


# Installation Guide for the TITAN TTCN-3 Test Executor

Jenő Balaskó, Ádám Knapp

Version 11.2.1, 2026-08-05

# Table of Contents

1. Introduction . . . 2
   1.1. Overview . . . 2
   1.2. Target Groups . . . 2
   1.3. Typographical Conventions . . . 2
   1.4. Prerequisites . . . 2
   1.5. Installing Prerequisites on Cygwin (on Windows) . . . 3
2. Installing only for Ericsson Users Working in E2C with AFS Service . . . 6
3. Installing from a pre-built binary package . . . 7
   3.1. Downloading the Software . . . 7
   3.2. Installing the Package . . . 7
   3.3. Install TITAN with Clang . . . 8
4. Building Titan from source code . . . 10
   4.1. Obtaining the source code to your local machine . . . 10
5. Setting the User Environment . . . 11
   5.1. Environment Variables . . . 11
   5.2. Modification of the User Login Script . . . 11
   5.3. Modifying Makefile Library . . . 12
6. Licensing (Only for Ericsson users) . . . 13
   6.1. Obtaining License Key (Only for Ericsson users) . . . 13
   6.2. Installing the License Key . . . 14
7. References . . . 16

## Abstract

This document describes the detailed information of installing TITAN TTCN-3 Test Executor and all of its components.

### Copyright

Copyright (c) 2000-2026 Ericsson Telecom AB.
All rights reserved. This program and the accompanying materials are made available under the terms of the Eclipse Public License v2.0 that accompanies this distribution, and is available at https://www.eclipse.org/org/documents/epl-2.0/EPL-2.0.html

### Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

# Chapter 1. Introduction

## 1.1. Overview

This document describes obtaining the TITAN TTCN-3 Test Executor software, installing the Test Executor and all of its components, setting the user environment, and licensing mechanism.

## 1.2. Target Groups

This document is for all audience who intend to create and execute simulations.

## 1.3. Typographical Conventions

This document uses the following typographical conventions:

- **Bold** is used to represent graphical user interface (GUI) components such as buttons, menus, menu items, dialog box options, fields and keywords, as well as menu commands. Bold is also used with '+' to represent key combinations. For example, **Ctrl+Click**
- The "/" character is used to denote a menu and sub-menu sequence. For example, **File / Open**.
- `Monospaced` font is used represent system elements such as command and parameter names, program names, path names, URLs, directory names and code examples.
- **`Bold monospaced font`** is used for commands that must be entered at the Command Line Interface (CLI), For example, **`ttcn3_start`**

## 1.4. Prerequisites

The supported platforms are Solaris, Linux and Cygwin (on Windows platforms).

The following are required for proper operation of the TITAN:

- Openssl-devel 0.9.8k or higher (on Ubuntu: libssl-dev)
- Libxml2-devel 2.7.1 or higher
- JDK 1.5.0_10 or later (only required if the JNI-based executor in the Executor plug-in is used. JNI cannot be used on Cygwin.) - *obsolete, will be removed in later releases*

| | |
|---|---|
| **NOTE** | If the platform has other, but compatible, version of above tools, TITAN will be built with those. |

On Linux, the platform-supplied versions of OpenSSL-devel and libxml2-devel are used. OpenSSL is usually installed by default. The libxml2 package and the development packages may need to be installed manually.

The development packages should be called openssldev (or devel) or libopenssldev (or devel) and libxml2dev (or devel) respectively.

To deploy the prerequisites is special on Cygwin therefore it is discussed below.

# 1.5. Installing Prerequisites on Cygwin (on Windows)

To deploy the prerequisites is special on Cygwin therefore it is discussed below.

Titan is always built on the newest Cygwin version available.

