www.phoronix-test-suite.com

Phoronix Test Suite v10.4.0 (Ibestad)

User Manual

Getting Started

Overview

The Phoronix Test Suite is the most comprehensive testing and benchmarking platform available for Linux, Solaris, macOS, Windows, and BSD operating systems. The Phoronix Test Suite allows for carrying out tests in a fully automated manner from test installation to execution and reporting. All tests are meant to be easily reproducible, easy-to-use, and support fully automated execution. The Phoronix Test Suite is open-source under the GNU GPLv3 license and is developed by Phoronix Media in cooperation with partners. Version 1.0 of the Phoronix Test Suite was publicly released in 2008.

The Phoronix Test Suite client itself is a test framework for providing seamless execution of test profiles and test suites. There are more than 400 tests available by default, which are transparently available via **OpenBenchmarking.org** integration. Of these default test profiles there is a range of sub-systems that can be tested and a range of hardware from mobile devices to desktops and worksrtations/servers. New tests can be easily introduced via the Phoronix Test Suite's extensible test architecture, with test profiles consisting of XML files and shell scripts. Test profiles can produce a quantitative result or other qualitative/abstract results like image quality comparisons and pass/fail. Using Phoronix Test Suite modules, other data can also be automatically collected at run-time such as the system power consumption, disk usage, and other software/hardware sensors. Test suites contain references to test profiles to execute as part of a set or can also reference other test suites. Test suites are defined via an XML schema.

Running the Phoronix Test Suite for the first time can be as simple as issuing a command such as *phoronix-test-suite benchmark c-ray*, which would proceed to install a simple CPU test, execute the test, and report the results. Along with the results, the system's hardware/software information is collected in a detailed manner, relevant system logs, and other important system attributes such as compiler flags and system state. Users can optionally upload their results to OpenBenchmarking.org for sharing results with others, comparing results against other systems, and to carry out further analysis.

OpenBenchmarking.org

OpenBenchmarking.org is an open, collaborative testing platform that makes the Phoronix Test Suite an even more extensible platform for conducting automated tests with complete integration into Phoronix Test Suite test client. OpenBenchmarking.org serves as a repository for storing test profiles, test suites, and result data. Test profiles and suites are stored in the OpenBenchmarking.org cloud to allow for new/updated tests to be seamlessly obtained via the Phoronix Test Suite without needing to manually update the Phoronix Test Suite client. OpenBenchmarking.org also makes it easy to facilitate side-by-side comparisons with any other results stored in the OpenBenchmarking.org cloud. Any Phoronix Test Suite user is permitted to upload their test results, test profiles, and suites to OpenBenchmarking.org.

When finding a set of results on OpenBenchmarking.org (e.g. **an example result file**), it's as easy as running the Phoronix Test Suite with that OpenBenchmarking.org ID to perform an automated side-by-side comparison (e.g. *phoronix-test-suite benchmark 1203160-BY-NVTEGRA3785*).

Thanks to the wealth of test data (results, system logs, etc) from crowd-sourced benchmarking via the

Phoronix Test Suite, a plethora of analytical features are also available from OpenBenchmarking.org.

Phoromatic

Phoromatic is a remote management system for the Phoronix Test Suite that allows the automatic scheduling of tests, remote installation of new tests, and the management of multiple test systems all through an intuitive, easy-to-use web interface. Tests can be scheduled to automatically run on a routine basis across multiple test systems. Phoromatic can also interface with revision control systems to offer support for issuing new tests on a context-basis, such as whenever a Git commit has been pushed or new daily image available. The test results are then available from this central, secure location.

Phoromatic is an add-on to the Phoronix Test Suite that's primarily intended for enterprise users when facilitating tests across a wide-spectrum of hardware within a test lab or when needing to carry out tests on a routine basis.

User Options

The following options are currently supported by the Phoronix Test Suite client. A list of available options can also be found by running *phoronix-test-suite help.*

System

diagnostics

This option will print information that is useful to developers when debugging problems with the Phoronix Test Suite and/or test profiles and test suites.

interactive

A simple text-driven interactive interface to the Phoronix Test Suite.

php-conf

This option will print information that is useful to developers when debugging problems with the Phoronix Test Suite and/or test profiles and test suites.

shell

A simple text-driven shell interface / helper to the Phoronix Test Suite. Ideal for those that may be new to the Phoronix Test Suite

system-info

Display the installed system hardware and software information as detected by the Phoronix Test Suite Phodevi Library.

system-properties

Display various hardware/software system properties detected by the Phoronix Device Interface (Phodevi) library.

system-sensors

Display the installed system hardware and software sensors in real-time as detected by the Phoronix

Test Suite Phodevi Library.

Test Installation

force-install *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will force the installation (or re-installation) of a test or suite. The arguments and process is similar to the install option but even if the test is installed, the entire installation process will automatically be executed. This option is generally used when debugging a test installation problem or wishing to re-install test(s) due to compiler or other environmental changes.

install *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will install the selected test(s) inside the testing environment directory. The install process from downloading of the test files to the installation is fully automated. The install option needs to be supplied with the test name or suite as an argument. Optionally, a OpenBenchmarking.org ID or the name of a saved results file can be supplied as well and the test(s) to install will automatically be extracted from that information. If the test is already installed and was run by the latest version of the installation process, no action will be taken. Multiple arguments can be supplied to install additional tests at the same time.

install-dependencies *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will install the external dependencies needed by the selected test(s) using the distribution's package management system. For example, some tests depend upon GCC for compiling code. If GCC is not detected on the system, the Phoronix Test Suite will attempt to install GCC using the distribution's package management system. If you are running this command as a local user, you may be prompted for the root password while the process is running. For unsupported distributions, the dependency names will be displayed along with common names for the package. The install-dependencies option needs to be supplied with the test name or suite as an argument. When using the install option, the external dependencies are automatically checked.

make-download-cache

This option will create a download cache for use by the Phoronix Test Suite. The download cache is created of test files already downloaded to the local system. If passing any test/suite names to make-download-cache, the needed files for those test profiles will first be automatically downloaded before creating the cache.

remove-installed-test *[Test]*

This option will permanently remove a installed test by the Phoronix Test Suite.

Testing

benchmark *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will install the selected test(s) (if needed) and will proceed to run the test(s). This option is equivalent to running phoronix-test-suite with the install option followed by the run option. Multiple arguments can be supplied to run additional tests at the same time and save the results into one file.

estimate-install-time *[Test | Suite | OpenBenchmarking ID | Test Result]*

This option will provide estimates for test install/setup time length.

estimate-run-time *[Test | Suite | OpenBenchmarking ID | Test Result]*

This option will provide estimates for test run-time / length.

finish-run *[Test Result]*

This option can be used if a test run had not properly finished running all tests within a saved results file. Using this option when specifying a saved results file where all tests had not completed will attempt to finish testing on the remaining tests where there are missing results.

run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will run the selected test(s).

run-random-tests

This option will query OpenBenchmarking.org to run random benchmarks and result comparisons on the system. This test can be used for simply supplying interesting results from your system onto OpenBenchmarking.org, stressing your system with random workloads, seeding new OpenBenchmarking.org results, etc. Basic options are provided at start-up for tuning the randomness of the testing when running this command.

run-subset *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will run the selected test(s) but prompt the user when passing any test suites or result files what subset / test(s) contained within there to run rather than running all passed tests/suites/results.

run-tests-in-suite *[Suite]*

This option can be used if you wish to run all of the tests found in a supplied suite, but you wish to re-configure each of the test options rather than using the defaults supplied by the suite.

stress-batch-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will run the passed tests/suites in the multi-process stress-testing mode while behaving by the Phoronix Test Suite batch testing characteristics. The stress-batch-run mode is similar to the stress-run command except that for any tests passed to it will run all combinations of the options rather than prompting the user for the values to be selected.

stress-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will run the passed tests/suites in the multi-process stress-testing mode. The stress-run mode will not produce a result file but is rather intended for running multiple test profiles concurrently to stress / burn-in the system. The number of tests to run concurrently can be toggled via the PTS_CONCURRENT_TEST_RUNS environment variable and by default is set to a value of 2.

strict-benchmark *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option is equivalent to the `benchmark` option except it enables various options to run benchmarks an extended number of times for ensuring better statistical accuracy if enforcing strict controls over the data quality, in some cases running the benchmarks for 20+ times.

strict-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option is equivalent to the `run` option except it enables various options to run benchmarks an extended number of times for ensuring better statistical accuracy if enforcing strict controls over the data quality, in some cases running the benchmarks for 20+ times.

Batch Testing

batch-benchmark *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option and its arguments are equivalent to the benchmark option, but the process will be run in the Phoronix Test Suite batch mode.

batch-install *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

If you wish to run the install process in the Phoronix Test Suite batch mode but do not wish to run any tests at this time. Running the install process in the batch mode will use the default values and not prompt the user of any possible options, to ensure the process is fully automated.

batch-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option and its arguments are equivalent to the run option, but the process will be run in the

Phoronix Test Suite batch mode.

batch-setup

This option is used to configure the batch mode options for the Phoronix Test Suite, which is subsequently written to the user configuration file. Among the options are whether to automatically upload the test results to OpenBenchmarking.org and prompting for the saved file name.

default-benchmark *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will install the selected test(s) (if needed) and will proceed to run the test(s) in the defaults mode. This option is equivalent to running phoronix-test-suite with the install option followed by the default-run option.

default-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will run the selected test(s). The name of the test or suite must be supplied or the OpenBenchmarking.org ID or saved local file name. Multiple arguments can be supplied to run additional tests at the same time and save the results in a suite-like fashion. Unlike the normal run option, the default-run will not prompt the user to select from the available test options but will instead use the default options as automatically set by pts-core or the test profile. Use batch-run to automatically test all of the available options.

dry-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option and its arguments pre-set the Phoronix Test Suite batch run mode with enforcing of defaults to not save any results and other behavior intended for a dry/test run. This option is primarily intended for testing/evaluation purposes.

internal-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option and its arguments pre-set the Phoronix Test Suite batch run mode with sane values for carrying out benchmarks in a semi-automated manner and without uploading any of the result data to the public OpenBenchmarking.org.

OpenBenchmarking.org

clone-result *[OpenBenchmarking ID] ...*

This option will download a local copy of a file that was saved to OpenBenchmarking.org, as long as a valid public ID is supplied.

enable-repo

This option is used if wanting to add a new OpenBenchmarking.org account/repository to your system for enabling third-party/unofficial test profiles and test suites.

list-recommended-tests

This option will list recommended test profiles for benchmarking sorted by hardware sub-system. The recommended tests are determined via querying OpenBenchmarking.org and determining the most popular tests for a given environment based upon the number of times a test profile has been downloaded, the number of test results available on OpenBenchmarking.org for a given test profile, the age of the test profile, and other weighted factors.

make-openbenchmarking-cache

This option will attempt to cache the test profile/suite meta-data from OpenBenchmarking.org for all linked repositories. This is useful if you're going to be running the Phoronix Test Suite / Phoromatic behind a firewall or without any Internet connection. Those with unrestricted Internet access or not utilizing a large local deployment of the Phoronix Test Suite / Phoromatic shouldn't need to run this command.

openbenchmarking-changes

This option will list recent changes to test profiles of enabled OpenBenchmarking.org repositories.

openbenchmarking-login

This option is used for controlling your Phoronix Test Suite client options for OpenBechmarking.org and syncing the client to your account.

openbenchmarking-refresh

This option is used for refreshing the stored OpenBenchmarking.org repostory information and other data. The Phoronix Test Suite will automatically refresh this data every three days or when other thresholds are exceeded, but this command can be used to manually refresh/updates the data.

openbenchmarking-repositories

This option will list the OpenBenchmarking.org repositories currently linked to this Phoronix Test Suite client instance.

openbenchmarking-uploads

This option will list any recent test result uploads from the system's IP address to

OpenBenchmarking.org.

recently-added-tests

This option will list the most recently added (newest) test profiles.

upload-result *[Test Result]*

This option is used for uploading a test result to OpenBenchmarking.org.

upload-test-profile

This option can be used for uploading a test profile to your account on OpenBenchmarking.org. By uploading your test profile to OpenBenchmarking.org, others are then able to browse and access this test suite for easy distribution in a seamless manner by other Phoronix Test Suite clients.

upload-test-suite *[Suite]*

This option can be used for uploading a test suite to your account on OpenBenchmarking.org. By uploading your test suite to OpenBenchmarking.org, others are then able to browse and access this test suite for easy distribution.

