

GT.M

Release Notes

V7.0-005

Empowering
the Financial World

FIS

Contact Information

GT.M Group
Fidelity National Information Services, Inc.
347 Riverside Drive
Jacksonville, FL 13220
United States of America

GT.M Support for customers: gtmsupport@fisglobal.com
Automated attendant for 24 hour support: +1 (484) 302-3248
Switchboard: +1 (484) 302-3160

Legal Notice

Copyright ©2022 Fidelity National Information Services, Inc. and/or its subsidiaries. All Rights Reserved.






Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts and no Back-Cover Texts.

GT.M™ is a trademark of Fidelity National Information Services, Inc. Other trademarks are the property of their respective owners.

This document contains a description of GT.M and the operating instructions pertaining to the various functions that comprise the system. This document does not contain any commitment of FIS. FIS believes the information in this publication is accurate as of its publication date; such information is subject to change without notice. FIS is not responsible for any errors or defects.

Revision History		
Revision 1.0	02 December 2022	V7.0-005

Table of Contents

V7.0-005	1
Overview	1
Conventions	1
Platforms	3
Platform support lifecycle	5
Additional Installation Instructions	5
Recompile	6
Rebuild Shared Libraries or Images	6
Compiling the Reference Implementation Plugin	6
Re-evaluate TLS configuration options	7
Upgrading to GT.M V7.0-005	7
Managing M mode and UTF-8 mode	8
Setting the environment variable TERM	9
Installing Compression Libraries	10
Change History	11
V7.0-005	11
Database	13
Language	15
System Administration	17
Other	19
Error and Other Messages	21
GDELOGFAIL 	21
SOCKCLOSE 	21
STACKCRIT 	21
STACKOFLOW 	21
TPFAIL 	22

This page is intentionally left blank.

V7.0-005

Overview

V7.0-005 adds audit logging facilities for DSE, GDE, and LKE. In addition, there are restriction settings for ZLINK, ZRUPDATE and SET \$ZROUTINES, an option to pin shared memory, and statistics for triggers, as well as various fixes. For more information, refer to the Change History section.

Items marked with the 🟢 symbol document new or different capabilities.

Please pay special attention to the items marked with the 🟡 symbol. as those document items that have a possible impact on existing code, practice or process. Please be sure to recompile all objects to ensure all the updates are in place.



Note

While FIS keeps message IDs and mnemonics quite stable, messages texts change more frequently as we strive to improve them, especially in response to user feedback. Please ensure you review any automated scripting that parses GT.M messages.

Conventions

This document uses the following conventions:

Flag/Qualifiers	-
Program Names or Functions	upper case. For example, MUPIP BACKUP
Examples	lower case. For example: mupip backup -database ACN,HIST /backup
Reference Number	A reference number is used to track software enhancements and support requests. It is enclosed between parentheses ().
Platform Identifier	Where an item affects only specific platforms, the platforms are listed in square brackets, e.g., [AIX]



Note

The term UNIX refers to the general sense of all platforms on which GT.M uses a POSIX API. As of this date, this includes: AIX and GNU/Linux x86_64.

The following table summarizes the new and revised replication terminology and qualifiers.

Pre V5.5-000 terminology	Pre V5.5-000 qualifier	Current terminology	Current qualifiers
originating instance or primary instance	-rootprimary	originating instance or originating primary instance. Within the context of a replication connection between two instances, an originating instance is referred to as source instance or source side. For example, in an B<-A->C replication configuration, A is the source instance for B and C.	-updok (recommended) -rootprimary (still accepted)
replicating instance (or secondary instance) and propagating instance	N/A for replicating instance or secondary instance. -propagateprimary for propagating instance	replicating instance. Within the context of a replication connection between two instances, a replicating instance that receives updates from a source instance is referred to as receiving instance or receiver side. For example, in an B<-A->C replication configuration, both B and C can be referred to as a receiving instance.	-updnotok
N/A	N/A	supplementary instance. For example, in an A->P->Q replication configuration, P is the supplementary instance. Both A and P are originating instances.	-updok

Effective V6.0-000, GT.M documentation adopted IEC standard Prefixes for binary multiples. This document therefore uses prefixes Ki, Mi and Ti (e.g., 1MiB for 1,048,576 bytes). Over time, we'll update all GT.M documentation to this standard.

✔ denotes a new feature that requires updating the manuals.

⚠ denotes a new feature or an enhancement that may not be upward compatible and may affect an existing application.

⊖ denotes deprecated messages.

⚠ denotes revised messages.

⊕ denotes added messages.


