or AUDITDUMP NONE specification that has been established by a previous SET AUDITTRAIL command.

`EXT`

displays the current creation values for the primary and secondary extent sizes of audit-trail files.

`MAXFILES`

displays the current creation value for the maximum number of files that can be maintained concurrently on the disc for any audit-trail sequence.

`MINFILES`

displays the current creation value for the the number of files that a transaction can span in any audit-trail sequence without being aborted.

`USAGE`

displays the current USAGE attribute default, if one has been set. See the description of this attribute for the ADD AUDITTRAIL command.

`*`

displays all of the creation values listed above. A question mark (?) is displayed for any required creation value (such as USAGE) that has not been set.

Considerations

This command displays only those creation values that have been established by SET AUDITTRAIL and that have not been reset by RESET AUDITTRAIL.

SHOW DUMPS

The SHOW DUMPS command displays the current creation values for catalog entries of dumps, as established by the SET DUMPS command. The SHOW DUMPS command syntax is:

```
             ┌ ┌ MEDIUM ┐ ┌ , MEDIUM ┐     ┐
             │ { SERIAL } │ , SERIAL │ ... │
SHOW DUMPS   │ └ TYPE   ┘ └ , TYPE   ┘     │
             └   *                         ┘
```

where

MEDIUM

displays any previously set tape-reel or disc-pack identifiers.

SERIAL

displays the dump serial number previously set (if any).

TYPE

displays any previous specification of the type of files being catalogued (audit dump or online dump).

*

displays all of the attributes listed above.

Considerations

The considerations for ADD DUMPS apply to SHOW DUMPS.

SHOW MEDIA

The SHOW MEDIA command displays the current creation values for catalog references of tapes and discs, as established by the SET MEDIA command. This command replaces the former SHOW TAPES command, but TMFCOM still recognizes SHOW TAPES and interprets it as SHOW MEDIA with TYPE TAPE. The SHOW MEDIA command syntax is:

```
              ┌ ⎧ DRIVE    ⎫ ┌, DRIVE    ┐     ┐
              │ ⎪ LABEL    ⎪ │, LABEL    │     │
              │ ⎨ OPERATOR ⎬ │, OPERATOR │     │
SHOW MEDIA    │ ⎪ RELABEL  ⎪ │, RELABEL  │...  │
              │ ⎪ STATUS   ⎪ │, STATUS   │     │
              │ ⎩ TYPE     ⎭ └, TYPE     ┘     │
              └   *                            ┘
```

where

`DRIVE`

displays any previous specification of which tape drives the tapes are to be mounted on.

`LABEL`

displays the LABEL setting.

`OPERATOR`

displays any previous OPERATOR OFF/ON specification.

`RELABEL`

displays the RELABEL setting.

`STATUS`

displays any previous STATUS specification.

`*`

displays all of the attributes listed above.

`TYPE`

displays any previous TYPE specification.

Considerations

The considerations for ADD MEDIA apply to SHOW MEDIA.

START TMF

START TMF activates TMF after it has been configured or after it has been shut down. TMF transaction processing cannot begin until START TMF has been issued. You can, however, start TMF without permitting transaction processing; this starts the various TMF processes so you can use most of the operational, catalog, dumping, and recovery commands. The START TMF command syntax is:

```
START TMF [ , TRANSACTION {OFF | ON}                               ]
          [              { volume                          }       ]
          [ , VOLUMES    { ( volume [ , volume ]... )      }       ]  (II only)
          [              { *                               }       ]
          [        { volume                        }               ]
          [ , NOT  { ( volume [ , volume ]... )    }               ]  (II only)
          [        { *                             }               ]
```

where

`TRANSACTION OFF`

disallows TMF transaction processing. The default setting is TRANSACTION ON. Using START TMF with TRANSACTION OFF is precisely equivalent to following START TMF immediately with a STOP TRANSACTION command.

`TRANSACTION ON`

allows transaction processing. This is the default setting.

`VOLUMES` (II only)

limits the disc volumes for which transaction processing is to be enabled. If VOLUMES is used, the START TMF command applies to all volumes in its *volume* list except those in the NOT list. If VOLUMES is not used, the START TMF command applies to all volumes in the system except those in the NOT list.

`volume`

is a disc volume name, such as $DATA.

`NOT` (II only)

further limits the disc volumes for which transaction processing is to be enabled. The START TMF command applies to all volumes in the VOLUMES list except those in the NOT list. If VOLUMES is not used, the START TMF command applies to all volumes in the system except those in the NOT list.

Considerations

On a NonStop system, START TMF automatically performs autorollback for any implicitly or explicitly specified volume that requires it.

START TMF will fail in these situations:

- The monitor audit trail has not been configured.
- Audit trails have not been defined for all audit processes (on a NonStop 1+ system) or disc processes (on a NonStop system).
- An audit trail is configured for automatic dumping and no operator has been configured (by the ALTER TMF command).
- An audit trail is configured for automatic dumping and an ALTER TMF, RECOVERY OFF command has been given.
- TMF is already running (that is, START TMF has already been issued) or a STOP TMF command is in progress.
- A failure has occurred that requires you to cold-load the system before restarting TMF and recovering the crashed files.

Even if START TMF does not fail, some disc files on a NonStop system may remain unavailable for TMF transaction processing. In some situations, this is because autorollback is insufficient for the specific problem and rollforward must be used (see RECOVER FILES). Situations requiring rollforward include:

- A media failure, such as a disc drive's head crash, has occurred.
- A crashed file has failed a previous attempt at autorollback.

Some other possible reasons for a volume's unavailability after START TMF include:

- You used the NOT *volume* option with START TMF.
- Unresolved network transactions, involving audited files on this volume, remain.
- A disc controller failure has occurred.
- The data volume's related audit-trail volume is unavailable.

After correcting the underlying problem, you can use the ENABLE VOLUMES command to make the volume available.

Examples

The command

```
START TMF, TRANSACTION OFF
```

enables TMF operations such as online dumps, audit-trail dumps, rollforward, or STATUS commands without enabling transaction processing.

The command

```
START TMF, VOLUMES *, NOT $DATA
```

or

```
START TMF, NOT $DATA
```

enables transaction processing on all volumes in the system except $DATA. All transactions affecting any audited file on $DATA will return file-system error 82 until $DATA has been enabled. If any file on $DATA needs autorollback recovery, you will have to use the ENABLE VOLUMES command.

The command

```
START TMF, VOLUMES ($DATA, $ACCT, $SALES)
```

enables transaction processing for audited files on all three of the specified volumes. If the system contains any other disc volume, all transactions affecting any audited file on that volume will return file-system error 82 until that volume has been started by an ENABLE VOLUMES command.

START TRANSACTION

START TRANSACTION allows TMF transaction processing to begin. This command is used after a START TMF command with the TRANSACTION OFF option or a STOP TRANSACTION command. The START TRANSACTION command syntax is simply:

```
START TRANSACTION
```

with no options.

Considerations

While the rollforward process is recovering data from a crash, new transactions can manipulate the files that were not affected by the crash. Similarly, TMF can record online dumps while transactions are manipulating the data base. This concurrent transaction activity, however convenient, can slow both the recovery (or dumping) and the transaction processing. In some circumstances, it can be more efficient to disable transaction processing until the recovery or dumping is complete. If you give either a START TMF command with the TRANSACTION OFF option or a STOP TRANSACTION command, no transactions can begin, although all other TMF functions remain enabled. From this state, START TRANSACTION reenables processing of transactions.

STATUS AUDITDUMP

STATUS AUDITDUMP displays the current status of one or more audit-dump processes. For each process that is actively dumping an audit trail, TMF displays

- the copy number and file identifier of the dump in progress
- the tape drive and reel to which the dump is being written
- how many files have been dumped to the current tape reel (including the current file)
- how many I/O blocks of the current file (including the current block) have been written.

The STATUS AUDITDUMP command syntax is:

```
                 { process                        }
STATUS AUDITDUMP { ( process [ , process ]... )   } [ ,{BRIEF | DETAIL} ]
```

where

`process`

names an audit-dump process, on the local system, whose status you want displayed.

`*`

displays the status of all audit-dump processes on the local system.

`BRIEF` or `DETAIL`

determines how much information is to be displayed. BRIEF displays the identifier, number of copies, WAIT status, tape drive (if any), and current activity for each audit-dump process. For each process that is not idle, it also tells the current activity's progress. DETAIL also lists CPUs and priority of each process. If you do not specify either BRIEF or DETAIL, TMF displays the detailed report.

Examples

If audit-dump process $ADP0 is in the process of dumping a file to tape drive $TAPE, the command

```
STATUS AUDITDUMP $ADP0, DETAIL
```

provides a display similar to this:

```
AUDITDUMP PROCESSES:
    AUDITDUMP $ADP0, CPUS (0:1), PRI 148, COPIES 1, WAIT OFF,
         OUT $DATA.TMF.DUMPRPT, DRIVE ($TAPE )
         CURRENT ACTIVITY: DUMPING AN AUDITTRAIL
         TRANSFER STATUS:
              TAG OF REQUEST: 0
              NUMBER OF COPY BEING REPORTED ON: 1
              FILE BEING TRANSFERED: $AUDIT.AP.AA000027
              DRIVE BEING USED: $TAPE
              REEL BEING USED: JER456
              FILE NUMBER ON REEL: 1
              BLOCK NUMBER ON REEL: 2
```

In this example, process $ADP0 is currently dumping the only copy of audit-trail file $AUDIT.AP.AA000027, using tape JER456 on drive $TAPE. This is the only file written on that tape, and the process is currently writing the second I/O block. The "TAG OF REQUEST" line can be ignored.

The command

```
STATUS AUDITDUMP $ADP0, BRIEF
```

provides the same display, but without the CPUs and priority.

STATUS AUDITTRAIL

STATUS AUDITTRAIL displays the sequence numbers and configuration attributes of selected audit trails. The STATUS AUDITTRAIL command syntax is:

```
STATUS AUDITTRAIL { generic-file-set                                 }
                  { ( generic-file-set [ , generic-file-set ]... )   }

  [ , {BRIEF | DETAIL} ]
```

where

`generic-file-set`

identifies the sequence of files that form an audit trail. The identifier is defined in the "Generic File-Set Identifiers" subsection near the beginning of this section.

With this command, you can substitute an asterisk (*) for one or more elements of the *generic file-set* identifier, as with standard GUARDIAN file identifiers.

Each *generic file-set* identifier must have been specified in a previous ADD AUDITTRAIL command.

`BRIEF` or `DETAIL`

determines how much information is to be displayed. BRIEF displays the generic file names of the audit trails in the *generic file-set* list. DETAIL also displays the sequence number of each current audit trail and all audit-trail configuration attributes. If you do not specify either BRIEF or DETAIL, TMF displays the detailed report.

Examples

The command

```
STATUS AUDITTRAIL *.*.*, BRIEF
```

displays the names of all audit trails in the system, in this format:

```
AUDITTRAILS:
    AUDITTRAIL $AUDIT.AP.AA
    AUDITTRAIL $AUDIT.AP.MN
    AUDITTRAIL $DATA.JERAUDIT.AA
```

The command

```
STATUS AUDITTRAIL $AUDIT.AP.*, DETAIL
```

displays information for all audit trails belonging to volume $AUDIT and subvolume AP, in this format:

```
AUDITTRAILS:
    0 AUDITTRAIL $AUDIT.AP.MN, EXT (350,250), MINFILES 2,
         MAXFILES 4, AUDITDUMP AUTOMATIC $ADPO, USAGE MONITOR,
         SEQUENCE NUMBER 000003
    1 AUDITTRAIL $AUDIT.AP.AA, EXT (800,500), MINFILES 2,
         MAXFILES 4, AUDITDUMP AUTOMATIC $ADPO, USAGE DISCPROCESS *,
         SEQUENCE NUMBER 000014
```

Among other data in this display, we can see that the current file of the monitor audit trail is $AUDIT.AP.MN000003 and the current file of the data audit trail is $AUDIT.AP.AA000014.