Information

check-tests *[Test]*

This option will perform a check on one or more test profiles to determine if there have been any vendor changes to the filename, filesize, url location, md5 and sha256 checksums.

info *[Test | Suite | OpenBenchmarking ID | Test Result]*

This option will show details about the supplied test, suite, virtual suite, or result file.

intersect *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will print the test profiles present in all passed result files / test suites. Two or more results/suites must be passed and printed will be all of the common test profiles.

list-all-tests

This option will list all test profiles that are available from the enabled OpenBenchmarking.org

repositories. Unlike the other test listing options, list-all-tests will show deprecated tests, potentially broken tests, or other tests not recommended for all environments. The only check in place is ensuring the test profiles are at least compatible with the operating system in use.

list-available-suites

This option will list all test suites that are available from the enabled OpenBenchmarking.org repositories.

list-available-tests

This option will list all test profiles that are available from the enabled OpenBenchmarking.org repositories where supported on the system and are of a verified state. If the system has no Internet access, it will only list the test profiles where the necesary test assets are available locally on the system or on an available network cache (the same behavior as using the list-cached-tests sub-command), unless using the list-all-tests option to override this behavior.

list-available-virtual-suites

This option will list all available virtual test suites that can be dynamically created based upon the available tests from enabled OpenBenchmarking.org repositories.

list-cached-tests

This option will list all test profiles where any needed test profiles are already cached or available from the local system under test. This is primarily useful if testing offline/behind-the-firewall and other use-cases where wanting to rely only upon local data.

list-installed-dependencies

This option will list all of the packages / external test dependencies that are already installed on the system that the Phoronix Test Suite may potentially depend upon by test profiles.

list-installed-suites

This option will list all suites that are currently installed on the system.

list-installed-tests

This option will list all test profiles that are currently installed on the system.

list-missing-dependencies

This option will list all of the packages / external test dependencies that are missing from the system that the Phoronix Test Suite may potentially need by select test profiles.

list-not-installed-tests

This option will list all test profiles that are supported and available but presently NOT installed on the system.

list-possible-dependencies

This option will list all of the packages / external test dependencies that are are potentially used by the Phoronix Test Suite.

list-saved-results

This option will list all of the saved test results found on the system.

list-test-usage

This option will list various details about installed tests and their usage.

list-unsupported-tests

This option will list all available test profiles that are available from the enabled OpenBenchmarking.org repositories but are NOT SUPPORTED on the given hardware/software platform. This is mainly a debugging option for those looking for test profiles to potentially port to new platforms, etc.

search

This option provides command-line searching abilities for test profiles / test suites / test results. The search query can be passed as a parameter otherwise the user is prompted to input their search query..

test-to-suite-map

This option will list all test profiles and any test suites each test belongs to.

Asset Creation

build-suite

This option will guide the user through the process of generating their own test suite, which they can then run. Optionally, passed as arguments can be the test(s) or suite(s) to add to the suite to be created, instead of being prompted through the process.

create-test-profile

This option can be used for creating a Phoronix Test Suite test profile by answering questions about the test for constructing the test profile XML meta-data and handling other boiler-plate basics for getting started in developing new tests.

debug-benchmark *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option is intended for use by test profile writers and is identical to the run option but will yield more information during the run process that can be used to debug issues with a test profile or to verify the test profile is functioning correctly.

debug-install *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option is intended for use by test profile writers and is identical to the install option but will yield more information during the run process that can be used to debug issues with a test profile installer or to verify the test profile is functioning correctly.

debug-result-parser *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option is intended for use by test profile writers and is used for debugging a result parser. No test execution is done, but there must already be PTS-generated .log files present within the test's installation directory.

debug-test-download-links *[Test | Suite | OpenBenchmarking ID | Test Result]*

This option will check all download links within the specified test profile(s) to ensure there are no broken URLs.

download-test-files *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This will download the selected test file(s) to the Phoronix Test Suite download cache but will not install the tests.

inspect-test-profile *[Test]*

This option can be used for inspecting a Phoronix Test Suite test profile with providing inside details on test profiles for debugging / evaluation / learning purposes.

rebuild-test-suite *[Suite]*

This option will regenerate the local test suite XML file against the OpenBenchmarking.org specification. This can be used to clean up any existing XML syntax / styling issues, etc.

result-file-to-suite *[Test Result]*

This option will guide the user through the process of generating their own test suite, which they can then run, that is based upon an existing test results file.

validate-result-file

This option can be used for validating a Phoronix Test Suite result file as being compliant against the OpenBenchmarking.org specification.

validate-test-profile *[Test]*

This option can be used for validating a Phoronix Test Suite test profile as being compliant against the OpenBenchmarking.org specification.

validate-test-suite *[Suite]*

This option can be used for validating a Phoronix Test Suite test suite as being compliant against the OpenBenchmarking.org specification.

Result Management

analyze-run-times *[Test Result]*

This option will read a saved test results file and print the statistics about how long the testing took to complete.

auto-sort-result-file *[Test Result]*

This option is used if you wish to automatically attempt to sort the results by their result identifier string.

compare-results-to-baseline *[Test Result] [Test Result]*

This option will allows you to specify a result as a baseline (first parameter) and a second result file (second parameter) that will offer some analysis for showing how the second result compares to the first in matching tests.

compare-results-two-way *[Test Result]*

This option will allows you to specify a result file and from there to compare two individual runs within that result file for looking at wins/losses and other metrics in a head-to-head type comparison.

edit-result-file *[Test Result]*

This option is used if you wish to edit the title and description of an existing result file.

extract-from-result-file *[Test Result]*

This option will extract a single set of test results from a saved results file that contains multiple test results that have been merged. The user is the prompted to specify a new result file name and select which result identifier to extract.

keep-results-in-result-file *[Test Result]*

This option is the inverse of the remove-results-from-result-file sub-command. If you wish to remove all results but those listed from a given result file, this option can be used. The user must specify a saved results file and then they will be prompted to provide a string to search for in keeping those results in that given result file but removing all other data.

merge-results *[Test Result] ...*

This option will manually merge multiple sets of test results generated by the Phoronix Test Suite.

remove-result *[Test Result]*

This option will permanently remove the saved file set that is set as the first argument.

remove-result-from-result-file *[Test Result]*

This option is used if there are test results (benchmarks) to be dropped from a given result file. The user must specify a saved results file and then they will be prompted to select the tests/benchmarks to remove.

remove-results-from-result-file *[Test Result]*

This option is used if there are test results (benchmarks) to be dropped from a given result file. The user must specify a saved results file and then they will be prompted to provide a string to search for in removing those results from that given result file.

remove-run-from-result-file *[Test Result]*

This option is used if there is a set of test results you wish to remove/delete from a saved results file. The user must specify a saved results file and then they will be prompted to select the results identifier associated with the results they wish to remove.

rename-identifier-in-result-file *[Test Result]*

This option is used if you wish to change the name of the identifier in a test results file that is shown in the Phoronix Test Suite Results Viewer and the contained graphs.

rename-result-file *[Test Result]*

This option is used if you wish to change the name of the saved name of a result file.

reorder-result-file *[Test Result]*

This option is used if you wish to manually change the order in which test results are shown in the Phoronix Test Suite Results Viewer and the contained graphs. The user must specify a saved results file and then they will be prompted to select the results identifiers one at a time in the order they would like them to be displayed from left to right.

result-file-raw-to-csv *[Test Result]*

This option will read a saved test results file and output the raw result file run data to a CSV file. This raw (individual) result file output is intended for data analytic purposes where the result-file-to-csv is more end-user-ready.

result-file-to-csv *[Test Result]*

This option will read a saved test results file and output the system hardware and software information along with the results to a CSV output. The CSV (Comma Separated Values) output can then be loaded into a spreadsheet for easy viewing.

result-file-to-json *[Test Result]*

This option will read a saved test results file and output the basic result information to JSON (JavaScript Object Notation).

result-file-to-pdf *[Test Result]*

This option will read a saved test results file and output the system hardware and software information along with the results to a PDF file.

result-file-to-text *[Test Result]*

This option will read a saved test results file and output the system hardware and software information to the terminal. The test results are also outputted.

show-result *[Test Result]*

Open up the test results in the Phoronix Test Suite Result Viewer or on OpenBenchmarking.org.

workload-topology *[Test Result]*

This option will read a saved test results file and print the test profiles contained within and their arrangement within different test suites for getting an idea as to the workload topology/make-up / logical groupings of the benchmarks being run.

Other

commands

This option will display a short list of possible Phoronix Test Suite commands.

debug-dependency-handler

This option is used for testing the distribution-specific dependency handler for external dependencies.

debug-render-test

This option is used during the development of the Phoronix Test Suite software for testing of the result and graph rendering code-paths This option will download a large number of reference test results from LinuxBenchmarking.com.

debug-self-test

This option is used during the development of the Phoronix Test Suite software for testing of internal interfaces, commands, and other common code-paths. The produced numbers should only be comparable for the same version of the Phoronix Test Suite, on the same hardware/software system, conducted on the same day of testing. This isn't intended as any scientific benchmark but simply to stress common PHP code-paths and looking for hot areas to optimize, etc.

help

This option will display a list of available Phoronix Test Suite commands and possible parameter types.

version

This option will display the Phoronix Test Suite client version.

Modules

auto-load-module

This option can be used for easily adding a module to the AutoLoadModules list in the Phoronix Test Suite user configuration file. That list controls what PTS modules are automatically loaded on start-up of the Phoronix Test Suite.

list-modules

This option will list all of the available Phoronix Test Suite modules on this system.

module-info *[Phoronix Test Suite Module]*

This option will show detailed information on a Phoronix Test Suite module such as the version, developer, and a description of its purpose.

module-setup *[Phoronix Test Suite Module]*

This option will allow you to configure all available end-user options for a Phoronix Test Suite module. These options are then stored within the user's configuration file. Not all modules may have options that can be configured by the end-user.

test-module *[Phoronix Test Suite Module]*

This option can be used for debugging a Phoronix Test Suite module.

unload-module

This option can be used for easily removing a module from the AutoLoadModules list in the Phoronix Test Suite user configuration file. That list controls what modules are automatically loaded on start-up of the Phoronix Test Suite.

User Configuration

enterprise-setup

This option can be run by enterprise users immediately after package installation or as part of an in-house setup script. Running this command will ensure the phoronix-test-suite program is never interrupted on new runs to accept user agreement changes and defaults the anonymous usage reporting to being disabled and other conservative defaults.

network-info

This option will print information detected by the Phoronix Test Suite around the system's network configuration.

network-setup

This option allows the user to configure how the Phoronix Test Suite connects to OpenBenchmarking.org and other web-services. Connecting through an HTTP proxy can be configured through this option.

user-config-reset

This option can be used for resetting the Phoronix Test Suite user configuration file to its default state.

user-config-set

This option can be used for setting an XML value in the Phoronix Test Suite user configuration file.

Result Analysis

executive-summary *[Test Result]*

This option will attempt to auto-generate a textual executive summary for a result file to highlight prominent results / averages.

result-file-confidence *[Test Result]*

This option will read a saved test results file and display various statistics on the confidence of the results with the standard deviation, three-sigma values, and other metrics while color-coding "passing" results in green.

result-file-stats *[Test Result]*

This option is used if you wish to analyze a result file by seeing various statistics on the result data for result files containing at least two sets of data.

wins-and-losses *[Test Result]*

This option is used if you wish to analyze a result file to see which runs produced the most wins/losses of those result identifiers in the saved file.

Phoromatic

start-phoromatic-server

Start the Phoromatic web server for controlling local Phoronix Test Suite client systems to facilitate automated and repeated test orchestration and other automated features targeted at the enterprise.

Result Viewer

start-result-viewer

Start the web-based result viewer.

