# NonStop AutoTMF FAQ

To obtain more detailed information on the topics mentioned in the following entries, please consult the NonStop AutoTMF User's Guide, available in the HPE NonStop Technical Library. For more specific technical questions not covered here, contact the HPE GNSC support organization. If you are registered as an HPE Passport user, you can also consult the NonStop eServices Portal to access the Knowledge database.

## USAGE

*Why do I need AutoTMF?*
If you want to replicate your database for disaster recovery purposes using NonStop RDF or HPE Shadowbase, your database must be audited, because these tools use the TMF audit trails to perform the replication and keep the primary and backup databases in sync.

If your database is not audited, you need AutoTMF to audit your database without having to recode your application programs.

AutoTMF also provides other advantages of auditing, such as improved insert performance and online dumping and file recovery to last transaction.

## PREREQUISITES

*Does AutoTMF require TMF?*
Yes. AutoTMF operates on audited files so the TMF subsystem must be active prior to auditing files and running AutoTMF-enabled applications.

*Does AutoTMF require RDF?*
No. AutoTMF allows programs that do not use TMF transactions to access audited files. For replication purposes, it is compatible with any software product that is audit trail-based, such as NonStop RDF and HPE Shadowbase.

*What release of Guardian is required for AutoTMF?*
AutoTMF an Independent product (IP) and works on all NonStop hardware and OS versions supported by HPE.

*Does AutoTMF require the use of a super ID?*
No. AutoTMF contains no privileged code and does not require the use of a super ID. It integrates with the security in place for your application.

## SCOPE OF FUNCTIONALITY

*How is AutoTMF functionality enabled?*
Once AutoTMF is installed, there are 2 steps that enable AutoTMF:

- A “prepare” of the application object files sets up the interception of Enscribe calls by the AutoTMF runtime.
- The auditing of the files for which AutoTMF will be generating transactions, referenced as “automatic” transactions.

Depending on the application, it might be necessary to configure some attributes to maintain compatibility with non-audited behavior, such as duration of transaction, number of updates per transaction, number of concurrent transactions, etc.

*Does AutoTMF require any change to the application logic? Must I recompile my programs?*
No. The AutoTMF “preparation” step operates strictly on object files. Even stripped object files can be prepared.

*How do application programs invoke AutoTMF?*
AutoTMF does not require any invocation by applications. Once the applications have been prepared, accesses to audited files are monitored and AutoTMF supplies TMF transactions whenever necessary.

*What type of files can be audited and managed by AutoTMF?*
AutoTMF supports structured audited Enscribe files: Key-sequenced, entry-sequenced and relative files, as well as queue files. It also supports audited unstructured files but is not recommended for edit files or object files.

*What application languages are supported by AutoTMF?*
AutoTMF can be used with applications written in Cobol, C and TAL. It operates on all types of object types: TNS (100), TNS/E (800) and TNS/X (500).

*Can any of the HPE utilities use AutoTMF to access audited files?*
The following utilities can be enabled for AutoTMF:

- FUP
- Spooler
- SORTPROG
- FTP

See paragraph “Preparing HPE Utilities” in the “Usage Guidelines” chapter of the NonStop AutoTMF User’s Guide for instructions on how to prepare these utilities for AutoTMF.

*Can I audit my Queue files?*
Yes, AutoTMF provides basic support for auditing Queue files. See section “Auditing Enscribe Queue Files” in chapter “Usage Guidelines” of the NonStop AutoTMF User’s Guide for details.

*What about NonStop process pairs?*
AutoTMF is compatible with NonStop process pairs. See section “NonStop AutoTMF and Process Pairs” in chapter “Usage Guidelines” of the NonStop AutoTMF User’s Guide for details.

*What type Enscribe file operations does AutoTMF support?*
All Enscribe I/O procedure calls are intercepted and processed. These include all the various CREATE, OPEN, POSITION, READ, and WRITE operations.

In addition, renaming audited files and replicating LOAD operations can be configured as advanced features. See chapter "Advanced Replication Features" in the NonStop AutoTMF User's Guide for details on replicating RENAME and LOAD.

*Does AutoTMF support transactions for audited SQL tables?*
No, AutoTMF only supports audited Enscribe files.

