www.phoronix-test-suite.com

Phoronix Test Suite v10.8.5

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 an automated test framework for providing seamless execution of test profiles and test suites. There are more than 650 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 workstations/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, 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.

A Phoromatic server can be started using *phoronix-test-suite start-phoromatic-server* (or the included systemd phoromatic-server service file). Clients can connect to the server using the *phoronix-test-suite phoromatic.connect* command as well as a phoromatic-client systemd service. See the Phoromatic section of the documentation for more information on setting up Phoromatic.

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

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 / resume testing on the remaining tests where there are missing results to be completed.

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.

dump-suites-to-git

This option will create a Git repository of OpenBenchmarking.org test suites.

dump-tests-to-git

This option will create a Git repository of OpenBenchmarking.org test profiles.

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.

ob-test-profile-analyze

This option is intended for test profile creators and generates a range of meta-data and other useful information that can be submitted to OpenBenchmarking.org to provide more verbose information for users of your test profiles.

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 repository 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

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 necessary 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-status

This sub-command provides a verbose look at all tests installed/uninstalled on the system and whether any errors were encountered at install-time or run-time and other test installation/runtime metrics for

complementing other Phoronix Test Suite sub-command outputs.

list-test-usage

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

print-tests *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will print the test identifiers of the specified result file(s), test suite(s), OpenBenchmarking.org ID(s), or other runnable object(s).

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.

dump-documentation

This option is used for re-generating the Phoronix Test Suite documentation.

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.

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

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. Alternatively, if using the environment variable "SORT_BY" other sort modes can be used, such as SORT_BY=date / SORT_BY=date-desc for sorting by the test run-time/date.

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.

prune-empty-results *[Test Result]*

This option is used if there are test results (benchmarks) to be dropped from a given result file that had no successful runs. I.e. if any run attempt(s) only errored out without producing any result for any of the test run(s) saved in that file. The user must specify a saved results file.

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

This option is used if there are test results (benchmarks) to be dropped from a given result file for having incomplete data, either a test run did not attempt to run that benchmark or failed to properly run. The user must specify a saved results file and the command will then attempt to find any results with incomplete/missing data and prompt the user with confirmation to remove them.

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.

show-result *[Test Result]*

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

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.

Result Analysis

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.

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.

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.

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.

Debugging

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.

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.

dump-file-info

This option will dump the MD5 / SHA256 hashes and file size for a given file.

dump-openbenchmarking-indexes

This option is used for dumping the parsed output of OpenBenchmarking.org index files (metadata).

dump-phodevi-smart-cache

This option is used for displaying the contents of the Phodevi smart cache on the system.

dump-possible-options

This option will print all possible phoronix-test-suite sub-commands.

dump-unhandled-dependencies

This option will list missing entries in the external dependencies XML file for the operating system under test. This option is used if wanting to help find missing dependency XML data to fill in for contributing to upstream Phoronix Test Suite.

list-failed-installs

This option will list all test profiles that were attempted to be installed on the local system but failed to be installed. Where applicable, the possible error(s) from the test installation are also reported to assist in debugging.

list-test-errors

This sub-command is complementary to list-failed-installs. Rather than listing test installation errors, list-test-errors is used for displaying past test run-time errors. This option will list all test profiles that produced an error previously when running the test profile / benchmark. If a test profile later successfully ran the test with any given option(s) without errors, the error is then removed from the archive. This option is intended to be helpful in debugging test profile issues later on for having a persistent collection of run-time errors.

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.

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.

variables

This option will print all of the official environment variables supported by the Phoronix Test Suite for user configuration purposes. These environment variables are also listed as part of the official Phoronix Test Suite documentation while this command will also show the current value of the variables if currently set.

Result Export

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. The outputted file appears in the user home directory or can otherwise be controlled via the OUTPUT_DIR and OUTPUT_FILE environment variables.

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

This option will read a saved test results file and output the system hardware and software information along with the results to pure HTML file. No external files are required for CSS/JavaScript or other assets. The graphs are rendered as inline SVG. This is a pure HTML-only representation of the results for emailing or other easy analysis outside of the Phoronix Test Suite. The outputted file appears in the user home directory or can otherwise be controlled via the OUTPUT_DIR and OUTPUT_FILE environment variables.

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). The outputted file appears in the user home directory or can otherwise be controlled

via the OUTPUT_DIR and OUTPUT_FILE environment variables.

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. The outputted file appears in the user home directory or can otherwise be controlled via the OUTPUT_DIR and OUTPUT_FILE environment variables.

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.

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.

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 similar 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

System Maintenance / Cleanup

This module can be used for system maintenance cleanup tasks around the Phoronix Test Suite. Currently implemented is support for automatically un-installing tests that have not been run in a period of time. When the module is loaded via the REMOVE_TESTS_OLDER_THAN environment variable, it will be automatically invoked at the end of running any benchmarks. Or this module can be manually invoked with the command: phoronix-test-suite cleanup.tests.

phoronix-test-suite cleanup.tests

This module utilizes the following environment variables: REMOVE_TESTS_OLDER_THAN.

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.

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 environment variables: PTS_FLUSH_CACHES.

Phoronix Test Suite v10.8.5

Test Client Documentation

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 environment 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. Depending upon system permissions you may be limited to using perf as root or adjusting the /proc/sys/kernel/perf_event_paranoid setting.

This module utilizes the following environment 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 environment 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

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 environment 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 environment 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 v10.8.5

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 environment 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 environment variables: RUN_TIMES_ARE_A_BENCHMARK, INSTALL_TIMES_ARE_A_BENCHMARK, INSTALL_SIZES_ARE_A_BENCHMARK.