The numbers preceding each audit trail's information (0 and 1 in this example) are used in certain console messages to identify the audit trails. For instance, if audit trail $AUDIT.AP.AA has filled MAXFILES – 1 files, TMF prohibits processing of new transactions for that audit trail and displays this console message:

```
...TMF SUSPENDED BEGIN-TRANSACTIONS: AUDIT TRAIL 1 AT 'MAXFILES'.
```

Then, when a file is purged, a similar message notifies the operator that the suspension is lifted.

This example is for a NonStop system, but it applies equally well to a NonStop 1+ system with AUDITPROCESS substituted for DISCPROCESS in the audit-trail usage.

STATUS BACKOUT

STATUS BACKOUT displays the name, execution priority, and processors for the backout process. The STATUS BACKOUT command syntax is:

```
STATUS BACKOUT [ , {BRIEF | DETAIL} ]
```

where

`BRIEF` or `DETAIL`

determines how much information is to be displayed. BRIEF displays only the name of the backout process. DETAIL displays the name, execution priority, and processors of the backout process. If you do not specify either BRIEF or DETAIL, TMF displays the detailed report.

Examples

The command

```
STATUS BACKOUT, BRIEF
```

provides a display similar to this:

```
BACKOUT PROCESS:
    BACKOUT $Z063
```

The command

```
STATUS BACKOUT, DETAIL
```

provides a display similar to this:

```
BACKOUT PROCESS:
    BACKOUT $Z063, CPUS (0:1), PRI 148
```

STATUS CATALOG

STATUS CATALOG displays the current status of the catalog. The STATUS CATALOG command syntax is:

```
STATUS CATALOG [ , {BRIEF | DETAIL } ]
```

where

`BRIEF` or `DETAIL`

determines how much information is to be displayed. BRIEF displays the catalog process's name, RETAINDEPTH and RELABEL attributes, and status, plus the number of requesters that have it open and how much disc space is allocated for it. DETAIL adds to this the process's CPU numbers and priority. If you do not specify either BRIEF or DETAIL, TMF displays the detailed report.

Examples

The command

```
STATUS CATALOG, DETAIL
```

provides a display similar to this:

```
TMF CATALOG:
    CATALOG $Z062, CPUS (0:1), PRI 148, RETAINDEPTH 3, RELABEL ON
    THE CATALOG PROCESS IS ACTIVELY PROCESSING A REQUEST OR DUMP.
    NUMBER OF ACTIVE REQUESTORS: 4
    TOTAL NUMBER OF DISC BYTES USED: 30784
```

When the report says the catalog process is "UP", it is alive and well but idle.

The command

```
STATUS CATALOG, BRIEF
```

provides the same display, but without the CPUs and priority.

STATUS TMF

STATUS TMF displays the configuration attributes of audit trails and of the backout, audit-dump, and catalog processes. It also tells whether TMF is currently started, stopped, or changing from one state to the other. The STATUS TMF command syntax is:

```
STATUS TMF [ , {BRIEF | DETAIL} ]
```

where

`BRIEF` or `DETAIL`

determines how much information is to be displayed. If you do not specify either BRIEF or DETAIL, TMF displays the detailed report. For each process and for the audit trails, the amount of information displayed is as described for the STATUS AUDITDUMP, STATUS AUDITTRAIL, STATUS BACKOUT, and STATUS CATALOG commands.

Considerations

The report could show TMF in any one of these states:

- *ACTIVE FOR TRANSACTION PROCESSING* is the normal state when TMF has been started.
- *STOPPED* is the normal state when TMF has not been started or has been stopped.
- *STARTING* is the state if a START TMF command is in progress.
- *STOPPING* is the state if a STOP TMF command is in progress.
- *STOPPING TRANSACTION PROCESSING* is the state if a STOP TRANSACTION command is in progress.
- *ACTIVE. BEGIN TRANSACTION IS DISABLED* is the state if a STOP TRANSACTION command has completed.

Examples

The command

```
STATUS TMF, BRIEF
```

provides a display similar to this:

```
TMF STATUS FOR SYSTEM \SAA AT 6-Mar-1981 14:00:58
    THE TMF AUDIT SUBSYSTEM IS ACTIVE FOR TRANSACTION PROCESSING.
    CURRENT CRASH COUNT FOR THIS SYSTEM IS 1
BACKOUT PROCESS:
    BACKOUT $Z007
AUDITTRAILS:
    AUDITTRAIL $AUDIT.AP.MN
    AUDITTRAIL $AUDIT.AP.AA
AUDITDUMP PROCESSES:
    AUDITDUMP $ADP0, COPIES 1, WAIT OFF
         CURRENT ACTIVITY: IDLE
TMF CATALOG:
    CATALOG $Z062, RETAINDEPTH 3, RELABEL ON
    THE CATALOG PROCESS IS UP.
    NUMBER OF ACTIVE REQUESTORS: 4
    TOTAL NUMBER OF DISC BYTES USED: 30784
```

The command

```
STATUS TMF, DETAIL
```

provides a display similar to this:

```
TMF STATUS FOR SYSTEM \SAA AT 6-Mar-1981 14:00:58
    THE TMF AUDIT SUBSYSTEM IS ACTIVE FOR TRANSACTION PROCESSING.
    CURRENT CRASH COUNT FOR THIS SYSTEM IS 1
BACKOUT PROCESS:
    BACKOUT $Z007, CPUS (0:1), PRI 149
AUDITTRAILS:
    0 AUDITTRAIL $AUDIT.AP.MN, EXT (20,20), MINFILES 2, MAXFILES 5,
           AUDITDUMP AUTOMATIC $ADP0, USAGE MONITOR,
           SEQUENCE NUMBER 000003
    1 AUDITTRAIL $AUDIT.AP.AA, EXT (90,90), MINFILES 2, MAXFILES 5,
           AUDITDUMP AUTOMATIC $ADP0, USAGE DISCPROCESS ($VOL1),
           SEQUENCE NUMBER 000008
AUDITDUMP PROCESSES:
    AUDITDUMP $ADP0, CPUS (0:1), PRI 148, COPIES 1, WAIT OFF,
         OUT $DATA.TMF.DUMPRPT
         CURRENT ACTIVITY: DUMPING AN AUDITTRAIL
         TRANSFER STATUS:
              TAG OF REQUEST: 0
              NUMBER OF COPY BEING REPORTED ON: 1
              FILE BEING TRANSFERED: $AUDIT.AP.AA000007
              DRIVE BEING USED: $TAPE
              REEL BEING USED: JER456
              FILE NUMBER ON REEL: 2
              BLOCK NUMBER ON REEL: 1
TMF CATALOG:
    CATALOG $Z062, CPUS (0:1), PRI 148, RETAINDEPTH 3, RELABEL ON
    THE CATALOG PROCESS IS ACTIVELY PROCESSING A REQUEST OR DUMP.
    NUMBER OF ACTIVE REQUESTORS: 4
    TOTAL NUMBER OF DISC BYTES USED: 30784
```

This example is for a NonStop system, but it applies equally well to a NonStop 1+ system with AUDITPROCESS substituted for DISCPROCESS in the audit-trail usage.

STATUS TRANSACTION

STATUS TRANSACTION displays the transaction identifier and status of one or more transactions in the system in which TMFCOM is executing. It can provide the transaction identifiers and status (active, ending, or aborting) of all transactions known to the system, or you can select specific transactions.

You can select those transactions that were begun by one or more specified processes, or you can specify a set of transaction identifiers based on the system name or number, sequence number, CPU number, state of completion, or any combination of these.

The STATUS TRANSACTION command syntax is:

```
STATUS TRANSACTION

 [ transaction-identifier                                      ]
 [ ( transaction-identifier   , transaction-identifier  ... )  ]

 [        { ACTIVE   }                                    ]
 [ ,STATE { ENDING   }                                    ]
 [        { ABORTING }                                    ]
 [           { process                                 }  ]...
 [ ,PROCESS  { ( process [ , process]... )             }  ]
 [           { ( cpu , pin )                           }  ]
 [           { ( ( cpu , pin) [ , ( cpu , pin ) ]... ) }  ]
```

where

`transaction-identifier`

identifies the transaction to be displayed. Its form is defined in the “Transaction Identifiers” subsection near the beginning of this section.

An asterisk (*), substituted for the appropriate element of the *transaction identifier*, matches that element for any transaction identifier still in the system.

If you omit the *transaction identifier* list, TMF reports on all transactions still in the system (as if you had specified *(*).*.*).

`STATE`

limits the display to active transactions (STATE ACTIVE), to transactions that are in the process of ending (STATE ENDING), or to transactions that are in the process of aborting (STATE ABORTING). If you do not use the STATE option, transactions in all states are selected for display.

`PROCESS`

uses the identity of the process that issued BEGIN-TRANSACTION for each transaction to determine whether data should be displayed about that transaction.

→

```
process
```

identifies the process that issued BEGIN-TRANSACTION for transactions to be selected for display. This is a standard GUARDIAN process name, such as $DOOIT or \KEOKUK.$DOOIT.

```
( cpu , pin )
```

is the CPU number and process identification number of the process that issued BEGIN-TRANSACTION for transactions to be selected for display.

Considerations

You can use any combination of transaction-identifier components to select transactions for display. If you use the STATE or PROCESS option, only those transactions whose identifiers match the options will be selected.

If communication with the specified system is not available, an asterisk is assumed for the *crash count*.

Only one STATE option can be included in any one STATUS TRANSACTION command.

Examples

The command

```
STATUS TRANSACTION 3.*
```

displays the transaction identifier and status for each transaction whose home node is the local system and that originated in CPU 3. The display is similar to this:

```
\HOME(0).3.177, PROCESS $ADDREC(3,16), STATE ENDING
\HOME(0).3.225, PROCESS $DELREC(3,11), STATE ACTIVE
\HOME(0).3.246, PROCESS $MODREC(3,22), STATE ABORTING
```

transaction identifier and *$MODREC(3,22)* is the name, CPU, and PIN of an application process. The process name for an aborting transaction could be wrongly reported if the process that began the transaction has died and some other process has been started in the same CPU with the same PIN.

The command

```
STATUS TRANSACTION \SF.*.*
```

displays the transaction identifier and status for each transaction whose home node is \SF.

The command

```
STATUS TRANSACTION *.*, STATE ABORTING
```

displays the transaction identifiers and status for each transaction that is aborting in the system where this command is executed.

The command

```
STATUS TRANSACTION *.*, STATE ACTIVE, PROCESS $HOPE
```

displays the transaction identifier and status of each active transaction started by the process $HOPE, whether or not $HOPE is still running.

STATUS VOLUMES

The STATUS VOLUMES command is valid on NonStop systems only, not on NonStop 1+ systems.

STATUS VOLUMES tells which disc volumes are enabled for TMF processing. The STATUS VOLUMES command syntax is:

```
                 ┌ volume                      ┐
STATUS VOLUMES   │ ( volume [ , volume ]... )  │ (II only)
                 └ *                           ┘
```

where

`volume`

is a disc volume name, such as $DATA.

`*`

requests information about all disc volumes in the system.

