Rocket UniVerse

New Features Guide

Version 11.3.1

October 2016
UNV-1131-NF-01

New features in v11.3.1

This section summarizes the significant changes and updates for Rocket UniVerse. If you are familiar with previous releases of UniVerse, and you want to know the new features and enhancements for newer versions, you can use this section to assist you.

- Video-documented features

  In this release, videos are available to help you understand some of the important new features in UniVerse 11.3.1.
- 64-bit builds

  Starting at UniVerse 11.3.1, UniVerse builds are 64-bit only on all platforms. As a consequence, Windows 2003 Server, Windows XP, and Windows Vista are no longer supported platforms.
- IPv6

  UniVerse 11.3.1 supports the IPv6 protocol.
- U2 Audit Logging

  There are several enhancements to U2 Audit Logging in UniVerse 11.3.1.
- U2 Python

  Support for the Python development language has been added to UniVerse 11.3.1 on Linux and Windows only.
- Rocket SystemCure for UniVerse

  UniVerse 11.3.1 includes support for SystemCure. Rocket SystemCure for UniVerse 1.1.0 is a new product that offers simplified discovery, monitoring, visualization, and triggering of actions or notifications for UniVerse system events.
- FIPS 140-2

  The Federal Information Processing Standard (FIPS) Publication 140-2 is a United States Federal Government computer security standard used to accredit cryptographic modules. FIPS 140-2 support is essential for a software product to be eligible for procurement by U.S. Federal Government agencies, as well as other government and non-government entities.
- UniVerse user number assignment and management

  UniVerse users now share a common pool of memory segments across all users. Previously, users had their own printer segment.
- U2 Data Replication

  There are several U2 Data Replication enhancements added at 11.3.1.
- Dynamic linking

  UniVerse 11.3.1 includes a change from static to dynamic linking of core UniVerse functions on UNIX and Linux platforms. Previously, dynamic libraries were only used for XML, External Database Access (EDA), or BASIC Call Interface (BCI). Also at this release, the General Calling Interface (GCI) has changed from statically linking the functions inside new executables (such as `uvsh.new`) to a dynamically linked shared library called `libu2gci.so` or `libu2gci.sl`.
- The fixtool_sa utility

  The `fixtool_sa` utility is a stand-alone version of the `fixtool` utility that allows you to verify and repair UniVerse files.
- Updated uvconfig file

  The `uvconfig` file has been updated at 11.3.1 to accommodate new parameters and update values of existing parameters.

Video-documented features

In this release, videos are available to help you understand some of the important new features in UniVerse 11.3.1.

Click the link to view the video on YouTube.

Link to video **Introducing U2 Python**

Create new programs in a modern language, while still accessing existing BASIC programs that were created in the multivalue UniVerse database. U2 Python provides the ability to use open source support and third-party packages using the Python community that can access U2 resources such as data, subroutines, query tools, and so on.

Link to video **Starting and running Python programs in UniVerse**

Start a Python prompt at UniVerse TCL by entering the `PYTHON` command; run a Python program from TCL by entering the `RUNPY` command; and start and run Python programs from the Windows command prompt. To follow along with this video in a lab environment, you must have the 11.3.1 UniVerse beta release that has Python licensed for use.

Link to video **U2 Audit Logging**

U2 Audit Logging is a comprehensive, flexible, secure solution that tracks the usage of database resources and related authentication and authorization operations. UniVerse 11.3.1 includes enhancements that improve the overall system performance of U2 Audit Logging.

Link to video **Rocket SystemCure for UniVerse**

Rocket SystemCure for UniVerse is a new Rocket product that offers simplified discovery, monitoring, visualization, and notifications of events in your UniVerse database and its server. Rocket SystemCure is a subset of the Rocket NetCure product. Unlike NetCure, where you can discover and see everything at the network engineer level, SystemCure focuses directly to the database level and anything that can affect its performance.

Parent topic: New features in v11.3.1

64-bit builds

Starting at UniVerse 11.3.1, UniVerse builds are 64-bit only on all platforms. As a consequence, Windows 2003 Server, Windows XP, and Windows Vista are no longer supported platforms.

The Linux base platform is now RHEL 6.0, so 5.x is no longer supported.

Parent topic: New features in v11.3.1

IPv6

UniVerse 11.3.1 supports the IPv6 protocol.

IPv6 compatibility provides the ability to create more IP addresses, as IPv4 addresses are running out. In addition, IPv6 utilizes a more secure data package. However, UniVerse device licensing is not supported on IPv6 protocols at this time. If you are connected using an IPv6-only client, the `uvlictool` command displays 127.0.0.1 (localhost) as the IP address.

