SMASER

Major New Features in TRIP 8

TRIPsystem
Product Documentation

End User License Agreement

All rights to this software, its documentation and logotypes of the TRIP product family and software (altogether “Software”) supplied by Smaser AG (Smaser) are exclusively owned by Smaser.

The transfer of this Software, solutions or parts thereof requires the prior written agreement of Smaser. Furthermore, the customer has the right to use licensed Software and / or process solutions supplied by Smaser to the extent specified in his contract with Smaser.

The free-to-use non-commercial version doesn’t require a prior written agreement with Smaser but such customers, organizations and/or third parties agree by using the software and / or solution of Smaser to be strongly obliged to keep all rights to this software, documentation and logotypes of the TRIP product family absolutely uninfringed and protected.

Table of Contents

Introduction

This document describes briefly the major new features of TRIP version 8.0. More information can be found in the specific documentation.

TRIP 8.2-0

- TRIP on Linux now requires RHEL8
- Improved session encryption
- Docker log file rotation
- CCL file directory configuration for networked sessions

TRIP 8.1-0

- Long names for databases, clusters, fields users and groups
- Increased file block size

TRIP 8.0-0

- Search set size limit
- 64-bit integers and numbers
- Field groups
- Login tickets
- TRIPxml features integrated with TRIPsystem
- TRIPsql features integrated with TRIPsystem
- JSON document store
- New design for JSON and XML databases
- Formatted values
- New search function GROUP()
- New database and cluster design APIs
- Default database character set
- Storage location for new databases

TRIP 8.0-4

- Automatic LDAP user import

TRIP 8.2-0

TRIP on Linux now requires RHEL8

The Linux build of TRIPsystem and associated server-side components such as TRIPcof has now require RedHat Enterprise Linux version 8. The target distribution for the previous main releases of TRIP 8 was CentOS7.

Improved session encryption

Session encryption for connections to TRIPsystem from TRIPjxp, TRIPnxp and TRIPmanager is now based on the AES algorithm. This improves security as well as it makes using encrypted sessions more performant.

To benefit from this change, TRIPmanager, TRIPnxp and/or TRIPjxp version 8.2 or later must be used. The change in TRIPsystem is backward compatible, so remains possible to use older versions of TRIPmanager, TRIPnxp and TRIPjxp if an upgrade of those components is not immediately feasible.

Docker log file rotation

A new tool “logmaint” is now available in the TRIPsystem Docker image. This tool is automatically run at a predefined interval and removes old log files that meet the conditions for removal. The log files handled are those from tbserver, Toolkit API, Kernel and XPI, as well as session index files (SIF) and batch job logs for PRINT, UPDATE, INDEX, LOAD and LOADIX tasks.

A log file is removed when either the maximum number of log files of its type has been exceeded or when is older than the predefined number of days. The maximum age is 30 days by default, with the exception of batch log for successful jobs that have a default max of 7 days.

The parameters for the logmaint tool can be set in the tdbs.conf file used by the container or at the command line when run interactively. The relevant tdbs.conf symbols are:

Symbol	Default	Description
TDBS_LOGPRUNE_INTERVAL	60	Interval in minutes between scans for log files to remove.
TDBS_LOGPRUNE_MAX_LOGS	100	The maximum number of log files of each type to retain (unless they are older than max age).
TDBS_LOGPRUNE_MAXAGE	30	The maximum number of days to retain log files.
TDBS_LOGPRUNE_MAXAGE_BATCH	7	The maximum number of days to retain logs for successful batch logs.

The logmaint tool is also available with the regular, non-Docker TRIPsystem distributions for Linux and Solaris.

CCL file directory configuration for networked sessions

The new tbserver-specific configuration symbols TBS_CCL_FILE_DIR and TBS_CCL_NO_RELATIVE controls how files read and written by the CCL commands EXPORT, IMPORT, LOAD and PRINT are handled.