Considerations

STATUS VOLUMES can be used only while TMF is running (that is, after START TMF).

Examples

The command

```
STATUS VOLUMES
```

or

```
STATUS VOLUMES *
```

lists each disc volume in the system and tells whether the volume is enabled for transaction processing. The display is similar to this example:

```
VOLUME      CONFIG TYPE       STATUS
$ACCT       AUDITED            ENABLED
$BEANS      AUDITTRAIL         DISABLED
$CITY       UNKNOWN            EXCLUDED
$DATA       NOT CONFIGURED
$SALES      AUDITED            DISABLED
```

In the center ("CONFIG TYPE") column, "NOT CONFIGURED" occurs when the a DP1 disc volume is in a DP2 environment or vice versa. "AUDITTRAIL" indicates a volume containing only audit trails. "AUDITED" indicates a volume containing audited data files. If a DP1 volume contains both audited data files and audit trails, "AUDITED" will be displayed.

The command

```
STATUS VOLUMES ($ACCT, $SALES)
```

lists only the two specified volumes:

```
VOLUME         CONFIG TYPE        STATUS
$ACCT          AUDITED             ENABLED
$SALES         AUDITED             DISABLED
```

STOP TMF

STOP TMF shuts down TMF in an orderly manner. Its command syntax is:

```
STOP TMF [ , WAIT {OFF | ON} ]
```

where

`WAIT OFF`

specifies that TMFCOM should not wait for quiescence, but should immediately return the TMFCOM prompt (~) for additional user input. This is the default setting.

`WAIT ON`

specifies that TMFCOM should not return the TMFCOM prompt until all TMF processes have ceased activity. The default setting is WAIT OFF.

Considerations

Before TMF can be shut down, the system must be *transaction-quiescent*; that is, it must have no currently active transactions. If the system is not quiescent, the STOP TMF command prohibits BEGIN-TRANSACTION from being issued in any processor, then tests again for quiescence. Only when the system is quiescent will STOP TMF stop the TMF processes.

If you specify WAIT ON and TMF detects active transactions, TMF periodically displays the status of all transactions. When the status is displayed, you can either allow TMFCOM to wait for quiescence, or use the BREAK key. Using the BREAK key during execution of STOP TMF does not cease the shutdown sequence; it only returns the TMFCOM prompt (~) for additional user input. This lets you exit TMFCOM or issue other commands while the shutdown is proceeding; for example, you may want to execute the ABORT TRANSACTION command on the home node for each of the active transactions (thereby forcing quiescence).

If an audit dump is pending during the shutdown procedure and you give the media-handling response STOP to that dump process, the audit-dump process retries the dump after TMF is restarted. STOP TMF cannot complete until all pending dump activity has ceased, either by finishing the dumps or by your use of the STOP response.

STOP TMF does not cause TMF to advance from one audit-trail sequence number to the next. For example, if audit trail $AUDIT.AP.MN is in the middle of file $AUDIT.AP.MN000088 when TMF is shut down, it will still be in file $AUDIT.AP.MN000088 when TMF is restarted.

If you issue STOP TMF while a START TMF command is in progress, TMF displays an error message and ignores the shutdown request.

When the STOP TMF command completes, TMF displays a console message showing a shutdown serial number. If you expect to use the SHUTDOWN option of RECOVER FILES, you should keep a record of the time, date, and serial number from each occurrence of this message. The INITIALIZE CATALOG command resets the shutdown serial counter, so a TMF shutdown after catalog initialization can have the same serial number as a pre-initialization shutdown.

STOP TRANSACTION

STOP TRANSACTION disallows processing of new TMF transactions. Its command syntax is simply:

```
STOP TRANSACTION
```

with no options.

Considerations

While the rollforward process is recovering data from a crash, new transactions can manipulate the files that were not affected by the crash. Similarly, TMF can record online dumps while transactions are manipulating the data base. This concurrent transaction activity, however convenient, can slow both the recovery (or dumping) and the transaction processing. In some circumstances, it can be more efficient to disable transaction processing until the recovery or dumping is complete. If you issue either a START TMF command with the TRANSACTION OFF option or a STOP TRANSACTION command, no transactions can begin, although all other TMF functions remain enabled. From this state, START TRANSACTION reenables processing of transactions.

SECTION 3

MEDIA-HANDLING RESPONSES

TMF sometimes asks the operator to load a tape or a disc dump, or to supply a tape-drive identifier or other information. TMF has a separate set of commands, administered by the TMFTAPE utility, to deal with these media-handling situations.

This media-handling facility uses an at-sign (@) to prompt you for a response. Depending on the situation, your response may include one or more of the commands described in this section: DISABLE, DRIVE, ENABLE, EXIT, FC, HELP, NEXT, REEL, RELEASE, STATUS, STOP, and WAIT.

You can specify more than one response, separated by semicolons (;), on one line. The effect is the same as when you type a carriage return between response options. In either case, TMF does not act on a DISABLE, DRIVE, ENABLE, NEXT, REEL, or RELEASE command until you enter EXIT.

This operator interaction can occur before automatic dumping of an audit trail and when you use the TMFCOM commands DUMP FILES, RECOVER FILES, ADD MEDIA, and ALTER MEDIA.

DISABLE

DISABLE terminates the specified copy of a dump. Its syntax is:

```
DISABLE copy
```

where

`copy`

identifies the dump copy to be terminated. The *copy* number must be between 1 and the *quantity* currently configured for the audit-dump process.

Considerations

After you have issued the TMFCOM command DUMP FILES to make two or more simultaneous copies of an online dump, DISABLE lets you terminate some of those copies. This may be necessary, for example, if you do not have enough scratch tapes or tape drives available to handle all of the scheduled copies.

If you use DISABLE and RELEASE in the same response, DISABLE is applied first.

If you attempt to disable all pending copies of the dump, TMF displays the message "All copies have been disabled. Please enable at least one copy." It then reprompts you, so you can use the ENABLE command to enable one or more copies.

To abort an online dump, you can use the STOP command. You cannot abort an audit dump, but you can use the WAIT command to defer it. STOP does not abort an audit dump although, if execution of a STOP TMF command is in progress, it does defer the audit dump until TMF is restarted.

You can also use the TMFCOM command CONTROL AUDITDUMP's DISABLE option to abort unwanted audit-dump copies, but that affects all subsequent audit dumps until TMF is stopped. The media-handling command DISABLE affects only the current dump.

Example

In this example, audit-dump process $AUD1 was configured to produce two copies of each dump but the operator has only one tape drive available:

```
$AUD1: For dump of file $AUDIT.AP.00000034 using serial 283, copy 1,
$AUD1: reel number 1,
$AUD1: please mount tape JER614 on drive $TAPE
$AUD1: and respond 'exit' when ready:
$AUD1: @EXIT
$AUD1: For dump of file $AUDIT.AP.00000034 using serial 283, copy 2,
$AUD1: reel number 1,
$AUD1: please mount tape JER615 on a free drive;
$AUD1: specify the drive and respond 'exit' when ready:
$AUD1: @DISABLE 2; EXIT
```

After the operator has mounted tape JER614 and responded with EXIT, $AUD1 prompts for another tape for copy 2. Because TMF writes all copies simultaneously and only one tape drive is available, the operator disables the second copy. Each subsequent dump by this process will again attempt to make two copies. The TMFCOM command

```
CONTROL AUDITDUMP $AUD1, COPIES 1
```

would change this until TMF can be shut down and the process reconfigured.

DRIVE

The DRIVE command specifies the tape drives on which you have mounted the required tapes. Its syntax is:

```
DRIVE {device                      }
      {(device [ , device ]... )}
```

where

`device`

identifies a tape drive. This is a standard GUARDIAN device name, such as $TAPE or \KEOKUK.$TAPE2.

Considerations

DRIVE commands, unlike REEL, are cumulative; that is, the command sequence

```
DRIVE $TAPE1; DRIVE $TAPE2
```

is equivalent to

```
DRIVE ($TAPE1, $TAPE2)
```

If you use RELEASE and DRIVE in the same response, RELEASE is applied first.

Before TMF can dump a file to tape, either by an audit-dump process or in an online dump, it checks the tape drives which were specified in the ADD AUDITDUMP (or CONTROL AUDITDUMP) or DUMP FILES command, respectively. If none of the specified tape drives is available, TMF displays a message similar to this:

```
FILE OPEN ERROR - $TAPE2 - FILE IN USE (012)
For dump of file $DATA.RV.FU using serial 365, copy 1, reel number 1,
please mount tape JER283 on a free drive;
specify the drive and respond 'exit' when ready:
@
```

The "file" referred to in the first line of the message is tape drive $TAPE2. If you have another tape drive available, you can mount a scratch tape on it, then use the DRIVE command to switch the dump to that drive.

When (as in the example above) TMF prompts you to "please mount tape ... on a free drive," you must use the DRIVE command to specify a drive.

For another example, assume that you are using the TMFCOM command ADD MEDIA to relabel a new tape and add it to the catalog. After displaying a date-and-system identification line, TMF asks you to mount the specified tape and to specify the drive. When you have done so and terminated your response with an EXIT command, TMF performs the requested action and describes its results, as shown here:

```
~ADD MEDIA TAPE JER456, LABEL ON, RELABEL ON, OPERATOR OFF, DETAIL
TMFCOM - T9016E05 - (15OCT82)     System \RA      5-Nov-1982 18:46:05
For labelling tapes,
please mount tape JER456 on a free drive;
specify the drive and respond 'exit' when ready:
@DRIVE $TAPE1; EXIT
Tape JER456 labelled on drive $TAPE1.
~
```

ENABLE

ENABLE reenables the specified copy of a dump which has been disabled by a DISABLE command or by the DISABLE option of the TMFCOM command CONTROL AUDITDUMP. Its syntax is:

```
ENABLE copy
```

where

`copy`

identifies the dump copy to be reenabled. The *copy* number must be between 1 and the *quantity* currently configured for the audit-dump process.

Considerations

If you attempt to disable all pending copies of the dump, TMF displays the message "All copies have been disabled. Please enable at least one copy." It then reprompts you, so you can use the ENABLE command to enable one or more copies.

EXIT

After a dumping or recovery process has prompted the operator for some tape-related action or information, the EXIT command tells TMF to proceed with the pending operation (such as writing a dump to tape). Its syntax is simply:

```
EXIT
```

with no options.

Considerations

When a TMF process requires you to mount a tape or specify some information, TMF displays an at-sign (@) prompt and the process waits for your response. It will continue to accept the media-handling responses described in this section, then prompt you again, until you issue the EXIT command. The EXIT command terminates this operator interaction mode and enables all media-handling commands you have made during this spate of interaction.

You may want to use a semicolon to put the EXIT command on the same line as another command to avoid a second prompt, as in this audit-dump example:

```
$ADMP: For dump of file $DATA.SMITH.MN000045 using serial 377, copy 1,
$ADMP: reel number 1,
$ADMP: please mount tape JER432 on drive $TAPE1
$ADMP: and respond 'exit' when ready:
$ADMP: @DRIVE $TAPE2;EXIT
```

FC

FC provides the ability to edit or repeat the preceeding command line. The FC command syntax is simply:

```
FC
```

with no options.

Considerations

Just as in TMFCOM and other Tandem software products, the FC command displays the previous command line and prompts for editing input with a period (.). At this point, it accepts the R, I, or D subcommands, to replace, insert, or delete letters in the command string. For a more complete discussion of this command, see Appendix B.