Result Notifier

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 environment 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 v10.8.5

Test Client Documentation

- cpu.freq.cpu6
- cpu.freq.cpu7
- cpu.freq.cpu8
- cpu.freq.cpu9
- cpu.freq.cpu10
- cpu.freq.cpu11
- cpu.freq.cpu12
- cpu.freq.cpu13
- cpu.freq.cpu14
- cpu.freq.cpu15
- 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.cpu8
- cpu.usage.cpu9
- cpu.usage.cpu10
- cpu.usage.cpu11
- cpu.usage.cpu12
- cpu.usage.cpu13
- cpu.usage.cpu14
- cpu.usage.cpu15
- 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 environment 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 environment 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 environment variables: HALT_SCREENSAVER.

Linux Turbostat Dumper

Setting TURBOSTAT_LOG=_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. If the TURBOSTAT_LOG= value does not point to a directory, the TurboStat output will be appended to the test run log files.

This module utilizes the following environment variables: TURBOSTAT_LOG.

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 environment 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 environment variables: WATCHDOG_SENSOR, WATCHDOG_SENSOR_THRESHOLD, WATCHDOG_MAXIMUM_WAIT.

Phoronix Test Suite v10.8.5

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, Alpine Linux, Void Linux, Intel Clear Linux, and Amazon Linux EC2.

Among the tested BSD distributions are FreeBSD, NetBSD, OpenBSD, and DragonflyBSD. Tested Solaris distributions include Oracle Solaris 11, OpenIndiana, and Illumos.

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 and PHP 8 are also fully supported by the Phoronix Test Suite.

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.

The Phoronix Test Suite does not need to be installed system-wide but can simply be run from the extracted phoronix-test-suite folder as the local user.

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.

Phoronix Test Suite v10.8.5

Test Client Documentation

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.

Phoronix Test Suite v10.8.5

Test Client Documentation

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

Phoronix Test Suite v10.8.5

Test Client Documentation

The Phoronix Test Suite is fully supported on Apple's macOS operating system. PHP ships with macOS by default on macOS 12 and older 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. On newer versions of macOS not shipping with PHP by default, **Homebrew** can be used for installing PHP or building PHP from source. The Phoronix Test Suite also supports making use of Homebrew for acquiring necessary Phoronix Test Suite dependencies on macOS.

Phoronix Test Suite v10.8.5

Test Client Documentation

Phoronix Test Suite On Windows

Introduction

Phoronix Test Suite 8.0 introduced rewritten Windows support that is at near feature parity to the program's long-standing support for Linux, macOS, and BSD operating systems.

The Phoronix Test Suite Windows support currently targets **Windows 10 x64**, **Windows 11 x64** and **Windows Server 2016 x64** and later. 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. 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 or Windows Server, deploying the Phoronix Test Suite is designed to be as easy and straight-forward as possible:

1. Download the Phoronix Test Suite from **Phoronix-Test-Suite on 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 2021, around 100 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 stockfish** 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.

Phoronix Test Suite v10.8.5

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
Gentoo
Linux Embedded Development Environment
Linux Mint
Mac OS X
MacPorts
Mageia
Mandriva
Microsoft Windows
MidnightBSD
NetBSD
OpenBSD
OpenIndiana
OpenMandriva
OpenMandrivaLinux
OpenSolaris
Optware
Oracle Server
PCLinuxOS
Pardus Linux
Red Hat Enterprise
Red Hat Enterprise Server

Phoronix Test Suite v10.8.5

Test Client Documentation

SUSE Enterprise Linux
SUSE Linux
Scientific
ScientificSL
Solus
Solus Linux
Termux
Ubuntu
Void Linux
Zenwalk
macOS Brew
openSUSE
openSUSE Leap
openSUSE Tumbleweed

Phoronix Test Suite v10.8.5

Test Client Documentation

Configuration

User Files & Folders

These files/folders are the default locations when running as a non-root Phoronix Test Suite user. When running as root, the paths may appear in standard system paths like */etc/phoronix-test-suite.xml*.

~/.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. When run as root, this path is */etc/phoronix-test-suite.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.json* 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/

Phoronix Test Suite v10.8.5

Test Client Documentation

This is the directory where test profiles are stored.

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

This is the directory where test suites are stored.

Phoronix Test Suite v10.8.5

Test Client Documentation

Main Configuration File

The main configuration file is *user-config.xml* (located at *~/.phoronix-test-suite/user-config.xml* or */etc/phoronix-test-suite.xml* when running as root/admin) contains the user configuration options for the Phoronix Test Suite. To edit any option, open the configuration file within your preferred text editor. Alternatively, you can use the *user-config-set* option with the Phoronix Test Suite to update settings. For example, to set the download cache with the Phoronix Test Suite, execute *phoronix-test-suite user-config-set CacheDirectory=~/cache-directory/*.

OpenBenchmarking Options

AnonymousUsageReporting

If this option is set to *TRUE*, anonymous usage information and statistics, like the tests that are run and their length of run, will be reported to **OpenBenchmarking.org** for analytical reasons. All submitted information is kept anonymous. For more information on the anonymous usage reporting, read the Phoronix Test Suite documentation.

IndexCacheTTL

The time to live for OpenBenchmarking.org index caches. This is an integer representing the number of days before an index cache should be automatically refreshed from OpenBenchmarking.org. The default value is *3* while setting the value to *0* will disable automatic refreshing of caches (caches can be manually updated at anytime using the respective command).

AlwaysUploadSystemLogs

If this option is set to *TRUE*, the system logs (i.e. dmesg, lspci, lsusb, Xorg.0.log) will always be uploaded to OpenBenchmarking.org when uploading your test results. Otherwise the user is prompted whether to attach the system logs with their results.

AllowResultUploadsToOpenBenchmarking

This option defines whether to allow/support result uploads to OpenBenchmarking.org. If set to *FALSE*, the user will not be prompted to allow uploading of test results to the public site.

General Options

DefaultBrowser

The Phoronix Test Suite will automatically attempt to launch the system's default web browser when needed. This is done first by checking for x-www-browser and then xdg-open. If neither command is available, the Phoronix Test Suite will fallback to checking for Firefox, Epiphany, Mozilla, or the open

command. If you wish to override the default browser that the Phoronix Test Suite selects, set this tag to the command name of the browser you wish to use. Leaving this tag empty will have the Phoronix Test Suite determine the default web browser.