The TBS_CCL_FILE_DIR symbol accepts a directory name as value. This directory will then be used for any file specified without a directory in a CCL command. For example, an order such as “`PRINT TFO=mydata.tfo NOW`” would otherwise write to whatever current directory the server-side tbserver process was using. With this configuration symbol in use, such files will instead be written to the configured directory.

The function of the TBS_CCL_NO_RELATIVE symbol is two-fold. Used alone, it restricts file names used with aforementioned CCL commands to fully qualified paths only. When the TBS_CCL_FILE_DIR symbol is also set, an attempt use a relative file path will be overridden so that the file instead will be written to the TBS_CCL_FILE_DIR directory.

Any deployment of TRIP that serves networked applications that may be expected to issue EXPORT, IMPORT, LOAD or PRINT should define these symbols.

If TBS_CCL_FILE_DIR is defined, the associated directory should regularly be cleaned up in order to not keep a potentially large number of files lingering unused in this directory. Use a scheduled job (e.g. cron) for this purpose.

TRIP 8.1-0

Long names

The maximum length for databases, clusters and fields have been increased to 64 bytes, users and groups to 128 bytes and passwords to 128 bytes.

Databases, clusters, fields, users and groups with long names can be fully managed via TRIPmanager version 8.1, and programmatically via TRIPjxp and TRIPnxp.

The maximum lengths are described by the following table, which can also be found in the TRIPmanager administrator's guide:

Category	Content Alpha-numeric?	Length[1]	First Letter Alpha-betic?	Allowable Punctuation
File	Yes	128	Yes	Underscore
Database	Yes	64 (16)	Yes	Underscore
Field	Yes	64 (16)	No	Underscore
Procedure	Yes	64 (16)	No	Underscore
Output Format	Yes	64 (16)	No	Underscore
Entry Form	Yes	64 (16)	No	Underscore
Search Form	Yes	64 (16)	No	Underscore
Group	Yes	128 (32)	No	Underscore
User	Yes	128 (32)	No	Underscore
Password	Yes	128 (32)	No	Underscore

[1] maximum length in characters (including file paths where applicable). The values in parentheses apply to programs using the pre-8.0 database design APIs (such as TRIPclassic).

As noted in the table above, long names are not supported for older interfaces such as TRIPclassic, TRIPclient or TRIPjtk. Even if found to be working in some cases, they must not be relied upon for production use via such interfaces.

Increased file block size

In previous versions of TRIP, the proprietary file format used by the BAF, BIF, VIF and SIF files have used a block size of 2048 bytes. Modern operating systems have file systems that support larger block sizes. Using blocks of the operating system's default size or a multiple thereof will result in a noticeable performance improvement when reading and writing said files.

TRIP 8.1 introduces support for larger file block sizes. To ensure backward compatibility, the default size remains at 2048 bytes. This can be changed by setting the tdbs.conf logical name TDBS_BLOCK_SIZE to the desired value (must be a multiple of 2048 bytes). A good optimal value on most contemporary systems is 4096 bytes. Currently, only the values 2048 and 4096 are supported by TRIP.

The optimal block size can on Windows be found out by running the following command as administrator in a command prompt:

```
fsutil fsinfo ntfsinfo DRIVE
```

Where "DRIVE" is the drive where your database files are located (e.g. C:). Look for the value labeled "Bytes Per Physical Sector".

On Linux, the optimal block size can be found out by running the following command as any user:

```
stat -fc %s /path/to/mount
```

Where "/path/to/mount" is the path to the mount directory for the file system on which the TRIP database files are located. The output is the block size in bytes.

The information about the block size used with a particular BAF, BIF, VIF or SIF file is stored in the file header. This means that you can use different block sizes for different databases should you wish to do so.

Note that if you set TDBS_BLOCK_SIZE to any other value than 2048, the resulting files cannot be used with older TRIPsystem versions. Trying to do so may cause TRIP to crash. The 8.0 series contain a safety control introduced in version 8.0-10 that blocks opening such files. The same type of safety control was added to the 7.2 series in version 7.2-6:3. All other older versions will become unstable with files using non-2048 byte block sizes.