Module Options

The following list is the modules included with the Phoronix Test Suite that are intended to extend the functionality of pts-core. Some of these options have commands that can be run directly in a similiar manner to the other Phoronix Test Suite user commands. Some modules are just meant to be loaded directly by adding the module name to the AutoLoadModules tag in ~/.phoronix-test-suite/user-config.xml or via the PTS_MODULES environment variable. A list of available modules is also available by running *phoronix-test-suite list-modules.*

Backup Creation + Restore

This is a module for creating backups of the Phoronix Test Suite / Phoromatic and allows for restoring of created backups. The backup will be in ZIP or TAR format. If only a path is specified, the file-name will be auto-generated with a current time-stamp.

phoronix-test-suite backup.create

phoronix-test-suite backup.restore

Dummy Module

This is a simple module intended for developers to just demonstrate some of the module functions.

phoronix-test-suite dummy_module.dummy-command

This is a simple module intended for developers to just demonstrate some of the module functions.

Generate Perf FlameGraphs For Tests

Setting FLAME_GRAPH_PATH=<path to flamegraph path> will auto-load and enable this Phoronix Test Suite module. The module will generate a Linux perf FlameGraph for each test run during the benchmarking process. Details on FlameGraph @ https://github.com/brendangregg/FlameGraph

This module utilizes the following environmental variables: FLAME_GRAPH_PATH.

Flush Caches

Loading this module will ensure caches (page cache, swap, etc) automatically get flushed prior to running any test.

This module utilizes the following environmental variables: PTS_FLUSH_CACHES.

Graphics Override

This module allows you to override some graphics rendering settings for the ATI and NVIDIA drivers while running the Phoronix Test Suite.

This module utilizes the following environmental variables: FORCE_AA, FORCE_AF.

Result Exporter To HTML

This module allows basic exporting of results to HTML for saving either to a file locally (specified using the EXPORT_RESULTS_HTML_FILE_TO environment variable) or to a mail account (specified using the EXPORT_RESULTS_HTML_EMAIL_TO environment variable). EXPORT_RESULTS_HTML_EMAIL_TO supports multiple email addresses delimited by a comma.

This module utilizes the following environmental variables: EXPORT_RESULTS_HTML_EMAIL_TO, EXPORT_RESULTS_HTML_FILE_TO.

Linux Perf Framework Reporter

Setting LINUX_PERF=1 will auto-load and enable this Phoronix Test Suite module. The module also depends upon running a modern Linux kernel (supporting perf) and that the perf binary is available via standard system paths.

This module utilizes the following environmental variables: LINUX_PERF.

Dynamic Result Viewer

This module pre-loads the HTTP dynamic result viewer for Phoronix Test Suite data.

phoronix-test-suite load_dynamic_result_viewer.start

Log Exporter

This module allows for easily exporting test run logs and system logs to external locations via specifying the directory paths via the COPY_TEST_RUN_LOGS_TO and COPY_SYSTEM_LOGS_TO environment variables.

This module utilizes the following environmental variables: COPY_TEST_RUN_LOGS_TO, COPY_SYSTEM_LOGS_TO.

MATISK

My Automated Test Infrastructure Setup Kit

phoronix-test-suite matisk.run

phoronix-test-suite matisk.template

OpenBenchmarking.org Auto Comparison

Phoronix Test Suite v10.4.0

Test Client Documentation

This module prints comparable OpenBenchmarking.org results in the command-line for reference purposes as tests are being run. OpenBenchmarking.org is automatically queried for results to show based on the test comparison hash and the system type (mobile, desktop, server, cloud, workstation, etc). No other system information or result data is transmitted.

phoronix-test-suite ob_auto_compare.debug

Performance Per Dollar/Cost Calculator

Setting the COST_PERF_PER_DOLLAR= environment variable to whatever value of the system cost/component you are running a comparison on will yield extra graphs that calculate the performance-per-dollar based on the test being run. The COST_PERF_PER_DOLLAR environment variable is applied just to the current test run identifier. Set the COST_PERF_PER_UNIT= environment variable if wishing to use a metric besides dollar/cost. The COST_PERF_PER_HOUR value can be used rather than COST_PERF_PER_DOLLAR if wishing to calculate the e.g. cloud time or other compute time based on an hourly basis.

phoronix-test-suite perf_per_dollar.add

This module utilizes the following environmental variables: COST_PERF_PER_DOLLAR, COST_PERF_PER_UNIT, COST_PERF_PER_HOUR.

Performance Tip Prompts

This module alerts the user if the system configuration may not be the right one for achieving the best performance with the target benchmark(s). This initial version of the module actually cares only about the BFQ I/O scheduler and powersave governor checks.

phoronix-test-suite perf_tips.show

This module utilizes the following environmental variables: SUPPRESS_PERF_TIPS.

This module alerts the user if the system configuration may not be the right one for achieving the best performance with the target benchmark(s). This initial version of the module actually cares only about the BFQ I/O scheduler: it gives a warning if BFQ is being used with an incorrect configuration in a disk benchmark, and suggests the right configuration to use. For the moment it only works for existing, throughput-based tests. It will need to be extended for responsiveness and soft real-time-latency tests.

Benchmarking Compiler PGO Impact

This module makes it easy to test a compiler PGO (Profile Guided Optimization) performance impact by running a test without PGO optimizations, capturing the PGO profile, rebuilding the tests with the PGO profile generated, and then repeat the benchmarks.

phoronix-test-suite pgo.benchmark

Phoronix Test Suite v10.4.0

Test Client Documentation

Phoromatic Client

The Phoromatic client is used for connecting to a Phoromatic server (Phoromatic.com or a locally run server) to facilitate the automatic running of tests, generally across multiple test nodes in a routine manner. For more details visit http://www.phoromatic.com/. This module is intended to be used with Phoronix Test Suite 5.2+ clients and servers.

phoronix-test-suite phoromatic.connect

phoronix-test-suite phoromatic.explore

phoronix-test-suite phoromatic.upload-result

phoronix-test-suite phoromatic.set-root-admin-password

phoronix-test-suite phoromatic.list-results

phoronix-test-suite phoromatic.clone

phoronix-test-suite phoromatic.export-results-for-account-schedules

The Phoromatic module contains the client support for interacting with Phoromatic and Phoromatic Tracker services.

Pushover.net

Submit notifications to your iOS/Android mobile devices of test results in real-time as push notifications, etc. Using the Pushover.net API.

This module utilizes the following environmental variables: PUSHOVER_NET_USER.

Report Test Time Graphs

Setting the RUN_TIMES_ARE_A_BENCHMARK=1 environment variable will automatically create additional graphs for each test run plotting the run-time needed for each test being executed. Setting the INSTALL_TIMES_ARE_A_BENCHMARK=1 environment variable will automatically create additional graphs for each test run plotting the time required for the test installation. Setting the INSTALL_SIZES_ARE_A_BENCHMARK=1 environment variable will automatically create additional graphs for each test run plotting the size of the installed test directory.

This module utilizes the following environmental variables: RUN_TIMES_ARE_A_BENCHMARK, INSTALL_TIMES_ARE_A_BENCHMARK, INSTALL_SIZES_ARE_A_BENCHMARK.

Result Notifier

Phoronix Test Suite v10.4.0

Test Client Documentation

A notification module.

Custom Result Export Methods

A simple example module about interfacing with Phoronix Test Suite core for dumping result files in a custom format.

phoronix-test-suite results_custom_export.nf

System Monitor

This module contains sensor monitoring support.

This module utilizes the following environmental variables: MONITOR, PERFORMANCE_PER_WATT, PERFORMANCE_PER_SENSOR, MONITOR_INTERVAL, MONITOR_PER_RUN.

Monitoring these sensors is as easy as running your normal Phoronix Test Suite commands but at the beginning of the command add: MONITOR=<selected sensors>. For example, this will monitor the CPU temperature and voltage during tests:

MONITOR=cpu.temp,cpu.voltage phoronix-test-suite benchmark universe

For some of the sensors there is an ability to monitor specific device, e.g. cpu.usage.cpu0 or hdd.read-speed.sda. If the PERFORMANCE_PER_WATT environment variable is set, a performance per Watt graph will also be added, assuming the system's power consumption can be monitored. PERFORMANCE_PER_SENSOR= will allow similar behavior but for arbitrary sensors. Below are all of the sensors supported by this version of the Phoronix Test Suite.

Supported Options:

- all
- all.ambient
- ambient.temp
- all.cgroup
- cgroup.cpu-usage
- all.cpu
- cpu.fan-speed
- cpu.freq
- all.cpu.freq
- cpu.freq.cpu0
- cpu.freq.cpu1
- cpu.freq.cpu2
- cpu.freq.cpu3
- cpu.freq.cpu4
- cpu.freq.cpu5

Phoronix Test Suite v10.4.0

Test Client Documentation

- cpu.freq.cpu6
- cpu.freq.cpu7
- cpu.peak-freq
- cpu.power
- cpu.temp
- cpu.usage
- all.cpu.usage
- cpu.usage.cpu0
- cpu.usage.cpu1
- cpu.usage.cpu2
- cpu.usage.cpu3
- cpu.usage.cpu4
- cpu.usage.cpu5
- cpu.usage.cpu6
- cpu.usage.cpu7
- cpu.usage.summary
- cpu.voltage
- all.gpu
- gpu.fan-speed
- gpu.freq
- gpu.memory-usage
- gpu.power
- gpu.temp
- gpu.usage
- gpu.voltage
- all.hdd
- hdd.read-speed
- all.hdd.read-speed
- hdd.read-speed.sda
- hdd.read-speed.sdb
- hdd.read-speed.nvme0n1
- hdd.temp
- all.hdd.temp
- hdd.temp.sda
- hdd.temp.sdb
- hdd.temp.nvme0n1
- hdd.write-speed
- all.hdd.write-speed
- hdd.write-speed.sda
- hdd.write-speed.sdb
- hdd.write-speed.nvme0n1
- all.memory
- memory.temp
- memory.usage
- all.swap

- swap.usage
- all.sys
- sys.fan-speed
- sys.iowait
- sys.power
- sys.temp
- sys.voltage
- all.sys.voltage

NOTE: Use the "system-sensors" command to see what sensors are available for monitoring on the system.

Test Timeout

This module allows killing a test if it exceeds a defined threshold, such as if the test is hung, etc. TEST_TIMEOUT_AFTER= environment variable can be used for controlling the behavior. When this variable is set, the value will can be set to "auto" or a positive integer. The value indicates the number of minutes until a test run should be aborted, such as for a safeguard against hung/deadlocked processes or other issues. Setting this to a high number as a backup would be recommended for fending off possible hangs / stalls in the testing process if the test does not quit on its own for whatever reason. If the value is "auto", it will quit if the time of a test run exceeds 3x the average time it normally takes the particular test to complete its run.

This module utilizes the following environmental variables: TEST_TIMEOUT_AFTER.

Timed Screenshot

This is a module that will take a screenshot of the system at a pre-defined interval. ImageMagick must be installed onto the system prior to using this module.

This module utilizes the following environmental variables: SCREENSHOT_INTERVAL.

Toggle Screensaver

This module toggles the system's screensaver while the Phoronix Test Suite is running. At this time, the GNOME and KDE screensavers are supported.

This module utilizes the following environmental variables: HALT_SCREENSAVER.

Linux Turbostat Dumper

Setting TURBOSTAT_LOG_DIR=_DIR_ will auto-load and enable this Phoronix Test Suite module. The module will -- if turbostat is installed on the system and the user is root -- allow dumping of the TurboStat data to the specified directly on a per-test basis. This allows easily collecting of turbostat logs for each test being run.

Phoronix Test Suite v10.4.0

Test Client Documentation

This module utilizes the following environmental variables: TURBOSTAT_LOG_DIR.

Update Checker

This module checks to see if the Phoronix Test Suite -- and its tests and suites -- are up to date plus also handles message of the day information.

Utilize Wine On Linux Benchmarking

This module when activated via the USE_WINE environment variable on Linux systems will override the test profile OS target to Windows and attempt to run the (Windows) tests under Wine, if installed on the system. USE_WINE can be either set to the name of the desired wine command or the absolute path to the wine binary you wish to use for benchmarking.

This module utilizes the following environmental variables: USE_WINE.

System Event Watchdog

This module has support for stopping/interrupting tests if various system issues occur, like a temperature sensor exceeds a defined threshold.

This module utilizes the following environmental variables: WATCHDOG_SENSOR, WATCHDOG_SENSOR_THRESHOLD.

