

CLIF Performance Testing Plugin

Plugin Information

View CLIF Performance Testing [on the plugin site](#) for more information.

Enables running performance tests defined with [CLIF](#) from Jenkins, and getting a variety of automatically generated performance reports and charts.

Useful for: automated performance testing, performance testing in continuous integration, providing a simple web user interface for CLIF, monitoring QoS or applications QoE and possibly send alerts in case of bad responsiveness.

[CLIF](#) is an open (source), extendible performance testing software supporting a variety of load injection protocols: TCP, UDP, HTTP(S), FTP, DNS, LDAP (S), SIP, RTP, GIT, SVN, JDBC, JMS...

CLIF also embeds a probe framework to monitor arbitrary resources available on the network (via SNMP, JMX...) or on computing nodes (CPU, RAM, disks, network...)

CLIF comes with a variety of integrations and user interfaces: command-line, simple standalone GUI, full-fledged GUI integrated to Eclipse, Maven plug-in and... Jenkins plug-in.

Thanks to this plug-in, you can import CLIF test projects (bundled scenarios, test plans and test data sets), typically written using CLIF's main GUI, as Jenkins jobs. Then, a detailed [performance report](#) is generated for each test run, including a global statistical analysis as well as graphs ([scatter chart](#), [moving statistics](#), [histograms](#) and [quantile](#) distribution analysis) about load injectors' responses times and throughput. The CLIF performance report also provides statistics and [charts about probes measurements](#).

A [performance trend chart](#) is also maintained through test executions.

Changelog

v1.0.1

- FIX: missing icons and help sections
- IMPROVEMENT: removed useless dependency with ant plug-in

Installation

Installing a CLIF runtime

Prior to running or importing any CLIF test plans, you need to install at least one CLIF runtime, typically a 2.3.3+ "CLIF server" distribution, available for download from [here](#).

Once you have unzipped this CLIF distribution, make sure you have a Java 8+ execution environment installed.

To check your CLIF installation, run this command line:

Example on Windows 7 (cmd.exe)

```
C:\> C:\path\to\clif-2.3.3-server\bin\clifcmd version
Caption=Microsoft Windows 7 Enterprise
OSArchitecture=64-bit
Name=Intel(R) Core(TM) i7-5600U CPU @ 2.60GHz
java version "1.8.0_121"
Java(TM) SE Runtime Environment (build 1.8.0_121-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.121-b13, mixed mode)
clif 2.3.3 compiled 2016-Aug-30, 15:27 GMT+0200 by xxxxxxxx
```

Example on Linux

```
$ /path/to/clif-2.3.3-server/bin/clifcmd version
Linux 4.4.0-59-generic #80-Ubuntu SMP Fri Jan 6 17:47:47 UTC 2017 x86_64
openjdk version "1.8.0_111"
OpenJDK Runtime Environment (build 1.8.0_111-8u111-b14-2ubuntu0.16.04.2-b14)
OpenJDK 64-Bit Server VM (build 25.111-b14, mixed mode)
clif 2.3.3 compiled 2016-Aug-30, 17:11 GMT+0200 by xxxxxxxx
```

In case Java can't be found, or the default Java environment is not appropriate, change your PATH environment variable so that the bin directory of the right Java runtime is found first.

Adding a CLIF runtime to Jenkins tools

Once the CLIF Performance Testing plug-in is installed in Jenkins, you can add the installed CLIF runtime to the Jenkins tools configuration page (Manage Jenkins>Global Tool Configuration).

Go to the Clif section, click on button **Clif installations...** and fill the fields:

- name (say, "CLIF 2.3.3")
- home directory (CLIF runtime's root directory, say "/path/to/clif-2.3.3-server")
- enter specific Java options to be set in CLIF's Java Virtual Machines running the tests (optional)
- Notes:
 - don't care about checkbox "Install automatically" (not implemented yet)
 - checkbox "Run with ProActive Scheduler?" is documented in [this page](#)

Click on **Save** or **Apply** buttons at the page bottom.

Note: you may declare as many CLIF runtimes as you need, including several instances of the same runtime (same CLIF home directory) but with different names and Java properties.

Creating a CLIF job

The CLIF Performance Testing plug-in provides two ways of creating a CLIF test job:

- either from a source code management system, as usual for Jenkins job;
- or by using the import wizard provided by the plug-in and available in Jenkins' main menu.

Via Source Code Management

As mentioned at the top of this page, CLIF's main GUI is based on Eclipse. CLIF scenarios, test plans and possible data sets are grouped together in a CLIF project. It is straightforward to use a source code management (Svn, CVS, Git...) client provided by Eclipse to push this project to an SCM repository.

