

GT.M

Release Notes

V7.0-003

Empowering
the Financial World



Contact Information

GT.M Group
Fidelity National Information Services, Inc.
200 Campus Drive
Collegeville, PA 19426
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
Website: <http://fis-gtm.com>

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 History		
Revision 1.0	24 June 2022	V7.0-003

Table of Contents

GT.M Release Notes	1
V7.0-003	1
Overview	1
Conventions	1
Platforms	2
Additional Installation Instructions	4
Upgrading to GT.M V7.0-003	6
Managing M mode and UTF-8 mode	6
Setting the environment variable TERM	7
Installing Compression Libraries	8
Change History	8
V7.0-003	8
Language	9
System Administration	10
Other	10
Error and Other Messages	11
DEVICEOPTION 	11


GT.M Release Notes

V7.0-003

Overview

V7.0-003 adds support for TLS v1.3 with the OpenSSL 1.1.1 series and provides the new `OPTIONS deviceparameter` to adjust the socket behavior. It also includes other fixes and enhancements including the support for ECDHE ciphers with OpenSSL 1.0.2 and better reporting of OpenSSL errors for TLS replication and database encryption. For more information, refer to the Change History section.

Items marked with  document new or different capabilities.

Please pay special attention to the items marked with the symbols  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.	-updok (recommended)

GT.M Release Notes

Pre V5.5-000 terminology	Pre V5.5-000 qualifier	Current terminology	Current qualifiers
		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.	-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.	-updnok
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.

GT.M Release Notes

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; Ubuntu 18.04 LTS, 20.04 LTS, and 22.04 LTS	<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> <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>

Platform	Supported Versions	Notes
		<p>Ubuntu 21.10 is Supportable.</p> <div data-bbox="753 296 837 380">  </div> <div data-bbox="873 289 1544 373"> <p>To use TLSv1.3 with OpenSSL 1.1.1 and up, you must recompile the reference encryption plugins</p> </div> <div data-bbox="753 428 837 512">  </div> <div data-bbox="873 422 1544 506"> <p>RHEL 8 includes compat-openssl10.x86_64 for binaries compiled against OpenSSL 1.0.2 on RHEL 7</p> </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 V7.0-003 in a Filesystem Hierarchy Standard compliant location such as /usr/lib/fis-gtm/V7.0-003_arch (for example, /usr/lib/fis-gtm/V7.0-003_x86_64 on Linux systems). A location such as /opt/fis-gtm/V7.0-003_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***.

GT.M Release Notes

- 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 libcrypt, libpgpme, 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 libcrypt11-dev libpgpme11-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 V7.0-003

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

GT.M Release Notes

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_V7.0-003_i686, then gtmprofile sets \$gtm_dist to /usr/lib/fis-gtm/gtm_V7.0-003_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 sect1 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(el), columns(cols), cursor_address(cup), cursor_down(cud1),
cursor_left(cub1), cursor_right(cuf1), cursor_up(cuu1), eat_newline_glitch(xenl), 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(smkn), 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

V7.0-003

Fixes and enhancements specific to V7.0-003:

Id	Prior Id	Category	Summary
GTM-DE257948	-	Language	Appropriate handling for square-bracket (nonstandard) extended reference syntax and NUMOFLOW in expressions made up of literals
GTM-DE276621	-	Language	Literal FALSE postconditionals don't cause failures when they suppress a command with a local variable in their argument that appears later in the line
GTM-DE294185	-	Admin	Receiver Server continues to operate after an unexpected termination of a TLS connection
GTM-DE294187	-	Language	Protect \$ZKey from damage during heap management
GTM-F134692	-	Admin	🔴 MUPIP INTEG and MUPIP DUMPFHEAD support the user-specified region order 🟢
GTM-F134877	GTM-8402	Other	GT.M TLS reference encryption plugin provides support for ECDHE ciphers
GTM-F135169	GTM-8991	Language	Provide additional options for SOCKET devices 🟢
GTM-F135258	GTM-9085	Other	GT.M TLS reference encryption plugin provides support for TLS v1.3 🟢
GTM-F135313	-	Other	Retry interruptions on database file creation or extension with NODEFER_ALLOCATE
GTM-F135334	GTM-9237	Other	Error reporting from the GT.M database and the TLS reference encryption plugins don't interfere with each other
GTM-F135416	GTM-9430	Admin	MUPIP FREEZE -ON does not hang while an online rollback is in progress