Phoronix Test Suite v10.4.0

Test Client Documentation

Installation Instructions

Setup Overview

The Phoronix Test Suite supports Linux, Apple macOS, Microsoft Windows, Solaris, Hurd, BSD, and other operating system environments. The only Linux distribution-specific code deals with the external dependencies support feature that are set by individual test profiles. If you are not running one of the supported Linux distributions, Solaris, BSD, or macOS, you may need to install a package manually (as instructed by the Phoronix Test Suite) in order for a test to run. An example of an external dependency would be GCC and the OpenGL Utility Toolkit being needed for test profiles that build an OpenGL benchmark from source-code.

Among the distributions where the Phoronix Test Suite has been officially tested include Ubuntu, Fedora, Mandriva / Mageia, Gentoo, PCLinuxOS, Arch Linux, Pardus, OpenSuSE, Optware, webOS, Zenwalk, CentOS, Red Hat Enterprise Linux, Oracle Linux, Scientific Linux, Debian, Mint, MEPIS, Alpine Linux, Void Linux, Intel Clear Linux, and Amazon Linux EC2.

Among the tested BSD distributions are FreeBSD, PC-BSD, NetBSD, OpenBSD, Debian GNU/kFreeBSD, and DragonflyBSD. Tested Solaris distributions include OpenSolaris, Solaris Express 11, Oracle Solaris 11, OpenIndiana, Illumos, and Nexenta.

Dependencies

The only required dependency for the Phoronix Test Suite is PHP 5.3 or newer. On Linux distributions, the needed package is commonly called *php5-cli* or *php-cli* or *php7* or *php*. It is important to note that only PHP for the command-line is needed and not a web server (Apache) or other packages commonly associated with PHP and its usage by web-sites. The PHP5 version required is PHP 5.3+ and can also be found at **www.php.net**. PHP 7 is fully supported by the Phoronix Test Suite as well as HHVM.

The *phoronix-test-suite.bat* Windows launcher for the Phoronix Test Suite will automatically download and setup PHP on the local system if PHP is not present already.

As part of the PHP requirement, the following PHP extensions are required and/or highly recommended in order to take advantage of the Phoronix Test Suite capabilities:

PHP DOM is needed for XML operations and must be installed for the Phoronix Test Suite to function.

PHP ZIP is needed for file compression and decompression and specifically dealing with test profiles and suites obtained via OpenBenchmarking.org or when uploading such tests and suites.

PHP OpenSSL is used for enabling HTTPS communication with Phoronix Test Suite /

OpenBenchmarking.org servers.

PHP GD is highly recommended for rendering of test data to JPEG and PNG image formats and is selectively used in other image operations.

PHP Zlib is highly recommended for greater data compression when dealing with remote OpenBenchmarking.org assets.

PHP PCNTL is used for multi-threaded system sensor monitoring support during the testing process and other threaded tasks by the Phoronix Test Suite module framework.

PHP POSIX is used for reliably obtaining more system information in an efficient manner.

PHP CURL is supported as an alternative networking library for improved network performance in downloading test files and other operations.

PHP FPDF is used to generate PDF reports of test data.

Without all of these extensions, some capabilities of the Phoronix Test Suite will not be available. Many of these packages are enabled by default and do not require any additional installation steps on most Linux distributions, otherwise they are often found in the package vendor's repository.

Notes

General

You may need to modify the *php.ini* file on your system in order to support uploading results to OpenBenchmarking.org or logging into your OpenBenchmarking.org account. The *allow_url_fopen*, *file_uploads*, and *allow_url_include* options must be set to true in the PHP configuration.

Major updates to the Phoronix Test Suite are released on a quarterly basis. The latest stable and development versions of the Phoronix Test Suite are available at **Phoronix-Test-Suite.com**. The Git repository where the latest Phoronix Test Suite code is provided is hosted at **GitHub.com/phoronix-test-suite** and can be cloned/pulled from the *https://github.com/phoronix-test-suite/phoronix-test-suite.git* repository location. The latest upstream development code is housed in the master tree while older Phoronix Test Suite releases are available in their respective Git branches based upon the release's code-name.

If building the PHP package from upstream sources, it should just be a matter of running *./configure* with the *--enable-zip* flag (all other requirements should be apart of the stock PHP configuration) to satisfy the PHP needs of the Phoronix Test Suite.

File Structure

If manually changing the location of the *phoronix-test-suite* launcher file, the *PTS_USER_PATH* environment variable must be adjusted inside the file to reflect the absolute location that leads to the root directory of the *pts* and *pts-core* folders. The *pts-core* directory contains the "engine" of the Phoronix Test Suite.

Running Locally

The Phoronix Test Suite can be simply extracted from the downloaded *.tar.gz* or *.zip* file or it can also be installed system-wide. If you just wish to run the Phoronix Test Suite without installing it, open a terminal and run *./phoronix-test-suite <options>* from the same directory.

Generic Installation

Running *install-sh* from the root directory of the Phoronix Test Suite will install the software for system-wide access. By default the *phoronix-test-suite* executable is in */usr/bin/*, the Phoronix Test Suite files in */usr/share/phoronix-test-suite/*, and the documentation in */usr/share/doc/phoronix-test-suite/*. Root access is required. The default installation prefix is /usr/ but can be adjusted as the first argument (example: *install-sh /home/user/* to install the Phoronix Test Suite in your home directory).

Debian/Ubuntu Installation

Debian/Ubuntu users are able to follow the Generic Installation instructions or can obtain a Debian Package from the Phoronix Test Suite web-site. The package contains the *phoronix-test-suite* executable in */usr/bin/*, the Phoronix Test Suite files in */usr/share/phoronix-test-suite/*, and the documentation in */usr/share/doc/phoronix-test-suite/*.

Fedora / Red Hat Installation

The Phoronix Test Suite can be installed on Fedora, Red Hat Enterprise Linux, and CentOS systems using the generic installation method. Alternatively, a *phoronix-test-suite* package is available in recent versions of the Fedora repository and in the EPEL (Extra Packages for Enterprise Linux) repository for Red Hat Enterprise Linux. However, at times this package may be out-of-date compared to upstream stable.

BSD Installation

The Phoronix Test Suite also supports *BSD operating systems. However, like the Solaris support, not all test profiles are compatible with BSD operating systems, but should run well on the likes of FreeBSD and DragonFlyBSD.

MacOS Installation

The Phoronix Test Suite is fully supported on Apple's macOS operating system. PHP ships with macOS by default so it's simply a matter of downloading the Phoronix Test Suite package, extracting it, and

running the executable. For tests that rely upon a compiler, Apple's XCode with GCC and LLVM can be utilized.

Phoronix Test Suite v10.4.0

Test Client Documentation

Phoronix Test Suite On Windows

Introduction

Phoronix Test Suite 8.0 features rewritten Windows support that is at a near feature parity to the program's long-standing support for Linux, macOS, BSD and Solaris operating systems. To make it abundantly clear, if you are using a Phoronix Test Suite version pre-8.0, you are best upgrading or ideally using Phoronix Test Suite Git as the Windows support remains in very active development at the moment as of early 2018.

The Phoronix Test Suite Windows support currently targets **Windows 10 x64** and **Windows Server 2016 x64**. Earlier versions of Windows, namely Windows Server 2012 and Windows 8, may work to some extent but some hardware/software reporting features and other capabilities may be missing or report warning messages. The Phoronix Test Suite Windows support is also exclusively focused on x86 64-bit support: the Phoronix Test Suite itself will run on x86 32-bit but many of the program dependencies are configured for making use of 64-bit binaries.

Windows Setup / Dependencies

As with Phoronix Test Suite on Linux and other operating systems, the principal dependency is on PHP (PHP v5.3 or newer, including PHP 7.x). Running the *phoronix-test-suite.bat* file launcher for the Phoronix Test Suite on Windows will attempt to download and setup PHP on the system under *C:\PHP* as the default location should PHP support not be found within your system's *Program Files* directories. The PHP Windows build does depend upon Microsoft Visual C++ redistributable libraries, which the Windows launcher will also attempt to download and install if needed.

The Phoronix Test Suite on Windows does depend upon **Cygwin** for its Bash interpreter and other basic utilities to ease the process of porting test profiles to Windows with being able to use many of the same test installation scripts on Windows/Linux/macOS/BSD/Solaris then largely unmodified. Most of the Windows tests depend upon their respective native Windows applications/binaries while this Cygwin support is a convenience for handling these Bash setup scripts and also some test profiles that depend upon a GNU toolchain. The Phoronix Test Suite will attempt to download and setup Cygwin on the system if Cygwin isn't found in its default location of *C:\cygwin64*.

Various test profiles may depend upon other "external dependencies" like Python, PERL, Steam, and Java, as examples. The Phoronix Test Suite as with its support for other operating systems and Linux distributions will attempt to install these needed dependencies on a per-test basis when needed if existing support is not detected on the system.

Running The Phoronix Test Suite On Windows

The Phoronix Test Suite can run from its local directory and does not need to be "installed" to a system path or any other "setup" process prior to execution. On a clean install of Windows 10 x64 or Windows

Phoronix Test Suite v10.4.0

Test Client Documentation

Server 2016, deploying the Phoronix Test Suite is designed to be as easy and straight-forward as possible:

1. Download the Phoronix Test Suite 8.0+ or **Phoronix-Test-Suite from GitHub** (**zip file**).

2. From the Command Prompt or PowerShell, enter the *phoronix-test-suite* directory whether it be from Git or a zipped download.

3. Run the *phoronix-test-suite.bat* file that should proceed to run the Phoronix Test Suite just as you would on any other operating system. If needed the Phoronix Test Suite will try to initially download and setup PHP if needed followed by the attempted automatic Cygwin setup, etc.

4. Any of the Phoronix Test Suite commands from other operating systems should work on Windows. If you are new to the Phoronix Test Suite, you may enjoy a bit more guided experience by running the **phoronix-test-suite shell** command.

Test Profiles On Windows

As of March 2018, around 50 of the test profiles are currently compatible with the Phoronix Test Suite on Windows. This includes many of the popular benchmarks and other interesting test cases. Over time more test profiles will continue to be ported to Windows where applicable and there are also some Windows-only tests also supported for execution by the Phoronix Test Suite.

Getting Started

Besides **phoronix-test-suite shell** and **phoronix-test-suite help**, there is also **phoronix-test-suite interactive** for helping new users understand Phoronix Test Suite benchmarking. Long story short, it should be as easy as running **phoronix-test-suite benchmark c-ray** or **phoronix-test-suite benchmark crafty** as some examples for carrying out automated, cross-platform benchmarks in a side-by-side and fully-reproducible manner.

Support

Community technical support is available via **GitHub**. For enterprise inquiries, commercial support, and custom engineering services, **contact us**.

Phoronix Test Suite v10.4.0

Test Client Documentation

External Dependencies

The Phoronix Test Suite has a feature known as "External Dependencies" where the Phoronix Test Suite can attempt to automatically install some of the test-specific dependencies on supported distributions. If running on a distribution where there is currently no External Dependencies profile, the needed package name(s) are listed for manual installation.

Below are a list of the operating systems that currently have external dependencies support within the Phoronix Test Suite for the automatic installation of needed test files.

Alpine Linux
Amazon
Angstrom
Arch Linux
Clear Linux
ClearOS
ClearOS Core Server
Debian
DragonFlyBSD
Fedora
Fluxbuntu
GNU KFreeBSD
Gentoo
Goobuntu
HP
Joli Cloud
Linaro
Linux Embedded Development Environment
Linux Mint
MEPIS
Mac OS X
MacPorts
Mageia
Mandriva
MeeGo
Microsoft Windows
MidnightBSD
Moblin
Mythbuntu
NetBSD
Nexenta Core

Phoronix Test Suite v10.4.0

Test Client Documentation

OLPC
OpenIndiana
OpenMandriva
OpenMandrivaLinux
OpenSolaris
OpenSuSE
Optware
Oracle Server
PCLinuxOS
Palm
Pardus Linux
Red Hat Enterprise
Red Hat Enterprise Server
SUSE
SUSE Linux
Scientific
ScientificSL
Solus
Solus Linux
Termux
Ubuntu
Void Linux
Zenwalk
gNewSense
macOS Brew

Phoronix Test Suite v10.4.0

Test Client Documentation

Configuration

User Files & Folders

~/.phoronix-test-suite/user-config.xml

This is a per-user configuration file. Among the information stored here is the test options, locations for storing files, and batch mode options. This file is formatted in XML.