Once your CLIF project is available on an SCM repository, create a new Freestyle project (New Item... > Freestyle Project). Go to the Source Code Management section and provide all necessary details about your project repository.

Then, go to the Build section, and add an **Invoke Clif** build step:

- choose your CLIF runtime, as previously configured in Jenkins tools
- enter your CLIF test plan file name - this file must be provided by the CLIF project that will be checked-out from the specified SCM repository
- click on **Save** or **Apply** button

Finally, you may be interested in adding a post-build action to automatically generate on each CLIF job successful completion. Refer to section below dedicated to [CLIF reports configuration](#).

Via the Import wizard

Another convenient way to proceed is to export the CLIF project as a zip archive from CLIF's Eclipse-based main GUI, and then to import it from Jenkins using wizard **Import a Clif zip**.

Once you have selected your CLIF project's zip file (beware of automatic refresh that tends to discard your file selection), the wizard makes you choose the CLIF runtime for the new job(s), as well as the test plans to import. One job will be created per imported test plan. However, note that all these jobs will share a common workspace, that will become visible as soon as you run the created jobs.

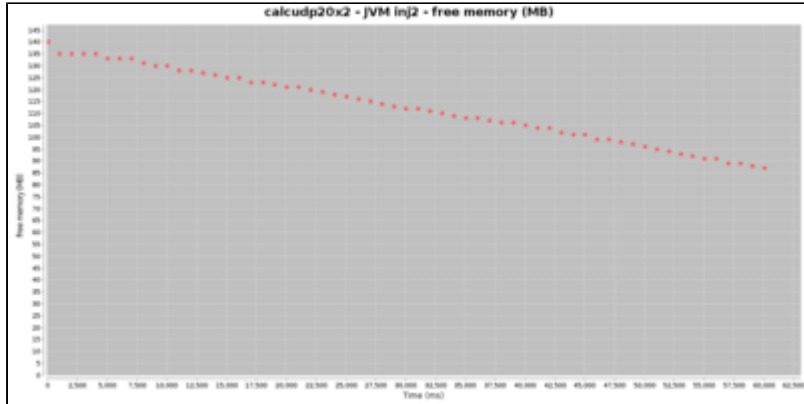
Click on **Validate** button.

Note: the import wizard also adds a Publish Clif performance report post-build action. Refer to next section to configure CLIF reports.

CLIF Reports Configuration

To get a performance report automatically generated on each successful CLIF job run, just add post-build action **Publish CLIF performance report** to the Post-build Actions section. You shall keep the default value for parameter *Clif report directory*, unless you changed it for some reason in the advanced settings of the *Invoke Clif* build action. CLIF performance reports give for each job execution:

- for each load injector, as well as for all load injectors combined:
 - an array of response times statistics for each request type, as well as average throughput and total number of errors
 - a collection of graphs about response times and throughput: scatter graph, moving statistics, distribution (histograms and quantiles)
- for each probe:
 - an array of statistics about each monitored value
 - a scatter graph for each monitored value
- for all probes and load injectors: an array of alarms



Build Performance Report

calcup20x2 - Tue Jan 24 15:33:21 CET 2017

Server - cli2

Probe - jvm (Argument=1000 60 Comments)

Label	Sampler
free memory (MB)	61
used memory %	61
free usable memory %	61

Injector - in2 - IsacRunner (Argument=calcup20.xls Comments)

Type	Requests	Success	Errors	Average	N
UDP CONNECT	20	20	0	36	
UDP RECEIVE	8427	8,265	162	1429	
computation error	883	0	883	0	
UDP SEND	8427	8,427	0	36	

Server - cli1

Probe - jvm (Argument=1000 60 Comments)

