Test Environment Toolkit

Release Notes for TETware Release 3.8
TET3-RN-3.8

Released: December 2005

The Open Group

The information contained within this document is subject to change without notice.

This document is produced by The Open Group at:

Thames Tower
Station Road
Reading
Berkshire
RG1 1LX
United Kingdom

++++++++++++++++++++++++++ TET END USER LICENCE ++++++++++++++++++++++++++

BY OPENING THE PACKAGE, YOU ARE CONSENTING TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT AGREE TO ALL OF THE TERMS OF THIS AGREEMENT, DO NOT INSTALL THE PRODUCT AND RETURN IT TO THE PLACE OF PURCHASE FOR A FULL REFUND.

TETWARE RELEASE 3.8 END USER LICENCE
REDISTRIBUTION NOT PERMITTED

This Agreement has two parts, applicable to the distributions as follows:

A. Free binary evaluation copies – valid for 90 days, full functionality – no warranty.

B. Free binary restricted versions – no warranty, limited functionality.

C. Licenced versions – full functionality, warranty fitness as described in documentation, includes source, binary and annual support.

PART I (A & B above) – TERMS APPLICABLE WHEN LICENCE FEES NOT (YET) PAID (LIMITED TO EVALUATION, EDUCATIONAL AND NON-PROFIT USE).

GRANT.

The Open Group grants you a non-exclusive licence to use the Software free of charge if

a. you are a student, faculty member or staff member of an educational institution (K-12, junior college, college or library) or an employee of an organisation which meets The Open Group's criteria for a charitable non-profit organisation; or

b. your use of the Software is for the purpose of evaluating whether to purchase an ongoing licence to the Software.

The evaluation period for use by or on behalf of a commercial entity is limited to 90 days; evaluation use by others is not subject to this 90 day limit. Government agencies (other than public libraries) are not considered educational or charitable non-profit organisations for purposes of this Agreement. If you are using the Software free of charge, you are not entitled to hard-copy documentation, support or telephone assistance. If you fit within the description above, you may use the Software for any purpose and without fee.

DISCLAIMER OF WARRANTY.

Free of charge Software is provided on an ''AS IS'' basis, without warranty of any kind.

THE OPEN GROUP DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE OPEN GROUP BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

PART II (C above) – TERMS APPLICABLE WHEN LICENCE FEES PAID.

GRANT.

Subject to payment of applicable licence fees, The Open Group grants to you a non-exclusive licence to use the Software and accompanying documentation (''Documentation'') as described below.

LIMITED WARRANTY.

The Open Group warrants that for a period of ninety (90) days from the date of acquisition, the Software, if operated as directed, will substantially achieve the functionality described in the Documentation. The Open Group does not warrant, however, that your use of the Software will be uninterrupted or that the operation of the Software will be error-free or secure.

SCOPE OF GRANT.

Permission to use for any purpose is hereby granted. Modification of the source is permitted. Redistribution of the source code is not permitted without express written permission of The Open Group. Distribution of sources containing adaptations is expressly prohibited.

Redistribution of binaries or binary products containing TETware code is permitted subject to the following conditions:

— this copyright notice is included unchanged with any binary distribution;

— the company distributing binary versions notifies The Open Group;

— the company distributing binary versions holds an annual TET support agreement in effect with The Open Group for the period the product is being sold, or a one off binary distribution fee equal to four years annual support is paid.

Modifications sent to the authors are humbly accepted and it is their prerogative to make the modifications official.

Portions of this work contain code and documentation derived from other versions of the Test Environment Toolkit, which contain the following copyright notices:

The unmodified source code of those works is freely available from `ftp.rdg.opengroup.org`. The modified code contained in TETware restricts the usage of that code as per this licence.

++

1. Introduction

1.1 Preface

These release notes accompany TETware Release 3.8.