HELP

HELP displays syntax information about the specified response option. Its syntax is:

```
       ┌ *       ┐
       │ DISABLE │
       │ DRIVE   │
       │ ENABLE  │
       │ EXIT    │
       │ FC      │
HELP   │ HELP    │
       │ NEXT    │
       │ REEL    │
       │ RELEASE │
       │ STATUS  │
       │ STOP    │
       └ WAIT    ┘
```

where

*

displays the HELP reports for all media-handling commands.

each other option

specifies the command whose information is to be displayed.

Examples

HELP with no option displays this general message:

```
Ask for help about a command using
     HELP <command name> [ <option name> ]
Ask for help about a non-terminal using
     HELP "<" <characters>... ">"
Ask for help about commands in general via
     HELP <syntax>
Action Commands:
     DISABLE      DRIVE       ENABLE      NEXT       REEL
     RELEASE      STATUS      STOP        WAIT
Utility Commands:
     EXIT         FC          HELP
```

In this message, the reference to a "non-terminal" is for any text which a HELP message returns within angle brackets, such as <*copy number*> in the next example.

The command

```
HELP DISABLE
```

displays this message:

```
DISABLE <copy number>
     This command, which can only be used when TMF is writing tapes,
disables the writing of the specified copy. <copy number> is an
integer in the range 1 to 99, identifying the specific copy to be
disabled.
```

The command

```
HELP <COPY NUMBER>
```

displays this message:

```
<copy number>
     This is an integer in the range 1 to 99, identifying a specific
dump copy being read or written.
```

NEXT

NEXT tells TMF to choose another tape reel. Its syntax is simply:

```
NEXT

   with no options.
```

Considerations

When you issue the NEXT command in response to TMF's request for a tape reel, TMF indicates its next choice and prompts you to mount that tape.

When dumping a file, for instance, if TMF has specified a certain scratch tape and that tape is not available, you can use NEXT to force TMF to select a different scratch tape. In this situation, you could also use the REEL command to indicate a specific scratch tape.

If you have previously specified a list of scratch tapes, by using either the REEL command or a TMFCOM command's REEL option, NEXT causes TMF to select the next tape in that specified list. When it has exhausted that list, TMF then selects the first scratch tape in the media directory.

During rollforward, the NEXT command causes TMF to select another, equivalent copy of the tape TMF had requested. This lets you switch to the second copy of an online dump or audit dump if the first is unavailable or unreadable.

For example, assume that you have two copies of an online dump, each of which spans two reels, or *parts*, as shown in this INFO DUMPS, DETAIL output:

```
TMF CATALOG DETAILED DUMP INFORMATION

    $DUMPS.ABC.DONNE , SERIAL 365
         TIME: 1-Feb-1982 21:47:49
         TYPE: AUDITDUMP
         MEDIUM: TAPE
         BAD OFF
         INVALID OFF
             PART  COPY   MEDIA ID  STATUS
               1     1     ABC876   READABLE
               1     2     ABC542   READABLE
               2     1     ABC568   READABLE
               2     2     ABC183   READABLE
```

If the rollforward process has prompted you to mount tape ABC876 (copy 1 of part 1) and you respond with the NEXT command, you are then prompted to mount tape ABC542 (copy 2 of the same part of the dump). When that tape has been read, TMF asks for tape ABC568 (copy 1 of part 2).

If you repeatedly enter NEXT until no more copies of a given part of the dump remain, then TMF asks again for copy 1. If none of the copies are available and readable, you can use the STOP command to terminate the rollforward operation, then use DELETE DUMPS on that online dump, then restart rollforward with another dump.

REEL

REEL specifies the next reels to be used. Its syntax is:

```
REEL { reel                    }
     { ( reel [, reel ]... )   }
```

where

`reel`

is a reel identifier, as defined in the “Tape Terminology” subsection in Section 1 of this manual.

Considerations

The REEL command is often used when TMF has prompted you to mount a specific scratch tape for a dump and that tape is not available.

This replaces any previous specification, except that any reels already written or otherwise in current use are kept.

REEL is for use with dumps, not rollforward recovery.

TMF does not check the specified *reel* identifier against the catalog until it has read the tape label. Then, if the tape already contains a valid dump, TMF displays a message like this one:

```
$Z004: Error: Reel JER345 is already in use.
```

and unloads the tape. If you specify a tape which is labelled as a scratch tape but is not in the catalog, TMF uses and catalogs it. If you mount a tape other than one specified, TMF repeats the prompt asking you to mount the specified tape.

RELEASE

If any of the specified tape drives is currently assigned to the currently prompting process (perhaps by a previously issued DRIVE command), RELEASE frees them for reassignment. The command syntax is:

```
RELEASE { device                    }
        { (device [, device ]... )  }
```

where

```
  device
```

identifies a tape drive. This is a standard GUARDIAN device name, such as $TAPE or \KEOKUK.$TAPE2.

Considerations

If process $AUD tries to use a drive that another process is already using, TMF displays a message similar to this:

```
$AUD: FILE OPEN ERROR - $TAPE2 - FILE IN USE (012)
$AUD: For dump of file $DATA.ACCTING.JONES using serial 95, copy 1,
$AUD: reel number 1,
$AUD: please mount tape JER283 on a free drive;
$AUD: specify the drive and respond 'exit' when ready:
$AUD: @
```

The "file" referred to in the first line of the message is tape drive $TAPE2. If drive $TAPE2 is not actually in use but just waiting for a tape to be mounted for a different TMF process, you might respond with a WAIT command like

```
WAIT 40
```

so you can communicate with the process owning the drive and then, when the owner prompts you, tell it

```
RELEASE $TAPE2;EXIT
```

to free the drive for use by the other process.

If any of the specified drives is in the middle of a dump, TMF displays a console message and refuses to release the drive until the copy being written on that drive is disabled or completed.

If you use DISABLE and RELEASE in the same response, DISABLE is applied first. If you use RELEASE and DRIVE in the same response, RELEASE is applied first.

Whenever TMF releases a tape drive, whether due to the completion of a command or due to a RELEASE command, any tape mounted on the drive is rewound and unloaded, to prevent accidental damage to or misuse of the tape. Similarly, if an assigned tape (i.e., a tape containing a valid dump) is on a drive specified for a new dump, TMF rewinds and unloads it.

STATUS

STATUS displays the current activity, the names of affected files, and other information pertaining to the activity which is causing the interaction. Its syntax is:

```
STATUS [ , {BRIEF|DETAIL} ]
```

where

`BRIEF or DETAIL`

determines how much information is to be displayed. DETAIL displays all the information shown in the example below. BRIEF omits the tape file number and tape block number. If you do not specify either BRIEF or DETAIL, TMF displays the brief report.

Considerations

At times, two or more dumps may be ready at the same time, with all of them prompting you for tapes. In such a situation, you may want to use the STATUS command to determine which one the current prompt belongs to. You can also use the STATUS report to help you decide which process's demand to handle first and which processes to give WAIT commands to.

Example

This example shows a STATUS report during an online dump:

```
$Z017: For dump of file $DATA.ACCT.SMITH using serial 666, copy 2,
$Z017: reel number 1,
$Z017: please mount tape JER678 on a free drive;
$Z017: specify the drive and respond 'exit' when ready:
$Z017: @STATUS, DETAIL
$Z017: TMFCOM - T9016E05 - (15OCT82) System \RA 5-Nov-1982 19:05:27
$Z017: Status for process $Z017
$Z017: Current function is DUMP FILES
$Z017: File set is $DATA.ACCT.*
$Z017: Total number of copies is 2
$Z017: Current file is $DATA.ACCT.SMITH
$Z017: Current dump serial number is 136
$Z017: Tape file number is 0
$Z017: Tape block number is 0
$Z017: Drive $TAPE1 is accessing reel JER345
$Z017: Tape reels not yet used are (JER678,JER939,JER123)
$Z017: @
```

In this example, $Z017 is the name of the currently active process.

The "Tape reels not yet used..." line shows which reels have been specified by REEL, either as a media-handling command or as a TMFCOM option. If no reels have been specified, it shows which scratch tape TMF has picked from the media catalog.

STOP

STOP terminates the current TMFCOM command and displays a message to that effect on the terminal from which the TMFCOM command was issued. The STOP command syntax is simply:

```
STOP
```

with no options.

Considerations

The STOP command is useful when you want to abort a DUMP FILES or RECOVER FILES command.

You cannot abort an audit dump, but you can use the WAIT command to defer it. STOP does not abort an audit dump; however, if a TMF shutdown (resulting from TMFCOM's STOP TMF command) is in progress and TMF is attempting to dump an audit-trail file, STOP terminates the attempted audit dump so the shutdown can complete. When TMF is restarted, the audit-dump process will again attempt to dump that audit-trail file.

If you use STOP on an audit dump when that dump is the only one pending (and TMF is not stopping), the audit-dump process immediately makes another attempt to dump the file. A better way to defer an audit dump is to use the WAIT command (unless TMF is stopping).

If you use the STOP command during a DUMP FILES operation of two or more files, any dumps which already have been completed remain valid. Any dump which was in progress is invalid and not catalogued.

Similarly, if you use the STOP command during a RECOVER FILES operation, any files which already have been completely rolled forward are usable. Any file whose recovery was in progress must be recovered again.

If you use the STOP command while relabelling two or more tapes (by ADD MEDIA or ALTER MEDIA), any tape whose relabelling has been completed already will have a valid entry in the catalog.

Whenever the operator responds with STOP to a prompt resulting from some TMFCOM command, the message

```
WARNING: The operation in progress has been stopped by the operator.
```

is displayed on the terminal from which the command was entered. If this happens during your TMFCOM command, you should contact the operator before using FC to correct your command (if necessary) and resubmit it.

WAIT

WAIT tells TMF to wait for a specified time and repeat the prompt, thus letting the operator defer a request temporarily while handling others. Its syntax is:

```
WAIT [ seconds ]
```

where

`seconds`

specifies how many seconds TMF should wait before reprompting you. The active process is suspended during that interval.

Considerations

For an audit-dump process, *seconds* is optional. For other processes, there is no default and you must specify *seconds*.

WAIT without the *seconds* option causes an audit-dump process to wait until it is reawakened by the TMFCOM command CONTROL AUDITDUMP with the WAIT OFF option. All waiting audit-dump processes reawaken automatically when the TMFCOM command STOP TMF is executed; at that time, you can use STOP to defer each audit-dump process until TMF is restarted.

The WAIT command is useful when several dumps are pending at the same time, with all of them prompting you for tapes. In such a situation, you may want to use the STATUS command to help you decide which one to handle first and which to give WAIT commands.

WAIT is also useful when an audit dump becomes active during an online dump, or when a rollforward operation calls simultaneously for more tape drives than you have available.

SECTION 4

RELATED FUP COMMANDS

In a NonStop system, TMF can audit any data-base file on any disc volume. In a NonStop 1+ system, any data-base file to be audited by TMF must be on a disc volume which was configured as audited during system generation.

TMF will write, to the configured audit trails, the before- and after-images of all changes to these files. This lets TMF back out changes to audited files if the transaction making the changes aborts, or roll forward to recover from a multiple-component failure. You can designate files as audited or change the status of a file between audited and non-audited by using the GUARDIAN Operating System's CREATE procedure or one or more of the File Utility Program (FUP) commands ALTER, CREATE, RESET, and SET.

This section describes the special options of these FUP commands which let you designate audited files. For complete details on the commands, see the *GUARDIAN Operating System Utilities Manual* for your system.

ALTER

The ALTER command changes certain attributes of an existing file. One of these attributes governs whether TMF is to audit the file. If you use the AUDIT option, the file is marked as an audited file. If you specify NO AUDIT, the file is marked as a non-audited file.