UsePhodeviCache

If this option is set to *TRUE*, the Phoronix Test Suite will use the Phodevi smart cache (if available). The Phodevi smart cache will automatically cache relevant system hardware/software attributes that can be safely stored and will be used until the system's software/hardware has changed or the system rebooted. Enabling this option will speed up the detection of installed hardware and software through the Phoronix Test Suite. If this option is set to *FALSE*, Phodevi will not generate a smart cache. The default value is *TRUE*.

DefaultDisplayMode

This option affects how text is displayed on the command-line interface during the testing process. If this option is set to *DEFAULT*, the text interface will be the traditional Phoronix Test Suite output. If this option is set to *CONCISE*, the display mode is shorter and more concise. This is the default mode used during batch testing. The default value is *DEFAULT*.

PhoromaticServers

This option can be used to specify the IP address(es) and port(s) of any Phoromatic Servers you wish to connect to for obtaining cached data, connecting to Phoromatic as a client test system, etc. The Phoronix Test Suite will attempt zero-conf network discovery but if that fails you can add the *IP:port* (the Phoromatic Server's HTTP port) to this element for targeted probing by the Phoronix Test Suite. Multiple Phoromatic Servers can be added if delimited by a comma; e.g. *IP:port,IP:port, IP:port*.

Modules Options

AutoLoadModules

This tag contains a string of the names of the Phoronix Test Suite modules to load by default when running the Phoronix Test Suite. Multiple modules can be listed when delimited by a comma. Modules that load via setting an environment variable can also be specified here (i.e. *FORCE_AA=8* as an option in this string to load the *graphics_override* module with the 8x forced anti-aliasing). The default value is *toggle_screensaver, update_checker*.

Installation Options

RemoveDownloadFiles

If this option is set to *TRUE*, once a test has been installed the downloaded files will be removed. Enabling this option will conserve disk space and in nearly all circumstances will not result in any problems. However, if a test profile directly depends upon a file that was downloaded (as opposed to

something extracted from a downloaded file during the installation process), enabling this option will cause issues. If this option is set to *FALSE*, the downloaded files will not be removed unless the test is uninstalled. The default value is *FALSE*.

SearchMediaForCache

If this option is set to *TRUE*, when installing a test it will automatically look for a Phoronix Test Suite download cache on removable media that is attached and mounted on the system. On the Linux operating system, the Phoronix Test Suite looks for devices mounted within the */media/* or */Volumes/* directories. If a download cache is found (a *download-cache/* folder within the drive's root directory) and a file it is looking for with matching MD5/SHA256 check-sum, the file will be automatically copied. Otherwise the standard download cache is checked. If this option is set to *FALSE*, removable media devices are not checked. The default value is *TRUE*.

SymLinkFilesFromCache

If this option is set to *TRUE*, during the test installation process when a file is found in a Phoronix Test Suite download cache, instead of copying the file just provide a symbolic link to the file. Enabling this option will conserve disk space and in nearly all circumstances will not result in any issues, permitting the download cache files are always mounted during testing and are not located on removable media. If this option is set to *FALSE*, the files will be copied from the download cache. The default value is *FALSE*.

PromptForDownloadMirror

If this option is set to *TRUE*, when downloading a test file the user will be prompted to select a mirror when multiple mirrors available. This option is targeted for those in remote regions or where their download speed may be greatly affected depending upon the server. If this option is set to *FALSE*, the Phoronix Test Suite will randomly pick a mirror. The default value is *FALSE*.

EnvironmentDirectory

This option sets the directory where tests will be installed to by the Phoronix Test Suite. The full path to the directory on the local file-system should be specified, though ~ is a valid character for denoting the user's home directory. The default value is *~/.phoronix-test-suite/installed-tests/*.

CacheDirectory

This option sets the directory for the main download cache. The download cache is checked when installing a test while attempting to locate a needed test file. If the file is found in the download cache, it will not be downloaded from there instead of an Internet mirror. When running *phoronix-test-suite make-download-cache*, files are automatically copied to this directory. The full path to the directory should be specified, though ~ is a valid character for denoting the user's home directory. Specifying an HTTP or FTP URL is valid. The default value is *~/.phoronix-test-suite/download-cache/*. Multiple cache directories can be specified as of Phoronix Test Suite 2.2 with each directory being delimited by a

colon.

Testing Options

SleepTimeBetweenTests

This option sets the time (in seconds) to sleep between running tests. The default value is *6*.

SaveSystemLogs

If this option is set to *TRUE*, when saving the results from a test it will also save various system details and logs to a sub-directory of the result file's location. Among the logs that will be archived include the X.Org log, dmesg, and lspci outputs. These system details may also be saved if a test suite explicitly requests this information be saved. If this option is set to *FALSE*, the system details / logs will not be saved by default. The default value is *FALSE*. When running in batch mode or using a Phoronix Certification and Qualification Suite, the logs will be saved regardless of this user setting.

SaveInstallationLogs

If this option is set to *TRUE*, when saving the results from a test it will archive the complete output generated by the test during its earlier installation process. The log(s) are then saved to a sub-directory of the result file's location. If this option is set to *FALSE*, the full test logs will not be saved. The default value is *FALSE*. When running in batch mode or using a Phoronix Certification and Qualification Suite, the logs will be saved regardless of this user setting.

RemoveTestInstallOnCompletion

If this option is set to *TRUE*, after a test has been completed, if that test profile is no longer present later in the test queue, the test installation will be removed from the disk. If the test is to be run at a later time, it will need to be re-installed. This is useful for embedded environments or Live CD/DVDs where the available memory (RAM) for storage may be limited.

SaveTestLogs

If this option is set to *TRUE*, when saving the results from a test it will archive the complete output of each test's run generated by the application under test itself. The default value is *FALSE*.