- If Cygwin has been installed already, refresh your Cygwin installation. Start the Cygwin setup utility (see below). It will refresh your installed Cygwin packages to the newest versions.
- If Cygwin hasn't been installed yet:
  1. Download and execute the latest Cygwin installer utility depending on your platform and the Titan package to be downloaded:
     32-bit version: https://cygwin.com/setup-x86.exe
     64-bit version: https://cygwin.com/setup-x86_64.exe
  2. Select Install from Internet (recommended to save local disk place)
  3. Choose Cygwin installation root directory (C: is recommended).
  4. Select All users or Just Me.
  5. Select "Local Package Directory" (typically the same directory, where the setup...exe Cygwin installer utility is stored).
  6. Use Internet Explorer Proxy Settings (recommended).
  7. Select a download mirror site.
  8. In the package selection dialog you can select different views to find the required packages easier and you can search the packages via the search field. The Cygwin installer will automatically select the packages which the manually selected ones are depending on. Do not deselect any automatically selected package! There are three hierarchical levels of minimally required packages, depending on Your task:
     a. test execution only (from command line or from Eclipse Titan Executor):
        Base: <All packages> (Default setting of the installer)
        Net: openssl
        Tcl: expect
     b. Test case development: in addition to the above select the following packages:
        Devel: binutils
        Devel: gcc-g++
        Devel: make
        Libs: libxml2-devel
        Net: openssl-devel (automatically installs Net:openssl as well, if selected
     c. To compile your own Titan Cygwin binary: in addition to the above, select the following packages:
        Devel: bison
        Devel: ctags (optional)
        Devel: diffstat
        Devel: flex

Devel: gcc-core
Devel: perl
Devel: git
Editors: <any editor of your preference> e.g vi, nedit, xemacs, gedit, nano and so on
Libs: libncurses-devel
Libs: libreadline-devel
Libs: libexpat1
Libs: libiconv, libiconv-devel, libiconv2

d. To contribute to Titan, test port or protocol module development: Devel: git-review If, after selecting the required packages and clicking on the "Next" button, a "Resolving Dependencies" window lists further required packages, ensure that the "Select required packages (RECOMMENDED)" checkbox is checked and click on the "Next" button.

9. Select the `Create` icon on the Desktop checkbox

10. Optional
Your "unix" home directory, by default is: `<your cygwin installation directory>/home/<yourUserId>`.
If you are (also) working in command line mode, it is a good practice to change this to the folder where your TTCN-3 projects are located.
In older `cygwins`:
Edit the file `<your cygwin installation directory>/etc/passw`:
In the line: `<ourUserId>:unused:<xxxxxx>:<yyyyy>:U-<yourDomain><yourUserId>, S-1-5-21-nnnnnn···nnnnnn:/home/<yourUserId>:/bin/bash`
replace `/home/<yourUserId>` with the folder of your preference.

Starting with Cygwin 1.7.34 or later, set `db_home` in file `/etc/nsswitch.conf`.
For example set:
`db_home: /cygdrive/c/Users/<yourUserId>`.

| | |
|---|---|
| **NOTE** | You can access all Windows drives from Cygwin as `/cygdrive/<windowsDriveLetter>`. Example: to set your "unix" home directory to the `My_Home` folder within your Windows Documents folder, you should replace `/home/<yourUserId>` by `/cygdrive/c/Users/<yourUserId>/Documents/My_Home`. |

| | |
|---|---|
| **WARNING** | The path of your "unix" home directory shall not contain any space! It is not a requirement, but is a kind of best practice to place Titan into a subfolder within your "unix" home directory. |

1. When installation is finished, add the
`$CYGWIN_INSTALL_DIRECTORY\bin` and
`$CYGWIN_INSTALL_DIRECTORY\usr\bin` directories to the `PATH`
environment variable of Windows, so Eclipse will access the shell commands.
For example, if the cygwin root is `C:\cygwin64` then `Path` should contain `C:\cygwin64\bin;C:\cygwin64\usr\bin`.

2. To check if your installation is correct, open either a Cygwin shell (use the desktop icon created during Cygwin installation or start `bash.exe` from the Windows `Start` menu) or start `cmd.exe`

from the Windows Start menu and type: `bash.exe`.

# Chapter 2. Installing only for Ericsson Users Working in E2C with AFS Service

1. In CLI or in your batch file add the required module containing the full pre-installed titan version, for example:
   `module add ttcn/7.1-pl0`
2. Set the environment variables according to the next chapter, except setting `$TTCN3_DIR`. It has been set by the previous command.

# Chapter 3. Installing from a pre-built binary package

This chapter describes obtaining the software and installing it.

## 3.1. Downloading the Software

The Titan package can be installed from the provided download sites.