A change in the audited/non-audited attribute of the file is automatically applied to any associated (automatically updated) alternate-key files and any secondary partitions.

If the specified file is an audited file whose crash-open flag is on, the NO AUDIT option of FUP's ALTER command resets that flag. A subsequent ALTER command with the AUDIT option would leave the flag off.

On a NonStop 1+ system, file error 80 is returned if *file name*, any associated (automatically updated) alternate-key file, or any secondary partition resides on a volume that has not been configured as audited. This situation does not apply to the NonStop system.

You cannot change the audited/non-audited attribute of the file if TMF is not active.

If an audited file is made unaudited, then all information about that file is removed from TMF's catalog. Any audit-trail dump or online dump pertaining to only that file becomes empty, and its media therefore become scratch.

If you make an audited file unaudited, then make it audited again, the catalog has no valid dumps for that file. In this situation, you should immediately dump the file to provide a basis for rollforward recovery.

CREATE

The CREATE command creates a file, using the current creation parameter values as specified by any previous SET commands issued in FUP. Any of the creation parameter values can be overridden in the CREATE command. If you use the AUDIT option, the file is marked as an audited file. If you specify NO AUDIT, the file is marked as a non-audited file. The default is NO AUDIT.

All automatically updated and automatically created alternate-key files for *file name* are designated as audited. On a NonStop 1+ system, file error 80 is returned if one of these files resides on a non-audited volume.

RESET

RESET cancels any creation values that have been established by the FUP command SET and restores these attributes to their original values, one of which is NO AUDIT.

SET

SET establishes one or more creation parameter values for subsequent use of the CREATE command within the same iteration of FUP. The command sequence

```
CREATE NEWFILE1
SET AUDIT
CREATE NEWFILE2
CREATE NEWFILE3
```

creates three files, the first non-audited and the others audited (assuming there was no previous SET AUDIT command in this iteration of FUP).

SECTION 5

RELATED PUP COMMANDS

TMF recognizes two forms of online dump. One is to tape, effected by the TMFCOM command DUMP FILES. The other is, essentially, just the removal of a disc pack from its drive. The basic disc online dump procedure is:

1. Use the Peripherals Utility Program (PUP) command REMOVEAUDITED to refresh one member of a mirrored pair of discs and to prepare it for removal.

2. Take that disc out of the drive and replace it with a scratch disc.

3. Use the PUP command REVIVE to copy the contents of the other mounted disc onto the new disc, thereby creating a new mirror and returning the system to its previous state.

4. Store the removed disc in a safe place until it is needed either for rollforward or to replace an online-dump disc.

PUP commands must be issued from the GUARDIAN Command Interpreter environment, not from TMFCOM.

PUP commands REMOVEAUDITED and REVIVE, described in this section, are discussed in detail in the *GUARDIAN Operating System Utilities Reference Manual* for your system.

REMOVEAUDITED

The PUP command REMOVEAUDITED creates a disc online dump by letting you physically remove a disc pack. This command refreshes one member of a mirrored pair of discs, prepares it for removal, and writes a description of its contents to the TMF catalog. The command syntax is:

```
REMOVEAUDITED [/OUT file/] volume { -P } , PACK pack
                                  { -M }
```

where

`/OUT file/`

specifies a file, terminal, or other destination for output from this command. The default is the terminal from which the command is issued.

`volume`

is a standard GUARDIAN volume name or logical device number, such as $5 or \KEOKUK.$DATA or $DATA.

`-P`

specifies that the primary volume is to be removed.

`-M`

specifies that the mirror volume is to be removed.

`PACK pack`

specifies the identifier by which the physical disc pack will be known to the catalog and the rollforward process. The *pack* identifier, comprising six or fewer alphanumeric characters, is encoded on the disc pack when REMOVEAUDITED creates an online dump; it should also be written visibly on the disc pack's cover. Note that the *pack* identifier names the physical disc, not the data recorded on it or the volume name of the disc drive.

This identifier will be erased if the pack is subjected to either the rollforward process or the PUP commands FORMAT, LABEL, or MOUNT!. While a pack bears this identifier, only SUPER-group users can use the PUP command MOUNT on it.

Considerations

If you use the REMOVEAUDITED command on a NonStop 1+ system's non-audited disc, an error message is displayed and the command terminates. (A NonStop system has no non-audited discs.) If you use the PUP command REMOVE on an audited disc, the operation proceeds but, when it has completed, this warning is printed:

```
THE REMOVE WAS SUCCESSFUL BUT IT IS NOT A TMF DUMP
```

Unlike the REMOVE command, REMOVEAUDITED proceeds regardless of whether any files on the disc are open. Therefore, the operation can be performed without interrupting the processing of transactions.

The REMOVEAUDITED command fails if the disc pack contains one or more audit trails, if it is not audited, or if it is not mirrored.

The TMFCOM configuration command ALTER TMF, RECOVERY OFF makes rollforward recovery impossible until TMF is reconfigured. REMOVEAUDITED operates normally in a TMF installation configured with this command, but the resulting dump can never be used.

Example

The command

```
REMOVEAUDITED $DATA-M, PACK DATA01
```

- encodes the identifier DATA01 on the disc pack in drive $DATA-M
- writes TMF catalog entries for the dump
- prepares for physical removal of the disc pack
- displays, on the terminal from which the command was given, a list of dumped files, similar to this one:

```
TMFCOM - T9016E03 - (01DEC81)     System \QA     5-Feb-1982 19:03:23
  File Name                       MATSN    FATSN    Error
[Dump serial number is 366.]
[Pack DATA01 used for copy 1.]
$DATA.ACCT.BATES                  9         34
$DATA.ACCT.JONES                  9         34
$DATA.ACCT.RUSH                   9         34
$DATA.ACCT.SMITH                  9         34
$DATA.MKT.HEATH                   9         34
$DATA.MKT.GARFIELD                9         34
$DATA.MKT.MORRIS                  9         34
[Total files dumped = 7.]
```

In the list's headline, *MATSN* stands for "monitor audit-trail sequence number" and *FATSN* stands for "file audit-trail sequence number"—the sequence number of the data audit trail.

REVIVE

The PUP command REVIVE, when used after REMOVEAUDITED, copies the contents of the remaining disc onto a scratch disc, thereby restoring the disc volume's status as a mirrored pair. The command syntax is:

```
REVIVE volume [ , tracks [ , interval ] ]
```

where

`volume`

is a standard GUARDIAN volume name, such as \KEOKUK.$DATA or $DATA.

`tracks`

specifies the number of tracks to be copied at each copying *interval.* The default is one track per *interval.*

`interval`

specifies the number of 10-millisecond units between each copying operation. The default is 1000, or 10 microseconds.

Considerations

The REVIVE command does not check the contents of the scratch disc pack before overwriting it. It is your responsibility to ensure that the newly inserted disc pack is indeed a scratch pack.

Example

If you have just used REMOVEAUDITED to remove the disc pack from $DATA-M, then replaced that pack with a scratch pack, the PUP command

```
REVIVE $DATA
```

copies the contents of the pack in drive $DATA-P onto the scratch pack and brings it into use.

SECTION 6

RELATED PATHCOM COMMANDS

PATHCOM, the command utility for PATHWAY, includes several features to help you configure PATHWAY systems to use TMF. You can

- specify a global *transaction restart limit* that defines the maximum number of times a failed transaction can be restarted automatically. The PATHCOM command to do this is SET PATHWAY. Note that if the terminal control process (TCP) does not have a backup process and the primary process fails, transactions will not be automatically restarted.

- define server classes that can update audited files. PATHCOM commands to do this are ADD SERVER, ALTER SERVER, RESET SERVER, and SET SERVER.

- specify which terminal program units (if any) will not be allowed to operate in TMF transaction mode. The PATHCOM commands ADD TERM, ALTER TERM, RESET TERM, and SET TERM configure the terminal program units. Another PATHCOM command, STATUS TERM, can display the status of terminals.

This section discusses the syntax of these PATHCOM commands. For complete details on the commands, see the *PATHWAY System Management Reference Manual.*

ADD SERVER

ADD SERVER enters a description of a server class into the PATHWAY configuration, using the current server parameters as specified by any previous SET SERVER commands.

For TMF, this command specifies whether members of the server class operate on audited or non-audited files. If you use the TMF ON option, the servers can lock and update records in audited files: they are TMF servers. With TMF OFF, the servers cannot lock or update records in audited files: they are non-TMF servers. The default is TMF OFF.

If the terminal is in transaction mode when a SEND statement is executed and the object of the SEND statement is a non-TMF server, the terminal is "suspended for pending abort."

Execution of a SEND statement to a TMF server is allowed while the terminal is not in transaction mode. However, if the server has any audited files open with write or read/write access, an attempt by the server to lock the file or to write or lock a record in the file is rejected with an error 75 ("no current transid").

If the terminal is not in transaction mode, checkpoints are done before and after each SEND statement to a non-TMF server and the terminal control process (TCP) does not automatically retry I/O errors. Checkpoints are *not* done before and after each SEND statement to a TMF server because this type of server is assumed to have "read only" access to its files; that is, its I/O requests are infinitely retryable. If the SEND statement does not have an ON ERROR clause, I/O errors are retried up to the limit specified in the MAXTMFRESTARTS values specified in the SET PATHWAY command. When the limit is reached, the terminal will be suspended but restartable.

ADD TERM

ADD TERM enters a description of a terminal program unit into the PATHWAY configuration, using the current parameters as specified by any previous SET TERM commands within the same iteration of PATHCOM.

This command's TMF option specifies actions to be taken for the TMF-oriented SCREEN COBOL verbs. If you use TMF OFF, these verbs appear to invoke TMF but actually do not. With TMF ON, the SCREEN COBOL verbs do invoke TMF. The default is TMF ON.

Under TMF OFF, the BEGIN-TRANSACTION verb causes the special register TRANSACTION-ID to be set to LOW-VALUE.

ALTER SERVER

ALTER SERVER changes the characteristics of a previously defined server class. The server class must have no running server processes when this command is issued.

For TMF, this command can alter the characteristic which lets servers operate on audited or non-audited files. If you use the TMF ON option, the servers can lock and update records in audited files: they are TMF servers. With TMF OFF, the servers cannot lock or update records in audited files: they are non-TMF servers.

See the discussion of ADD SERVER for other considerations pertaining to this attribute.

ALTER TERM

ALTER TERM changes the characteristics of a previously defined terminal program unit. The TMF option specifies actions to be taken for the TMF-oriented SCREEN COBOL verbs. If you use TMF OFF, these verbs appear to invoke TMF but actually do not. With TMF ON, the SCREEN COBOL verbs do invoke TMF. The default is TMF ON.

See the discussion of ADD TERM for other considerations pertaining to this attribute.

RESET SERVER

RESET SERVER resets one or more attributes for a server to their initial values, one of which is TMF OFF. See the discussion of ADD SERVER for considerations pertaining to this attribute.

RESET TERM

RESET TERM resets one or more attributes for a terminal program unit to their initial values, one of which is TMF ON. See the discussion of ADD TERM for considerations pertaining to this attribute.

SET PATHWAY

SET PATHWAY configures the PATHWAY monitor. PATHMON must be in the starting state before this command is issued.

The MAXTMFRESTARTS option of this command sets a value for the global *transaction restart limit* for transactions in the PATHWAY monitor system. This limit represents the maximum number of times that a transaction can be restarted automatically if it aborts because of a failure. It also represents the maximum number of times that a SEND statement to a TMF server will be automatically retried when (1) the terminal is not in transaction mode and (2) the SEND verb does not have an ON ERROR clause.