Platforms

Over time, computing platforms evolve. Vendors obsolete hardware architectures. New versions of operating systems replace old ones. We at FIS continually evaluate platforms and versions of platforms that should be Supported for GT.M. In the table below, we document not only the ones that are currently Supported for this release, but also alert you to our future plans given the evolution of computing platforms. If you are an FIS customer, and these plans would cause you hardship, please contact your FIS account executive promptly to discuss your needs.

Each GT.M release is extensively tested by FIS on a set of specific versions of operating systems on specific hardware architectures, we refer to the combination of operating system and hardware architecture as a platform. We deem this set of specific versions: Supported. There may be other versions of the same operating systems on which a GT.M release may not have been tested, but on which the FIS GT.M Group knows of no reason why GT.M would not work. We deem this larger set of versions: Supportable. There is an even larger set of platforms on which GT.M may well run satisfactorily, but where the FIS GT.M team lacks the knowledge to determine whether GT.M is Supportable and therefore deem them: Unsupported. Contact FIS GT.M Support with inquiries about your preferred platform.

As of the publication date, FIS supports this release on the hardware and operating system versions below. Contact FIS for a current list of Supported platforms. The reference implementation of the encryption reference plugin has its own additional requirements.

Platform	Supported Versions	Notes
IBM Power Systems AIX	7.1 TL 5, 7.2 TL 5	<p>Only 64-bit versions of AIX with POWER7 as the minimum required CPU architecture level are Supported.</p> <p>While GT.M supports both UTF-8 mode and M mode on this platform, there are problems with the AIX ICU utilities that prevent FIS from testing 4-byte UTF-8 characters as comprehensively on this platform as we do on others.</p> <p>Running GT.M on AIX 7.1 requires APAR IZ87564, a fix for the POW() function, to be applied. To verify that this fix has been installed, execute instfix -ik IZ87564.</p> <p>Only the AIX jfs2 filesystem is Supported. Other filesystems, such as jfs1 are Supportable, but not Supported. FIS strongly recommends use of the jfs2 filesystem on AIX; use jfs1 only for existing databases not yet migrated to a jfs2 filesystem.</p>
x86_64 GNU/Linux	Red Hat Enterprise Linux 7.9, 8.6, 8.7; Ubuntu 18.04 LTS, 20.04 LTS, and 22.04	<p>To run 64-bit GT.M processes requires both a 64-bit kernel as well as 64-bit hardware.</p> <p>GT.M should also run on recent releases of other major Linux distributions with a contemporary Linux kernel (2.6.32 or later), glibc (version 2.12 or later) and ncurses (version 5.7 or later).</p>

Platform	Supported Versions	Notes
	LTS; Amazon Linux 2	<p>Due to build optimization and library incompatibilities, GT.M versions older than V6.2-000 are incompatible with glibc 2.24 and up. This incompatibility has not been reported by a customer, but was observed on internal test systems that use the latest Linux software distributions from Fedora (26), Debian (unstable), and Ubuntu (17.10). In internal testing, processes either hung or encountered a segmentation violation (SIG-11) during operation. Customers upgrading to Linux distributions that utilize glibc 2.24+ must upgrade their GT.M version at the same time as or before the OS upgrade.</p> <p>GT.M requires a compatible version of the libtinfo library. On Red Hat, the ncurses-libs and ncurses-compat-libs packages contain the libtinfo library. On Debian/Ubuntu, libtinfo5 and libncurses5 packages contain the libtinfo library. If any of these packages is not already installed on your system, please install using an appropriate package manager.</p> <p>To support the optional WRITE /TLS fifth argument (the ability to provide / override options in the tlsid section of the encryption configuration file), the reference implementation of the encryption plugin requires libconfig 1.4.x.</p> <p>Only the ext4 and xfs filesystems are Supported. Other filesystems are Supportable, but not Supported. Furthermore, if you use the NODEFER_ALLOCATE feature, FIS strongly recommends that you use xfs. If you must use NODEFER_ALLOCATE with ext4, you must ensure that your kernel includes commit d2dc317d564a46dfc683978a2e5a4f91434e9711 (search for d2dc317d564a46dfc683978a2e5a4f91434e9711 at https://www.kernel.org/pub/linux/kernel/v4.x/ChangeLog-4.0.3). The Red Hat Bugzilla identifier for the bug is 1213487. With NODEFER_ALLOCATE, do not use any filesystem other than ext4 and a kernel with the fix, or xfs.</p> <p>Ubuntu 21.10 is Supportable.</p> <div style="display: flex; align-items: flex-start;"> <div style="margin-right: 10px;">  </div> <div> <p>Note</p> <ul style="list-style-type: none"> ● To use TLSv1.3 with OpenSSL 1.1.1 and up, you must recompile the reference encryption plugins ● RHEL 8 includes compat-openssl10.x86_64 for binaries compiled against OpenSSL 1.0.2 on RHEL 7 </div> </div>



Important

Effective V7.0-003, GT.M is no longer Supportable on the 32 bit x86 platform. Please contact your FIS account manager if you need ongoing support for GT.M on this platform.