ResultsDirectory

This option sets the directory where test results will be saved by the Phoronix Test Suite. The full path to the directory on the local file-system should be specified, though ~ is a valid character for denoting the user's home directory. The default value is *~/.phoronix-test-suite/test-results/*.

AlwaysUploadResultsToOpenBenchmarking

This option defines whether test results should always be uploaded to OpenBenchmarking.org upon their completion. If this value is set to *FALSE*, the user will be prompted each time whether the results should be uploaded to OpenBenchmarking.org, unless running in batch mode where the value is pre-defined. The default value is *FALSE*.

AutoSortRunQueue

This option defines whether the Phoronix Test Suite should sort the queue of tests to run based upon their title and category of tests. If *FALSE*, the run queue won't be sorted and they will be run in the order they were added.

ShowPostRunStatistics

If *TRUE*, the Phoronix Test Suite will show various test run statistics / comparison data based upon the test results / result file being tested after the testing has finished.

TestResultValidation Options

DynamicRunCount

If this option is set to *TRUE*, the Phoronix Test Suite will automatically increase the number of times a test is to be run if the standard deviation of the test results exceeds a predefined threshold. This option is set to *TRUE* by default and is designed to ensure the statistical signifiance of the test results. The run count will increase until the standard deviation falls below the threshold or when the total number of run counts exceeds twice the amount that is set to run by default from the given test profile. Under certain conditions the run count may also increase further.

LimitDynamicToTestLength

If *DynamicRunCount* is set to *TRUE*, this option sets a limit on the maximum length per trial run that a test can execute (in minutes) for the run count to be adjusted. This option is to prevent tests that take a very long amount of time to run from consuming too much time. By default this value is set to *20* minutes.

StandardDeviationThreshold

This option defines the overall standard deviation threshold (as a percent) for the Phoronix Test Suite to dynamically increase the run count of a test if this limit is exceeded. The default value is *3.50*.

ExportResultsTo

This option can specify a file (either the absolute path or relative if contained within *~/.phoronix-test-suite/* where a set of test results will be passed as the first argument as a string with each of the test results being delimited by a colon. If the executed script returns an exit status of *0* the results are considered valid, if the script returns an exit status of *1* the Phoronix Test Suite will request

the test be run again.

ResultViewer Options

WebPort

The default HTTP web port to use for launching the web-based result viewer. If the value is set to *RANDOM*, a random open web port will be used.

LimitAccessToLocalHost

If this value is set to *TRUE* (default), the web-based result viewer is only accessible by the local host. If the value is *FALSE*, anyone with access to the IP/port can access the result viewer.

AccessKey

An access key / password can be optionally supplied as a basic precaution particularly for web-accessible result viewers that aren't limited to the local host. Set the string value here of the desired key/password that the user will be prompted to enter when trying to access the result viewer.

AllowSavingResultChanges

This allows saving result file changes (notes, modifying result files, etc) of result files from the web-based result viewer. Besides needing to be set to *TRUE*, the result file directory must also be write-enabled.

AllowDeletingResults

This option is similar to *AllowSavingResultChanges* but controls the behavior of whether results can be permanently removed. Besides needing to be set to *TRUE*, the result file directory must also be write-enabled.

Batch Mode Options

The batch mode options are only used when using either the *batch-run* or *batch-benchmark* options with the Phoronix Test Suite. This mode is designed to fully automate the operation of the Phoronix Test Suite except for areas where the user would like to be prompted. To configure the batch mode options, it is recommended to run *phoronix-test-suite batch-setup* instead of modifying these values by hand.

SaveResults

If this option is set to *TRUE*, when running in batch mode the test results will be automatically saved.

OpenBrowser

If this option is set to *TRUE*, when running in batch mode the web-browser will automatically open when displaying test results. If this option is set to *FALSE*, the web-browser will not be opened.

UploadResults

If this option is set to *TRUE*, when running in batch mode the test results will be automatically uploaded to **OpenBenchmarking.org**.

PromptForTestIdentifier

If this option is set to *TRUE*, when running in batch mode the user will be prompted to enter a test identifier. If this option is set to *FALSE*, a test identifier will be automatically generated.

PromptForTestDescription

If this option is set to *TRUE*, when running in batch mode the user will be prompted to enter a test description. If this option is set to *FALSE*, the default test description will be used.

PromptSaveName

If this option is set to *TRUE*, when running in batch mode the user will be prompted to enter a test name. If this option is set to *FALSE*, a test name will be automatically generated.

Networking Options

NoInternetCommunication

If you wish to disable Internet communication within the Phoronix Test Suite by default, set this option to *TRUE*. The default value is *FALSE*. Setting this to *FALSE* will still allow Phoromatic to communicate with network servers such as for intranet-based download caches or a Phoromatic Server. Internet support is generally required for downloading test profiles from OpenBenchmarking.org, acquiring necessary test files from their respective sources, etc.

NoNetworkCommunication

If you wish to disable network support (including Internet access) entirely within the Phoronix Test Suite, set this option to *TRUE*. The default value is *FALSE*.

Timeout

This is the read timeout (in seconds) for network connections. The default value is *20*.

ProxyAddress

If you wish to use a HTTP proxy server to allow the Phoronix Test Suite to communicate with

OpenBenchmarking.org and other web services, enter the IP address / server name of the proxy server in this tag. If the proxy address and port tags are left empty but the *http_proxy* environment variable is set, the Phoronix Test Suite will attempt to use that as the proxy information.

ProxyPort

If using a proxy server, enter the TCP port in this tag.

Server Options

RemoteAccessPort

If you wish to allow remote access to the built-in web-based interface to the Phoronix Test Suite when running its built-in web server, set the port number for remote access here. Port 80 is the common HTTP port but the Phoronix Test Suite web-interface can be easily set to other port numbers. If you do not wish to allow remote access, use the default value of *FALSE* or *-1*. If the value is set to *RANDOM*, a random port number will be chosen.