The transaction restart limit can be between 1 and 32767 to indicate the number of restarts, zero to indicate no restarts, or – 1 to indicate unlimited restarts. If this option is omitted, the default is 5.

SET SERVER

SET SERVER establishes values for the characteristics of a server class before a subsequent use of the ADD SERVER command. For TMF, this command can establish whether subsequently created server classes operate on audited or non-audited files. If configured with the TMF ON option, the servers can lock and update records in audited files: they are TMF servers. With TMF OFF, the servers cannot lock or update records in audited files: they are non-TMF servers. The default is TMF OFF.

See the discussion of ADD SERVER for other considerations pertaining to this attribute.

SET TERM

SET TERM establishes values for the characteristics of a terminal program unit before a subsequent use of the ADD TERM command within the same iteration of PATHCOM.

The TMF option specifies actions to be taken for the TMF-oriented SCREEN COBOL verbs. If you configure the terminal program unit with TMF OFF, these verbs appear to invoke TMF but actually do not. With TMF ON, the SCREEN COBOL verbs do invoke TMF. The default is TMF ON.

See the discussion of ADD TERM for other considerations pertaining to this attribute.

STATUS TERM

The STATUS TERM command displays the current state of specified terminals. When the command is used with its DETAIL option, the status display includes associated program unit names and any associated TMF transaction identifiers, and the number of restarts associated with each of those transaction identifiers.

For example, the command

```
STATUS $TERM1, DETAIL
```

displays, among other data, any TMF transaction identifiers associated with terminal $TERM1 and the number of restarts associated with each of those transaction identifiers.

SECTION 7

RELATED SCREEN COBOL VERBS AND REGISTERS

SCREEN COBOL includes four verbs which enable programming of PATHWAY applications for TMF:

- ABORT-TRANSACTION aborts and backs out a transaction.
- BEGIN-TRANSACTION starts a transaction.
- END-TRANSACTION ends a transaction.
- RESTART-TRANSACTION backs out, then restarts a transaction from the BEGIN-TRANSACTION point.

SCREEN COBOL also defines three special registers for use with TMF. All the SCREEN COBOL verbs and registers are discussed in detail in the *PATHWAY Programming Manual.*

SPECIAL REGISTERS

The SCREEN COBOL compiler automatically defines three special registers for TMF users: TRANSACTION-ID, TERMINATION-STATUS, and RESTART-COUNTER. Their uses and implicit declarations are described below.

TRANSACTION-ID

Executing BEGIN-TRANSACTION sets TRANSACTION-ID to the value of the transaction identifier. Executing END-TRANSACTION or ABORT-TRANSACTION sets it to SPACES.

TRANSACTION-ID has this declaration:

```
01 TRANSACTION-ID PIC X(8).
```

TERMINATION-STATUS

Executing BEGIN-TRANSACTION sets the value of TERMINATION-STATUS to indicate the outcome of BEGIN-TRANSACTION. The possible values are:

1—The transaction was started or restarted.

2—TMF is not installed. If there is no ON ERROR phrase, the default system action is to "suspend the terminal for pending abort."

3—TMF is not started. If there is no ON ERROR phrase, the default system action is to suspend the terminal, but the terminal can be restarted by the PATHCOM command RESUME.

4—A fatal error has occurred. If there is no ON ERROR phrase, the default system action is to "suspend the terminal for pending abort."

TERMINATION-STATUS has this declaration:

```
01 TERMINATION-STATUS PIC 9999 COMP.
```

RESTART-COUNTER

Executing BEGIN-TRANSACTION sets RESTART-COUNTER to the number of times the transaction has been restarted. RESTART-COUNTER is reset to zero when BEGIN-TRANSACTION is first executed for a particular transaction.

RESTART-COUNTER has this declaration:

```
01 RESTART-COUNTER PIC 9999 COMP.
```

See the explanation of the BEGIN-TRANSACTION verb, elsewhere in this section, for an example of how to use RESTART-COUNTER to selectively limit the number of times that a transaction is retried.

ABORT-TRANSACTION

Generally, ABORT-TRANSACTION is used when the SCREEN COBOL program detects an unrecoverable error and decides to abandon the transaction. When this verb is executed, the transaction is aborted and all updates made by the transaction to audited data files are backed out. The aborted transaction will not be restarted automatically.

The ABORT-TRANSACTION verb syntax is simply:

```
ABORT-TRANSACTION

with no options.
```

Execution of ABORT-TRANSACTION causes the terminal to leave transaction mode and sets the special register TRANSACTION-ID to SPACES.

If the terminal is not in transaction mode when ABORT-TRANSACTION is executed, the terminal is "suspended for pending abort" and terminal execution cannot be resumed with a RESUME command.

If a fatal error occurs while the transaction is being aborted, and the current BEGIN-TRANSACTION verb did not have an ON ERROR clause, then the terminal is "suspended for pending abort"; the current transaction is backed out and terminal execution cannot be resumed with a RESUME command. If the BEGIN-TRANSACTION verb did have an ON ERROR clause, that clause is executed and the terminal is not suspended.

BEGIN-TRANSACTION

BEGIN-TRANSACTION starts a new transaction; it identifies the beginning of a sequence of operations that will be treated by TMF as a single transaction. When this verb is executed,

- the terminal enters transaction mode
- TMF is requested to start a new transaction
- the transaction identifier for the new transaction is assigned to the TRANSACTION-ID special register
- special registers RESTART-COUNTER and TERMINATION-STATUS are reset to zero for the first occurrence of the transaction.

The BEGIN-TRANSACTION verb syntax is:

```
BEGIN-TRANSACTION [ ON ERROR imperative-statement ]
```

where

```
ON ERROR imperative-statement
```

specifies the statement to be executed if the transaction is being restarted or if an error occurs.

Considerations

The BEGIN-TRANSACTION verb indicates the restarting point to be used if a failure occurs while the terminal is in transaction mode. If the transaction fails for any reason, its data-base changes are backed out and (with the exception of the SCREEN COBOL program issuing ABORT-TRANSACTION) execution of the SCREEN COBOL program can be restarted at that point if these conditions are met:

- If ON ERROR is absent, the number of times that the transaction has been restarted is compared with the global restart limit specified by the MAXTMFRESTARTS option of PATHWAY's SET PATHWAY command. If the number of restarts is less than that limit, the transaction is restarted with a new transaction identifer, the RESTART-COUNTER special register is incremented by one, and the TERMINATION-STATUS special register remains set to one. If the number of restarts equals the transaction restart limit, the terminal is suspended but its execution can be resumed.
- If ON ERROR is present, the transaction is restarted, RESTART-COUNTER is incremented by one, TERMINATION-STATUS remains set to one, and the ON ERROR branch is executed. You can then determine whether or not the transaction should be restarted in the ON ERROR branch of the SCREEN COBOL program; for example, RESTART-COUNTER can be compared to a local restart limit established within the program.

If the terminal is already in transaction mode when BEGIN-TRANSACTION is issued, the terminal is "suspended for pending abort"; the current transaction is backed out and terminal execution cannot be resumed with a RESUME command.

Example

The code sequence

```
enter data
.
.
ACCEPT screen
BEGIN-TRANSACTION
   ON ERROR PERFORM check error.
IF abort-flag NOT = 0
   GO TO enter-data.
.
.
SEND .....
END-TRANSACTION
.
.
stop trans.
  GO TO enter data
.
.
check error.
  MOVE 0 TO abort-flag.
IF TERMINATION-STATUS = 1
   IF RESTART-COUNTER > 2
      ABORT-TRANSACTION
      DISPLAY "Nope" IN MSG
      MOVE 1 TO abort-flag.
```

- accepts input data from the operator
- starts a new transaction
- in the event of an error, checks to determine if this transaction has been restarted more than two times. If it has, the transaction is aborted and the operator is asked to enter the data again. If not, another attempt is made to process the transaction.

END-TRANSACTION

END-TRANSACTION indicates that the transaction is complete. When this verb is successfully executed, the data-base updates made by the transaction become permanent, the terminal leaves transaction mode, and the special register TRANSACTION-ID is set to SPACES.

If TMF rejects END-TRANSACTION, the SCREEN COBOL program is restarted at the BEGIN-TRANSACTION point.

The END-TRANSACTION verb syntax is simply:

```
END-TRANSACTION
```

with no options.

If the terminal is not in transaction mode when END-TRANSACTION is executed, it is "suspended for pending abort"; terminal execution cannot be resumed with a RESUME command.

RESTART-TRANSACTION

RESTART-TRANSACTION is used when the SCREEN COBOL program detects an error that may be temporary, and decides to abandon the current attempt and retry the transaction. When this verb is executed,

- the current execution of the transaction will be backed out
- the transaction will be restarted at the BEGIN-TRANSACTION point with a new transaction identifier
- the special register RESTART-COUNTER will be incremented by one.

The RESTART-TRANSACTION verb syntax is simply:

```
RESTART-TRANSACTION
```

with no options.

The restart due to executing RESTART-TRANSACTION counts as a restart for purposes of the global transaction restart limit.

If the terminal is not in transaction mode when RESTART-TRANSACTION is executed, the terminal is "suspended for pending abort"; terminal execution cannot be resumed with a RESUME command.

SECTION 8

RELATED GUARDIAN PROCEDURES

Programmers writing or adapting applications for TMF must use at least some of these callable procedures:

- ABORTTRANSACTION aborts a transaction.
- ACTIVATERECEIVETRANSID is used when programming $RECEIVE-queuing servers; it restores the transaction identifier associated with a queued message request (for which REPLY has not been executed) that was previously acquired by reading $RECEIVE.
- BEGINTRANSACTION causes TMF to create a new transaction identifier.
- ENDTRANSACTION causes the data-base changes associated with a transaction identifier to be committed.
- GETTMPNAME obtains the dummy device name of the transaction monitor process (TMP).
- GETTRANSID obtains the transaction identifier of the calling process.
- RESUMETRANSACTION is used when programming NonStop requester processes or requester processes which have multiple concurrent active transactions; it restores a transaction identifier associated with a previous call on BEGINTRANSACTION.

The GUARDIAN operating system defines a set of literals for programmers to use in place of the error numbers returned by these procedures. Each of these literals begins with FE (for "file error"). For example, the literal for error 0 (no error) is FEOK and that for error 30 (failure to obtain an LCB) is FENOLCB. All of these literals are in the system file LDECLARE (for a NonStop 1+ system) or DERROR (for a NonStop system).

TRANSACTION IDENTIFIERS

In a Tandem system with TMF, each transaction is a uniquely-identified entity. Each transaction is distinguished from other transactions by a four-word transaction identifier, which is created by a successful call to the BEGINTRANSACTION procedure.

For programming purposes (as opposed to TMFCOM commands), the form of the transaction identifier is:

`transid[0].<0:7>`	contains 1 plus the EXPAND system number of the system in which BEGINTRANSACTION was called. (Note: this is for the *internal* form only; TMFCOM commands accept and display the actual system number.) This number is 0 for a system which is not part of a network. The system number identifies the home node of the transaction.
`transid[0].<8:15>`	contains the number of the processor in which BEGINTRANSACTION was executed.
`transid[1-2]`	contains a doubleword sequence number to make the transaction identification unique.
`transid[3]`	contains a "crash count" indicating the number of times the home node (of the transaction) has had a total system failure since the last time the TMFCOM command INITIALIZE TMF was issued on the home node.