Parent topic: New features in v11.3.1

U2 Audit Logging

There are several enhancements to U2 Audit Logging in UniVerse 11.3.1.

For more information about the following new features, see the chapter about U2 Audit Logging in the *Security Features Guide*.

- New `audman` configuration commands have been added. Use the `audman` utility to manage U2 Audit Logging and send requests to the `uvaudd` daemon. When using the `uvaudd` related commands from the `audman` utility, the actions are logged into the `uvaudd.log` file. You can alternatively use XAdmin starting at version 4.12.0 to configure U2 Audit Logging.
- New `uvconfig` file parameters for configuring U2 Audit Logging have been added. See Updated uvconfig file, on page 9.
- Three audit log file types are now available: hashed log file, sequential log file (default), and the operating system `syslog` file (for UNIX and Linux only).

Parent topic: New features in v11.3.1

U2 Python

Support for the Python development language has been added to UniVerse 11.3.1 on Linux and Windows only.

Python is a dynamic programming language that interfaces with UniVerse 11.3.1 as a native language, similar to BASIC. UniVerse is bundled with Python 3.4.1, but it can work with other Python 3.4.x feature releases by modifying the `.pyconfig` file to specify an existing Python installation that you want U2 to use. Other Python versions such as 3.5 are not allowed to import the u2py module in the Python environment.

To license U2 Python on UniVerse, perform one of the following actions:

- On Windows platforms, select the **Python** check box during installation.
- In XAdmin, from the Admin Tasks pane, double-click **License**. From the **Update** tab, click the **PYTHON** check box, or use the wizard and select **PY**.
- On Linux platforms, add `PY:1` to the Linux license screen on the package line.
- On either Windows or Linux platforms, using `uvregen`, enter `bin/uvregen -p PY:1` in $UVHOME.

There are several enhancements related to U2 Python in UniVerse 11.3.1. For more information about the following new features, see the new *U2 Python User Guide*.

- Two TCL commands `RUNPY` and `PYTHON` have been added. `RUNPY` runs a Python program from TCL. `PYTHON` launches Python's interactive shell and can execute Python commands.
- A new BASIC variable type, PYOBJECT is available. PYOBJECT is used internally to receive a Python object.
- Three new @variables have been added: @PYEXCEPTIONMSG, @PYEXCEPTIONTRACEBACK, @PYEXCEPTIONTYPE.
- Several U2 BASIC API functions have been added, including PyCall, PyCallFunction, PyCallMethod, PyGetAttr, PyImport, and PySetAttr.
- The u2py extension module has been added to make use of the modified UniObjects API and provide access to the U2 server.

Parent topic: New features in v11.3.1

Rocket SystemCure for UniVerse

UniVerse 11.3.1 includes support for SystemCure. Rocket SystemCure for UniVerse 1.1.0 is a new product that offers simplified discovery, monitoring, visualization, and triggering of actions or notifications for UniVerse system events.

SystemCure focuses on UniVerse instances, providing real-time dashboards and reports, as well as monitoring and analysis for availability, performance, trends, correlations, and capacity planning for global and user-session metrics. Events can be set up to trigger notifications and script execution automatically. SystemCure provides system-wide information for earlier recognition and quicker diagnosis of degradations and problems.

In order to access the UniVerse database, a SystemCure add-on license will be needed in Rocket Business Connect for each serial/license number.

To add the license on Windows, select the **SystemCure** check box during installation. On the UNIX license screen, add the `CURE:1` option to the `Package List` line. When using the `uvregen` tool from the command line, add the `-p CURE:1` option. The performance memory segment and SystemCure agents will not start unless the CURE licensing add-on is enabled and licensed. Additionally, the PERF_MON_MODE value is set to 1 by default to enable performance monitoring. This allows UniVerse and the SystemCure agents to communicate with each other.

Parent topic: New features in v11.3.1

FIPS 140-2

The Federal Information Processing Standard (FIPS) Publication 140-2 is a United States Federal Government computer security standard used to accredit cryptographic modules. FIPS 140-2 support is essential for a software product to be eligible for procurement by U.S. Federal Government agencies, as well as other government and non-government entities.

With FIPS 140-2 support, you can:

- configure UniVerse 11.3.1 to run in FIPS mode by modifying the `uvconfig` file, utilizing TCL commands, or the Windows Registry.
- convert certain persistent data, such as SCRs and SPLs, to be able to run under the FIPS mode.

For more information about FIPS 140-2 support, see the chapter about FIPS 140-2 support in the *Security Features Guide*.