Id	Prior Id	Category	Summary
GTM-F135424	-	Language	Empty string rather than UNDEF error in case of a lost FOR control variable in NOUNDEF mode
GTM-F135428	GTM-9460	Language	\$ZSOCKET() returns an empty string when given an out of range index
GTM-F157495	-	Language	\$VIEW("DEVICE",<device-designation>) returns device status 

Language

- GT.M accepts square-bracket expressions as synonyms for the standard vertical bars. Previously, GT.M handled square-bracket expressions inappropriately, which could cause a fatal error. In addition, GT.M detects and recovers from certain numeric overflow errors. Previously, GT.M inappropriately handled such overflow errors when optimizing arithmetic expressions containing literals, which caused fatal errors. (GTM-DE257948)
- The GT.M compiler correctly handles lines with literal FALSE postconditionals. Due to a regression in V7.0-002, the compiler could encounter a segmentation violation (SIG-11) error when having to fetch a variable that earlier in the same line was the argument of a command with a literal FALSE postconditional. (GTM-DE276621)
- Reading from \$ZKey works as expected. Previously under rare circumstances, an access of \$ZKey could result in a segmentation violation (SIG-11). This was only seen in development and not reported by a customer. (GTM-DE294187)
- The OPEN and USE commands for SOCKET devices support assigning characteristics maintained with the POSIX setsockopt() service using the OPTIONS deviceparameter for the newly created socket or the current socket.

`OPTIONS=expr` Applies to: SOC

The argument (`expr`) is a string which contains a comma separated list of setsockopt options. If the option takes a value, it is given after an equal sign (=) following the option. The supported options are:

KEEPALIVE a positive value enables SO_KEEPALIVE. A zero value disables SO_KEEPALIVE.
KEEPIDLE sets the TCP_KEEPIIDLE socket value.
KEEPCNT sets the TCP_KEEPCNT socket value.
KEEPINTVL sets the TCP_KEEPIINTVL socket value.
SNDBUF sets the size of the socket's network send buffer (SO_SNDBUF) in bytes.

Examples:

`USE dev:OPTIONS="KEEPALIVE=1,KEEPIDLE=50"`

This enables SO_KEEPALIVE and set TCP_KEEPIIDLE to 50 seconds.

`USE dev:OPTIONS="KEEPALIVE=0"`

This disables SO_KEEPALIVE.




Note

For more information on the use of these options, please review the man page for setsockopt . On Linux, "man 7 socket" and "man 7 tcp" provide additional information.

The \$ZSOCKET() function supports an "OPTIONS" keyword which takes an index argument and returns a string of the OPTIONS previously specified for the selected socket. The string may not exactly match the string originally specified but has the same meanings.

GT.M Release Notes

The new keywords "KEEPALIVE", "KEEPCNT", "KEEPIDLE", "KEEPINTVL", and "SNDBUF" return the individual items. If the system's current value for the item doesn't match the value previously specified with the OPTIONS device parameter, both values are returned separated by a semicolon (";"): "uservalue;systemvalue".

The "ZIBFSIZE" keyword may return the system value for SO_RCVBUF in addition to the value from the ZIBFSIZE device parameter. Note that the operating system may modify the values specified for SO_RCVBUF and SO_SNDBUF so the returned values for those options obtained with POSIX getsockopt() service may be different than those specified using setsockopt(). (GTM-F135169) 


- In NOUNDEF mode, GT.M assigns the value of an empty string to an undefined FOR control variable. This may result in an unintended infinite loop if a process KILLS or NEWs a control variable. Previously, GT.M provided an UNDEF error for this condition even when the mode was NOUNDEF. (GTM-F135424)
- If the index for the \$ZSOCKET() function is outside the range of attached sockets for the specified SOCKET device, \$ZSOCKET() returns an empty string. Previously, if prior actions on the SOCKET device removed one or more sockets, \$ZSOCKET() could return stale or invalid information, or cause a segmentation violation (SIG-11). (GTM-F135428)
- The \$VIEW() keyword "DEVICE" returns the type and status, if any, for the device named by the second argument. If no device of that name exists, the function returns an empty string. The device type is based on what ZSHOW "D" shows.