ABORTTRANSACTION

ABORTTRANSACTION aborts and backs out a transaction. When this procedure is called by the process that issued BEGINTRANSACTION (or its backup), TMF backs out the data-base changes made for the process's current-transaction identifier. The syntax of the ABORTTRANSACTION procedure call is:

```
!INT: FUNCTION! ABORTTRANSACTION
```

ABORTTRANSACTION returns one of these file-error numbers:

0 (no error)

30 (failure to obtain an LCB)

75 (the requesting process has no current-transaction identifier)

76 (the transaction is in the process of ending)

78 (invalid or obsolete transaction identifier: the transaction was not begun by this process pair or is no longer in the system)

Any disc process (in a NonStop system) or audit process (in a NonStop 1+ system) that receives a subsequent I/O request for the aborting transaction identifier will reject the request with file error 97 (transaction aborted) or 78 (invalid or obsolete transaction identifier).

When ABORTTRANSACTION returns to the caller, the transaction has not been backed out, but the transaction's state has changed from active to aborting. Later, the backout process will back out the transaction by restoring its before-images to all disc files that it changed. When backout is complete, the locks held for that transaction identifier are released. This means that if the transaction is restarted with a new transaction identifier (after ABORTTRANSACTION returns) before the backout process has finished backing out the transaction, the new transaction will wait for backout to complete if it is queued on a record lock held by the aborted transaction identifier.

ACTIVATERECEIVETRANSID

ACTIVATERECEIVETRANSID is used in the coding of $RECEIVE-queuing servers; that is, servers that can read requests from $RECEIVE before replying to previously read $RECEIVE requests. When a server calls this procedure with a message tag obtained by a call to the GUARDIAN LASTRECEIVE or RECEIVEINFO procedures, the transaction identifier of the message associated with the tag becomes the current-transaction identifier for the server process. See the *System Procedure Calls Manual* or the *GUARDIAN Operating System User's Guide* for more information on LASTRECEIVE, RECEIVEINFO, and $RECEIVE-queuing servers.

The syntax of the ACTIVATERECEIVETRANSID procedure call is:

```
CALL ACTIVATERECEIVETRANSID ( message-tag )
```

where

```
  message-tag INT:value
```

identifies a message request from the group of requests that are currently queued by the server; it is the same parameter that will be passed by the server to the REPLY procedure. The *message tag* must be an integer between 0 and *receivedepth* – 1, inclusive, that is currently associated with a queued message.

A condition code setting of < (CCL) indicates an error. A condition code setting of = (CCE) indicates that ACTIVATERECEIVETRANSID was successful. A condition code setting of > (CCG) is never returned by ACTIVATERECEIVETRANSID.

ACTIVATERECEIVETRANSID returns a condition code only. A call to the FILEINFO procedure for more information on the $RECEIVE file returns one of these values as the file error:

0	(no error)
2	(*receivedepth* = 0)
16	($RECEIVE file not open)
74	(invalid *message tag* value)
75	(the message specified by the tag has no associated transaction identifier)

The FILEINFO procedure is described in the *System Procedure Calls Manual* for your system.

BEGINTRANSACTION

BEGINTRANSACTION starts a new transaction. When this procedure is called, TMF creates a new transaction identifier that becomes the current-transaction identifier for the process issuing BEGINTRANSACTION. The syntax of the BEGINTRANSACTION procedure call is:

```
!INT: function! BEGINTRANSACTION [ ( trans-begin-tag ) ]
```

where

`trans-begin-tag INT (32): ref`

> is passed a value that identifies the new transaction identifier among other transaction identifiers that the calling process pair has begun. This parameter is required by NonStop processes and by processes which have multiple concurrent active transactions.

BEGINTRANSACTION returns one of these file-error numbers:

0	(no error)
22	(parameter is out of bounds)
30	(failure to obtain an LCB)
82	(transaction processing disabled)
83	(the process has begun more concurrent transactions than can be handled)
84	(TMF is not configured)

The value returned to the *trans-begin-tag* can be passed to the RESUMETRANSACTION procedure to restore to currency a transaction identifier that was previously begun by this process (or its backup). See the explanation of RESUMETRANSACTION, in this section, for more details on this operation.

When BEGINTRANSACTION is executed, it increments by one or more the sequence-number counter of its processor. The value of the sequence-number counter is placed in words 1 and 2 of the transaction identifier. (See the introduction to this section for a breakdown of the transaction identifier's form.)

ENDTRANSACTION

ENDTRANSACTION commits the data-base changes associated with a transaction identifier. When this procedure is called by the process (or its backup) that issued BEGINTRANSACTION, TMF attempts to commit the transaction. If the action completes successfully, the changes made by the transaction will be permanent and the locks held for the transaction will be released. The syntax of the ENDTRANSACTION procedure call is:

```
!INT: FUNCTION! ENDTRANSACTION
```

ENDTRANSACTION returns one of these file-error numbers:

0 (no error)

30 (failure to obtain an LCB)

75 (the requesting process has no current-transaction identifier)

78 (invalid transaction identifier: transaction was not begun by this process pair or is no longer in the system)

81 (one or more no-wait I/O requests that transmitted the transaction being ended are outstanding)

90 (the transaction was aborted due to deletion of the BEGINTRANSACTION process)

92 (the transaction was aborted because the path to a participating network node is down)

93 (the transaction was aborted by the system for spanning too many audit files)

94 (the transaction was aborted by the system in response to an operator command)

97 (the transaction was aborted by a previous call to ABORTTRANSACTION)

ENDTRANSACTION is, by default, a waited operation unless the calling process has the transaction pseudofile (TFILE) open at the time of the ENDTRANSACTION call. In this case, it is a no-wait operation. If it is a no-wait operation, the errors that can be returned from the ENDTRANSACTION call itself are 30, 75, 78, and 81. Errors 90, 92, 93, 94, and 97 are returned as file errors on the TFILE after the call to AWAITIO.

GETTMPNAME

GETTMPNAME obtains the symbolic name of the TMP's logical device name. Opening the transaction pseudofile (TFILE) requires use of that symbolic name, which will normally be $TMP. The syntax of the GETTMPNAME procedure call is:

```
!INT: function! GETTMPNAME ( device-name )
```

where

```
device-name INT: ref: 12
```

is an array of 12 words to which GETMPNAME returns the name of the dummy device that the TMP is configured as. If the TMP is not configured, the *device name* will be all blanks.

GETTMPNAME returns one of these file-error numbers:

0 (no error)

22 (parameter is out of bounds)

84 (TMF is not configured)

The code fragment below demonstrates the use of GETTMPNAME in opening a TFILE:

```
INT        tmp^dummy^name [0:11];
INT        err^code, tfile^number;
err^code   := GETTMPNAME (tmp^dummy^name);      ! FEOK is the mnemonic
IF err^code=FEOK                                ! for file-error 0
  then CALL OPEN (tmp^dummy^name, tfile^number, ...);
  else ...
```

GETTRANSID

GETTRANSID obtains the transaction identifier of the calling process. When this procedure is executed, TMF returns the current-transaction identifier of the calling process. The syntax of the GETTRANSID procedure call is:

```
!INT: function! GETTRANSID ( transid )
```

where

```
  transid INT: ref 4
```

is an array of four words to which GETTRANSID returns the current-transaction identifier. (See the introduction to this section for a breakdown of the transaction identifier's form.)

GETTRANSID returns one of these file-error numbers:

0 (no error)

22 (a parameter is out of bounds)

75 (the requesting process has no current-transaction identifier)

On NonStop (not NonStop 1+) systems, GETTRANSID can also return one of these error numbers:

82 (transaction processing is disabled)

201 (the current path to the device is down)

RESUMETRANSACTION

RESUMETRANSACTION restores to currency a transaction identifier created by a previous call to BEGINTRANSACTION; it is called with the transaction tag returned by the call to BEGINTRANSACTION. The transaction identifier shown by the tag becomes the current-transaction identifier for the process calling RESUMETRANSACTION. The syntax of the RESUMETRANSACTION procedure call is:

```
!INT: function! RESUMETRANSACTION ( trans-begin-tag )
```

where

```
trans-begin-tag INT(32): value
```

is the value returned by the optional *trans-begin-tag* parameter of BEGINTRANSACTION. If the value of this parameter is 0D, the current-transaction identifier of the calling process is reset to 0, indicating no transaction identifier.

RESUMETRANSACTION returns one of these file-error numbers:

0 (no error)

76 (the transaction is in the process of ending)

78 (*trans-begin-tag* is invalid or obsolete, or the transaction was not begun by this process pair)

90 (the transaction was aborted due to deletion of the BEGINTRANSACTION process)

92 (the transaction was aborted because the path to a participating network node is down)

93 (the transaction was aborted by the system for spanning too many audit files)

94 (the transaction was aborted by the system in response to an operator command)

97 (the transaction was aborted by a previous call to ABORTTRANSACTION)

If the transaction identifier identified by *trans-begin-tag* was begun by the calling process or its backup, and is still active, it becomes the current-transaction identifier for the calling process even if an error is returned.

RESUMETRANSACTION does not change the current-transaction identifier for the backup of the calling process.

APPENDIX A

ERROR MESSAGES

The *System Messages Manual* lists the error messages that are returned by GUARDIAN procedures and the console error messages that are generated by TMF and other processes.

Console messages are displayed in a format similar to this:

```
89 11:43 21APR81 FROM 004,01,018 LDEV 0022 CU % 420 BACKOUT
ERROR #0011 TRANSACTION SEQ #00000238
```

In this example, 89 is the message number, and is followed by the time and date. The message was generated in system number 004, CPU number 01, and the process whose identification number (PIN) is 018. Logical device number (LDEV) 0022 and controller unit number (CU) 420 refer, in this case, to the TMP.

APPENDIX B

BASIC COMMANDS AND FILE IDENTIFIERS

This appendix describes the basic commands in detail. These commands are similar to those used in many Tandem products. Before reading these descriptions, however, it may be useful to review the form of a file identifier.

FILE-SET IDENTIFIERS

Each disc file in the GUARDIAN system is identified by a unique, symbolic file name. The name, and therefore the location, of a disc file is determined in several parts:

- The *system name* identifies a specific system within a network.
- The *volume name* identifies a physical disc pack mounted on a disc unit.
- The *subvolume name* identifies a related set of files, as defined by the user.
- The *disc file name* identifies a specific file within the subvolume.

File names are represented to Tandem subsystem programs by these parts, concatenated into a contiguous string with each part separated from the next by a period:

```
\system.$volume.subvolume.file
```

Note that the system name is immediately preceded by a backslash (\ ) and the volume name is immediately preceded by a dollar sign ($).

You can omit leading elements of the identifier, as

```
$volume.subvolume.file
```

or

```
subvolume.file
```

or even just

```
file
```

When only part of a file identifier is supplied as a command parameter, the internal representation of the file name is expanded into the full identifier. As a minimum, a partial identifier must include the file name. An omitted volume or subvolume name assumes the GUARDIAN Command Interpreter's current default volume or subvolume name. An omitted system name defaults to the GUARDIAN Command Interpreter's local system name.

In many instances, you can substitute asterisks for one or more elements of a file identifier, as in

```
\system.$volume.*.*
```

An asterisk indicates that all possible instances of the replaced identifier element are specified. Thus one identifier can specify a large set of files. This example specifies the set of all files in all subvolumes in the indicated volume on the indicated system.

Where an identifier element is specified as an asterisk, all elements to the right of that element must also be asterisks. Thus

```
$VOL1.*.FILE1
```

is not a valid identifier.

DEVICE AND PROCESS NAMES

Each process and each device, such as a tape drive, printer, or disc, is identified by a unique, symbolic name. This name is similar to the first half of a disc-file identifier:

- The *system name* identifies a specific system within a network.