Parent topic: New features in v11.3.1

UniVerse user number assignment and management

UniVerse users now share a common pool of memory segments across all users. Previously, users had their own printer segment.

Certain customer applications could be impacted if they capture the NET signature user number. The following changes have the most impact:

- The `@USERNO` value has been changed for sessions. The value for `@USERNO` is now the position in the LCT table for a given pid. The LCT table can be seen with `uvsms -L` and will show the pids that are currently logged in. Any entries reported as `-1` in the table indicate that the corresponding slot in the table is not currently in use.

For example, if slot 1 of the LCT table has a pid of 31755 and your pid matches, then the `@USERNO` for your session will be 1. If USE_FIXED_PORT in the `uvconfig` file is 0, the `unohist` file is no longer used. If USE_FIXED_PORT is set to a positive number, the `unohist` file is still used.

For phantoms and UniRPC connections, the `@USERNO` value is the negative representation of the position in the LCT table. Any interactive session will be a positive number. A value of zero is not used. Background and phantom processes show the @USERNO as a negative number.

- The `USERINFO(`*`code`*`, `*`value`*`, `*`userinfo`*`)` function has been added. This BASIC function allows you to get the pid of an existing user or get the user number based on the pid. For more information about syntax, see the *BASIC Commands Reference*.
- Previously, when performing the `SH` or `DOS` command to enter the operating system prompt and starting an additional UniVerse shell, the `LOGIN` paragraph was bypassed, allowing a potential security hole. This vulnerability has been addressed. Starting at this release, the `LOGIN` paragraph is executed by the second UniVerse shell unless the `LOGIN` execution is disabled. The additional shell now consumes its own LCT slot and a UniVerse license. In order for phantoms to bypass the `LOGIN` paragraph, add this example `LOGIN` paragraph (modified accordingly for your account):

```
Paragraph
IF @TTY = 'phantom' THEN GO END.OF.LOGIN
DISPLAY CALL_MY_MENU
END.OF.LOGIN:
```

- The `LISTUSER` command has been added. The following example illustrates this command.

```
# LISTUSER
UsrNo Pid.... UID.. UserName Type Acct................ LogonTime..............
    1   25161     0 root     Term /usr/uv              Mon Aug  1 12:31:41 2016
    2   25168     0 root     Term /usr/uv/HS.SALES     Mon Aug  1 12:32:30 2016

Total lines returned: 2
```

Parent topic: New features in v11.3.1

U2 Data Replication

There are several U2 Data Replication enhancements added at 11.3.1.

The following sections briefly describe some of these new features; for more information, see the *U2 Data Replication User Guide*.

Asynchronous CGTs

A cross-group transaction (CGT) is a transaction involved in multiple replication groups. At UniVerse 11.3.1, asynchronous CGT processing has been added to improve replication performance. In an asynchronous methodology, each replication group handles CGTs separately as a single-group transaction. Writer processes (RWs – uvrw for UniVerse, and udrw for UniData) only wait on logs from their own group to arrive and load before they apply the logs to the subscriber database. The subscriber process sends the sub-done acknowledgement if all the logs in its own group are committed, without waiting for other involved replication groups. This methodology improves the overall throughput on the subscriber as it eliminates the potential for some replication groups to pause processing while waiting for the other groups in a CGT to complete.

For more information, see the section called "Cross-group transaction" in the *U2 Data Replication User Guide*.

Delayed standby replication

Delayed replication allows updates to be delayed by a specified time on the subscribing system in U2 Data Replication. The publishing system can failover to the subscriber system up to a specified time. This will allow you to cancel or void some unwanted updates during the failover to protect the database from accidental misuse or malicious damage.

For more information, see the section called "Delayed replication" in the *U2 Data Replication User Guide*.

Replication pacing

Replication pacing allows U2 Data Replication to gracefully slow down the pace of publisher database updates when replication becomes overflowed. This reduces the likelihood of the replication logs overflowing and ultimately disabling U2 Data Replication.

Prior to UniVerse 11.3.1, when the total replication log file exceeded its limit, U2 Data Replication was disabled. Starting at 11.3.1, pacing gracefully slows down the database updates to prevent too many replication overflows to the log files. As the publisher process slowly paces the updates, the subscriber is able to catch up according to priorities set by the administrator.

Administrators can define a session priority level that guides the pacing mechanism in recognizing when to slow down and when to resume normal speeds. Instead of overflowing the log file, the number of data updates sent to the log file is reduced, minimizing the possibility of disablement.

For more information, see the section called "Replication pacing" in the *U2 Data Replication User Guide*.