There are two principle versions of TETware. One version is known as Distributed TETware. This version uses a client-server architecture and provides support for processing local, remote and distributed test cases. The other version is known as TETware-Lite. This version does not use a client-server architecture or require a network transport. It is more simple to set up than is Distributed TETware but does not provide support for the processing of remote or distributed test cases.

Both versions of TETware are fully supported on UNIX operating systems and also on the Windows NT/2000/XP operating system. This release of TETware has not been tested on Windows 95/98/Me, although it is expected that TETware-Lite will still be usable on them. However, no support is provided by The Open Group for such use.

Throughout this document, the Windows NT/2000/XP and Windows 95/98/Me operating systems are referred to collectively as **Win32 systems**. The individual system names are only used when it is necessary to distinguish between them.

1.2 Audience

This document is intended to be read by software engineers and/or systems administrators who will install TETware on their computer systems. A knowledge of system administration is assumed when TETware installation and configuration instructions are presented. In addition, a knowledge of network administration is assumed when TETware is to be built to use network transports.

1.3 Conventions used in this document

The following typographic conventions are used throughout this document:

- `Courier font` is used for function and program names, literals and file names. Examples and computer-generated output are also presented in this font.
- The names of variables are presented in *italic font*. You should substitute the variable's value when typing a command that contains a word in this font.
- **Bold font** is used for headings and for emphasis.

1.4 Related documents

Refer to the following documents for additional information about TETware:

- *Test Environment Toolkit: TETware Installation Guide*
  There is one version of this document for each operating system family on which TETware is implemented.
- *Test Environment Toolkit: TETware User Guide*
- *Test Environment Toolkit: TETware Programmers Guide*

- *Test Environment Toolkit: TETware Knowledge Base*

Source and postscript versions of these documents are included in the TETware distribution.

1.5 Problem reporting

If you have subscribed to TETware support and you encounter problems when building or executing TETware you should take a copy of the error reporting template contained in the file *tet-root*`/doc/tet3/err.template` in the distribution, fill in the details of the problem, and send it by electronic mail to:

`tet_support@opengroup.org`

Alternately you may use a web browser to complete and submit the HTML form included in the file *tet-root*`/doc/tet3/errtemp.html` in the distribution.

You should include sufficient information in your report to enable someone who is unfamiliar with your system to be of assistance in solving the problem.

Users are reminded that the user-contributed software which accompanies the TETware distribution is not covered by TETware support services.

2. New features and notable changes

The following features appear for the first time in this release of TETware:

- The Python API is now part of the supported release for UNIX systems (it was previously user-contributed software). Its use is dependent on shared-library versions of the TETware libraries being built.
- A Distributed Korn Shell API has been added.
- Remote systems can be disconnected within the last test purpose of a test case and be reconnected when the next test case is started. The main purpose of this feature is to allow remote systems to be rebooted, but it can be used in any situation where connections to `tccd` on a remote system are broken and need to be re-established. The feature must be enabled by adding indicators to the `:remote:` or `:distributed:` directive in the scenario file to specify which systems need to be reconnected, otherwise TETware will report disconnections as errors in the normal way.
- Timestamps in the journal can now be in two formats. The default is *HH*`:`*MM*`:`*SS* as in previous TETware releases. If the TET_FULL_TIMESTAMPS parameter is set to `True` then timestamps are generated in either the format *YYYY*`–`*MM*`–`*DD*`T`*HH*`:`*MM*`:`*SS*`.`*mmm* or the format *YYYY*`–`*MM*`–`*DD*`T`*HH*`:`*MM*`:`*SS*, depending on which part of TETware generated the timestamp and whether it was configured to support milliseconds when built.
- The `systems.equiv` file can now specify domains such as `.opengroup.org` and subnets such as `192.168.1.0/24`, not just individual host names and addresses.
- The main `configure` script now recognises MacOS X, Solaris 10 and QNX.
- When any test purpose produces a result code that has the `Abort` action specified in the `tet_code` file, `tcc` will abort the test run after processing the current test case.

More information about these features is contained in the relevant sections of the TETware Installation Guides, User Guide and Programmers Guide.