TRIP 8.0-0

Search set size limit

The record count and hit count values for search sets are now available as 64-bit integer values. This allows for hit and record counts larger than INT_MAX (2147483647). New APIs have been added (to the C toolkit, TRIPjxp and TRIPnxp) that expose these counts as signed 64-bit integer values. The older APIs are still available, and will for values larger than INT_MAX truncate the value to INT_MAX before returning it.

The new methods and properties in TRIPjxp and TRIPnxp for accessing the 64-bit versions of these counts are:

- TdbHistoryDetail

  - JXP: getLongRecordCount(), getLongHitCount()
  - NXP: LongRecordCount, LongHitCount
- TdbRecordSet
  - JXP: getLongRecordCount(), getLongHitCount()
  - NXP: LongRecordCount, LongHitCount
- TdbSearchSet
  - JXP: getLongRecordCount(), getLongHitCount()
  - NXP: LongRecordCount, LongHitCount

The C toolkit have the following new functions that return 64-bit values for the hit and record counts:

- TdbSearchInfo

64-bit counts are not available in TRIPclassic. Search sets with record and/or hit counts that do not fit into a 32-bit signed integer will in TRIPclassic be truncated to INT_MAX (2147483647).

64-bit integers and numbers

The value range for the INTEGER data type has been extended from signed 32-bit to signed 64-bit. Also increased to 64-bit is the value range for the NUMBER data type, giving it a higher precision.

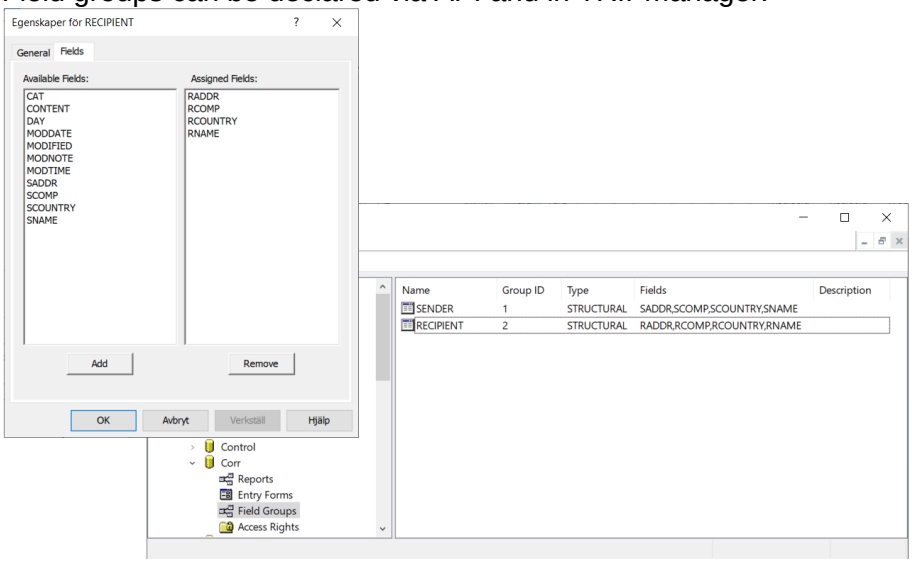
Existing databases with INTEGER and/or NUMBER fields can start using this larger value range without any modifications to the database design.

IMPORTANT: Databases populated in TRIP 8 with numeric field values larger than would fit into a signed 32-bit value cannot be used with older versions of TRIP without risking system stability.

Field groups

Field groups allow the grouping of fields in a database under a common name. A field group can be used like a VIEW (search in all fields in the group by specifying its name instead of a field name), and optionally be used to enforce a tuple structure.

Declaring Field Groups
Field groups can be declared via API and in TRIPmanager.