~/.phoronix-test-suite/graph-config.json

This is a per-user configuration file for storing graph attributes. The adjustable options include HTML hex color codes for different areas of the graph, dimensions of the graph, and font sizes. This file is formatted in JSON.

~/.phoronix-test-suite/download-cache/

This directory contains test packages that have been downloaded for test profiles. For more information on the download cache.

~/.phoronix-test-suite/installed-tests/

This directory is where tests are installed by default. Each test has its own directory within a sub-directory of *installed-tests/* based upon its OpenBenchmarking.org repository. In the test's folder is a *pts-install.xml* file used for managing the installation.

~/.phoronix-test-suite/test-results/

This directory is where tests results are saved by default. Each saved file has its own directory. In the saved directory is then a *composite.xml* file containing the useful results while in the *test-X.xml* files are back-ups of the results.

~/.phoronix-test-suite/modules-data/

This is the directory where any Phoronix Test Suite modules should save any files to, within a sub-directory of the module's name. The module configuration settings are also stored within this directory.

~/.phoronix-test-suite/test-profiles/

This is the directory where test profiles are stored.

~/.phoronix-test-suite/test-suites/

Phoronix Test Suite v10.4.0

Test Client Documentation

This is the directory where test suites are stored.

Environment Variables

TEST_TIMEOUT_AFTER

When this variable is set, the value will can be set to *auto* or a positive integer. The value indicates the number of minutes until a test run should be aborted, such as for a safeguard against hung/deadlocked processes or other issues. Setting this to a high number as a backup would be recommended for fending off possible hangs / stalls in the testing process if the test does not quit. If the value is *auto*, it will quit if the time of a test run exceeds 3x the average time it normally takes the particular test to complete its run. In the future, auto might be enabled by default in a future PTS release.

TEST_RESULTS_NAME

When this variable is set, the value will be used as the name for automatically saving the test results.

TEST_RESULTS_IDENTIFIER

When this variable is set, the value will be used as the test identifier when automatically saving the test results.

TEST_RESULTS_DESCRIPTION

When this variable is set, the value will be used as the test results description when saving the test results.

PRESET_OPTIONS

For setting any test option(s) from an environment variable rather than being prompted for the options when running a test. Example: *PRESET_OPTIONS="stream.run-type=Add" ./phoronix-test-suite benchmark stream*. Multiple options can be passed to this environment variable when delimited by a semicolon.

SKIP_TESTS

If there are any test(s) to exempt from the testing process, specify them in this variable. Multiple tests can be waived by delimiting each test identifier by a comma. A test hardware type (i.e. Graphics) can also be supplied for skipping a range of tests.

SKIP_TESTS_HAVING_ARGS

If any of the test(s) have an argument matching any strings contained in this environment variable, the test execution will be skipped. Multiple strings can be set when delimiting by a comma.

Phoronix Test Suite v10.4.0

Test Client Documentation

RUN_TESTS_IN_RANDOM_ORDER

Setting this environment variable will cause the tests to be run in a random order.

SKIP_TESTING_SUBSYSTEMS

If you are running a set of benchmarks (namely a result file) but wish to skip some of the tests that don't belong to a certain test type group, you can set the hardware types to test via this environment variable. E.g. setting *SKIP_TESTING_SUBSYSTEMS=Graphics* will skip all test profiles to run that are not of the graphics test group. Multiple types should be delimited by a comma.

PTS_MODULE_SETUP

This variable can be used to load Phoronix Test Suite module settings automatically when using the *module-setup* option. An example would be:
PTS_MODULE_SETUP="phoromatic.remote_host=http://www.phoromatic.com/; phoromatic.remote_account=123456; phoromatic.remote_verifier=ABCD" phoronix-test-suite module-setup phoromatic.

PTS_MODULES

If there are any Phoronix Test Suite modules to additionally load, they can be specified here. Multiple modules can be supplied by delimiting them with a comma. The more appropriate way of loading Phoronix Test Suite modules for longer periods of time is by using the *~/.phoronix-test-suite/user-config.xml* configuration.

NO_PHODEVI_CACHE

This is a debugging option to disable the Phodevi cache from being loaded of cached software/hardware information. Instead, all software/hardware will be polled from the Phodevi library without caching.

EXTERNAL_PHODEVI_CACHE

This option can be used for loading an external Phodevi cache. Such as loading the native hardware/software information from within a Windows Wine client from a native system host.

PTS_DISPLAY_MODE

If you wish to load a non-default display mode for a single instance, specify the mode in this variable.

TOTAL_LOOP_TIME

When running any test(s), if you would like the test(s) to continue running as a loop until a certain time has been reached, this variable can be used. The value should be the number of minutes to run the

testing process before the loop is ended. The testing will finish whenever the currently active test has finished once the time has elapsed. The minimum value allowed is 10 minutes.

LIMIT_ELAPSED_TEST_TIME

If you want to ensure that the time for a given Phoronix Test Suite process doesn't elapse past a certain number of minutes, specify the number of minutes for this environment variable. When the amount of time spent testing exceeds that amount, the testing will end prematurely while still saving the tests that were completed in time.

TOTAL_LOOP_COUNT

When running any test(s), if you would like the test(s) to continue running for a number of times, this variable can be used. The value should be the number of times to loop the testing process before ending.

FORCE_TIMES_TO_RUN

If you wish to override the number of times to run each test -- rather than the Phoronix Test Suite using the number of times specified in each test profile -- this variable can be used.

FORCE_TIMES_TO_RUN_MULTIPLE

This option allows specifying a multiple for increasing the number of times a test will run based upon the original TimesToRun value specified in the test definition. This allows for increasing the expected times to run based on a multiple of that default rather than a static value.

FORCE_MIN_TIMES_TO_RUN

This is similar to the FORCE_TIMES_TO_RUN option but will only be used if the test profile's run count is less than this defined value.

FORCE_MIN_TIMES_TO_RUN_CUTOFF

When used in conjunction with FORCE_MIN_TIMES_TO_RUN, the override value will only be applied to test profiles where its average run-time length (in minutes) is less than the value specified by FORCE_MIN_TIMES_TO_RUN_CUTOFF.

FORCE_MIN_DURATION_PER_TEST

This is similar to FORCE_MIN_TIMES_TO_RUN but allows specifying a time (in minutes) that each test should be run for. Each test will loop at least until that amount of time has elapsed. This can be useful for short-running tests if wanting to ensure each test is run long enough to rule out system noise.

IGNORE_RUNS

IGNORE_RUNS can be passed a comma-separated list of runs to skip on each benchmark. For example, IGNORE_RUNS=1 would always drop the first run from being recorded.

NO_FILE_HASH_CHECKS

To disable MD5/SHA256 check-sums from being checked when downloading test files, set this variable to 1. This variable used to be known as *NO_MD5_CHECKS*, which is still honored but was changed to *NO_FILE_HASH_CHECKS* to reflect other kind of file hash sum checks.

NO_HTTPS

Set this environment variable to 1 if you don't wish to use HTTPS download links for test profiles (or the system/network lacks HTTPS support). When enabled, HTTPS links will then be done over HTTP.

PTS_DOWNLOAD_CACHE

While non-standard Phoronix Test Suite download caches can be specified within the *user-config.xml* file, an additional directory to look for potential Phoronix Test Suite download files can be specified by this variable.

GRAPH_HIGHLIGHT

If this variable is set with a valid test identifer from a result file whether you are using the *refresh-graphs* command or any other related to the rendering of test results on a bar graph, the specified test identifier's result will be rendered in a different color than the other test results. Multiple identifiers can be specified when delimited by a comma. Additionally, for each key it is possible to provide the actual color value, or an index in the color palette. Example:
"will_be_different,group1a=1,group1b=1,blue=#0000ff"

TEST_EXEC_PREPEND

Set this variable to any command/environment variable that you may be passed prepended to the test execution string at runtime.

VIDEO_MEMORY

If Phodevi fails to detect the system's video memory capacity or is incorrectly detected, the video memory capacity (in MB) can be specified by this variable.

OVERRIDE_VIDEO_MODES

If Phodevi fails to detect all of the system's monitor video modes or a separate set of modes would be preferred, the modes can be specified in this variable. Example:
OVERRIDE_VIDEO_MODES=800x600,1024x768,1280x1024 phoronix-test-suite benchmark nexuiz.

Phoronix Test Suite v10.4.0

Test Client Documentation

SKIP_TEST_SUPPORT_CHECKS

If this environment variable is set, it will not honor the support checks made by individual test profiles. I.e. test profiles that would normally be considered un-supported on a given platform are attempted to install and run regardless.

SKIP_ALL_TEST_SUPPORT_CHECKS

If this environment variable is set, all tests will be permitted on the client for execution. SKIP_ALL_TEST_SUPPORT_CHECKS is more liberal than SKIP_TEST_SUPPORT_CHECKS in letting disk tests run on RAM-based file-systems, attempt to run 2D/3D tests on VESA display drivers, and other special cases.

DEFAULT_VIDEO_MODE

If Phodevi fails to detect the system's monitor standard / default resolution, the mode can be specified in this variable. Example: *DEFAULT_VIDEO_MODE=1680x1050 phoronix-test-suite benchmark nexuiz* .

SKIP_EXTERNAL_DEPENDENCIES

To skip the Phoronix Test Suite external dependency checking/installation when installing a test, set this environment variable to *1*. If wishing to skip only certain external dependencies, set this variable's value to the name of the external dependencies (the generic dependency names used by the Phoronix Test Suite) to not install. Multiple dependencies to skip can be delimited by a comma.

PTS_EXTRA_SYSTEM_LOGS_DIR

By default the Phoronix Test Suite collects common system logs (cpuinfo, lscpu, dmesg) during the benchmarking process when saving test results. If wanting to collect additional, arbitrary system log files specific to your operating environment or for other niche system information, the *PTS_EXTRA_SYSTEM_LOGS_DIR* environment variable can be set as a path to a directory containing such log files. Prior to running the Phoronix Test Suite simply set *PTS_EXTRA_SYSTEM_LOGS_DIR* to the directory where any text files should be captured from following test completion.

Phoronix Test Suite v10.4.0

Test Client Documentation

General Information

Frequently Asked Questions

Q: May I use the Phoronix Test Suite when running benchmarks for my own publication or blog? Are there any publishing restrictions?

A: Anyone is more than welcome to use the Phoronix Test Suite for their own publication or purpose. While the Phoronix Test Suite came out of our internal test tools for carrying out Linux hardware reviews at **Phoronix.com**, we invite other hardware review web-sites, technology journals, and independent publications to use our software too. While not required, we would just kindly ask that you mention in your review/article that the *Phoronix Test Suite* was used for carrying out your testing, and ideally to link to **www.phoronix-test-suite.com** so that your readers will know where to obtain the software if they are interested in running the tests. You are also more than welcome to upload your results to **OpenBenchmarking.org** so that others may compare their results against yours in an easy manner.

We also try to make the Phoronix Test Suite easy-to-use by independent publications. For example, if you would like to watermark your web-site's URL into the graphs containing your test results, that can be easily modified in *~/.phoronix-test-suite/graph-config.json*. The colors and other graph settings are also stored in this XML file. If you are a publication and run into any issues with the Phoronix Test Suite or have a feature request, please let us know.

Q: Why does the Phoronix Test Suite not use my distribution's package management system for acquiring all needed packages?

A: The tests themselves are generally downloaded from source and built locally on the machine, rather than fetching any distribution-specific packages. This is done to ensure more comparable results across operating systems / releases, etc. The distribution packager could be applying a number of unknown patches to the software, building the software with unique build options, or making other changes to the software that could skew the results.

Q: Besides being a developer, documentation writer, or having any other unique technical abilities, how else can I contribute to the Phoronix Test Suite?

A: Independent code contributions are very welcome as well as creating your own test profiles and suites. We also appreciate any feedback, comments, or other ideas either by emailing us, posting on GitHub, or sending a message to the mailing list.

Q: Do you offer technical support for the Phoronix Test Suite

A: Paid, professional support is available and is done via **our commercial services**. We also offer Phoromatic licenses for use within a corporate intranet and other custom services. Free, community

support is offered via our **mailing list**, IRC channel (*#phoronix* on *FreeNode.net*, and **GitHub**.