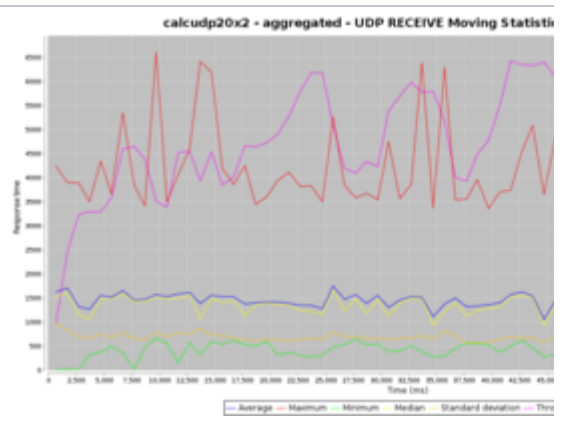
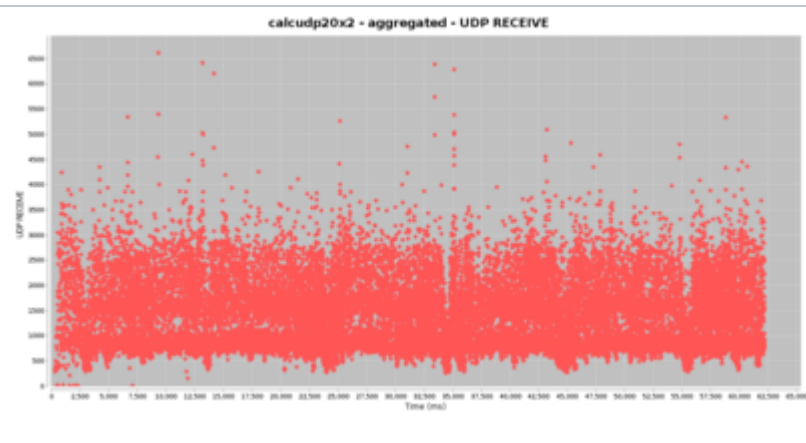
Label	Sampler
free memory (MB)	61
used memory %	61
free usable memory %	61

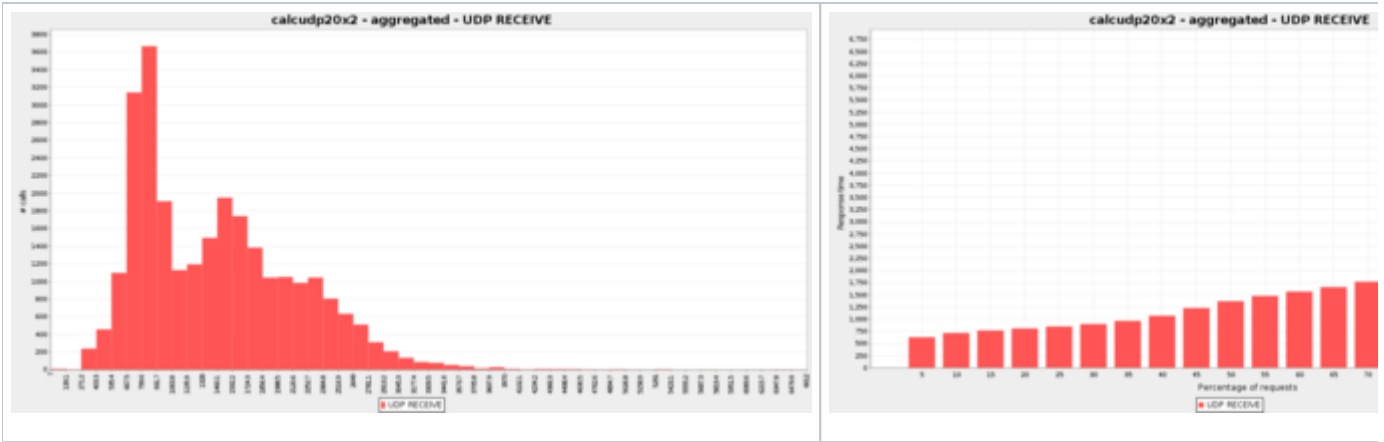
Injector - in1 - IsacRunner (Argument=calcup20.xls Comments)

Type	Requests	Success	Errors	Average	N
UDP CONNECT	20	20	0	43	
UDP RECEIVE	14750	14,652	98	1530	
computation error	1409	0	1409	0	
UDP SEND	14750	14,750	0	35	

Aggregated values

Type	Requests	Success	Errors	Average	N
UDP CONNECT	40	40	0	39	
UDP RECEIVE	23177	22,917	260	1494	
computation error	2292	0	2292	0	
UDP SEND	23177	23,177	0	35	





With the **Advanced** settings of *Publish Clif performance report* post-build action, you may set:

- aliases to requests, that make it possible to aggregate requests that would be considered of different types otherwise, as well as to get arbitrary request names in the report
- request success characterization, to override the test scenarios' success and failure specification
- a time sub-interval (to discard initial and/or final test phases)
- measurements statistical cleaning (to discard outstanding values)
- charts parameters (charts size, number or size of distributions categories, moving statistics time-window)

Beyond the detailed CLIF performance report generated for each CLIF job successful run, a Clif performance Trend report is also maintained to see the response time and error trends for each request type.