Tuple Lists as Field Groups
A tuple field group will validate the data in the fields and make sure that all included fields have the same number of values when records are being stored. There are also new APIs that allows the manipulation of rows (tuples) in a tuple field groups, helping applications to maintain data consistency when updating and deleting field values in a tuple field group.

Login tickets

A login ticket is a unique, generated, alternate temporary identity for a TRIP user. A login ticket can be requested via the APIs in TRIPjxp and TRIPnxp, and can, once obtained, be used instead of username and password to log in to TRIP as the user to which the ticket was assigned.

The purpose of this feature is to allow an SSO-like behaviour in web applications that use TRIP user logins. The user must explicitly log in once, at which time the application requests a login ticket and sets it as a cookie. At subsequent access of the application at a time when the application session has expired, the application can use the login ticket in the cookie to log in the user automatically.

To login using a login ticket, the application enters the login ticket as the password and leaves the username argument blank. Upon success, the name of the user is available via the session object (TdbSession) in TRIPjxp and TRIPnxp.

This feature is disabled by default. To enable it, set the configuration symbol TDBS_LOGIN_TICKETS=Y in the privileged section of tdbs.conf. The

A login ticket is valid for a configurable time, which resets each time the user is logging in. This time is determined by the tdbs.conf privileged section configuration symbol TDBS_TICKET_TIMEOUT, that (if not set) is 30 days by default. A valid value is a positive integer, with an optional suffix:

- **d** – Indicates number of days (default)
- **h** – Indicates number of hours
- **m** – Indicates number of minutes
- **s** – Indicates number of seconds

All assigned login tickets are automatically revoked when the TRIP daemon is restarted.

TRIPxml features integrated with TRIPsystem

The functionality in the add-on product TRIPxml is now available in TRIPsystem. TRIPxml is therefore no longer available as a separately installable add-on product, but remain separately licensable.

The command-line tools of TRIPxml have all been replaced with the single new command-line tool `jxtool`.

Upgrading from a TRIPxml 1.x installation is not supported. Please refer to the “Installation and Upgrade” section of the “JSON and XML Databases” document for more information on how to proceed in such a case.

TRIPsql features integrated with TRIPsystem

The functionality in the add-on product TRIPsql is now available in TRIPsystem. TRIPsql is therefore no longer available as a separately installable add-on product, but remain separately licensable.

Because of this change, the client drivers for JDBC, ODBC and ADO.NET are available as separately installable packages.

For more information, please refer to the “Installation and Upgrade” section in the “SQL Reference Manual” document.

JSON document store

XML databases now support the storage of JSON documents. The JSON import and export functionality is available via the TRIPxml-related tools and APIs. The XPath based query language is also supported for XML databases that contain JSON documents, or a mix or JSON and XML documents.

Refer to the “JSON and XML Databases” document for more information.

New design for JSON and XML databases

The XML database design has been changed to use less disk space and to accommodate the storage of JSON documents.

XML databases created with in older versions of TRIPsystem with TRIPxml 2.x and 3.x remain supported, but cannot be used to store JSON documents.

Formatted values

Output-format based reports can now be requested for each record in a search set as so-called “formatted values”. These values are exposed by the TRIPjxp and TRIPnxp APIs as a form of virtual field, where an instance of the TdbFieldTemplate class is used to specify the name of an output format instead of a regular field. This results in the formatted result to be returned to the application from each record via a TdbTextField instance.

Using formatted values is a convenient way to get a mix of output format reports and regular field values from a search result in one request.

New search function GROUP()

GROUP() is a new function in CCL. It resolves to the TRIP user groups the currently logged on user is a member of. It is intended for use in read and write scopes so that records can be given read and/or write access depending on which groups the currently logged on user is a member of.

This would, for example, match records in which the ACL_READ_GROUPS field have values that are the same as at least one of the groups the user is a member of:

```
FIND ACL_READ_GROUPS=GROUP()
```

New API functions

New C API functions, primarily for database and cluster design, have been added. These do not utilize the structures (e.g. base_spec_rec, etc) that the older database design API functions use, but instead work via opaque handles and property getters/setters. The old

versions of these API functions are marked as deprecated and will generate compiler warnings when used.