Platform support lifecycle

FIS usually supports new operating system versions six months or so after stable releases are available and we usually support each version for a two year window. GT.M releases are also normally supported for two years after release. While FIS attempts to provide support for customers in good standing on any GT.M release and operating system version, our ability to provide support diminishes after the two year window.

GT.M cannot be patched, and bugs are only fixed in new releases of software.

Additional Installation Instructions


To install GT.M, see the "Installing GT.M" section in the GT.M Administration and Operations Guide. For minimal down time, upgrade a current replicating instance and restart replication. Once that replicating instance is current, switch it to become the originating instance. Upgrade the prior originating instance to become a replicating instance, and perform a switchover when you want it to resume an originating primary role.



Caution

Never replace the binary image on disk of any executable file while it is in use by an active process. It may lead to unpredictable results. Depending on the operating system, these results include but are not limited to denial of service (that is, system lockup) and damage to files that these processes have open (that is, database structural damage).

- FIS strongly recommends installing each version of GT.M in a separate (new) directory, rather than overwriting a previously installed version. If you have a legitimate need to overwrite an existing GT.M installation with a new version, you must first shut down all processes using the old version. FIS suggests installing GT.M V7.0-005 in a Filesystem Hierarchy Standard compliant location such as /usr/lib/fis-gtm/V7.0-005_arch (for example, /usr/lib/fis-gtm/V7.0-005_x86_64 on Linux systems). A location such as /opt/fis-gtm/V7.0-005_arch would also be appropriate.
- Use the appropriate MUPIP command (e.g. ROLLBACK, RECOVER, RUNDOWN) of the old GT.M version to ensure all database files are cleanly closed.
- Make sure gtmsecshr is not running. If gtmsecshr is running, first stop all GT.M processes including the DSE, LKE and MUPIP utilities and then perform a **MUPIP STOP *pid_of_gtmsecshr***.

- Starting with V6.2-000, GT.M no longer supports the use of the deprecated \$gtm_dbkeys and the master key file it points to for database encryption. To convert master files to the libconfig format, please click  to download the CONVDBKEYS.m program and follow instructions in the comments near the top of the program file. You can also download CONVDBKEYS.m from <http://tinco.pair.com/bhaskar/gtm/doc/articles/downloadables/CONVDBKEYS.m>. If you are using \$gtm_dbkeys for database encryption, please convert master key files to libconfig format immediately after upgrading to V6.2-000 or later. Also, modify your environment scripts to include the use of gtmcrypt_config environment variable.

Recompile

- Recompile all M and C source files.

Rebuild Shared Libraries or Images

- Rebuild all Shared Libraries after recompiling all M and C source files.
- If your application is not using object code shared using GT.M's auto-relink functionality, please consider using it.

Compiling the Reference Implementation Plugin

If you plan to use the example / reference implementation plugin in support of database encryption, TLS replication, or TLS sockets, you must compile the reference plugin in order to match the shared library dependencies specific to your platform. The instructions for compiling the Reference Implementation plugin are as follows:

1. Install the development headers and libraries for libgcrypt, libgpgme, libconfig, and libssl. On Linux, the package names of development libraries usually have a suffix such as -dev or -devel and are available through the package manager. For example, on Ubuntu_x86_64 a command like the following installs the required development libraries:

```
sudo apt-get install libgcrypt11-dev libgpgme11-dev libconfig-dev libssl-dev
```

Note that the package names may vary by distribution / version.

2. Unpack \$gtm_dist/plugin/gtmcrypt/source.tar to a temporary directory.

```
mkdir /tmp/plugin-build
cd /tmp/plugin-build
cp $gtm_dist/plugin/gtmcrypt/source.tar .
tar -xvf source.tar
```

3. Follow the instructions in the README.
 - Open Makefile with your editor; review and edit the common header (IFLAGS) and library paths (LIBFLAGS) in the Makefile to reflect those on your system.

- Define the `gtm_dist` environment variable to point to the absolute path for the directory where you have GT.M installed
- Copy and paste the commands from the README to compile and install the encryption plugin with the permissions defined at install time



The encryption plugin currently uses functionality that is deprecated in OpenSSL 3.0. This will be fixed in a future release.