3. Problems fixed since the last release

The following problems have been fixed since the last TETware release:

- The main `configure` script no longer uses the non-portable `getopt` utility.
- By default the `defines.mk` file created on certified UNIX® systems defines the appropriate standard feature-test macros in the compiler options. This enables support for milliseconds when the new full-timestamps feature is used, and also enables the use of standards-compliant code in other areas mentioned below. The standard feature-test macros are also defined by default for other systems that recognise them but are not UNIX® certified, such as Linux.
- The POSIX Shell API was using some old-style `trap` reset commands with no dash (`trap signal_list`). These have been changed to use a dash (`trap - signal_list`).
- The `tet_vprintf()` function in the C API library no longer produces a bogus ‘‘unexpected EOF’’ warning message when `va_copy` is not defined and the formatted output is an empty string.
- `tccd` now always accepts connections from the host name `localhost`, so it no longer needs to be specified in `systems.equiv`.
- The address length argument to the `accept()`, `getsockname()` and `getpeername()` functions is now passed as a pointer to `socklen_t` if the standard feature-test macros for UNIX98, UNIX03 or POSIX01 are defined, and as a pointer to `size_t` if the standard feature-test macros for UNIX95 are defined. (Otherwise it is a pointer to `int` as before.)
- In TETware-Lite, `tcc` was not passing trace flags down to the TCM.
- The standard `poll()` function is now used if the standard feature-test macros for UNIX95, UNIX98 or UNIX03 are defined. (Previously it had to be enabled by defining SVID3_POLL.)
- The `tet_eaccess()` library function now passes a buffer large enough for NGROUPS_MAX+1 groups (which `getgroups()` can return on UNIX03 and POSIX01 systems). It has also been changed to avoid non-portable assumptions about mode values.
- The directory `/var/tmp` has been added to the default list of temporary directories that the `tet_mktfname()` library function tries to use. It has also been changed to report an error (and a suggestion to set the TMPDIR environment variable) if none of the directories it tries are usable.
- The standard `setsid()` function is now used if the feature-test macro _XOPEN_SOURCE is defined. Previously its use was only enabled by defining _POSIX_SOURCE, _POSIX_C_SOURCE or HAS_SETSID.
- In the Java API the `compareTPs()` function in `SimpleTestCase.java` has been changed to use a numeric comparison.
- In `src/tet3/inetlib/nbio.c` non-portable headers are now only included if no standard feature-test macros are defined.
- The bogus `endpwent()` call has been removed from `src/tet3/tccd/tccd.c`.

- In places where code previously tried to calculate the maximum time interval representable as a `time_t` value by using bit-shifting, it now uses a simple fixed value of 2147483647 seconds.
- A few other minor portability improvements have been made in various places in the C code, such as eliminating some implicit function declarations that had not been caught before.

4. Known problems in this release

There are no known problems in this release.

5. Building and installing TETware

5.1 Building and installation instructions

For information on how to build and install TETware, please following the instructions in the version of the TETware Installation Guide which is appropriate for your system.

Note: In this release the Java API may be built on Solaris, Linux and Win32 systems. In order to build the Java API on these systems a variable must be set in the `defines.mk` file which specifies where the Java Development Kit (JDK) has been installed on your machine.

On Solaris Release 7 and later the JDK is supplied with the operating system and so is installed in a standard place (`/usr/java`). This location is specified in the `defines.mk` files for such systems that are supplied in the distribution.

However, on other platforms the JDK might be installed anywhere, so it is necessary to customise your `defines.mk` file if you want to build the Java API. Refer to the section entitled ‘‘Support for Java’’ in the TETware Installation Guide for instructions on how to do this.

In the `defines.mk` file on UNIX systems it is necessary to specify the list of signals that are used by the Java Virtual Machine, so as to avoid conflict with the use of signals by the TETware Java API runtime support library. This signal list may change, depending on which JDK and/or operating system version you are using.