Download the Titan package for your platform, OS and GCC version from the provided download sites:

- For Ericsson users only: http://ttcn.ericsson.se/download. The usage of this version is conditioned by the presence of a license file and supported by the Titan support team.
- For users outside Ericsson: https://projects.eclipse.org/projects/tools.titan/downloads. This version is licensed under the Eclipse Public License.

A binary distribution, suitable for the used operating system (Solaris, Linux, FreeBSD), and for a C++ compiler, in a tar-gzip archive will be received. For Windows[1] users there is no pre-built version, but compiling the open-source version is possible.

| | |
|---|---|
| **WARNING** | the version of C++ compiler used is important. If the version difference between the system's compiler and the compiler that the basic TTCN–3 library was built with is large enough, the linking of executable test suites will fail with strange error messages. The reason is the different mapping of C++ class and (polymorphic) member function names into linker symbols. For example, this problem persists between versions 2.8.x and 2.95.x of GCC. Different C++ compilers (e.g. Sun Workshop and GCC) are, of course, totally incompatible. The solution for this problem is to use nearly the same version of the C++ compiler as the binary package was built with. |

Binaries for other operating systems or C++ compilers are available only on request.

## 3.2. Installing the Package

No administrator (root) privileges are required for installation, but the install directory must be readable for all users of the test executor. Perform the following steps to install TITAN:

1. Create an empty directory, for example, `/usr/local/TTCN3` or `/home/<UserId>/TTCN3`. This directory will be referred as `$TTCN3_DIR` in the further sections of this document.
2. Copy the `.tgz` file into this directory.
3. Unpack all files from the archive using any of the following commands (assuming GNU tar):
   `tar xvzf ttcn3-<version>-<platform>-<compiler>.tgz`
   or
   `gzip -dc ttcn3-<version>-<platform>-<compiler>.tgz | tar -xvf-`

The following sub-directories are created:

- `bin` contains the executable programs: The Compiler, the Makefile Generator, the Main Controller for parallel test execution and two log formatter utilities.
- `etc` contains a demo license key, which enables to use the parser parts of the Compiler by any user on any host, that is, without C++ code generation. The installation can be tested with this demo key until the personalized license key is received.
- `include` contains the C+/+ header files needed to compile the generated C++ code.
- `lib` contains the pre-compiled Base Library for use with the generated C++ code both for single and parallel mode in static and dynamic linking[2] formats.
- `man` contains UNIX manual pages (for the Compiler and the Makefile Generator).
- `demo` contains a simple TTCN–3 test suite ("Hello, world!") together with a sample test port and a compiled executable.
- `doc` contains this documentation in PostScript and PDF formats.

To complete the TITAN TTCN–3 Test Executor installation, some environmental variables should be set and the login script should be modified.

| | |
|---|---|
| **NOTE** | The C++ source code generated by this version (patch level) of Compiler is not compatible with older versions of TTCN–3 Base Library and vice versa.[3] If upgrading TITAN from an older version, all modules of existing test suites must be re-translated with the new compiler in order to make them running with the new libraries. |

It is recommended to make a backup copy of the older version of the distribution. There are some minor incompatibilities in the compiler's grammar that may cause many syntax errors in TTCN–3 modules that were translated correctly with earlier versions.

## 3.3. Install TITAN with Clang

Currently it is experimental to use TITAN with clang on Ubuntu operating system. It is tested only on Ubuntu. In order to use TITAN with clang on Ubuntu some steps must be done:

1. Install **clang-3.8** (3.8 is the required version) or **clang** version 6.0.0 on Ubuntu 18.04. For clang 6.0.0, the suffix `-3.8` shall be omitted in the steps listed below.
2. Go into your TITAN installation directory and open (or create) the Makefile.personal file and add the following lines:
   **CXX := clang++-3.8**
   **CC := clang-3.8**
3. If TITAN is already compiled run **make distclean** command
4. To compile TITAN with clang run **make** and **make install** commands.

There are some important notes about using clang with TITAN:

- The C++ source code generated and TITAN must be compiled with the same version of clang. See

section 2.2 note.