Re-evaluate TLS configuration options

The GT.M TLS reference encryption plugin implements a subset of options as documented in the OpenSSL 1.0.2 man page for `SSL_set_options` which modify the default behavior of OpenSSL. Future versions of the plugin will enable new options as and when the OpenSSL library adds them. To enable options not supported by the GT.M TLS reference plugin, it is possible to create an OpenSSL configuration for GT.M processes. See the OpenSSL man page for "config".

Upgrading to GT.M V7.0-005

There are two upgrade paths available when you upgrade to GT.M V7.0-003 from a V6 version:

- **V7 Upgrade Path 1:** Perform a MUPIP EXTRACT -FREEZE on an existing V6 database and then perform MUPIP LOAD on an empty V7 database using replication to minimize downtime. You can also use the MERGE command to move data from V6 to the new V7 database; as with MUPIP EXTRACT, you need to keep the source data in a stable state. If you are using triggers, extract the triggers from the V6 database and load them in the new V7 database. We recommend you use replication to further stage the conversion and minimize down time.
- **V7 Upgrade Path 2:** Continue using your V6 databases with GT.M V7.0-003. You may wish to keep your latest V6 version available to create new database file, in case you do not wish to operate with files of differing format, as V7 releases always create V7 format databases.

Choose V7 Upgrade Path 1 if you anticipate a database file to grow up to 16Gi blocks or require trees of up to 11 levels. Note that the maximum size of a V7 database file having 8KiB block size is 114688GiB (8KiB*16Gi).

Choose V7 Upgrade Path 2 if you do not anticipate a database file to grow beyond the V6 database limit of 994Mi blocks or a tree depth limit of 7 levels. Note that the maximum size of a V6 database file having 8KiB block size is 7936GiB (8KiB*992Mi).

Other than the new maximum database file size that comes with V7 Upgrade Path 1, there is no difference between V7 Upgrade Path 1 and V7 Upgrade Path 2. You can choose V7 Upgrade Path 2 first and then later choose V7 Upgrade Path 1 if a need arises.



Note

To upgrade your database from V5 to V7, you need to first upgrade your database from V5 to a V6 database and then choose an appropriate V7 upgrade path. Refer to the appropriate V6 release notes for the database upgrade instructions or consult your GT.M support channel.



Important

In an upcoming release, the FIS GT.M team plans to provide in-place database upgrade with minimal downtime.

Managing M mode and UTF-8 mode

With International Components for Unicode® (ICU) version 3.6 or later installed, GT.M's UTF-8 mode provides support for Unicode® (ISO/IEC-10646) character strings. On a system that does not have ICU 3.6 or later installed, GT.M only supports M mode.

On a system that has ICU installed, GT.M optionally installs support for both M mode and UTF-8 mode, including a utf8 subdirectory of the directory where GT.M is installed. From the same source file, depending upon the value of the environment variable `gtm_chset`, the GT.M compiler generates an object file either for M mode or UTF-8 mode. GT.M generates a new object file when it finds both a source and an object file, and the object predates the source file and was generated with the same setting of `$gtm_chset/$ZCHset`. A GT.M process generates an error if it encounters an object file generated with a different setting of `$gtm_chset/$ZCHset` than that processes' current value.

Always generate an M object module with a value of `$gtm_chset/$ZCHset` matching the value processes executing that module will have. As the GT.M installation itself contains utility programs written in M, their object files also conform to this rule. In order to use utility programs in both M mode and UTF-8 mode, the GT.M installation ensures that both M and UTF-8 versions of object modules exist, the latter in the utf8 subdirectory. This technique of segregating the object modules by their compilation mode prevents both frequent recompiles and errors in installations where both modes are in use. If your installation uses both modes, consider a similar pattern for structuring application object code repositories.

GT.M is installed in a parent directory and a utf8 subdirectory as follows:

- Actual files for GT.M executable programs (mumps, mupip, dse, lke, and so on) are in the parent directory, that is, the location specified for installation.
- Object files for programs written in M (GDE, utilities) have two versions - one compiled with support for UTF-8 mode in the utf8 subdirectory, and one compiled without support for UTF-8 mode in the parent directory. Installing GT.M generates both versions of object files, as long as ICU 3.6 or greater is installed and visible to GT.M when GT.M is installed, and you choose the option to install UTF-8 mode support. During installation, GT.M provides an option that allows placing the object code in shared libraries in addition to individual files in the directory.