See below for a list of the new API functions and what they deprecate:

New Function	Deprecates
TdbChangeUser	
TdbCloseBaseDesign	
TdbCloseBaseField	
TdbCloseClusterDesign	
TdbClusterAddMember	(struct cluster_spec_rec)
TdbClusterDelMember	(struct cluster_spec_rec)
TdbClusterDelMembers	(struct cluster_spec_rec)
TdbClusterEnumMembers	(struct cluster_spec_rec)
TdbClusterGetMember	(struct cluster_spec_rec)
TdbCopyBaseDesign	TdbCopyBaseDef
TdbCopyClusterDesign	(struct cluster_spec_rec)
TdbCopyThesDesign	TdbCopyThesDef
TdbDeleteBaseField	TdbDeleteFieldSpec
TdbGetBaseDesign	TdbGetBaseDef
TdbGetBaseField	TdbGetFieldName TdbGetFieldSpec
TdbGetBaseProperty	(struct base_spec_rec)
TdbGetClusterDesign	TdbGetDbCluster
TdbGetClusterProperty	(struct cluster_spec_rec)
TdbGetFieldProperty	(struct field_spec_rec)
TdbGetThesDesign	TdbGetThesDef
TdbLicenseInfo	
TdbPutBaseDesign	TdbPutBaseDef
TdbPutBaseField	TdbPutFieldSpec
TdbPutBaseProperty	TdbDefineGraph TdbDefineXml (struct base_spec_rec)
TdbPutFieldProperty	(struct field_spec_rec)
TdbPutClusterDesign	TdbPutDbCluster
TdbPutClusterProperty	(struct cluster_spec_rec)
TdbPutThesDesign	TdbPutThesDef

Default database character set

The character set for new databases is now set to the session character set per default. This means that a new database created in TRIPclassic will get the character set that TDBS_CHARS indicates, and a new database created in TRIPmanager, TRIPnxp or TRIPjxp will get Unicode UTF-8 unless otherwise specified.

Storage location for new databases

Previous versions of TRIP would use the current directory of the server-side TRIP process as the default storage location for new databases. This could result in databases created in location not suitable for such files, such as in the C:\Windows\System32 folder on Windows.

TRIP now uses the TDBS_BASES logical name as storage location for new databases. If not previously defined, this is set by the installation program to “C:\Program Files\infinIT Services\TRIP Databases” on Windows and to /usr/local/trip/data on Unix/Linux. The installer creates this directory if needed.

If the directory that TDBS_BASES is set to is unsuitable, it can easily be changed by editing the tdbs.conf file. Such customization will be retained across upgrades.

TDBS_BASES will be applied to new databases if no other location (logical name or directory path) is defined during database creation. If the definition of TDBS_BASES in tdbs.conf has been commented out or removed, TRIP will revert to its previous behaviour of using the current directory.

TRIP 8.0-4

Automatic LDAP user import

Authentication via LDAP integration enables users to log in to TRIP without the DBA having to maintain user passwords in TRIP. Using guest login will in addition eliminate the need to have dedicated user accounts in TRIP, which makes it easier to have a large number of user access TRIP.

However, with LDAP guest login, users will all be logged in to TRIP as the BUILTIN_GUEST user. Everyone will share the same identity in TRIP. For basic usage scenarios that may be OK, but if either of the following is a usage requirement, then guest login is not an option:

- The user base contains different kinds of users that must have different access to databases, fields or records.
- User-specific TRIP functionality such as CCL procedures (such as can be created by storing a search strategy), individual start procedures, etc, is required.

To help enable the above, TRIP version 8.0-4 introduces automatic import and synchronization of LDAP users. This means that a user logging in via LDAP will get a TRIP-specific user account created for it automatically. This feature must be enabled for use, and is disabled by default.

Read more about automatic LDAP user import in the TRIP LDAP Authentication white paper.