Q: May I put the Phoronix Test Suite logo on my company's web-site or on my product packaging?

A: **Contact us** for licensing information and details regarding the Phoronix Certification & Qualification Suite.

Q: How often is the Phoronix Test Suite updated?

A: We provide major updates on a quarterly basis with an occasional point release to fix outstanding bugs or address other issues. The latest work going into the Phoronix Test Suite is accessible via our Git repository at **GitHub.com/phoronix-test-suite**.

Tips & Tricks

General

- The desktop's screensaver will automatically be shutdown when a test is running and will be restored to its previous state upon the test's completion. This is supported for GNOME, KDE, and other XDG-supportive desktop environments.

- If you have many computers you wish to benchmark, once all of your tests have been downloaded, run *phoronix-test-suite make-download-cache* to generate a copy of these files at *~/.phoronix-test-suite/download-cache/*. A download cache is used for conserving time and bandwidth by eliminating the need for the Phoronix Test Suite to download files that have already been downloaded once. Copy this folder to the other systems or copy it to a DVD or USB hard drive, connect it to the next test system, and the Phoronix Test Suite will automatically use this local download cache. Or store these files on a local HTTP/FTP server and then update your *~/.phoronix-test-suite/user-config.xml* file to reflect the location of this download cache directory.

Running

- When running a test in batch mode (through the use of the *batch-run* or *batch-benchmark* options) that normally has end-user options (such as the sub-test to run or resolution), the Phoronix Test Suite will run the test with each unique combination of options possible, if configured appropriately.

- When running a test where you are prompted to enter any test options, multiple selections can be performed -- which will result in multiple test runs for each combination of the selected option(s) -- by separating each intended test option / number with a comma.

- When being prompted for the test identifier or the file name for saving the results, several user variables are supported. These include *$VIDEO_RESOLUTION*, *$VIDEO_CARD*, *$OPERATING_SYSTEM*, *$PROCESSOR*, *$MOTHERBOARD*, *$CHIPSET*, and *$KERNEL_VERSION*. If any of these variables are entered, the Phoronix Test Suite will replace them with their respective

values before saving the results.

- If *RemoveDownloadFiles* is set to *TRUE* within the *user-config.xml* file, once a test has been installed the originally downloaded files for that test will be automatically removed. This conserves disk space but will cause these files to be re-downloaded the next time the test needs to be re-installed. This will also not back up the downloaded files to the Phoronix Test Suite download cache. Enabling this option is just recommended for users with very limited disk space.

- If the amount of video memory for your graphics card is incorrectly reported by the Phoronix Test Suite (you can check by running *phoronix-test-suite diagnostics*), you can use the *VIDEO_MEMORY=* environment variable for overriding the video memory capacity (in Megabytes) used by the Phoronix Test Suite.

- If the *DISPLAY* environment variable is not set or *NO_GRAPHICS_TESTS* environment variable is set, tests of type *Graphics* will not be run. Likewise, if *NO_SYSTEM_TESTS* environment variable is set, tests of type *System* will not run. This applies to all test types where *NO_<TEST TYPE>_TESTS* is set.

- If while running multiple tests you want to quit the testing prematurely, in a new terminal window type *touch ~/.phoronix-test-suite/halt-testing*. All results for tests that had already run will be saved (permitting you opted to save the results), except for the test currently being run.

- If you wish to stop the current test run prematurely but continue the testing process, in a new terminal window type *touch ~/.phoronix-test-suite/skip-test*.

- If you want the specified test(s) to run in a loop for a set period of time, use the *TOTAL_LOOP_TIME* environment variable. For instance, running *TOTAL_LOOP_TIME=120 phoronix-test-suite benchmark ffmpeg* would keep running the ffmpeg test profile for 120 minutes.

- If you want the specified test(s) to run in a loop for a set number of times, use the *TOTAL_LOOP_COUNT* environment variable. For instance, running *TOTAL_LOOP_COUNT=3 phoronix-test-suite benchmark ffmpeg* would keep running the ffmpeg test profile three times.

- When any tests are being installed and when tests are being run, a lock is created in the system's temporary directory with the name *phoronix-test-suite.active* (i.e. */tmp/phoronix-test-suite.active*) and is removed upon completion. Thus if you have any system scripts that you wish to run when tests are not running or being installed as to not impact the results, one simple way to handle this is by having the script check for the existence of this lock.

Troubleshooting

- If a test profile fails immediately after starting, check the test profile's directory in *~/.phoronix-test-suite/installed-tests/* to confirm that the needed files are present. On platforms without External Dependencies support (Windows), it may be necessary to download the files manually and place them in this directory. If this is the case, you will notice that the "Downloading" phase of test

installation completes instantly.

- Inspect the scripts inside the above test profile's directory and confirm that directories or search paths for the test correspond to those on your system

- Try running the test profile with the *debug-benchmark* command, or reinstalling with the *debug-install* command and make note of any unusual output.

Configuration

- The user configuration options for the Phoronix Test Suite are stored in *~/.phoronix-test-suite/user-config.xml*. The batch mode options are also stored within this file and those can be adjusted by running *phoronix-test-suite batch-setup*.

- The colors, size, and other attributes for the graphs found within the Phoronix Test Suite Results Viewer can be modified via the file *~/.phoronix-test-suite/graph-config.json*.

Test / Suite Writing

- The Phoronix Test Suite recursively determines tests/suites and allows a suite to call another suite.

Module Writing

- By writing a Phoronix Test Suite module, you can easily extend the functionality of the Phoronix Test Suite. Some examples are being able to email the results upon completion, controlling the system's screensaver, updating a text LCD panel with the current test status, etc.

Phoronix Test Suite v10.4.0

Test Client Documentation

Virtual Test Suites

Virtual test suites are not like a traditional test suite defined by the XML suite specification. Virtual test suites are dynamically generated in real-time by the Phoronix Test Suite client based upon the specified test critera. Virtual test suites can automatically consist of all test profiles that are compatible with a particular operating system or test profiles that meet other critera. When running a virtual suite, the OpenBenchmarking.org repository of the test profiles to use for generating the dynamic suite must be prefixed.

Virtual test suites can be installed and run just like a normal XML test suite and shares nearly all of the same capabilities. However, when running a virtual suite, the user will be prompted to input any user-configuration options for needed test profiles just as they would need to do if running the test individually. When running a virtual suite, the user also has the ability to select individual tests within the suite to run or to run all of the contained test profiles. Virtual test suites are also only supported for an OpenBenchmarking.org repository if there is no test profile or test suite of the same name in the repository. Below is a list of common virtual test suites for the main Phoronix Test Suite repository, but the dynamic list of available virtual test suites based upon the enabled repositories is available by running *phoronix-test-suite list-available-virtual-suites.*

All Tests In Pts *pts/all*

This is a collection of all supported test profiles found within the specified OpenBenchmarking.org repository.

Application Tests *pts/application*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a application software test.

Benchmark Tests *pts/benchmark*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a benchmark software test.

BLAS (Basic Linear Algebra Sub-Routine) Tests *pts/blas*

This is a collection of test profiles having an external dependency on BLAS (Basic Linear Algebra Sub-Routine)

Phoronix Test Suite v10.4.0

Test Client Documentation

C++ Boost Tests *pts/boost*

This is a collection of test profiles having an external dependency on C++ Boost

Bsd Operating System Tests *pts/bsd*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the bsd Operating System.

Disk Subsystem Tests *pts/disk*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the disk sub-system.

Everything In Pts *pts/everything*

This is a collection of all test profiles found within the specified OpenBenchmarking.org repository, including unsupported tests, etc.

Fortran Tests *pts/fortran*

This is a collection of test profiles having an external dependency on Fortran

Game Tests *pts/game*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a game software test.

Go Language Tests *pts/golang*

This is a collection of test profiles having an external dependency on Go Language

Graphics Subsystem Tests *pts/graphics*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the graphics sub-system.

Phoronix Test Suite v10.4.0

Test Client Documentation

Installed Tests *pts/installed*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository that are already installed on the system under test.

Java Tests *pts/java*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing java.

LAPACK (Linear Algebra Pack) Tests *pts/lapack*

This is a collection of test profiles having an external dependency on LAPACK (Linear Algebra Pack)

Linux Operating System Tests *pts/linux*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the linux Operating System.

Macosx Operating System Tests *pts/macosx*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the macosx Operating System.

Memory Subsystem Tests *pts/memory*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the memory sub-system.

Network Subsystem Tests *pts/network*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the network sub-system.

Node.js + NPM Tests *pts/node-npm*

This is a collection of test profiles having an external dependency on Node.js + NPM

Phoronix Test Suite v10.4.0

Test Client Documentation

OpenCV Tests *pts/opencv*

This is a collection of test profiles having an external dependency on OpenCV

Openmpi Tests *pts/openmpi*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing openmpi.

Os Subsystem Tests *pts/os*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the os sub-system.

Processor Subsystem Tests *pts/processor*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the processor sub-system.

Python Tests *pts/python*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing python.

Ruby Tests *pts/ruby*

This is a collection of test profiles having an external dependency on Ruby

Rust Tests *pts/rust*

This is a collection of test profiles having an external dependency on Rust

Scientific Tests *pts/scientific*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a scientific software test.

Phoronix Test Suite v10.4.0

Test Client Documentation

Simulator Tests *pts/simulator*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a simulator software test.

Solaris Operating System Tests *pts/solaris*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the solaris Operating System.

System Subsystem Tests *pts/system*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the system sub-system.

Utility Tests *pts/utility*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a utility software test.

Windows Operating System Tests *pts/windows*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the windows Operating System.

Smp Tests *pts/smp*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing smp.

Opencl Tests *pts/opencl*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing opencl.

Cuda Tests *pts/cuda*

Phoronix Test Suite v10.4.0

Test Client Documentation

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing cuda.

Mpi Tests *pts/mpi*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing mpi.

Openmp Tests *pts/openmp*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing openmp.

Cloud Tests *pts/cloud*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing cloud.

Docker Tests *pts/docker*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing docker.

Go Tests *pts/go*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing go.

Optix Tests *pts/optix*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing optix.

Vdpau Tests *pts/vdpau*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing vdpau.

Phoronix Test Suite v10.4.0

Test Client Documentation

Video Tests *pts/video*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing video.

Responsiveness Tests *pts/responsiveness*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing responsiveness.

All Tests In System *system/all*

This is a collection of all supported test profiles found within the specified OpenBenchmarking.org repository.

Application Tests *system/application*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a application software test.

Benchmark Tests *system/benchmark*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a benchmark software test.

BLAS (Basic Linear Algebra Sub-Routine) Tests *system/blas*

This is a collection of test profiles having an external dependency on BLAS (Basic Linear Algebra Sub-Routine)

C++ Boost Tests *system/boost*

This is a collection of test profiles having an external dependency on C++ Boost

Bsd Operating System Tests *system/bsd*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where

the test profile is specified as being compatible with the bsd Operating System.

Disk Subsystem Tests *system/disk*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the disk sub-system.

Everything In System *system/everything*

This is a collection of all test profiles found within the specified OpenBenchmarking.org repository, including unsupported tests, etc.

Game Tests *system/game*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a game software test.

Graphics Subsystem Tests *system/graphics*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the graphics sub-system.

Linux Operating System Tests *system/linux*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the linux Operating System.

Macosx Operating System Tests *system/macosx*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the macosx Operating System.

Network Subsystem Tests *system/network*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the network sub-system.

OpenCV Tests *system/opencv*

Phoronix Test Suite v10.4.0

Test Client Documentation

This is a collection of test profiles having an external dependency on OpenCV

OpenMPI Tests *system/openmpi*

This is a collection of test profiles having an external dependency on OpenMPI

Processor Subsystem Tests *system/processor*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the processor sub-system.

Python Tests *system/python*

This is a collection of test profiles having an external dependency on Python

Scientific Tests *system/scientific*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a scientific software test.

Solaris Operating System Tests *system/solaris*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the solaris Operating System.

System Subsystem Tests *system/system*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the system sub-system.

Utility Tests *system/utility*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a utility software test.

Windows Operating System Tests *system/windows*

Phoronix Test Suite v10.4.0

Test Client Documentation

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the windows Operating System.

Opencl Tests *system/opencl*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing opencl.

Cuda Tests *system/cuda*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing cuda.