Password

If you wish to require a password when entering the web-based interface to the Phoronix Test Suite -- either locally or remotely -- specify the password here using the password's SHA256 sum as the value.

WebSocketPort

The default port to use when running a WebSocket server. If no port is assigned or *RANDOM* is set, a random port will be chosen.

AdvertiseServiceZeroConf

If this option is set to *TRUE* when starting a Phoromatic Server instance, the software will attempt to broadcast its service using zeroconf networking (Avahi on Linux assuming *avahi-publish* is present).

AdvertiseServiceOpenBenchmarkRelay

If this option is set to *TRUE* when starting a Phoromatic Server instance, the software will broadcast the local IP/port of the server to a private OpenBenchmarking.org service so that if any other user on the local IP block from the same global IP address is in search of a Phoromatic Server, the IP address will be relayed. This is an alternative or complementary to the zero-conf/Avahi option above to help systems running the Phoronix Test Suite client on a LAN discover a Phoromatic Server for easy setup and/or download cache support for faster test setup/installation.

PhoromaticStorage

The location for the Phoromatic Server to store test results of connected systems, account information,

etc. The default location is *~/.phoronix-test-suite/phoromatic/*.

Phoronix Test Suite v10.8.5

Test Client Documentation

Environment Variables

DONT_BALANCE_TESTS_FOR_SUBSYSTEMS

If this value is true, the Phoronix Test Suite stress-run manager will not attempt to distribute the selected test(s) among available hardware subsystems. For stress runs with tests covering multiple subsystems (e.g. CPU, GPU, RAM), the default behavior is try to ensure the tests to run concurrently are as balanced across the tested subsystems as possible.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: stress-run mode.

DONT_TRY_TO_ENSURE_TESTS_ARE_UNIQUE

When running in the stress-run mode, the default behavior will try to ensure when tests are running concurrently that as many unique tests as possible are being run. Setting this value to try will avoid that check and just attempt to truly randomize the tests being run concurrently without regard for trying to avoid duplicates.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: stress-run mode.

FORCE_ABSOLUTE_MIN_TIMES_TO_RUN

*This option is similar to FORCE_MIN_TIMES_TO_RUN but is *absolute* in ensuring each test will run at least that number of times and not subject to change of any timed cut-offs or other factors.*

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.

FORCE_MIN_DURATION_PER_TEST

This option can be used to specify the minimum number of times to run a given benchmark. Rather than relying on a static times-to-run count, the test will keep looping until the time has exceeded this number (in minutes).

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.

Phoronix Test Suite v10.8.5

Test Client Documentation

FORCE_MIN_TIMES_TO_RUN

This option is similar to FORCE_TIMES_TO_RUN but is used for specifying the minimum possible number of times to run. Unlike FORCE_TIMES_TO_RUN, the run count can still exceed this value if the deviation between results or other factors are too high.

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.

FORCE_MIN_TIMES_TO_RUN_CUTOFF

Used in conjunction with the FORCE_MIN_TIMES_TO_RUN, the FORCE_MIN_TIMES_TO_RUN_CUTOFF can be used for specifyingg the amount of time (in minutes) before foregoing additional runs. This allows cutting off the testing early if this time threshold has been reached.

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.

FORCE_TIMES_TO_RUN

This option can be used to override the default number of times a given test is run. Rather than being specified by the individual test profile, FORCE_TIMES_TO_RUN allows for specifying the number of times to run each benchmark.

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.

FORCE_TIMES_TO_RUN_MULTIPLE

This option is similar to FORCE_TIMES_TO_RUN but the value is a multiple for how many times the test profile should be run respective to its default value. If the value is set to 2 and a given test profile by default is set to run 3 times, it would now instead be run a total of 6 times. This can be used for increasing the statistical significance of test results by using a multiple of the default rather than a static number as is the case with FORCE_TIMES_TO_RUN.

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.

GRAPH_HIGHLIGHT

If automatically generating an HTML or PDF result file from the command-line and wanting to highlight desired result identifier(s), GRAPH_HIGHLIGHT can be set to a comma delimited list of result identifiers to highlight / color differently than the rest.

The value can be of type: string.
The variable is relevant for: result output generation.

IGNORE_RUNS

This option can be used if wanting the Phoronix Test Suite to automatically toss out a specified result position when running a test profile multiple times. E.g. setting this value to 1 will toss out automatically the first run of each test profile or a value of 3 will toss out the third run of a given test. This overrides the IgnoreRuns option also available to individual test profiles. Multiple values for runs to ignore can be specified by delimiting with a comma.

The value can be of type: string.
The variable is relevant for: test execution / benchmarking.

LIMIT_ELAPSED_TEST_TIME

This option can be used for limiting the amount of time the benchmarking process runs. The value specified is the number of minutes to allow for benchmarking. After a test finishes if that number of minutes has been exceeded, the testing process will abort early and not run any remaining tests.

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.

LINUX_PERF

This option allows providing additional complementary per-test graphs looking at various Linux perf subsystem metrics such as cache usage, instructions executed, and other metrics. This requires you to have Linux's perf user-space utility already installed and performance counter access.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: test execution / benchmarking.
The variable depends upon functionality provided by the Phoronix Test Suite module: linux_perf.

LOG_CLI_OUTPUT

[EXPERIMENTAL] When this option is enabled, the Phoronix Test Suite standard output from the terminal will be logged to any relevant Phoronix Test Suite / Phoromatic log file. This is mainly useful for

debugging purposes and if wishing to always archive the standard output as part of Phoronix Test Suite logs.

The value can be of type: boolean (TRUE / FALSE).

MONITOR

This option can be used for system sensor monitoring during test execution. The Phoronix Test Suite system_monitor module can monitor various exposed sensors and record them as part of the result file and present them as additional graphs / metrics in the result viewer. The exposed sensors varies by platform hardware/software. This functionality also requires PHP PCNTL support and thus is not available for some platforms (i.e. Windows).