- The utf8 subdirectory has files called mumps, mupip, dse, lke, and so on, which are relative symbolic links to the executables in the parent directory (for example, mumps is the symbolic link ../mumps).
- When a shell process sources the file gtmprofile, the behavior is as follows:
 - If \$gtm_chset is "m", "M" or undefined, there is no change from the previous GT.M versions to the value of the environment variable \$gtmroutines.
 - If \$gtm_chset is "UTF-8" (the check is case-insensitive),
 - \$gtm_dist is set to the utf8 subdirectory (that is, if GT.M is installed in /usr/lib/fis-gtm/gtm_V7.0-005_i686, then gtmprofile sets \$gtm_dist to /usr/lib/fis-gtm/gtm_V7.0-005_i686/utf8).
 - On platforms where the object files have not been placed in a libgtmutil.so shared library, the last element of \$gtmroutines is \$gtm_dist(\$gtm_dist/..) so that the source files in the parent directory for utility programs are matched with object files in the utf8 subdirectory. On platforms where the object files are in libgtmutil.so, that shared library is the one with the object files compiled in the mode for the process.

For more information on gtmprofile, refer to the Basic Operations chapter of GT.M Administration and Operations Guide.

Although GT.M uses ICU for UTF-8 operation, ICU is not FIS software and FIS does not support ICU.

Setting the environment variable TERM

The environment variable TERM must specify a terminfo entry that accurately matches the terminal (or terminal emulator) settings. Refer to the terminfo man pages for more information on the terminal settings of the platform where GT.M needs to run.

- Some terminfo entries may seem to work properly but fail to recognize function key sequences or fail to position the cursor properly in response to escape sequences from GT.M. GT.M itself does not have any knowledge of specific terminal control characteristics. Therefore, it is important to specify the right terminfo entry to let GT.M communicate correctly with the terminal. You may need to add new terminfo entries depending on your specific platform and implementation. The terminal (emulator) vendor may also be able to help.
- GT.M uses the following terminfo capabilities. The full variable name is followed by the capname in parenthesis:

```
auto_right_margin(am), clr_eos(ed), clr_eol(e1), columns(cols), cursor_address(cup),
cursor_down(cud1), cursor_left(cub1), cursor_right(cuf1), cursor_up(cuu1),
eat_newline_glitch(xen1), key_backspace(kbs), key_dc(kdch1),key_down(kcud1),
key_left(kcub1), key_right(kcuf1), key_up(kcuu1), key_insert(kich1),
keypad_local(rmkx),keypad_xmit(smxx), lines(lines).
```

GT.M sends keypad_xmit before terminal reads for direct mode and READs (other than READ *) if EDITING is enabled. GT.M sends keypad_local after these terminal reads.

Installing Compression Libraries

If you plan to use the optional compression facility for replication, you must provide the compression library. The GT.M interface for compression libraries accepts the zlib compression libraries without any need for adaptation. These libraries are included in many UNIX distributions and are downloadable from the zlib home page. If you prefer to use other compression libraries, you need to configure or adapt them to provide the same API as that provided by zlib.

If a package for zlib is available with your operating system, FIS suggests that you use it rather than building your own.

By default, GT.M searches for the libz.so shared library in the standard system library directories (for example, /usr/lib, /usr/local/lib, /usr/local/lib64). If the shared library is installed in a non-standard location, before starting replication, you must ensure that the environment variable LIBPATH (AIX) or LD_LIBRARY_PATH (GNU/Linux) includes the directory containing the library. The Source and Receiver Server link the shared library at runtime. If this fails for any reason (such as file not found, or insufficient authorization), the replication logic logs a DLLNOOPEN error and continues with no compression.

Although GT.M uses a library such as zlib for compression, such libraries are not FIS software and FIS does not support any compression libraries.

Change History

V7.0-005

Fixes and enhancements specific to V7.0-005:

Id	Prior Id	Category	Summary
GTM-DE309281	-	Other	Protect getaddrinfo and freeaddrinfo calls from interrupts
GTM-DE326986	-	Admin	Improved management of shared memory associated with MUPIP INTEG -ONLINE
GTM-DE327593	-	DB	Fix name-level \$ORDER(gvn,-1)
GTM-DE340860	-	Admin	Clean up temporary backup files following multi-region backup copy errors
GTM-DE345399	-	DB	Prevent inappropriate TPFALL with forth retry code of X
GTM-F134396	GTM-7577	DB	More context for TPFALL errors ✔
GTM-F135278	GTM-9129	Language	GT.M performs certain cases of string concatenation more efficiently
GTM-F135288	GTM-9140	Admin	✔ Environment variables for huge pages and shared memory pinning ✔
GTM-F135319	GTM-9208	Other	Improve handling of stack limit detection and reporting ✔
GTM-F135355	GTM-9297	Other	GT.M Internal signal DRVLONGJMP no longer exposed
GTM-F135370	GTM-9345	Admin	Audit LKE facility ✔
GTM-F135376	GTM-9365	Admin	Automatically reestablish replication connection on detecting various network errors
GTM-F135382	GTM-9376	Admin	Audit GDE facility ✔
GTM-F135383	GTM-9377	Admin	Audit DSE facility ✔
GTM-F135406	GTM-9413	DB	✔ Trigger statistics for ZSHOW and ^%YGBLSTAT ✔
GTM-F135418	GTM-9435	Admin	Extend restriction facility to ZLINK, ZRUPDATE & SET \$ZROUTINES ✔
GTM-F170998	-	Language	\$ZAUDITLOG() function for possible application audit logging and audit GDE facility ✔
GTM-F171004	-	Admin	✔ GT.M audit logging puts standard input labeled with "tty=" and reports more error information ✔

