LoadGen 1.0

1.Introduction

LoadGen is a web application which is designed to load test client/server software (such as a web application). It is developed based on fibers which is a lightweight thread implementation in Java. This implementation helps in generating very high load without consuming much resources. It can be used for HTTP, database and web service testing.

2.Download and installation instructions

- Download the latest release of LoadGen: https://github.com/stanlyjohn2/LoadGenerator and extract it. Set environment variable LOADGEN_HOME to the LoadGenerator directory.
- Download Apache maven: https://maven.apache.org/download.cgi and extract it. Set environment variable M2_HOME to the Maven directory.
- Download Apache Tomcat 8.0.23: https://archive.apache.org/dist/tomcat/tomcat-8/v8.0.23/bin/ and extract it. Set environment variable CATALINA_HOME to the tomcat home directory.
- `Download` *`comsat-tomcat-loader-0.7.0-jdk8.jar`* `https://www.1maven.com/findpomandjar/co.paralleluniverse:comsat-tomcat-loader:0.5.0 and put it in $CATALINA_HOME/lib directory.`
- Download Jdk 1.8: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html and extract it. Set environment variable JAVA_HOME to the jdk directory.

  Example: export LOADGEN_HOME=/home/stanly/LoadGenerator
  export JAVA_HOME=/usr/lib/jvm/jdk1.8.0_91
  export M2_HOME=/usr/apache/apache-maven-3.3.9
  export CATALINA_HOME=/usr/apache/apache-tomcat-8.0.23

- Set the path variables

  Example: export PATH=${PATH}:${M2_HOME}/bin:${CATALINA_HOME}/bin

- Run the following command in Loadgen home directory: `mvn -Ptomcat8x dependency:properties package cargo:run`

- Go to http://localhost:8080/LoadGen for the homepage of the loadgen.

3. Usage

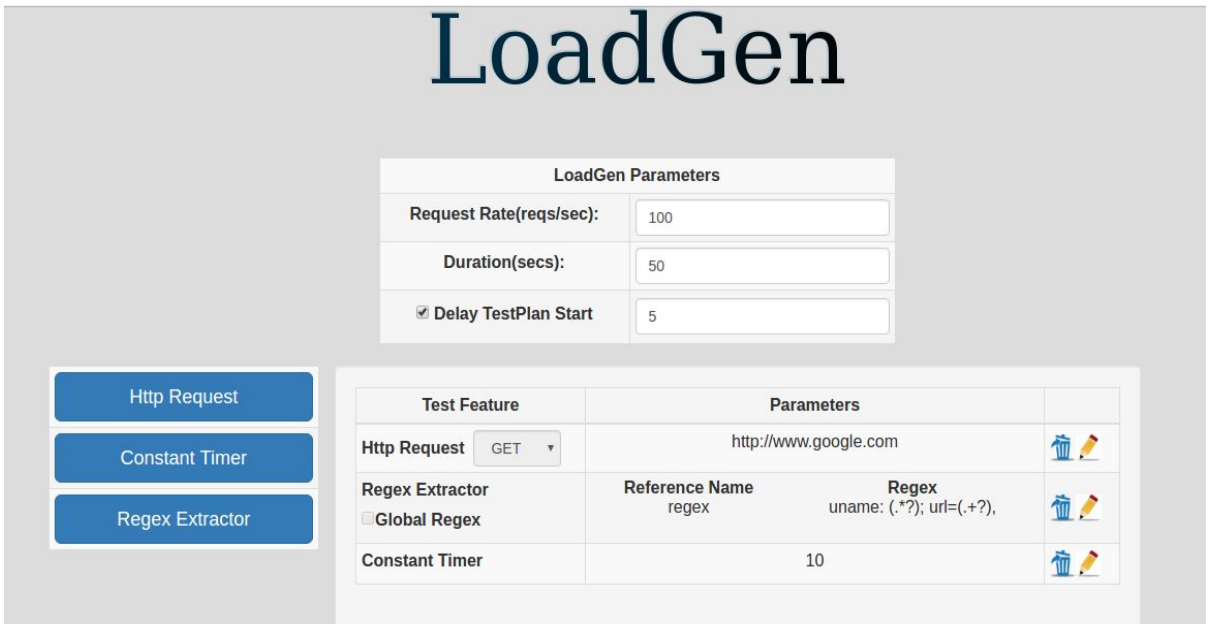
3.1. Creating a Test Plan

- Select the test plan type (Normal HTTP, Random HTTP, Database, Web Service)

  Home Page:

- Add the required parameters for the selected test.