The value can be of type: enumeration (all, cpu.peak-freq, cpu.temp, cpu.power, cpu.usage, gpu.freq, gpu.power, gpu.temp, hdd.temp, memory.usage, swap.usage, sys.power, sys.temp)
Multiple options can be supplied when delimited by a comma..
The variable is relevant for: test execution / benchmarking.
The variable depends upon functionality provided by the Phoronix Test Suite module: system_monitor.

NO_COLOR

This option when enabled will force-disable the CLI/TUI text coloring. By default the Phoronix Test Suite will attempt to use CLI/TUI text colors and bolding of text for supported terminals.

The value can be of type: boolean (TRUE / FALSE).

NO_COMPILER_MASK

By default the Phoronix Test Suite attempts to determine the intended system code compilers (namely C / C++ / Fortran) and to intercept the arguments being passed to them during test installation in order to record the prominent compiler flags being used. If this behavior causes problems for your system, NO_COMPILER_MASK can be enabled for debugging purposes to avoid this compiler intercepting/symlinking behavior.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: test installation.

NO_DOWNLOAD_CACHE

Enable this option if the Phoronix Test Suite should not attempt to discover and use any local/remote Phoronix Test Suite download cache when installing tests and attempting to find those files locally or on

a LAN resource.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: test installation.

NO_EXTERNAL_DEPENDENCIES

Enabling this option will have the Phoronix Test Suite skip over attempting to detect and install any system/external dependencies needed to run desired test profiles. This should just be used in case of testing/evaluation purposes and may leave some tests unable to successfully build/install.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: test installation.

NO_FILE_HASH_CHECKS

Enable this option if you want to skip the MD5 / SHA256 file hash checks after downloading files with known MD5/SHA256 hashsums for verification. This is namely useful for select debugging scenarios and other situations where a file may have been trivially changed / re-packaged and wishing to still install a test even though the hash no longer matches until the test profile has been updated.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: test installation.

NO_HTTPS

Enable this option if wanting the Phoronix Test Suite when downloading resources to attempt to only use HTTP without any HTTPS connections. Note: some downloads may fail for servers that only support HTTPS.

The value can be of type: boolean (TRUE / FALSE).

NO_PHODEVI_CACHE

This option will disable use of the built-in Phodevi (Phoronix Device Interface) cache of system software/hardware details. When enabled, the information is not cached and will be re-computed on each query. This is mainly useful for debugging purposes.

The value can be of type: boolean (TRUE / FALSE).

Phoronix Test Suite v10.8.5

Test Client Documentation

OUTPUT_DIR

When exporting a result file, this option can be used for specifying the writable directory path where the exported result files should be saved to. The file-name will be automatically generated.

The value can be of type: string.
The variable is relevant for: result output generation.

OUTPUT_FILE

When exporting a result file, this option can be used for specifying the file name / file path and name of where to save the exported result file to rather than assuming the user home directory.

The value can be of type: string.
The variable is relevant for: result output generation.

PHODEVI_SANITIZE

This option can be used for stripping out part of a string on Phodevi (Phoronix Device Interface) hardware/software properties. Namely around the reported hardware/software information in result files if wanting any values / portions of strings stripped out from that information, such as for confidential hardware strings or other privacy concerns, PHODEVI_SANITIZE can be set. The value will be removed from read Phodevi hardware/software properties if set. Multiple strings to search for can be set by delimiting with a comma. If wanting to limit the sanitization to a particular property, the property value can be specified such as [property]=[value] to sanitisze like a value of "motherboard=ABCVENDOR" or CPU=ENGINEERING-SAMPLE to delete those strings rather than simply the string to remove that will look for matches in any property."

The value can be of type: string.

PRESET_OPTIONS

PRESET_OPTIONS can be used for seeding the values of test profile run options from the environment (though the preferred approach for pre-configuring tests in an automated manner would be by constructing your own local test suite). 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".

The value can be of type: string.
The variable is relevant for: test execution / benchmarking.

Phoronix Test Suite v10.8.5

Test Client Documentation

PRESET_OPTIONS_VALUES

This option is similar to PRESET_OPTIONS and uses the same syntax but rather than seeding the selected run option it uses the value verbatim as for what is passed to the test profile run option.

The value can be of type: string.
The variable is relevant for: test execution / benchmarking.

PTS_CONCURRENT_TEST_RUNS

This option is used in the stress run/benchmarking mode to indicate the number of tests to run concurrently as part of the stress run process.

The value can be of type: positive integer.
The variable is relevant for: stress-run mode.

PTS_DISPLAY_MODE

If you wish to load a non-default display mode for a single instance, specify the mode in this variable as an alternative to adjusting the user configuration file.

The value can be of type: enumeration (BASIC, BATCH, CONCISE, SHORT, DEFAULT).

PTS_DOWNLOAD_CACHE

PTS_DOWNLOAD_CACHE can be used for setting a path to a directory on the system containing a Phoronix Test Suite download cache if located outside one of the default locations.

The value can be of type: string.
The variable is relevant for: test installation.

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, this option 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 files should be captured from following test completion.

The value can be of type: string.

The variable is relevant for: test execution / benchmarking.

PTS_IGNORE_MODULES

Enabling this option can be used for temporarily disabling Phoronix Test Suite modules from being loaded on a given run. This is primarily for debugging purposes.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: modules.

PTS_MODULES

This option can be used for specifying a comma-separated list of Phoronix Test Suite modules to load at start-time, complementary to the modules specified in the user configuration file. PTS_MODULES is namely used for development purposes or wanting to temporarily enable a given module.

The value can be of type: string.
The variable is relevant for: modules.

PTS_MODULE_SETUP

This option can be used for seeding a module's settings when running the phoronix-test-suite module-setup command. 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".

The value can be of type: string.
The variable is relevant for: modules.

PTS_SILENT_MODE

This option when enabled will yield slightly less verbose Phoronix Test Suite terminal output by silencing unnecessary messages / prompts.