REPLOGGER

The REPLOGGER utility is now installed automatically with UniVerse 11.3.1. REPLOGGER is a BASIC program that allows you to monitor replication. REPLOGGER collects information from the replication system using the replication administration uvreptool and writes the information to a sequential log file. The information logged by REPLOGGER describes how the replication system performs over a period of time. Using REPLOGGER helps you understand and analyze how your replication system responds to changing workloads. REPLOGGER can produce a large amount of data, so to aid in analyzing and visualizing the data, a Java tool called the U2 Replication Analyzer is available on github to graph the result sets.

For more information, see the section called "Monitoring replication with REPLOGGER" in the *U2 Data Replication User Guide*.

Parent topic: New features in v11.3.1

Dynamic linking

UniVerse 11.3.1 includes a change from static to dynamic linking of core UniVerse functions on UNIX and Linux platforms. Previously, dynamic libraries were only used for XML, External Database Access (EDA), or BASIC Call Interface (BCI). Also at this release, the General Calling Interface (GCI) has changed from statically linking the functions inside new executables (such as `uvsh.new`) to a dynamically linked shared library called `libu2gci.so` or `libu2gci.sl`.

> **Note:** On Linux platforms, the shared library extension is `.so` – shared object. This is the same as most UNIX platforms, except on HP-UX, which uses the `.sl` extension – shared library.

Dynamic linking allows the C code routines to be integrated with UniVerse and called by BASIC programs. With dynamic linking, GCI utilities do not have to be statically linked to each new release, allowing easier installation of product updates and reducing system administration.

Changing from static to dynamic linking offers the following benefits:

- One copy of each function is stored in memory and shared among all users.
- The UniVerse installation image size has been reduced.
- UniVerse executable program sizes have been reduced.
- Patching is easier in some cases as only library changes might be necessary.
- Linking is now performed at run time.

For more information, see the section called "Dynamic linking" in *Administering UniVerse*.

UniVerse functionality changes due to dynamic linking

The uv script

Starting at UniVerse 11.3.1, the `uv` executable is now a script. This change was primarily done for `setuid` bit issues, but has an added benefit of allowing dynamic library variables to be set. If one of the `uv -admin` functions is needed, the script calls the `uvadmin` executable in UniVerse `bin`; otherwise, it calls the `uvsh` executable.

If calling the `uv` script, the dynamic linking environment variables are set to include the UniVerse `bin` directory before executing `uvadmin` or `uvsh`. This change was done to reduce the chances of a missing library error. If the variables were previously set, the UniVerse `bin` directory is included at the end of the variables.

If you are using U2 Python, the UniVerse Python `lib` directory is included after the UniVerse `bin` path.

UniVerse scripts

The following scripts in UniVerse `bin` have been modified to include the dynamic link libraries:

- `uv`
- `uv.rc`
- `showuv`
- `uvdiag` (v5.1.0+)
- `uv.load`
- `uv.install`
- `uvipcrm`
- `updatercs.sh`

For `uv` and `uv.rc`, if the variables were set previously, the UniVerse `bin` is added to the end of the variables. For the other scripts noted, the UniVerse `bin` directory is added to the front of the variable definition. If the environment variables were not previously set, the UniVerse `bin` directory is added as the only value. On AIX platforms, LIBPATH is used. On HP-UX platforms, SHLIB_PATH is used. For all platforms, LD_LIBRARY_PATH is set.

GCI

Previously, when running `GCI.ADMIN` option 4 to make a new UniVerse shell, the process would create a `uvsh.new` file and other executables in the UniVerse home directory. This process allowed administrators to test the GCI subroutine changes before updating the `uvsh` executable in the UniVerse `bin` directory.

With the changes to dynamic linking, this process no longer works as the `uvsh.new` executable would still reference the default `libu2gci.so` file. If the `libu2gci.so` process found first in the search order for your platform does not include referenced GCI subroutines, the error in the following example is seen:

```
>RUN BP TEST
Program "TEST": Line 4, Invalid GCI subroutine.
```

Option 4 of the GCI Administration menu has been changed at this release to create the `libu2gci.so/.sl` file in the `$UVHOME` directory. To test these changes, set LD_LIBRARY_PATH, LIBPATH, and/or SHLIB_PATH to include `$UVHOME` before you make any references to UniVerse `bin`. Any new `uvsh` calls will include the proper GCI library. No `*.new` files are created in `$UVHOME` at 11.3.1. If they do exist, then they were from versions 11.2.x and earlier and should be removed.