- The *device name* or *process name* identifies the specific device or process.

This name is expanded in a manner similar to that used for a file identifier; if no system name is specified, the device or process is assumed to be on the local system. For instance,

```
\TSB.$TAPE1
```

might specify a particular tape drive on system \TSB, but if you are on that system you could just use $TAPE1.

DESCRIPTIONS OF BASIC COMMANDS

Basic commands are associated with control of the GUARDIAN Command Interpreter. Table B-1 summarizes the basic commands and their functions.

Table B-1. TMFCOM Basic Commands

Command	Function
CMDSYS	sets the default system for expansion of any file names, with the exception of OBEY file names.
CMDVOL	sets the default volume and subvolume for expansion of any file names, with the exception of OBEY file names.
EXIT	terminates execution of TMFCOM.
FC	fixes a command, using facilities of the FIXSTRING procedure.
HELP	displays information about TMFCOM commands.
OBEY	reads commands from a command file.
OBEYSYS	sets the default system for expansion of OBEY file names.
OBEYVOL	sets the default volume and subvolume for expansion of OBEY file names.
OUT	redirects listing output.
SYSTEM	sets the default system for expansion of all file names.
VOLUME	sets the default volume and subvolume for expansion of all file names.

CMDSYS Command

The CMDSYS command sets the default system name for file references; otherwise, the default system is the system currently in use. This command does not affect references to OBEY files. The syntax of this command is:

```
CMDSYS [ \system ]

where

   system

      is a GUARDIAN system name.
```

If the CMDSYS command is not issued, file references are expanded according to the default settings in effect when the GUARDIAN Command Interpreter was started. If the system name is not included in the command, the default is reset for local file-name expansion. This is not the same as specifying the local system name: omitting the system name in this command permits references to long device names, which are prohibited in network-form file names.

If the specified system name is invalid, or would be invalid when combined with the current CMDVOL setting, an error occurs. An error message is displayed and the CMDSYS setting is not changed.

For example, the command

```
CMDSYS \NY
```

specifies that the default system for future file references will be system \NY.

CMDVOL Command

The CMDVOL command sets the default volume and subvolume names for future file references. This command does not affect references to OBEY files. The syntax of this command is:

```
CMDVOL { $volume                 }
       { [ $volume. ] subvolume }
```

where

`volume`

is a GUARDIAN volume name.

`subvolume`

is a GUARDIAN subvolume name.

If the CMDVOL command is not issued, file references are expanded according to the default settings in effect when the GUARDIAN Command Interpreter was started. If either the volume name or subvolume name is omitted, the previous setting applies.

If either part of the new specification is invalid, or would be invalid when combined with the current defaults and the current CMDSYS setting, an error occurs. An error message is displayed and the CMDVOL defaults are not changed.

For example, the command

```
CMDVOL $MKT.ABC
```

specifies that the default volume for future file references will be $MKT and the default subvolume will be ABC.

EXIT Command

The EXIT command terminates the current OBEY file or command. The EXIT command syntax is simply:

```
EXIT
```

with no options.

When EXIT appears in a command string or OBEY file, TMFCOM terminates.

FC Command

The FC command provides the ability to edit or repeat a command line. The syntax of the FC command is simply:

```
FC
```

with no options.

When this command executes, it displays the previous command line and prompts for editing input with a period (.). FC accepts three subcommands:

```
R replacement-string
```

which replaces one or more characters

```
I insertion-string
```

which inserts one or more characters

```
D
```

which deletes one character.

Subcommands and their associated strings are entered beneath the displayed command line and terminated with a carriage return. Replacement, insertion, and deletion begins with the character positioned directly above the subcommand (*R*, *I*, or *D*).

Subcommand *R* replaces characters in the command line with *replacement-string* on a one-for-one basis. Subcommand *I* inserts characters in the command line with *insertion-string* on a one-for-one basis. Subcommand *D* deletes the character above it in the command line; the subcommand can be repeated for each character that is to be deleted. If a string but no subcommand is entered, *R* is assumed.

After the line is edited, FC again displays the command line and prompts for another subcommand. FC terminates when it receives only a carriage return; the corrected command line is then executed.

A BREAK anywhere terminates the FC command without execution. A double slash (//) in the first two character positions, immediately followed by a carriage return, terminates the FC command without execution.

Examples of FC subcommand positioning:

```
the subbsysten
      d            delete the extra letter b
   i enable        insert the word enable
             rm    replace the letter n with m
```

HELP Command

The HELP command displays the syntax of subsystem commands. The syntax of the HELP command is:

```
HELP { command-name      }
     { < symbol-name >   }
```

where

`command-name`

is the name of a TMFCOM command whose syntax is to be displayed.

`<symbol-name>`

is the name of a parameter used in the description provided by the HELP command. The enclosing set of angle brackets is required.

If parameters are omitted, the names of all TMFCOM commands are displayed.

OBEY Command

The OBEY command causes commands to be read from a specified file. Its syntax is:

```
OBEY file
```

where

`file`

is a GUARDIAN file identifier, specifying a file which contains one or more valid TMFCOM commands.

If you do not supply the *file* system, volume, or subvolume with the OBEY command, TMFCOM uses the defaults supplied by the OBEYSYS and OBEYVOL commands or by the SYSTEM and VOLUME commands.

Commands are read from the named file and processed until the end of the file, at which point the OBEY file is closed and command input reverts to the file from which the OBEY command was read. Additional OBEY commands can appear within an OBEY file; OBEY files can be nested to a depth of four.

If default settings such as VOLUME or OBEYVOL are changed in an OBEY file, these settings are not automatically returned to their previous states.

If any part of the specification is invalid, if the file does not exist, or if the file cannot be opened, an error occurs. An error message is displayed and the current source for command input is not changed.

If an error occurs during execution of a command from an OBEY file, the OBEY file is terminated; if it contains any subsequent commands, they are not executed.

OBEYSYS Command

The OBEYSYS command sets the default system name for OBEY file-name expansion. The syntax of the OBEYSYS command is:

```
OBEYSYS [ \system ]
```

where

`system`

is a GUARDIAN system name.

If the OBEYSYS command is not issued, the default settings in effect when TMFCOM was started are used. If the *system* is omitted, the default is set for local file name expansion. Note that this is not the same as specifying the local system name: issuing OBEYSYS without the *system* option permits references to long device names, which are prohibited in network-form file names.

If the specified system name is invalid, or would be invalid when combined with the current OBEYVOL setting, an error occurs. An error message is displayed and the OBEYSYS setting is not changed.

OBEYVOL Command

The OBEYVOL command sets the default volume and subvolume names for OBEY file-name expansion. The syntax of the OBEYVOL command is:

```
OBEYVOL { $volume                  }
        { [ $volume. ] subvolume   }
```

where

`volume`

is a GUARDIAN volume name.

`subvolume`

is a GUARDIAN subvolume name.

If the OBEYVOL command is not issued, the default settings in effect when TMFCOM was started are used. If either the volume name or subvolume name is omitted, the previous setting applies. If both names are omitted and the file from which the OBEYVOL command was read is a disc file, the defaults for OBEY file name expansion are set to the system, volume, and subvolume of the current command input file. If both names are omitted and the file from which the OBEYVOL command was read is not a disc file, an error occurs; an error message is displayed and the OBEYVOL defaults are not changed.

If either part of the new specification is invalid, or would be invalid when combined with the current defaults and the current OBEYSYS setting, an error occurs. An error message is displayed and the OBEYVOL defaults are not changed.

OUT Command

The OUT command directs all output, other than operator prompts and console messages, to a specified destination. The OUT command syntax is:

```
{OUT file     }
{/ OUT file / }
```

where

`file`

is a GUARDIAN file identifier, specifying the new output destination. It can specify a disc file, a device (such as a printer), or a process (such as the spooler).

The first form of the OUT command, without the slash (/) delimiters, causes permanent redirection of the output. For example, the command sequence

```
OUT $S.#PRINT
INFO TMF
```

redirects INFO TMF output (and all subsequent output from commands issued during the current iteration of TMFCOM) to the spooler.

The other form, with the delimiters, causes temporary redirection of the output. You use this form as part of another TMFCOM command, by positioning it immediately after the first word of the other command name and before any other part of that command. For example,

```
INFO /OUT SMITH.QUERY/ TMF
```

directs INFO TMF output to file SMITH.QUERY on the current default disc volume. If the file identifier is invalid, or the file does not exist, or the file cannot be opened, an error message is displayed and the listing is not redirected.

SYSTEM Command

The SYSTEM command sets the default system for expansion of any file names. The syntax of the SYSTEM command is:

```
SYSTEM [ \system ]
```

where

`system`

is a GUARDIAN system name.

If the SYSTEM command is not issued, the default settings in effect when TMFCOM was started are used. If the system name is omitted, the default is set for local file-name expansion. Note that this is not the same as specifying the local system name. Omitting the system name in this command permits references to long device names, which are prohibited in network-form file names.

A SYSTEM command is equivalent to entering both an OBEYSYS and a CMDSYS command with the same specification.

If the specified system name is invalid, or would be invalid when combined with the current VOLUME setting, an error message is displayed and the SYSTEM setting (or the OBEYSYS and CMDSYS setting) is not changed.

VOLUME Command

The VOLUME command sets the default volume and subvolume names for subsequent file references. The syntax of the VOLUME command is:

```
VOLUME { $volume                 }
       { [ $volume. ] subvolume  }
```

where

`volume`

is a GUARDIAN volume name.

`subvolume`

is a GUARDIAN subvolume name.

If the VOLUME command is not issued, the default settings in effect when TMFCOM was started are used. If either the volume name or subvolume name is omitted, the previous setting applies.

A VOLUME command is equivalent to entering both an OBEYVOL and a CMDVOL command with the same specification. However, this does not mean that the settings for OBEYVOL and CMDVOL would necessarily be identical after a VOLUME command. For example, if the default for OBEYVOL is $V1.A and the default for CMDVOL is $V2.B, then a subsequent command of

```
VOLUME X
```

yields $V1.X for OBEYVOL and $V2.X for CMDVOL.

If either part of the new specification is invalid, or would be invalid when combined with the current defaults and the current SYSTEM setting (or with the current OBEYSYS and CMDSYS setting), an error occurs. An error message is displayed and the VOLUME setting (or OBEYVOL and CMDVOL setting) is not changed.

INDEX

READER COMMENT CARD

Tandem welcomes your comments on the quality and usefulness of its software documentation. Does this manual serve your needs? If not, how could we improve it? Your comments will be forwarded to the writer for review and action, as appropriate.

If your answer to any of the questions below is "no," please supply detailed information, including page numbers, under Comments. Use additional sheets if necessary.

▶ Is this manual technically accurate?	Yes ☐	No ☐
▶ Is information missing?	Yes ☐	No ☐
▶ Are the organization and content clear?	Yes ☐	No ☐
▶ Are the format and packaging convenient?	Yes ☐	No ☐

Comments

Name	Date
Company	
Address	
City/State	Zip

Transaction Monitoring Facility (TMF™) Reference Manual

NonStop™ Systems
NonStop 1+™ System

82541 A00

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 482 CUPERTINO, CA, U.S.A.

POSTAGE WILL BE PAID BY ADDRESSEE

Tandem Computers Incorporated
Attn: Manager—Software Publications
Location 01, Department 6350
19333 Vallco Parkway
Cupertino CA 95014-9990

TAPE TAPE

Tandem Computers Incorporated
19333 Vallco Parkway
Cupertino, CA   95014-2599