- Makefiles of TTCN-3 projects must be modified by hand(replace **CXX = g* with *CXX = clang3.8**). Or regenerated using **makefilegen**, to use clang compiler. TITAN's **makefilegen** can detect if it was compiled with clang and will generate makefiles with clang as default C++ compiler.
- Required clang version is **3.8**.

| **NOTE** | On Ubuntu 18.04 the default clang version is 6.0.0. |
|---|---|

[1] For using TITAN on Windows platforms, installing the Cygwin programming environment is required see chapter 1.5 Installing Prerequisites on Cygwin (on Windows)

[2] Note that not all platforms support dynamic linking.

[3] Sometimes even the linking fails; but a successful linking does not mean that everything is correct at all.

# Chapter 4. Building Titan from source code

## 4.1. Obtaining the source code to your local machine

The name of the source code repository of Titan is titan.core in Eclipse GitLab. Follow steps as follows.

1. First time execute these commands:
   ```
   cd ~/git
   git clone https://gitlab.eclipse.org/eclipse/titan/titan.core.git
   ```
   This way a folder "titan.core", the "titan repository" will be created with the TITAN source code and build system.
   To update the already existing repository execute these commands:
   ```
   cd ~/git/titan.core
   git pull https://gitlab.eclipse.org/eclipse/titan/titan.core.git
   ```
2. Follow the instructions in the file “titan.core/README.<your platform>”
3. Continue with the next paragraph of this document.

# Chapter 5. Setting the User Environment

This chapter describes the environment variables that must be set, and the modification of the user login scripts.

## 5.1. Environment Variables

The following environment variables should be set:

- With system administrator privileges, set the `$TTCN3_DIR` environment variable in the common `/etc/profile` and add the `$TTCN3_DIR/bin` directory to the system paths.
- All tools of TITAN, including the Executable Test Suites, require a shared library of OpenSSL (`libcrypto.so`) for execution. To avoid incompatibilities, the suitable shared object file is provided in `$TTCN3_DIR/lib`, so add `$TTCN3_DIR/lib` to the `LD_LIBRARY_PATH` environment variable.

  **WARNING** If this step is not performed, the compiler will not start!

- Add `$TTCN3_DIR/man` to the `$MANPATH` environment variable to reach the manual pages directly.
- If there is no valid license key, refer to Licensing. If upgrading from an older version with a license key valid for this version, skip this step.
- To run TITAN, ensure that the `$TTCN3_DIR` environmental variable has been set, for example, assuming a tcsh as login shell: `setenv TTCN3_DIR /usr/local/TTCNv3`
- To use the TTCN–3 keyword help feature in the GUI with a web browser other than the default Netscape, it is necessary to set the `$TTCN3_BROWSER` `environmental variable, for example, to specify Opera, type the following at the C-shell: `setenv $TTCN3_BROWSER opera`

After setting the environmental variables, the TITAN TTCN–3 Test Executor installation is complete.

## 5.2. Modification of the User Login Script

The following examples provide some help in modifying the login scripts. In case of using AFS for installing TITAN, the first command (setting TTCN3_DIR) must not apply because it already has been set.

**Example modifications of login script** assuming bash as login shell:

```
TTCN3_DIR=/usr/local/TTCNv3 # not for AFS
PATH=$TTCN3_DIR/bin:$PATH
LD_LIBRARY_PATH=$TTCN3_DIR/lib:$LD_LIBRARY_PATH
MANPATH=$MANPATH:$TTCN3_DIR/man
TTCN3_LICENSE_FILE=/home/tmpusr/license.dat
export TTCN3_DIR PATH LD_LIBRARY_PATH MANPATH TTCN3_LICENSE_FILE
```

**Example modifications of login script** assuming tcsh as login shell:

```
setenv TTCN3_DIR /usr/local/TTCNv3 # not for AFS
setenv PATH ${TTCN3_DIR}/bin:${PATH}
setenv LD_LIBRARY_PATH ${TTCN3_DIR}/lib:${LD_LIBRARY_PATH}
setenv MANPATH ${MANPATH}:${TTCN3_DIR}/man
setenv TTCN3_LICENSE_FILE /home/tmpusr/license.dat
```

## 5.3. Modifying Makefile Library

Make sure that the Makefile contains the following highlighted part:

```
SOLARIS8_LIBS = -lxnet -lxml2 -lresolv -lnsl -lsocket
LINUX_LIBS = -lxml2 -lpthread -lrt
```

# Chapter 6. Licensing (Only for Ericsson users)

This chapter describes how to obtain and install a TITAN license key.