When using only option 4 (staging) and the `libu2gci.so/sl` file is located in the UniVerse home directory, only `uvsh` can be used to test the changes by setting the LD_LIBRARY_PATH, LIBPATH, and/or SHLIB_PATH. UniRPC client connections cannot access the `$UVHOME/libu2gci.so` or `.sl` file; they can only access the shared libraries in the UniVerse `bin` directory.

When you are ready to put the new GCI library into production, run option 5 in the GCI Administration menu as was done previously. This copies the `libu2gci.so/.sl` file from `$UVHOME` to the UniVerse `bin` directory. It also copies the file to `$UVHOME/lib.d` if found. After running option 5, new UniVerse shells or UniRPC connections recognize the modified changes. Existing connections use the version of the GCI library loaded at the time of execution.

Parent topic: New features in v11.3.1

The fixtool_sa utility

The `fixtool_sa` utility is a stand-alone version of the `fixtool` utility that allows you to verify and repair UniVerse files.

`fixtool_sa` is independent from the version of UniVerse installed and can be copied and executed to verify and repair UniVerse files on similar hardware running earlier versions of UniVerse.

> **Warning:** `fixtool_sa` does not support locks set by other processes. `fixtool_sa` should not be used on a file that is actively being used by UniVerse processes as no concurrency locking is performed.

To repair a file (-fix) with `fixtool_sa` on a system where UniVerse is started, the -force option must be specified. Other than the -force option, the command syntax for `fixtool_sa` is identical to `fixtool`. Refer to the `fixtool` documentation in the *User Reference Guide* for additional information, including examples.

Parent topic: New features in v11.3.1

Updated uvconfig file

The `uvconfig` file has been updated at 11.3.1 to accommodate new parameters and update values of existing parameters.

For more information about the full contents of the `uvconfig` file, see *Administering UniVerse*.

The following table describes the new parameters added at 11.3.1.

Parameter	Description
AUDIT_LOG_TYPE	Specifies the audit log file type. The default is 2 (sequential file). Valid options are: ▪ 1 - Type-30 UniVerse dynamic file ▪ 2 - Operating system sequential file (default) ▪ 3 - Operating system `syslog` file (valid only for UNIX/Linux)

Parameter	Description
AUDIT_SEQ_BUF_SZ	Specifies the size of each memory buffer used for sequential log files in 1K blocks. UniVerse will have all buffers initialized, but only the first AUDIT_LOG_MAX ones activated. UniVerse processes (such as uvsh) randomly choose an active buffer each time a log record is produced to provide balanced load for all buffers and to accommodate buffer suspension/resumption. A larger size can improve audit performance. This parameter only applies to the sequential audit log file. The unit is 1024-byte block. The default is 1024 (1 MB).
AUDIT_SEQ_FILE_SWITCH	Specifies the amount of time in seconds before UniVerse must switch to a new audit log file. On a busy system, the log file can grow very rapidly. Decide the value carefully based on your application and system load. The format of this parameter is `uvaud.`*`ddmmyyyy.seq`*, where *seq* is a sequential number determined by UniVerse to guarantee the uniqueness of the log files. The timestamp is determined by UTC time. This parameter only applies to the sequential audit log file. The unit is 1 second. The default is 0 (no time switch limit).
AUDIT_SEQ_FILE_SZ	Specifies the maximum size of a sequential log file in 1K blocks. UniVerse will automatically switch to a new audit log file when its size has reached the maximum. UniVerse may switch to a new file even if the maximum is not reached. See AUDIT_SEQ_FILE_SWITCH for more information. This parameter only applies to the sequential audit log file. The unit is 1024-byte block. The default is 500000 (500 MB), and the minimum size should not be less than 1MB. The total size can be larger than 4GB.
AUDIT_SEQ_OUTBLK_SZ	Specifies the output block size in 1K blocks used by `uvaudd` daemon for sequential log files. `uvaudd` writes to a log file only after at least this size of data was available from the buffer. It is not necessarily equal to the system physical block size. Depending on platforms, a larger block size can improve performance but can cause more log data loss in the case of a system crash. This parameter only applies to the sequential audit log file. The unit is 1024-byte block. The default is 4 (4 KB). Any value less than 1 or greater than 32 will default to 4.
AUDIT_SEQ_SYNC_CNT	Specifies the number of cached log records when UniVerse must put the cached data into the system buffer. On a busy system, the log data can grow very rapidly. Decide the value carefully based on your application and system load. If the value is 0, data is not cached and the audit log record is immediately put into the system buffer. A larger count value may cause more log data loss in the case of a system crash. See AUDIT_SEQ_SYNC_TIME for more information. This parameter may or may not improve performance. This parameter only applies to the sequential audit log file. The default is 0 (no caching), and you can choose up to 1000.
AUDIT_SEQ_SYNC_TIME	Specifies the amount of time in milliseconds (up to 5,000) before UniVerse must put the cached audit log records into the system buffer. On a busy system, the log data can grow very rapidly. Decide the value carefully based on your application and system load. If the value is 0, data is not cached and the audit log record is immediately put into the system buffer. A longer sync time may cause more log data loss in the case of system crash. See AUDIT_SEQ_SYNC_CNT for more information. This parameter may or may not improve performance. This parameter only applies to the sequential audit log file. The unit is 1 millisecond. The default is 0 (no caching).