The `defines.mk` files for various UNIX platforms on which the Java API is supported contain signal lists for particular JDK versions. If you are using a different JDK version you may need to change this list. Please refer to the section entitled ‘‘Support for Java’’ in the TETware Installation Guide for UNIX Operating Systems for further details.

5.2 Installed platforms

5.2.1 UNIX systems

Both TETware-Lite and Distributed TETware versions have been installed and tested on the following platforms:

- HP-UX 11.00 (hppa)
- HP-UX 11.23 (ia64)
- Linux systems compliant with LSB 2.0 (i386)
- Solaris 8 using the Sun Studio 11 `c89` compiler (sparc-32)
- Solaris 10 using the Sun Studio 10 `c99` compiler (sparc-64)

5.2.2 Win32 systems

TETware-Lite and Distributed TETware have been installed and tested on Intel PCs running Windows 2000.

5.3 API status

5.3.1 Thread-safe APIs on Win32

On Win32 systems the thread-safe APIs must be used with the multi-threaded DLL version of the C runtime support library. Use with the multi-threaded static version of the C runtime support library is not supported.

5.3.2 Java

The Java API is supported on Linux, Solaris 8 and later, and Win32 systems.

5.3.3 Perl

The Perl API uses perl 5 syntax such that it executes without warnings when run under the control of `perl –w`. However, test case authors should note that perl test cases may still emit warnings, when run under the control of `perl –w`, about API interface variables defined in the test case only being used once. For example, consider the following trivial perl test case:

```
#!/usr/bin/perl

@iclist=("ic1");
@ic1=("tp1");

$tet'startup = "startup";
$tet'cleanup = "cleanup";

sub startup
{
        &tet'infoline("in startup function");
}

sub cleanup
{
        &tet'infoline("in cleanup function");
```

```
}

sub tp1
{
        &tet'infoline("This is tp1 in a simple perl test case");
        &tet'result("PASS");
}

require "$ENV{\"TET_ROOT\"}/lib/perl/tcm.pl";
```

When run with `perl –w`, the following warnings are generated from the test case source file:

```
Name "tet::startup" used only once: possible typo at simple.pl line 6.
Name "tet::cleanup" used only once: possible typo at simple.pl line 7.
Name "main::ic1" used only once: possible typo at simple.pl line 4.
Name "main::iclist" used only once: possible typo at simple.pl line 3.
```

When Perl version 5.6 or later is used, it is possible to suppress these warnings by use of the `our` keyword in the test case source file. For example:

```
#!/usr/bin/perl

@iclist=("ic1");
@ic1=("tp1");

$tet'startup = "startup";
$tet'cleanup = "cleanup";

our(@iclist, @ic1);                                          ←

{                                                            ←
        package tet;                                         ←
        our($startup, $cleanup);                             ←
}                                                            ←

rest of test case . . .
```

In this example the code that has been added in order to suppress warning messages is indicated by a ← character in the right margin.

5.4 Transport-specific status

5.4.1 Sockets network interface

The sockets network interface is the preferred configuration for Distributed TETware and its use is recommended over XTI on all systems that support sockets.

5.4.2 XTI network interface

There is a known problem with some SVR4 XTI implementations in which the `t_sync()` function does not work correctly. An XTI implementation may hold transport endpoint data either in kernel or in user address space. The `t_sync()` function is included in the XTI specification for the benefit of implementations which do not automatically detect when the transport endpoint data held in user space is lost. Examples of when such loss might occur are when the file descriptor underlying a transport endpoint is duplicated using `fcntl()` or when the process address space is overlaid by one of the `exec()` system calls.

Since the XTI version of TETware may perform both of these operations, it will not function when used with an XTI implementation which does not automatically detect the loss of data held in user address space and in which `t_sync()` does not perform the advertised function. A common symptom of this problem is when one of the servers `tetsyncd` and `tetxresd` fails with a `TBADQLEN` error associated with a `t_listen()` call soon after being started by `tcc`.