*Is there an easy way to identify which programs would need to use AutoTMF to audit this file set?*
There is no benefit in trying to select which programs should be prepared. To simplify the object file management, it is recommended to prepare all application object files  AutoTMF will determine when automatic transactions are required. If a program does not access audited files, AutoTMF does not generate transactions.

*What if I already have TMF in my programs but not all files are audited?*
AutoTMF coexists with applications that are "TMF-aware". Files that were previously unaudited can be configured to be managed by AutoTMF as soon as they are audited. Since automatic transactions are always committed, updates to these files are committed even if the application fails or calls ABORTRANSACTION, maintaining compatibility with the program's prior behavior.

*None of our data files are audited today. To use AutoTMF, do we have to audit all files accessed by our programs?*
No. You choose which files to audit. AutoTMF generates automatic transactions for audited files whenever they are needed; access to unaudited files remains unchanged.

*We run Base24 in our shop. Can I use AutoTMF to audit the Base 24 files? If so, does it support the Full Refresh feature?*
Yes, there are many NonStop Base24 installations that use AutoTMF around the world. See chapter "Preparing Base24 Programs" in the NonStop AutoTMF User's Guide for details on how to enable AutoTMF for Base24.

AutoTMF also supports the Full Refresh feature as of SPR T0581ABE (16NOV2017). See paragraph "Replicating Files Managed by Base24 Full Refresh" in the "Advanced Replication Features" in the NonStop AutoTMF User's Guide for details on how to implement this type of replication.

## APPLICATION MANAGEMENT

*If I move or copy a prepared object to another location, do I need to prepare it again?*
No, preparation is more like acceleration, except that it takes a fraction of a second per object file and unless a program is recompiled, it does not need preparation again.

When a program is prepared, it will reference a user library or DLL. So if you move the prepared program to a different system, you must take care that the user library/DLL exists and has the same file name on the new system.

*How do I determine which object files are prepared for AutoTMF?*
Use the AutoTMF INFO PROGRAM command to generate a report on all object files in a file set.

*If a database file that is protected by AutoTMF is moved to another location, apart from setting the audit flag on the database file, is there any further action that needs to be taken?*

If no special AutoTMF attributes have been configured, then no further action is required. If some special AutoTMF attributes have been configured for the file, then the AutoTMF configuration must be updated to specify attributes for the new file location.

*If the user forgot to prepare on one or more of the object files, what happens when the program writes to an audited file?*
If a program is not prepared, a record lock, update or insert into an audited file will fail with an error 75 ("Requesting process has no current process transaction identifier") unless the program has a transaction of its own. TMF guarantees that no update can be performed on an audited file without an active transaction. The program will probably abend because it does not expect TMF-related I/O errors.

*AutoTMF is implemented as a user library. What happens if the application programs already point to a user library?*
If the user library performs I/O operations that access audited files, then the library must first be prepared

For TNS (code 100) programs that already have a user library:

- The AutoTMF runtime must be combined with the user library using Binder
- The prepare and binding operations must be performed before the programs using the library can be prepared.

For native application programs (code 800 and 500) no further processing is required

See paragraph “Preparing Programs that Have a User Library” in chapter “Preparing Programs” of the NonStop AutoTMF User’s Guide for details.

*How does AutoTMF know when to start a transaction?*
AutoTMF intercepts Enscribe I/O calls issued by an application program. If a transactional operation is issued against an audited file and no TMF transaction is active, AutoTMF starts an automatic transaction.

## AUTOMATIC TRANSACTIONS

*What is an automatic transaction?*
An automatic transaction is a TMF transaction that is generated by the AutoTMF runtime, as opposed to a process transaction, which is coded in the application.

*Does AutoTMF understand business transactions?*
AutoTMF does not claim to understand business transaction boundaries. However, the natural locking and unlocking behavior of an application normally causes AutoTMF to commit transactions at reasonable points in the processing of a business unit of work.

*How does AutoTMF know when to commit a transaction?*
In normal operation, AutoTMF will only commit a transaction when all records locked under that transaction are logically unlocked. So, it depends on consistent locking and unlocking of records in the application process. If a process does not unlock records in a timely manner, AutoTMF issues EMS warnings specifying the process, program and file that is not being unlocked.