Parameter	Description
FIPS_MODE	Specifies the default system-wide FIPS mode. UniVerse uses an embedded FIPS 140-2 validated cryptographic module provided by OpenSSL. When UniVerse is running with FIPS mode enabled (value 1), all its crypto operations are performed by the embedded FIPS module where only FIPS 140-2 compliant crypto algorithms are allowed. When FIPS mode is disabled (value 0), no such restrictions apply. Default value: 0.
HTTP_DEFAULT_VERSION	Specifies the default HTTP version for HTTP requests. Currently only two values are allowed: 1.0 and 1.1. If other values are specified, UniVerse defaults to 1.1.
MAX_REP_SHMSZ_GB	The maximum shared memory segment size, in GB, for a replication group. If this value is configured, then the maximum shared memory buffer size for a replication group is the sum of MAX_REP_SHMSZ_GB * 1GB + MAX_REP_SHMSZ.
NLSLCMODE	Specifies whether locales are enabled. A value of 0 indicates that locales are disabled. A value of 1 or 2 indicates that locales are enabled. When the TIME locale is on, the value of NLSLCMODE controls the first day of the week. When set to 1, Sunday is the first day of the week. A setting of 2 results in Monday being the first day of the week.
PERF_MON_FILE	Specifies the number of UniVerse files that performance monitoring needs to trace in each UniVerse session. This value is ignored if PERF_MON_MODE is disabled or Rocket SystemCure for UniVerse is not licensed. The default value is 200.
PERF_MON_MODE	Specifies whether to enable performance monitoring. The default value is 1. Valid values are: ▪ 0 - Disable performance monitoring ▪ 1 - Enable performance monitoring
REP_ASYNCHRONOUS_TP	Specifies how U2 Data Replication handles transactional updates when files span multiple replication groups. When set to 1, the cross-group transactions are handled in asynchronous mode. When set to the default value of 0, transactional updates are handled synchronously and the transaction is not committed on the subscriber until all replication groups have been updated.
REP_DISABLE_DISK_PCT	This parameter defines the system-wide limit of disk usage for replication logs. The value is the maximum percentage the replication log files can consume of the total space available on the file system in which the logs are configured. The default uvconfig parameter is 95%. U2 Data Replication is disabled immediately if this limit is reached, and a full resynchronization of the subscriber is required after this event. If the replication log disk is shared with data files or other applications, you should properly define this percentage to prevent application or database failure due to a growing replication log file size.
TANDEM_FLAG	Specifies whether sessions are allowed to be TANDEMized. The default value is 1, meaning sessions can be TANDEMized.

Parameter	Description
TP_COMMIT_LOGGING	Specifies whether to use transaction commit logging. The default value is 0. Valid values are: 0, 1, 2, or 3. ▪ 0 - Default; Transactions are not logged. ▪ 1 - The transaction attempts to write a log record during commit. The transaction itself will continue to commit in spite of any errors encountered in writing to the TRANS_COMMIT_LOG file. ▪ 2 - The transaction attempts to write a log record during commit. The entire transaction is aborted if the attempt to write to the TRANS_COMMIT_LOG file fails. ▪ 3 - The function behaves like option 1, except that if the TRANS_COMMIT_LOG file is a UVNet file, the updates to it are in no-wait mode.
USE_FIXED_PORT	Not available on Windows. Uses the UniVerse 11.2.x and prior method of assigning user numbers based on the relative position of a tty device in the `unohist` file. First entry in the `unohist` file is user 11, second user 12, and so on. The USE_FIXED_PORT value is the quantity of user numbers that will be reserved for assignment based on tty device. For example, USE_FIXED_PORT 5 reserves users 11-16 for assignment to the first five tty devices in the `unohist` file. This value cannot exceed (NUSERS - 10). Valid values are: ▪ 0 - Do not use fixed ports ▪ positive numbers - Use fixed port numbers. The first fixed port number starts at 11. The last fixed port is 10 + USE_FIXED_PORT.

The following table describes parameters whose default values have changed from 11.2.