  Sample Http Test Page:



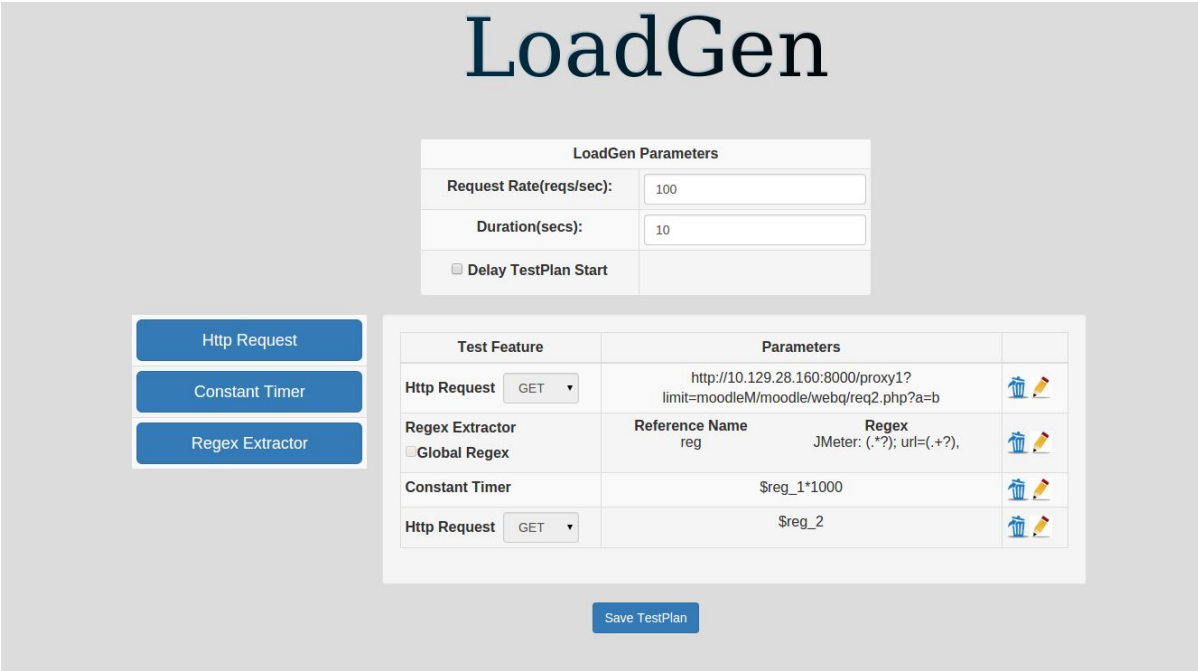
- Save the testplan using the saveplan button.
- Add as many testplans as needed and it will get executed parallely.
- Save the test for future purpose if needed.
- Start the test

3.2 Run the saved test

- Load the test xml file using the upload button

- Start the test

3.3 Using regular expression extractor and delay timer



- The above figure shows a sample usage of regex extractor. Once we start the test
- The regular expression extractor will have a reference name and a regex.
- The regex will be applied to the response of the previous HTTP request.
- The extracted content can be later used by referencing it using the reference name.

3.4 Results and Outputs

Once the test is started, test details such as throughput, response time and error rate will be displayed live in a table. Live graphs can be enabled by the graphs button.

Sample Output:

3.5 Test Summary

Once the test is finished, a *test summary* button will appear. Click on the button to download the test summary as a pdf file.