The value can be of type: boolean (TRUE / FALSE).

PTS_TEST_INSTALL_ROOT_PATH

This option can be used for overriding where tests are installed to on the system. An absolute writable

directory path can be the value if wanting to override the default (or user configuration file specified) test installation directory path.

The value can be of type: string.
The variable is relevant for: test installation, test execution / benchmarking, stress-run mode.

REMOVE_TESTS_OLDER_THAN

This option with the cleanup module can be used for automatically un-installing/removing installed tests if they have not been run in a period of time. The value for REMOVE_TESTS_OLDER_THAN is the number of days the test can be installed without running until this module will clean-up/remove older tests.

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.
The variable depends upon functionality provided by the Phoronix Test Suite module: cleanup.

REMOVE_TESTS_ON_COMPLETION

When this option is set to true, installed test profiles will be automatically removed/uninstalled when they are no longer in the current test execution queue. This is used for saving disk space / resources by automatically removing installed tests after they have been executed. For more persistent behavior is the RemoveTestInstallOnCompletion option within the Phoronix Test Suite user configuration file.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: test execution / benchmarking.

SKIP_EXTERNAL_DEPENDENCIES

Rather than NO_EXTERNAL_DEPENDENCIES to outright disable the Phoronix Test Suite external dependency handling, SKIP_EXTERNAL_DEPENDENCIES can be used with a value of a comma separated list of specific external dependencies to avoid. This is mostly useful for any external dependencies that may be out of date or fail to install on your platform.

The value can be of type: string.
The variable is relevant for: test installation.

SKIP_TESTING_SUBSYSTEMS

This option is similar to SKIP_TESTS but allows for specifying hardware subsystems (e.g. Graphics) to skip from installing/running any test profiles beloning to that subsystem type. Multiple subsystems can

be specified when delimited by a comma.

The value can be of type: string.
The variable is relevant for: test installation, test execution / benchmarking.

SKIP_TESTS

SKIP_TESTS will skip the test installation and execution of any test identifiers specified by this option. Multiple test identifiers can be specified, delimited by a comma.

The value can be of type: string.
The variable is relevant for: test installation, test execution / benchmarking.

SKIP_TESTS_HAVING_ARGS

SKIP_TESTS_HAVING_ARGS will skip the test installation and execution of any tests where the specified test arguments match the given string. E.g. if wanting to skip all Vulkan tests in a result file but run just the OpenGL tests or similar where wanting to limit the tests being run from within a result file. Multiple values can be specified when delimited by a comma.

The value can be of type: string.
The variable is relevant for: test installation, test execution / benchmarking.

SKIP_TEST_SUPPORT_CHECKS

This debugging/validation option will have the Phoronix Test Suite skip any test support checks for a test profile (architecture compatibility, OS compatibility, etc) and just assume all tests are supported.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: test installation, test execution / benchmarking.

SORT_BY

This option can be used for specifying the sort order for commands like auto-sort-result-file whether to sort by identifier name, test length, etc.

Default Value: identifier

The value can be of type: enumeration (date, date-asc, date-desc, identifier).

Phoronix Test Suite v10.8.5

Test Client Documentation

TERMINAL_WIDTH

This option is used for overriding the detected default of the terminal width for the CLI/TUI interface.

The value can be of type: positive integer.

TEST_EXECUTION_SORT

This option can be used for controlling the sort order that the test profiles / benchmarks are run in, whether sorted or not and in what manner.

The value can be of type: enumeration (none, random, dependencies, test-estimated-time, test-estimated-time-desc, test, default).
The variable is relevant for: test execution / benchmarking.

TEST_EXEC_PREPEND

This option can be used if wanting to specify a binary (e.g. sudo, cgroup or other resource limiting binaries or performance counters) to be called as the binary pre-pended prior to running a test profile binary/script. This option is namely used for specialized use-cases.

The value can be of type: string.
The variable is relevant for: test execution / benchmarking.

TEST_RESULTS_DESCRIPTION

This option can be used for specifying the result file description for saving that string and not be prompted for providing a description during the test execution process.

The value can be of type: string.
The variable is relevant for: test execution / benchmarking, stress-run mode.

TEST_RESULTS_IDENTIFIER

This option can be used for specifying the result identifier for distinguishing this run within the saved result file.

The value can be of type: string.
The variable is relevant for: test execution / benchmarking, stress-run mode.

Phoronix Test Suite v10.8.5

Test Client Documentation

TEST_RESULTS_NAME

This option can be used for specifying the result file name for saving the test/benchmark results automatically to the given name.

The value can be of type: string.
The variable is relevant for: test execution / benchmarking, stress-run mode.

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. This functionality requires system PHP PCNTL support (i.e. no Windows support).

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.
The variable depends upon functionality provided by the Phoronix Test Suite module: test_timeout.

TOTAL_LOOP_COUNT

This option is used to specify a multiple if wishing to run each test multiple times rather than just once per saved result file.

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.

TOTAL_LOOP_TIME

This option is used to specify the amount of time (in minutes) to loop the testing during the Phoronix Test Suite stress run or normal benchmarking process.

The value can be of type: positive integer.
The variable is relevant for: stress-run mode, test execution / benchmarking.

TURBOSTAT_LOG

This option allows attaching "turbostat" outputs to the end of archived benchmark/test log files if

interested in the Linux TurboStat information. This assumes you have turbostat available on the Linux system(s) and have permissions (root) for running turbostat.

The value can be of type: boolean (TRUE / FALSE).
The variable is relevant for: test execution / benchmarking.
The variable depends upon functionality provided by the Phoronix Test Suite module: turbostat.

WATCHDOG_MAXIMUM_WAIT

Used in conjunction with the WATCHDOG_SENSOR option, this is the maximum amount of time to potentially wait when the watchdog is triggered for surpassing the threshold value. The value is the maximum number of minutes to wait being above the threshold.