AutoTMF attempts to improve performance by allowing multiple record locks and unlocks to be managed by a single transaction, but this

can be controlled by various configuration options to force transaction commit after certain events. As indicated above, it does not attempt to determine "business" transaction boundaries.

AutoTMF always tries to commit all database updates in the event of process termination or failure, even if the process has not unlocked all records. It tries to emulate normal unaudited file behavior, which "commits" all updates as soon as they are processed by the disk process.

AutoTMF has different commit points:

Conditional Commit Events – subject to lock state

- Call to READUPDATE on $RECEIVE
- Transaction elapsed time or number of updates per transaction - whichever comes first (configurable)
- NORMAL or STRONG ISOLATION event such as SERVERCLASS_SEND_ or WRITEREAD to device that commit the Auto TX instead of merely suspending it

Unconditional Commit Events - not subject to lock states because implicit unlocks

- Call to STOP or ABEND – implicit unlock on all files
- CLOSE of all files participating in an auto TX– implicit unlock
- AutoCommit limit reached (before reaching TMF AutoAbort)
- Process stopped through AutoTMF STOP PROCESS command

See chapter "Configuring Automatic Transactions" in the NonStop AutoTMF User's Guide for detailed descriptions of automatic transaction attributes and how to configure them.

*What happens in the case of a unilateral abort?*
If an external failure or a STOP command causes the program to terminate, active transactions are aborted by TMF including those generated by AutoTMF because AutoTMF has no control over the transactions in such situations.

The uncommitted updates performed by the program are rolled back. There are a few configuration options that help mitigate the effects of unilateral aborts.

*Does locking behavior of a program change when enabled for AutoTMF?*
The lock protocols of audited and unaudited access are significantly different. For example, locks on updated records are not released until the transaction commits, even if the program has unlocked the records.

AutoTMF is engineered to avoid lock contention and deadlocks, but complex sequences of database operations between cooperating processes may result in increased lock contention. In these cases, the user should specify stronger isolation between processes.

See chapter "Problem Resolution" of the NonStop AutoTMF User's Guide for a comparison of locking rules.

## PERFORMANCE & RESOURCES

*I have a program that writes an unstructured file using Bulk I/O operations. Can I use AutoTMF to replicate these files? What about writes to structured files using Bulk I/O?*
TMF and the file system allow unstructured writes or updates to an audited unstructured file, and AutoTMF supports unstructured access, including large data transfers, to audited unstructured files.

*Does AutoTMF impact application performance?*
AutoTMF itself adds very little extra processing to the application program. AutoTMF performance is TMF performance in that there is a slight increase in CPU usage but decreased I/O latency. In general, sequential access is significantly improved.

*What is the operational impact of AutoTMF?*
Once configured, AutoTMF requires very little management. The AutoTMF monitor must be started after a system cold load, but is fault-tolerant. It can also be configured to run as a generic process.

The operational aspects of auditing files are no different with AutoTMF than they would be if the application had been rewritten to use TMF transactions. Users should conform to the operations guidelines recommended by HPE.

## DIAGNOSTIC TOOLS

*Where does AutoTMF log its messages?*
In addition to being displayed on the home terminal of the application programs, the internal errors detected by the AutoTMF runtime are logged to an EMS collector. The default collector is $0. An alternate collector can be configured by setting the EMSCOLLECTOR global parameter. The AutoTMF monitor messages are logged to the same EMS collector.

*How do I troubleshoot problems in my AutoTMF environment?*
The most useful tools to diagnose problems are:

- The EMS log, where error messages are collected. AutoTMF comes with EMS filters that can be applied to extract TMF and AutoTMF events from the log and facilitate its perusal.
- An enhanced LISTLOCKS command. The most often reported AutoTMF problems are related to locking, because locking requirements in an audited environment differ from those in non-audited environments. While the FUP LISTLOCKS command is useful, the AutoTMF LISTLOCKS offers additional features, such as a deadlock detector which are more useful to analyse locking behavior. Locking problems are usually resolved by configuring file or object attributes that circumvent them.
- The trace facility. Virtually all file system and TMF operations performed by a program can be traced along with the data that is read and written by the program. Tracing also displays the active DEFINEs, assigns, the startup message, and process termination messages.

See the “Problem Resolution” chapter in the NonStop AutoTMF User’s Guide for further details on troubleshooting tools