From version 1.1.pl8, TITAN can be used only with a valid license key.

## 6.1. Obtaining License Key (Only for Ericsson users)

The license keys are **free of charge** and can be ordered via an HTML form on the following URL: Request a Titan licence at:

http://ttcn.ericsson.se/license/

The personalized license key is a simple ASCII text file, which is sent as an e-mail attachment.

Example of license file:

```
—BEGIN TTCN-3 LICENSE FILE—
AAAAAUrhbm9zIFpvbHThbiBTemFi8wAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAFN6YWJvLkphbm9zQGV0aC5lcmljc3Nvbi5zZQAAAAAAAAAAAAAAAAAAAAAA
AAAAAENvbmZvcm1hbmNlIExhYiwgRXJpY3Nzb24gSHVuZ2FyeSBMdGQuAAAAAAAA
AAAAAEVUSC9STC9TAAAAAAAAAAA7ygrgPayP34CzP9B0bXBqc3oAAAAAAAEAAAAB
AAAAAAAAAAEAAABjAAAAYwAAAIEAAAAAAAAADAsAhRmeNSqfy5/3iEHFsBi1miR
+imw2AIUdRN/V3m6gDQzVeMS+wFUl3UEeKgAAA==
—END TTCN-3 LICENSE FILE—
```

The license key contains the following information encoded in PEM format of OpenSSL library:

A unique identifier (integer number). If the license needs to be renewed or there are problems with licensing, refer to this `Unique ID`.

- Personal data: user's name, e-mail address, company's name and department.
- The time interval of the license key validity.
- The host ID of the computer where the license is valid on (optional).
- The login name that is allowed to use the tool with this key (optional).
- The type of limitation, that is, host ID, login name or both.
- The version interval of the Test Executor that the license key is valid for.
- The list of features that are enabled by this key (in a bitmask).
- DSA digital signature[4], which is calculated on all information fields to protect data integrity and make it impossible to modify license information by the user.

## 6.2. Installing the License Key

Perform the following steps to install the license key:

- Save the license key somewhere in the user home directory. The recommended name for it is `license.dat`, but it can be named alternatively
- It is advised to change its permissions to read-only in order to avoid accidental modification or erasing.
- Set the `TTCN3_LICENSE_FILE` environment variable to point to the license file with full path name. Add this command to the login script to do this step automatically for each login.
- Check the validity of the license by issuing `$TTCN3_DIR/bin/compiler -v`. The compiler will print its version and the information contained in the license file. Also it checks the validity of the license key. Example printout:

```
$ compiler -v
TTCN-3 and ASN.1 Compiler for the TTCN-3 Test Executor
Version: 8.0.0
Build date: May 27 2021 13:49:06
Compiled with: GCC 10.2.0
Using OpenSSL 1.1.1f  31 Mar 2020

Copyright (c) 2000-2026 Ericsson Telecom AB

License information:
---------------------------------------------------------------
License file : /cygdrive/c/Users/ethbaat/license_98.dat
Unique ID    : 98
Licensee     : Jeno Balasko
E-mail       : jeno.balasko@ericsson.com
Company      : Ericsson Hungary
Department   : ETH/
Valid from   : Fri Sep 20 00:00:00 2002
Valid until  : Thu Nov  4 23:59:59 2021
Limitation   : USER
Host ID      : 00000000
Login name   : ethbaat
Versions     : from 1.1.pl0 until 1.99.pl99
Languages    : TTCN3 ASN1
Encoders     : RAW TEXT BER PER XER
Applications : CODEGEN TPGEN SINGLE MCTR HC LOGFORMAT
Max PTCs     : 10000
---------------------------------------------------------------
The license key is valid. +
```

The last line of the printout indicates the success or the problems with the license key.

If a host-limited key is needed, perform it in the same way but do it as system administrator. Copy it into a common directory, for example `$TTCN3_DIR/etc`, and set `TTCN3_LICENSE_FILE` in the common

login script of all users, for example, in `/etc/profile`.

[4] The public key required to check the DSA signature is compiled into all tools and libraries.

# Chapter 7. References


- [1] User Guide for TITAN TTCN-3 Test Executor
- [2] Programmers Technical Reference for TITAN TTCN-3 Test Executor