Examples:



```
WRITE $VIEW("DEVICE", "0")
TERMINAL : OPEN
```

This indicates the \$PRINCIPAL device is a terminal and it is open, which is usually the case for \$PRINCIPAL. \$ZPIN and \$ZPOUT intrinsic special variables used as the second argument select the corresponding side of a split \$PRINCIPAL device.

```
OPEN "f.txt"
CLOSE "f.txt":NODESTROY
WRITE $VIEW("DEVICE", "f.txt")
RMS:CLOSED
```

(GTM-F157495) 

System Administration

- The Receiver Server continues to operate after an unexpected termination of a TLS connection. Previously, and more frequently with OpenSSL 3.0, the Receiver Server would issue a TLSIOERR error message and terminate in response to network disruptions. (GTM-DE294185)
- When MUPIP INTEG and MUPIP DUMPFHEAD arguments specify a region list, MUPIP processes regions in the listed order, or, for names expanded by wildcard ("*"), alphabetically. Previously, MUPIP DUMPFHEAD and MUPIP INTEG ignored any user-specified order of regions, and processed regions in FTOK order, which tends to change with changes in operational conditions within the underlying file system. (GTM-F134692)  
- MUPIP FREEZE -ON operates while an online rollback is in progress. GT.M now checks for contention and ensures online rollback appropriately releases the required resources and other database accesses wait for this to occur. Previously in rare cases, it might hang because of shared resource contention. (GTM-F135416)

Other

- The GT.M TLS plugin library supports ECDHE ciphers with OpenSSL 1.0.2. OpenSSL 1.1.0 and above automatically support ECDHE ciphers. (GTM-F134877)

GT.M Release Notes


- The GT.M TLS reference encryption plugin supports TLS v1.3 with OpenSSL 1.1.1 series and up. OpenSSL negotiates that highest level of TLS available between both sides of a connection. The implication is that just using OpenSSL 1.1.1 and up does not guarantee the use of TLSv1.3 for all TLS sessions. Previously the TLS reference encryption plugin supported only up to TLS v1.2.



TLSv1.3 introduces a number of protocol changes. The concept of renegotiation was removed. GT.M used TLSv1.2 renegotiation to ensure appropriate refresh of session keys (for rationale, see Luykx, A. and K. Paterson, "Limits on Authenticated Encryption Use in TLS", 2016; <http://www.isg.rhul.ac.uk/~kp/TLS-AEbounds.pdf>). As a result, GT.M uses the term "renegotiation" for SOCKET devices where the mode is "server" and "renegotiation_interval" for the frequency of renegotiation (by the Source Server) during database replication. For TLSv1.3 sessions, the GT.M Reference TLS plugin treats the "renegotiation" as a request to update the session keys.



OpenSSL 3.0 by default does not allow client-side initiated TLSv1.2 renegotiation requests due to potential DoS attacks. Because of this, the GT.M Reference TLS plugin built with a pre-OpenSSL 3.0 version does not support client-initiated renegotiation when communicating with a plugin build with OpenSSL 3.0. This limitation only affects database replication and not SOCKET devices.

(GTM-F135258) 

- When an event interrupts a POSIX fallocate system call used for database file creation or extension with NODEFER_ALLOCATE, GT.M retries the call. Previously, such an interruption resulted in a PREALLOCATEFAIL message. (GTM-F135313)
- The GT.M reference encryption plugin correctly reports OpenSSL messages when a process reports an error message from either a TLS connection or database encryption. Previously processes leveraging the reference encryption plugin with OpenSSL could see incorrect (potentially empty) error messages. This was only seen in development and not reported by a customer. (GTM-F135334)

Error and Other Messages

DEVICEOPTION

DEVICEOPTION, Option xxxx on yyyy command: zzzz

Run Time Error: The xxxx option in an OPTIONS device parameter on a yyyy command was incorrectly specified as described by zzzz.

Action: Correct the option as indicated.