Id	Prior Id	Category	Summary
GTM-F188829	-	Admin	MUPIP shell works with MUPIP Audit facility 🟢
GTM-F188844	-	Other	Auto-Relink performance Improvement

Database

- GT.M handles name-level \$ORDER(gvn,-1)/\$ZPREVIOUS(gvn) correctly when searching across subscript-level mappings in the global directory. In V7.0-004, \$ORDER(gvn,-1), where gvn is an unsubscripted global, could return the same gvn instead of a previous global name or the null string. The workaround was to add global data to otherwise empty global maps between the specified gvn and its previous gvn, and optionally KILLing it afterwards, which leaves around sufficient residual information in the database to avoid the issue. (GTM-DE327593)
- TP transactions allocating blocks do not encounter TPFail errors with a fourth restart code of "X" due to a large number of such actions within a region. In V7.0 versions it would take a lifetime to encounter the issue but it has been completely eliminated. While customer update patterns vary and we do not have the means to replicate them, our limited testing indicates the revision may result in a modest reduction in bit map related TPRESTARTs with code: a. In V6.3-005 through V6.3-014, such a TPFail could arise in a region with a high rate of TP induced block allocations and a long period of up time. The workaround was to monitor tp_hint and restart or reset the monitored value; contact your GT.M support channel for additional information. (GTM-DE345399)
- TPFail errors include context similar to that provided for GVXXXXFAIL errors. Previously such errors did not supply context beyond the retry codes. (GTM-F134396) 🟢
- GT.M reports the STG, KTG and ZTG statistics as part of the ZSHOW and YGBLSTAT outputs. These accumulators provide additional details on invoked SET KILL and ZTRIGGER and triggers. Due to the nature of some statistics, their order of appearance in ZSHOW output may change. Previously, GT.M did not record these statistics in ZSHOW and YGBLSTAT. (GTM-F135406) 🟡 🟢

This page is intentionally left blank.

Language

- GT.M avoids copying strings in some cases involving the concatenation operator. Certain concatenation patterns can see significant performance improvements. Previously, GT.M sometimes unnecessarily recopied concatenation operands to the end of the heap (stringpool). (GTM-F135278)
- The \$ZAUDITLOG() function establishes a connection via a socket and sends its argument to a logger/listener process. It requires setting the AZA_ENABLE audit logging facility in the \$gtm_dist/restrict.txt file. The format for the \$ZAUDITLOG() function is:

`ZAUDITLOG(expr)`

- expr specifies the string to send for audit logging
- \$ZAUDITLOG() identifies its message with src=4, and like other GT.M logging facilities, records the location of GT.M distribution, uid, euid, pid, tty, and the command / argument(s).
- A return of: TRUE (1) indicates successful logging, FALSE (0) indicates logging is not enabled; a trappable RESTRICTEDOP error indicates logging is enabled but not working.
- If LGDE is specified as an option for the AZA_ENABLE facility, GDE logs all commands. GT.M ignores this option if specified with other A*_ENABLE audit logging facilities. When it fails to log a command, GDE issues a GDELOGFAIL error. The following table characterizes \$ZAUDITLOG() and GDE audit logging behavior:

\$ZAUDITLOG() / GDE logging Characteristics				
AZA_ENAB	LGDE	Logging success	GDE audit logging	\$ZAUDITLOG() result
Yes	Yes	Yes	Yes	1
Yes	No	Yes	No	1
Yes	Yes	No	GDELOGFAIL error	RESTRICTEDOP error
Yes	No	No	No	RESTRICTEDOP error
No	N/A	N/A	No	0

Previously, GT.M did not support the \$ZAUDITLOG() function. (GTM-F170998) 🟢

This page is intentionally left blank.