Smp Tests *system/smp*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing smp.

All Tests In Git *git/all*

This is a collection of all supported test profiles found within the specified OpenBenchmarking.org repository.

Bsd Operating System Tests *git/bsd*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the bsd Operating System.

Everything In Git *git/everything*

This is a collection of all test profiles found within the specified OpenBenchmarking.org repository, including unsupported tests, etc.

Linux Operating System Tests *git/linux*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the linux Operating System.

Phoronix Test Suite v10.4.0

Test Client Documentation

Macosx Operating System Tests *git/macosx*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the macosx Operating System.

Processor Subsystem Tests *git/processor*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the processor sub-system.

Rust Tests *git/rust*

This is a collection of test profiles having an external dependency on Rust

Utility Tests *git/utility*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a utility software test.

Smp Tests *git/smp*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing smp.

Phoronix Test Suite v10.4.0

Test Client Documentation

Component Testing

Compiler Testing & Masking

A majority of the test profiles provided by OpenBenchmarking.org to the Phoronix Test Suite are source-based tests. Relying upon the upstream source-code for each program under test allows for the tests to be easily brought to new platforms and architectures, avoids any out-of-tree / non-default packaging differences by different distributions and operating systems, and to allow the Phoronix Test Suite user to easily test new compilers and/or compiler options. For the source-based tests, the Phoronix Test Suite relies upon a compiler (e.g. GCC, LLVM/Clang, Sun Studio, Open64, et al) being present on the system under test. The Phoronix Test Suite does respect *CC/CXX* environment variables and test profiles are expected to honor *CFLAGS/CXXFLAGS* and other compiler settings.

As of Phoronix Test Suite 3.8, a compiler masking method is utilized for logging compiler options and other settings with each test profile installation. Prior to installing an open-source test, the Phoronix Test Suite determines the intended compiler to be used based upon the pre-set environment variables or the pre-set compiler(s) within the *PATH* environment variable. The Phoronix Test Suite then masks the compiler to ensure that any options/flags submitted to the compiler are first passed through pts-core so that they can be logged for later use, then is linked to the original, intended compiler. Additionally, other compiler binary names of the same type are blacklisted to prevent their un-intended use (i.e. if a test profile has hard-coded *gcc* in its build script, but *clang* is set as the compiler via *CC*, a sym-link will automatically be created from *gcc* to the masked *clang* for the duration of its test installation).

The logged compiler data is then used by the Phoronix Test Suite following the test execution process for automated result analysis. If there is a detected change in compiler settings, the differences are automatically reported to the test result graphs. Additionally, key compiler information (e.g. the compiler optimization level and key libraries that were linked at compile-time) is reported as a footnote on result graphs. The Phoronix Test Suite handles all of this in a fully automated manner; test profiles require no changes to take advantage of these compiler-reporting capabilities.

Separately, the Phoronix Test Suite attempts to automatically log the system compiler's build configuration (i.e. GCC's *gcc -v* "Configured with") output. If the Phoronix Test Suite detects there is a compiler build configuration change between test runs in a result file, it will report each compiler's build configuration to the system information table within the results viewer. Reported to the table is a reduced view of the build configuration options, with less relevant items being stripped away from this view to reduce verbosity. Upon clicking the text, the raw compiler information output can be viewed in full.

Disk / File-System Testing

By default tests are installed to *~/.phoronix-test-suite/installed-tests/*. However, the location can be updated from *~/.phoronix-test-suite/user-config.xml* or dynamically via the *PTS_TEST_INSTALL_ROOT_PATH* environment variable.

Phoronix Test Suite v10.4.0

Test Client Documentation

When any disk tests are executed, the Phoronix Test Suite will attempt to log the mount options and scheduler of the disk/partition being used for testing. This information is subsequently displayed within the system information table. If the scheduler / mount options are maintained the same throughout all test runs, only a single line is displayed otherwise the options for each test run will be automatically displayed. The file-system in use is always captured and shown in the system information table.

Phoronix Test Suite v10.4.0

Test Client Documentation

Phoronix Test Suite Phoromatic

Phoromatic Server

Introduction

Phoromatic is a remote management system for the Phoronix Test Suite. Phoromatic allows the automatic (hence the name *Phoro-matic*) scheduling of tests, remote installation of new tests, and the management of multiple test systems all through an intuitive, easy-to-use web interface. Tests can be scheduled to automatically run on a routine basis across multiple test systems. The test results are then available from this central, secure location.

Phoromatic was originally introduced with Phoronix Test Suite 2.0 via Phoromatic.com as a project going back to 2008~2009. Phoromatic.com debuted as a hosted instance with the option of behind-the-firewall licensing for use within organizations. With Phoronix Test Suite 5.2 the model shifted to offer a local, open-source version of Phoromatic built into the Phoronix Test Suite code-base. Thanks to continued enterprise development, with Phoronix Test Suite 5.4 is now a fully-functioning, built-in version of Phoromatic that's open-source and can be used for behind-the-firewall testing without needing to push results to OpenBenchmarking.org and the ability to keep all results private.

Phoromatic in Phoronix Test Suite 5.4 also has the ability to support zero-conf network discovery using Avahi and the automatic distribution of needed test profiles/suites and test files. Phoronix Test Suite 5.4's Phoromatic is a significant breakthrough for open-source testing particularly those running this GPL benchmarking software within test labs and other large organizations.

Features

Built atop the Phoronix Test Suite, Phoromatic offers many features for both enterprise and community/personal users:

Automated Scheduling

Whether it is every evening at 6:00PM, once every Thursday at 10:00AM or somewhere in between, Phoromatic can schedule tests to be run at user-defined intervals. The testing schedules can be updated through Phoromatic web interface. After the test(s) have run, the results will be immediately uploaded to Phoromatic.

Extensible

Any test profile or test suite that is compliant with the Phoronix Test Suite specification will work with Phoromatic. Phoromatic is able to leverage the hundreds of test profiles and test suites currently in the Phoronix Test Suite via OpenBenchmarking.org, along with any custom or proprietary test profiles you or your company utilize. Additionally, the Phoromatic interface allows the user to construct their own

test suite(s).

Remote Testing

Once the test system is setup, all testing and management of that system can be done remotely. There is no need to execute Phoronix Test Suite commands locally using the GUI or command line version, but instead nearly all of the same features are accessible from the Phoromatic interface.

Multi-System Support

A single Phoromatic account is able to manage multiple test systems running the Phoronix Test Suite. Phoromatic supports grouping together test systems, tagging, and other features to support effectively managing many test systems. From the Phoromatic interface, installed system hardware and software from a given system can also be viewed.

Turn-Key Deployment

No additional software needs to be installed to support Phoromatic; all that's needed is Phoronix Test Suite 5.4 or later for full compatibility. New test systems can easily be synced with a given Phoromatic account by running a single command from the Phoronix Test Suite client.

Result Management

Test results are automatically uploaded to the Phoromatic account and remain private unless you opt to upload them to OpenBenchmarking.org. From the Phoromatic interface, results from multiple test systems can easily be compared and multiple results from the same systems can be used to track performance over time. There are also options to look at the statistical significance of the results and other features to efficiently and effectively analyze the system's performance.

Decentralized

Once the Phoronix Test Suite running on the Phoromatic Server has been able to cache all of the OpenBenchmarking.org test files and the needed files for each test, Phoromatic with any Phoronix Test Suite clients on your LAN can run fully decentralized without the need for a constant stream of OpenBenchmarking.org communication or Internet connection for that matter. (The only exception would be if your local systems don't have all their needed external dependencies and your system's package manager would need to install components like a compiler or necessary system libraries.

Fully Open-Source

Phoromatic is now fully open-source within the Phoronix Test Suite code-base for fostering greater development and new capabilities. Patches are welcome and Phoronix Media is available to provide commercial support and custom engineering services around Phoromatic and the Phoronix Test Suite.

Phoromatic Server Setup

Phoronix Test Suite v10.4.0

Test Client Documentation

Phoromatic is built into the Phoronix Test Suite code-base and should be found in all packaged versions of the **phoronix-test-suite**. Starting the Phoromatic Server entails running phoronix-test-suite start-phoromatic-server after configuring the server information within *~/.phoronix-test-suite/user-config.xml*. The Phoromatic Server can with or without root permissions depending upon your firewall and the port numbers you wish to use for the server.

On the "client side", any up-to-date version of the Phoronix Test Suite can automatically communicate with the Phoromatic Server. If Avahi support is available (commonly in Linux distribution repositories as avahi-tools), there should be zero-conf discovery if the Phoromatic Server and client systems are on the same LAN. If a Phoronix Test Suite client discovers a Phoromatic Server, it will attempt to use it automatically as a local download cache. In the event of no Internet connection, it will also attempt to obtain the needed OpenBenchmarking.org test/suite meta-data from the Phoromatic Server based upon its archived meta-data. This allows the Phoronix Test Suite / Phoromatic deployment on the LAN to be self-sustaining without an Internet connection as long as the systems have all installed test dependencies.

Further configuration of the setup parameters for the Phoromatic Server and Phoronix Test Suite clients can be tuned via the *~/.phoronix-test-suite/user-config.xml* file. All control and configuration of the Phoromatic Server is done via the web-based interface when the Phoromatic Server is active.

The Phoromatic Server utilizes PHP/HHVM's built-in web-server capabilities and there's also a Phoronix Test Suite built-in WebSocket server that's also initiated for back-end processing. At this time there are no ports set by default for these services but must be defined within the user configuration file. With the Avahi zero-conf network discovery and other automated detection in place, there's little restrictions over the port selection.

Systemd and Upstart service files are shipped with the Phoronix Test Suite for those that wish to have the services automatically run as daemons. The only new requirements over the basic Phoronix Test Suite system requirements is having PHP-SQLite support installed and the newer version of PHP or HHVM is recommended for offering the best support.

Example Deployments

Use Case A: Unrestricted Internet Access, Local Result Storage

Systems on your network with unrestricted Internet access is the easiest and simplest deployment for the Phoronix Test Suite and Phoromatic. After installing the Phoronix Test Suite on the system you wish to designate the Phoromatic Server and have configured the *user-config.xml* file, simply run:

$ phoronix-test-suite start-phoromatic-server

Assuming you have no firewall or permission issues, the built-in web server and WebSocket server should proceed to initiate along with outputting the IP/port information for these services. Unless otherwise disabled from the user configuration file and if avahi-tools is present, the Phoromatic Server will be advertised with Avahi for zero-configuration networking.

Phoronix Test Suite v10.4.0

Test Client Documentation

From the Phoromatic web interface you are able to create an account and from there proceed with the creating of test schedules, updating settings, and connecting systems. From the "client systems" you wish to use as the benchmarking nodes, it's simply a matter of running **phoronix-test-suite phoromatic.connect** with zero-conf networking or otherwise follow the information from the Phoromatic web interface for manual setup with the IP/port information.

Use Case B: No Internet Available To Client Systems

It's possible to run the Phoronix Test Suite and Phoromatic Server without a persistent Internet connection as long as you are able to first download the necessary files to the Phoromatic Server. After installing the Phoronix Test Suite on the system you wish to designate the Phoromatic Server and have configured the *user-config.xml* file, a few commands from the system while having an Internet connection will be able to cache the needed data:

$ phoronix-test-suite make-download-cache x264 xonotic ffmpeg

This command will simply download all of the needed test files for the tests/suites passed to the sub-command. Alternatively you could also pass pts/all to cache all tests. It's important though to just cache the tests/suites you'll be using on your network. This will generate the test file download cache by default to *~/.phoronix-test-suite/download-cache/* or */usr/share/phoronix-test-suite/download-cache/* depending upon your write permissions. You can always run this command later with more test files. Alternatively, if you already have a number of tests installed on the system, simply running "phoronix-test-suite make-download-cache" will generate the cache based upon the currently installed tests.

$ phoronix-test-suite make-openbenchmarking-cache

This command will cache as much of the OpenBenchmarking.org meta-data as possible for test profiles and test suites. After the above commands, the Phoromatic Server should no longer need a persistent Internet connection.

$ phoronix-test-suite start-phoromatic-server

Proceed to start the Phoromatic Server and operate as normal.

For the test clients without an Internet connection, as long as they're able to reach the Phoromatic Server, the Phoromatic Server should be able to automatically serve all of the needed test files download cache and OpenBenchmarking.org meta-data to the systems locally.