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.
The variable depends upon functionality provided by the Phoronix Test Suite module: watchdog.

WATCHDOG_SENSOR

This option will enable the watchdog module that checks system sensor values pre/interim/post benchmark execution. If the selected sensor(s) exceed the static threshold level, testing will be paused before continuing to any additional tests so that the system can sleep. Ideally this will allow the system to return to a more suitable state before resuming testing after the sensor value is back below the threshold or after a pre-defined maximum time limit to spend sleeping. This module is mostly focused on pausing testing should system core temperatures become too elevated to allow time for heat dissipation.

The value can be of type: enumeration (cpu.temp, gpu.temp, hdd.temp, sys.temp)
Multiple options can be supplied when delimited by a comma..
The variable is relevant for: test execution / benchmarking.
The variable depends upon functionality provided by the Phoronix Test Suite module: watchdog.

WATCHDOG_SENSOR_THRESHOLD

Used in conjunction with the WATCHDOG_SENSOR option, the WATCHDOG_SENSOR_THRESHOLD specifies the threshold for the sensor reading when the testing should be paused (e.g. the Celsius cut-off temperature).

The value can be of type: positive integer.
The variable is relevant for: test execution / benchmarking.
The variable depends upon functionality provided by the Phoronix Test Suite module: watchdog.

Phoronix Test Suite v10.8.5

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**. Free, community support is offered via **GitHub**.

Phoronix Test Suite v10.8.5

Test Client Documentation

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.

Phoronix Test Suite v10.8.5

Test Client Documentation

- 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 v10.8.5

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.*

64-bit Arm / AArch64 Tests In pts *pts/aarch64*

This is a collection of test profiles where there have been successful benchmark results submitted to OpenBenchmarking.org from 64-bit Arm / AArch64 CPU architecture hardware, i.e. these tests are proven to be 64-bit Arm / AArch64 compatible though not necessarily all compatible test profiles for the given architecture - just those with submitted public results previously on OpenBenchmarking.org.

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*

Phoronix Test Suite v10.8.5

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 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)

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.

C/C++ Compiler Benchmark Workloads In pts *pts/compiler*

This is a collection of test profiles often useful for C/C++ compiler benchmarks and where the test profiles will respect CFLAGS/CXXFLAGS environment variables.

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*

Phoronix Test Suite v10.8.5

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 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.

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*

Phoronix Test Suite v10.8.5

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 a test of the memory sub-system.

Multi-Core/Multi-Threaded Workloads In pts *pts/multicore*

This is a collection of test profiles that have been detected to be CPU multi-threaded capable.

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

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*

Phoronix Test Suite v10.8.5

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 python.

RISC-V Tests In pts *pts/riscv*

This is a collection of test profiles where there have been successful benchmark results submitted to OpenBenchmarking.org from RISC-V CPU architecture hardware, i.e. these tests are proven to be RISC-V compatible though not necessarily all compatible test profiles for the given architecture - just those with submitted public results previously on OpenBenchmarking.org.

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.

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.

Single-Threaded Workloads In pts *pts/single-threaded*

This is a collection of test profiles that have been detected to be single-threaded or only very poorly CPU threaded.

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.

Phoronix Test Suite v10.8.5

Test Client Documentation

Steam Tests *pts/steam*

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

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.

Cuda Tests *pts/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.

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.

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.

Ai Tests *pts/ai*

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 ai.

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.

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.

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.

Video Tests *pts/video*

Phoronix Test Suite v10.8.5

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 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.

64-bit Arm / AArch64 Tests In system *system/aarch64*

This is a collection of test profiles where there have been successful benchmark results submitted to OpenBenchmarking.org from 64-bit Arm / AArch64 CPU architecture hardware, i.e. these tests are proven to be 64-bit Arm / AArch64 compatible though not necessarily all compatible test profiles for the given architecture - just those with submitted public results previously on OpenBenchmarking.org.

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

Phoronix Test Suite v10.8.5

Test Client Documentation

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*

Phoronix Test Suite v10.8.5

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 a test of the network sub-system.

OpenCV Tests *system/opencv*

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

RISC-V Tests In system *system/riscv*

This is a collection of test profiles where there have been successful benchmark results submitted to OpenBenchmarking.org from RISC-V CPU architecture hardware, i.e. these tests are proven to be RISC-V compatible though not necessarily all compatible test profiles for the given architecture - just those with submitted public results previously on OpenBenchmarking.org.

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*

Phoronix Test Suite v10.8.5

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 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*

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.

64-bit Arm / AArch64 Tests In git *git/aarch64*

This is a collection of test profiles where there have been successful benchmark results submitted to OpenBenchmarking.org from 64-bit Arm / AArch64 CPU architecture hardware, i.e. these tests are proven to be 64-bit Arm / AArch64 compatible though not necessarily all compatible test profiles for the given architecture - just those with submitted public results previously on OpenBenchmarking.org.

All Tests in git *git/all*

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

repository.

Application Tests *git/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.

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.

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

System Subsystem Tests *git/system*

Phoronix Test Suite v10.8.5

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 a test of the system sub-system.

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 v10.8.5

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.

The Phoronix Test Suite utilizes a compiler masking method 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 v10.8.5

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 v10.8.5

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.

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 / Offline Testing

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

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'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 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 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.

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:

Phoronix Test Suite v10.8.5

Test Client Documentation

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/l0SSJY*. 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

Phoronix Test Suite v10.8.5

Test Client Documentation

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 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

Packaged with the Phoronix Test Suite are basic *phoromatic-client* and *phoromatic-server* configurations for systemd. 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.

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*.

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

Phoronix Test Suite v10.8.5

Test Client Documentation

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*.

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 v10.8.5

Test Client Documentation

Confidential Testing / Avoiding Accidental Result Upload

Offline Enhancement Via Local Cache

Since Phoronix Test Suite 9.0, there have been 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.

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 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 v10.8.5

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, and nye_Xml are some of the related open-source projects provided by **Phoronix Media**.