Parameter	Description
NUSERS	The number of user sessions (such as uvsh and PHANTOM) that can run at the same time. The recommended value is greater than or equal to the number of licensed users. This will require tuning based on the number of UniVerse processes that are required to run on the system at the same time. If this parameter is too low, you see the "No More LCTs" error. If you set NUSERS to a value below the licensed user count, UniVerse will use the licensed user count in place of NUSERS. For example, an enterprise license of 150 users should have NUSERS set to 1875 (150 * 10) + 25%. (Note: The example assumes a maximum of 10 connections for each user. Adjust this value to your average number of connections per user.) ▪ Change SHM_GNTBLS to 32 and change SHM_GNPAGES to 64 to ensure 32 * 64 = 2048, which is greater than NUSERS. ▪ Check that the UNIX value of semmni is at least 1232. (NUSERS / (NSEM_PSET+1)) + 1024. ▪ Check that the UNIX value of semmns is at least 3933. (NUSERS + 10 + 2048) ▪ Check that the UNIX value of shmmni is at least 1880. (Greater than SHM_GNTBLS or NUSERS +5) ▪ Check that the UNIX value of semmnu is at least 1775 (NUSERS).

Parameter	Description
OPTMEM	Specifies the amount of memory allocated for the query optimizer's workspace. This is specified in 1K units. The default value is 256.
SHM_GNPAGES	Number of global pages in a shared memory segment. Tune (SHM_GNTBLS * SHM_GNPAGES) high enough to avoid "No More GCTs" error. This parameter should be a multiple of 32.
SHM_GNTBLS	Number of GCTs (global control tables) in CTL. Each shared memory segment is associated with a GCT. The GCT registers the use of global pages in its associated shared memory segment. Tune (SHM_GNTBLS * SHM_GNPAGES) high enough to avoid the "No More GCTs" error.

SSL_PROTOCOLS	**UniRPC Security Protocols Configuration.** Specifies the UniRPC daemon's allowed security protocols for secure connections. The allowed protocols are SSLv3, TLSv1, TLSv1.1, and TLSv1.2.
	The valid delimiters are comma(,), and the plus sign (+).
	Invalid protocols will be ignored. If the parameter is not specified, or the resultant string is empty, then SSLv3+TLSv1+TLSv1.1+TLSv1.2 will be the default.
	Note: The default secure protocols are TLSv1.1 and TLSv1.2. Setting the minimum value to TLSv1.1 and the maximum value to TLSv1.2 allows for the strongest possible security protocol.
	UniRPC connections will work with a TLSv1 setting if it is defined in the SSL_PROTOCOLS parameter in the `uvconfig` file.
	TELNET Security Protocol Configuration. For Windows users, you must update the Current-Version registry in order to use the SSL_OPTIONS and SSL_PROTOCOLS options. The CurrentVersion registry can be found in the following location:
	`HKEY_MACHINE/SOFTWARE/Rocket Software/UniVerse/CurrentVersion`
	UniVerse Telnet connections will work with TLSv1.1 and TLSv1.2 only. For UNIX and Linux platforms, If you require any other protocol, make the changes to the `inetd.conf` file (`xinetd.conf` on Linux platforms).
	To enable only the TLSv1 protocol, use the following `uvssltelnet` example:
	`uvssltelnet stream tcp nowait root /disk1/uv113/bin/uvtelnetd uvtelnetd -d3 -P tlsv1`

Parent topic: New features in v11.3.1

New features in v11.2.4

This section summarizes the significant changes and updates for Rocket UniVerse. If you are familiar with previous releases of UniVerse, and you want to know the new features and enhancements for newer versions, you can use this section to assist you.

Disablement of Telnet services

Starting at this release, Windows users can selectively disable the Telnet port or the SSL Telnet port using XAdmin. For more information, refer to *Administering UniVerse*.

Local time zone configuration

Prior to UniVerse 11.2.4, the date and time data stored in the audit log records was based on UTC only. Beginning at UniVerse 11.2.4, UniVerse adds the date and time data based on local timezone to audit log records. The data is stored in location 19 for each record. The dictionary name for this data field is TZINFO. For more information, see the *Security Features Guide*.

Group option added to the uv.load script

Beginning at this release, the -g option has been added to the `uv.load` script. This option specifies the group name or group number on installation files. Any group permissions set specifically during the `uv.install` script will remain with the administrator's default group value. The value entered for the group_name or group_number is not validated. If not used, the default group is 1.

UOEXEC_RETAIN_LOCK parameter added to the uvconfig file

Starting at this release, a new `uvconfig` file parameter has been added: UOEXEC_RETAIN_LOCK. This parameter enables a `READU` lock that is set in a BASIC subroutine called from the UniObjects application to remain persistent until it is explicitly released.