Use Case C: Phoromatic Across The Internet

If wishing to use the same Phoromatic Server across multiple geographic locations, it's easily possible -- you just lose out on the zero-conf networking ability. To let the Phoronix Test Suite client systems know about the remote Phoromatic Server, simply add the Phoromatic Server information to the client's *PhoromaticServers* element within the *user-config.xml*. Of course, make sure the Phoromatic Server

has a globally resolvable IP address and its Phoromatic HTTP/WebSocket ports are open. Once informing the client of the Phoromatic Server, the use cases as above apply in the same manner.

Client Setup

From Phoronix Test Suite client systems running on the LAN, the following command will report all available detected Phoromatic Servers along with important server and debugging information:

$ phoronix-test-suite phoromatic.explore

With the following example output on finding one successful server:

IP: 192.168.1.211
HTTP PORT: 5447
WEBSOCKET PORT: 5427
SERVER: PHP 5.5.9-1ubuntu4.4 Development Server
PHORONIX TEST SUITE: Phoronix Test Suite v5.4.0m1 [5313]
DOWNLOAD CACHE: 19 FILES / 2390 MB CACHE SIZE
SUPPORTED OPENBENCHMARKING.ORG REPOSITORIES:
pts - Last Generated: 05 Oct 2014 07:16

Phoromatic Servers are detected by the Phoronix Test Suite through Avahi or if manually configuring the Phoronix Test Suite clients to point to Phoromatic Servers. For networks without Avahi/auto-discovery support or for test systems that may be connecting from another network, the IP address and HTTP port number can be added to the local system's *~/.phoronix-test-suite/user-config.xml* with the *PhoromaticServers* element. Adding the *IP:port* (the Phoromatic Server's HTTP port) to the PhoromaticServers *user-config.xml* element for will perform targeted probing by the Phoronix Test Suite without any dependence on Avahi. Multiple Phoromatic Servers can be added if each IP:port is delimited by a comma.

To connect a Phoronix Test Suite system for benchmarking to an account, log into your Phoromatic account from the web-interface and on the main/system pages will be instructions along with a specially formed string to run, e.g. *phoronix-test-suite phoromatic.connect 192.168.1.211:5447/I0SSJY*. When running that command once on the system(s) to be synced to that account, as the administrator you'll be able to validate/approve the systems from the Phoromatic web interface. After that, whenever the system(s) are to be running benchmarks, simply have the **phoronix-test-suite phoromatic.connect** command running on the system (after the initial account has been synced, simply running **phoronix-test-suite phoromatic.connect** is enough for the system to find the server and its account).

Root Administrator

The root administrator account is able to manage the server-level settings, e.g. Phoromatic storage location and other global settings related to the Phoronix Test Suite / Phoromatic Server, from the web user-interface.

To enable the root administrator log-in, first from the server's command-line interface run **phoronix-test-suite phoromatic.set-root-admin-password** to set the password. Following that, you can log into the root administrator account via the web interface via the *rootadmin* user-name and the set password.

Other Advice

Disable Internet Precaution

If you have an Internet connection but want to ensure your Phoronix Test Suite client doesn't attempt to use it for any matter, via the *~/.phoronix-test-suite/user-config.xml* you can set *NoInternetCommunication* to *TRUE*. There's also a NoNetworkCommunication tag, but setting that to TRUE will disable any form of network communication -- including communication with the Phoromatic Server.

Ports / Services

The Phoromatic Server process currently relies upon a PHP/HHVM built-in web server process and a PTS-hosted WebSocket server. The web server process handles the web UI and much of the responsibilities of the Phoromatic Server. Over time the PTS WebSocket server will be increasingly utilized for bi-directional, real-time communication between the server and clients -- including for features like viewing real-time hardware sensors of client systems from the server UI.

Systemd / Upstart

Packaged with the Phoronix Test Suite are basic *phoromatic-client* and *phoromatic-server* configurations for both Upstart and systemd init systems. The *phoromatic-server* configuration will launch the Phoronix Test Suite's Phoromatic Server and the *phoromatic-client* service will attempt to connect to a pre-configured Phoromatic Server. The systemd service files will automatically be installed via the Phoronix Test Suite *install-sh* process while the Upstart jobs can be copied from *deploy/phoromatic-upstart/** to */etc/init*.

Cache Verification

To confirm the files accessible to Phoronix Test Suite client systems, from the Phoromatic Server web user-interface go to the *settings* page followed by the *cache settings* link to view information about the download and OpenBenchmarking.org caches. From the client systems, running **phoronix-test-suite phoromatic.explore** will also supply cache statistics.

Log Files

The Phoromatic Server will produce a log file of events / debugging information to *~/.phoronix-test-suite/phoromatic.log* or */var/log/phoromatic.log* depending upon the service's permissions. When running the Phoronix Test Suite Phoromatic client, the log will be written to one of the respective locations in *phoronix-test-suite.log*.

Phoronix Test Suite v10.4.0

Test Client Documentation

Multi-User Accounts

For each time a user account is made from the Phoromatic web UI's log-in page, all of the test schedules, systems, and other account information is separate to allow for a completely isolated multi-user system. If a main administrator (the one creating the account) wishes to have multiple users sharing the same account data, that user can create additional accounts from the *Users* tab of their account. The main administrator can make an additional administrator account or a "viewer" account that can consume the account's data but not create/modify the schedules, systems, or other account details.

File Locations

When running the Phoronix Test Suite Phoromatic Server as root, rather than using the *~/.phoronix-test-suite/* directory, the standard Linux file-system hierarchy standard is honored. The main storage path is */var/lib/phoronix-test-suite/*, the user configuration file is */etc/phoronix-test-suite.xml*, and */var/cache/phoronix-test-suite/* for cache files.

Uploading Other Test Results

Unscheduled test results and other results found on connected systems to a Phoromatic account can upload the data to the Phoromatic Server using the *phoronix-test-suite phoromatic.upload-result <result file identifier >* sub-command.

User Context File Logging

For those utilizing custom set context script files as part of the Phoromatic test schedule, any important notes / log information can be written to the file specified by the *PHOROMATIC_LOG_FILE* environment variable set while running the user context scripts. The contents of that file is then sent to the Phoromatic Server otherwise the standard output of the script's execution is submitted to the Phoromatic Server for logging. These logs can then be viewed by the Phoromatic Server along with the test results. Other environment variables accessible when running a user context script include *PHOROMATIC_TRIGGER*, *PHOROMATIC_SCHEDULE_ID*, and *PHOROMATIC_SCHEDULE_PROCESS*.

Phoronix Test Suite v10.4.0

Test Client Documentation

Offline Testing

Offline Testing/Benchmarking For Single System Environments

The Phoronix Test Suite ships with a cache (up to date as of release time) of the available OpenBenchmarking.org test profile / test suite metadata but external download files are necessary for the test profiles to function. Due to hundreds of different test profiles and consisting of software under test that is of varying software licenses, there is no centralized archive of all possible test files.

To obtain a cache of the files needed for the desired test(s), from a machine with a working Internet connection, run the **phoronix-test-suite make-download-cache** sub-command and pass the name of the tests/suites you wish to download. The make-download-cache will download the files for the desired test profiles so you can then transfer them for use on individual computer(s) lacking an Internet connection.

By default the files will be cached to *~/.phoronix-test-suite/download-cache* and when transferred to an offline system in the same location it should then be automatically utilized by the Phoronix Test Suite when going to install the test(s). The individual download-cache directory can be copied to the offline system or otherwise more broadly the *~/.phoronix-test-suite* directory can also be copied to the offline system(s). If running as root, the default download cache directory is **/var/cache/phoronix-test-suite/download-cache/**.

When the Phoronix Test Suite download cache is transferred to the offline system, the Phoronix Test Suite should begin automatically making use of those files when detected in the appropriate directory and having a matching hash-sum for the given file.

If not able to run the Phoronix Test Suite on a machine with a working Internet connection, manually downloading the files referenced within the test profiles XML metadata and placing them within the respective download-cache directory on a system is another manual alternative.

Note that the make-download-cache will only cache the files downloaded by the Phoronix Test Suite. Depending upon the test(s) and the state of your operating system, you may also need packages obtained from your package manager / distribution (e.g. compiler, dependency libraries, etc) that are not cached by the Phoronix Test Suite due to the diverse nature of different supported operating systems.

Phoronix Test Suite v10.4.0

Test Client Documentation

Offline Improvements + Confidential Testing / Avoiding A

Offline Enhancement Via Local Cache

Beginning with Phoronix Test Suite 9.0, there are improvements to improve the out-of-the-box experience if running the Phoronix Test Suite in a strictly offline environment / behind-the-firewall without access to OpenBenchmarking.org for being able to obtain test profiles / test suites. From Phoronix Test Suite 3.0 when OpenBenchmarking.org was introduced until Phoronix Test Suite 9.0, Internet connectivity was initially required for obtaining the test profiles/suites as the cloud/repository. OpenBenchmarking.org allows for tests to be updated independently of the Phoronix Test Suite releases as well as allowing new tests to be introduced on-demand. Aside from when new tests require explicit new PTS features, this allows tests/suites to be seamlessly used by older versions of the Phoronix Test Suite without any upgrade process required, assuming Internet connectivity is available.

Beginning with Phoronix Test Suite 9.0, a static snapshot of the official tests/suites is included as part of the Phoronix Test Suite package. The intention with this is to provide a static snapshot with all tests/suites as of release time, similar to the behavior with pre-3.0 releases. The benefit to including this static snapshot is helping those that are running strictly offline/isolated to be able to have at least recent tests/suites available without first needing to query OpenBenchmarking.org for this data. But Internet support is certainly desired in order to be able to obtain updated and new test profiles.

This static snapshot is provided in the *ob-cache/* Phoronix Test Suite folder. If this cache is not needed or wish to customize/extend it, it can be safely removed and or altered without causing issues. When the Phoronix Test Suite has Internet connectivity, it will continue to query OpenBenchmarking.org for new/updated tests and suites.

This local cache does provide current and previous versions of test profiles to allow users to continue running older versions of tests/results even when upgrading their Phoronix Test Suite offline copy.

Even with the local cache, there still is the need for obtaining any necessary files needed to run the selected test(s). For those wishing to optimize that workflow for offline usage, see the existing *phoronix-test-suite make-download-cache* sub-command documentation. The *phoronix-test-suite make-openbenchmarking-cache* sub-command may also be desirable depending upon setup.

Disabling OpenBenchmarking.org Result Upload Functionality

Phoronix Test Suite 9.0 also improved the workflow around disabling OpenBenchmarking.org result uploading functionality for those carrying out confidential tests or otherwise wish to provide safeguards for ensuring no results may be accidentally uploaded publicly.

Removal of OpenBenchmarking.org upload support can be done by deleting *phoronix-test-suite/pts-core/objects/pts_openbenchmarking_upload.php*. If that file is removed, the Phoronix Test Suite should respond gracefully and not prompt users about any upload and within that

file is the only logic for actually uploading the results to Openbenchmarking. So simply by removing that file you should be covered from any accidental uploading of results. Removal/disabling of this file also prevents any anonymous usage reporting.

For those without the ability to remove that file from their Phoronix Test Suite installation or as a secondary safeguard, from the Phoronix Test Suite user configuration file (*/etc/phoronix-test-suite.xml as root or ~/.phoronix-test-suite/user-config.xml for most users) is a "AllowResultUploadsToOpenBenchmarking" option. If setting that value to FALSE, it should apply the same behavior as if deleting the pts_openbenchmarking_upload file.*

If distributing a customized/local copy of the Phoronix Test Suite, the default behavior of the configuration file (in addition to deleting the pts_openbenchmarking_upload file) can be done via the user configuration defaults defined within *pts-core/static/user-config-defaults.xml*.

Phoronix Test Suite v10.4.0

Test Client Documentation

Development Credits

The Phoronix Test Suite is based upon the extensive testing and internal tools developed by **Phoronix.com** since 2004 along with support from leading tier-one computer hardware and software vendors. The principal architects of the Phoronix Test Suite are **Michael Larabel** and Matthew Tippett. The phoronix-test-suite, pts_Graph, Phoromatic, Phodevi, tandem_Xml, and nye_Xml are some of the related open-source projects provided by **Phoronix Media**.