System Administration

- GT.M processes detach from shared memory associated with MUPIP INTEG snapshots correctly. Previously, relatively idle GT.M processes could remain attached to such shared memory segments on unjournalled regions or when the journal file switched while the snapshot was active, which prevented GT.M process rundown from removing shared memory. The workaround was to use MUPIP RUNDOWN. (GTM-DE326986)
- MUPIP BACKUP does not leave temporary files when there is an error during the copy phase for a multi-region backup. Previously, MUPIP BACKUP incorrectly left temporary files when there was an error during the copy phase for any region except the first. (GTM-DE340860)
- When the `gtm_pinshm` environment variable is defined and evaluates to a non-zero integer or any case-independent string or leading substring of TRUE or YES in a process creating shared memory, GT.M attempts to pin such memory used for database global buffers, replication buffers, and routine buffers into physical memory. Huge pages are implicitly locked in physical memory, so GT.M does not attempt to pin shared memory buffers backed by huge pages. `gtm_pinshm` does not pin memory used by online INTEG (integ snapshot). Pinning may not succeed due to insufficient physical memory and/or OS configuration. When the `gtm_hugetlb_shm` environment variable is defined and evaluates to a non-zero integer or any case-independent string or leading substring of TRUE or YES in a process creating shared memory, GT.M attempts to back all such shared memory segments with huge pages, using the default huge page size. If huge pages cannot be used, GT.M tries to back the shared memory with base pages instead, and attempts to pin the shared memory if requested with `gtm_pinshm`. GT.M issues a SHMHUGETLB or SHMLOCK warning message to the system log when the system is unable to back shared memory with huge pages or is unable to pin shared memory to physical memory, respectively. Previously, GT.M did not support the `gtm_pinshm` option to pin memory, and `gtm_hugetlb_shm` replaces the use of `libhugetlbf`s for huge page functions, so GT.M no longer evaluates `libhugetlbf`s environment variables, e.g. `HUGETLB_SHM`, `HUGETLB_VERBOSE`, etc. [Linux] (GTM-F135288) 🟡🟢
- When the restriction file contains a line specifying `AL_ENABLE`, LKE establishes a connection, a via socket, to a logger/listener process, and sends all commands, designating `src=5`, and using the socket to the listener for audit logging. If sending succeeds, LKE executes the command. If the connection or the send fail, LKE issues a RESTRICTEDOP error and does not execute the command. `AL_ENABLE` supports TCP, TLS or UNIX socket types. By default, LKE commands execute without logging. Previously, LKE did not provide an audit logging option. (GTM-F135370) 🟢
- GT.M Replication resets replication connection and attempts to automatically reestablish connection on detecting various network errors. Previously, GT.M replication server processes could shut down on encountering some network errors. (GTM-F135376)
- When `LGDE` is specified as an option for the `AZA_ENABLE` facility, GDE logs all commands. For example, an entry like the following in `$gtm_dist/restrict.txt` enables GDE logging via a local socket:

```
AZA_ENABLE:LGDE:/path/to/socket/file/audit.sock
```

The AZA_ENABLE facility enables the use of the \$ZAUDITLOG() function which GDE uses for logging commands. Refer to GTM-F170998 for information on the \$ZAUDITLOG() function and for other possible use in application audit logging. Previously, GDE did not provide an audit logging option. (GTM-F135382) 🟢

- When the restriction file specifies AD_ENABLE, DSE establishes a connection via a socket, to a logger/listener process, and sends all commands, designating src=6, and using the socket to the listener for audit logging. If sending succeeds, DSE executes the command. If the connection or the send fail, DSE issues a RESTRICTEDOP error and does not execute the command. AD_ENABLE supports TCP, TLS or UNIX socket types. By default, DSE commands execute without logging. Previously, DSE did not provide an audit logging option. (GTM-F135383) 🟢
- The GT.M restrictions facility recognizes ZLINK, ZRUPDATE and SET \$ZROUTINES. When an explicit ZLINK(not auto-zlink), ZRUPDATE, or SET \$ZROUTINES restriction conditions are configured, the restricted command issues a RESTRICTEDOP error message. Previously, the restrictions facility did not support ZLINK, ZRUPDATE, or SET \$ZROUTINES. (GTM-F135418) 🟢
- GT.M audit logging facilities use tty to label the standard input of the process. GT.M places tty=ttyname before the command field in all audit log messages. If the standard input at process startup is not terminal device, GT.M logs tty=0. In addition, the audit facilities check for errors at the time of closing a socket / terminating a connection and report them with a GTM-E-SOCKCLOSE message to the operator log. The audit logger/listener sample programs (downloadable from the A&O Guide) switch their log files after receiving a SIGHUP signal. The switched log file has a suffix "_%Y%j%H%M%S" (yearjuliendayhoursminutesseconds) and the naming convention is similar to what GT.M uses for switching journal files. FIS recommends periodically switching logger files. Deleting an active audit log file makes it lost to new processes, while existing processes continue to use it, so FIS recommends taking such a step. The sample programs have a Makefile. Previously, the audit log facilities did not provide tty information, did not check and report on errors during socket close, the logger/listener programs did not implement a log file switching mechanism, and those programs had no Makefile. (GTM-F171004) 🟡🟢
- When the restriction file contains a line specifying AM_ENABLE, commands typed at the MUPIP prompt (MUIP>) are audit logged and executed the same as MUPIP shell commands. Note that MUPIP returns to the shell after each command. Previously, when this facility was enabled, all commands typed at the MUPIP prompt (MUIP>) produced the RESTRICTEDOP error. (GTM-F188829) 🟢