SSL support for XAdmin

XAdmin support for encrypted SSL configuration files has been added at this release and will be fully available for connections to UniData 8.1 or later and UniVerse 11.2.4 or later when released.

XAdmin now allows users to create self-signed SHA2 certificates in the SSL Configuration Admin Task.

TOXML now distinguishes between single and multivalued fields

Prior to this release, the TOXML report export keyword treated all A or S type dictionary items as multivalued fields. Beginning at this release, TOXML will use field 5 of an A or S type dictionary to determine if the field is Single or Multivalued. If field 5 of the dictionary item does not contain 'S' or 'M', the field will be treated as multivalued by default.

New features in v11.2.3

This section summarizes the significant changes and updates for Rocket UniVerse. If you are familiar with previous releases of UniVerse, and you want to know the new features and enhancements for newer versions, you can use this section to assist you.

PKCS #12 support

Starting at this release, UniVerse supports PKCS #12 (Microsoft pfx) file formats. This enhancement creates an environment that allows the U2 certificate to recognize the pfx file format and import certificates into that store. For more information about PKCS #12 support, refer to the *Security Features Guide* and the *BASIC Commands Reference*.

Data link compression

Beginning at UniVerse 11.2.3, the U2 Data Replication process allows data to be compressed during the network transfer phase in a process called data link compression. For more information about data link compression, see the *U2 Data Replication User Guide*.

New features in v11.2.0

This section summarizes the significant changes and updates for Rocket UniVerse. If you are familiar with previous releases of UniVerse, and you want to know the new features and enhancements for newer versions, you can use this section to assist you.

U2 Audit Logging

U2 Audit Logging has been added at this release. U2 Audit Logging tracks the usage of database resources and related authentication and authorization operations. For more about U2 Audit Logging, see the *Security Features Guide*.

U2 Dynamic Object

U2 Dynamic Object (UDO) has been added at this release. UDO provides an object structure in UniVerse. For more information about UDO, see the *BASIC Extensions Guide*.

EDA Replication

At this release, EDA Replication is introduced. For more information about EDA Replication, see the *External Database Access (EDA) Reference*.

Disabling U2 Data Replication

The ability to disable U2 Data Replication has been added at this release. For more information, see the *U2 Data Replication User Guide*.

Local functions and subroutines

Local functions and subroutines has been added at this release. For more information, see the *BASIC User Guide*.

UniVerse BASIC changes

An enhancement was made to the `XDOMLocate` function. The following functions were added at UniVerse 11.2:

- `XDOMLength`
- `XDOMItem`
- `XDOMGetElementById`
- `XDOMGetElementsByName`
- `XDOMGetElementsByTag`
- `XDOMGetChildNodes`
- `XDOMQuery`

Notices

Edition

Publication date: October 2016
Book number: UNV-1131-NF-01
Product version: Version 11.3.1

Copyright

Trademarks

Examples

This information might contain examples of data and reports. The examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

License agreement

Note: This product may contain encryption technology. Many countries prohibit or restrict the use, import, or export of encryption technologies, and current use, import, and export regulations should be followed when exporting this product.

Corporate information

Rocket Software, Inc. develops enterprise infrastructure products in four key areas: storage, networks, and compliance; database servers and tools; business information and analytics; and application development, integration, and modernization.

Website: www.rocketsoftware.com

Rocket Global Headquarters
77 4th Avenue, Suite 100
Waltham, MA 02451-1468
USA

To contact Rocket Software by telephone for any reason, including obtaining pre-sales information and technical support, use one of the following telephone numbers.

Country	Toll-free telephone number
United States	1-855-577-4323
Australia	1-800-823-405
Belgium	0800-266-65
Canada	1-855-577-4323
China	800-720-1170
France	08-05-08-05-62
Germany	0800-180-0882
Italy	800-878-295
Japan	0800-170-5464
Netherlands	0-800-022-2961
New Zealand	0800-003210
South Africa	0-800-980-818
United Kingdom	0800-520-0439

Contacting Technical Support

The Rocket Customer Portal is the primary method of obtaining support. If you have current support and maintenance agreements with Rocket Software, you can access the Rocket Customer Portal and report a problem, download an update, or read answers to FAQs. To log in to the Rocket Customer Portal or to request a Rocket Customer Portal account, go to www.rocketsoftware.com/support.

In addition to using the Rocket Customer Portal to obtain support, you can use one of the telephone numbers that are listed above or send an email to support@rocketsoftware.com.