Other

- GT.M handles the interruption of certain socket operations correctly. Previously, asynchronous events like MUPIP STOP while creating a socket connection could cause a segmentation violation (SIG-11). This was only seen in development and not reported by a customer. (GTM-DE309281)
- When a single operation exceeds both the STACKCRIT and STACKOFLOW limits, GT.M issues a STACKCRIT message to the operator log, and exits with a STACKOFLOW error. Note this condition is fatal and not trappable. If this occurs, look into the gtm_mstack_crit and gtm_mstack_size environment variables. Previously, when this occurred, the process failed with a segmentation violation (SIG-11). (GTM-F135319) 🟢
- GT.M prevents internal error ERR_DRVLONGJMP from becoming user visible after failure to open a statsDB. Previously, there were cases in which the error's mnemonic could be found in \$ZSTATUS following attempts to open statsDBs in nonexistent directories. (GTM-F135355)
- When configured for auto-relink, GT.M uses a lightweight operation to validate an existing relinkctl file. Previously, every GT.M process using a relinkctl file would use a heavyweight operation to validate it, which added to routine ZLINK latency on the process' first lookup in the auto-relink directory. In particular, this may have resulted in visible latency when starting a large number of JOB processes if the JOB routine search included one or more auto-relink directories. (GTM-F188844)

This page is intentionally left blank.

Error and Other Messages

GDELOGFAIL

GDELOGFAIL, GDE failed to log command. Check operator log for more information

GDE Error: This message appears when LGDE is specified with the AZA_ENABLE facility and there is a problem with logging the GDE commands. This message also prevents the execution of the GDE command.

Action: Review the operator log for the exact reason of the failure to log the GDE command

SOCKCLOSE

SOCKCLOSE, Error closing socket: (errno = aaaa) xxxx

Operator log/All GT.M Components Error: aaaa contains the OS error code and xxx indicates information about the error that occurred while closing a socket connection for the process attempting to log a command to the audit logging facility.

Action: Review the message to determine whether the logger programs are running or whether a change is required in the configuration of your audit logging facility.

STACKCRIT

STACKCRIT, Stack space critical

Run Time Error: This indicates that the process has consumed almost all of the available stack space.

Action: Look for infinite recursion. If you do not take immediate action to reduce your stack, GT.M is likely to produce a STACKOFLOW error, which terminates the process. Examine the stack with ZSHOW. Trim the stack using QUIT, ZGOTO, HALT or ZHALT. Note that if a single application call generates a stack frame larger than the stack space between the STACKCRIT and STACKOFLOW boundaries, the process puts a STACKCRIT message in the operator log, but processes the STACKOFLOW and terminates, so the application gets no chance to intervene and handle the error. Look into the gtm_mstack_crit_threshold and gtm_mstack_size environment variables.

STACKOFLOW

STACKOFLOW, Stack overflow

Run Time Fatal: This indicates that the process required more stack space than was available in memory.

Action: Reduce the stack when you get a STACKCRIT error. This error terminates the process. Note that if a single application call generates a stack frame larger than the stack space between the

STACKCRIT and STACKOFLOW boundaries, the process puts a STACKCRIT message in the operator log, but processes the STACKOFLOW and terminates, so the application gets no chance to intervene and handle the error. Look into the gtm_mstack_crit_threshold and gtm_mstack_size environment variables.

TPFAIL

TPFAIL, Transaction COMMIT failed.

Run Time Error: This indicates that GT.M attempted to process this transaction four times, but encountered an error every time. Additional accompanying messages indicate details of the failure.

Action: Report this database error to the group responsible for database integrity at your operation.