International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 05 | May 2020 www.irjet.net p-ISSN: 2395-0072

Comprehensive Review of Load Testing Tools

Siddhant Shrivastava[1], Prof. Prapulla SB[2]

[1] *Student, Department of Computer Science and Engineering, R.V. College of Engineering, Bengaluru, Karnataka, India*

[2] *Assistant Professor, Department of Computer Science and Engineering, R.V. College of Engineering, Bengaluru, Karnataka, India*

Abstract – *Load Testing is a type of non-functional testing to recognize an application's actions when it swarmed with a large number of user hits and check how well it handles traffic. This is essential to do before the site goes live because if the website is unable to manage traffic later, substantial damage and repair work will be incurred. Gatling is a Scala, Akka toolkit, and Netty based open-source framework. It is a highly capable, easy-to-use software that comes with high maintenance and performance. It comes with excellent support for the HTTP protocol, but as the core engine is protocol agnostic, support for other protocols is perfectly possible. Locust is a Python-based load testing tool, considered to be simple to use and used for distributed testing. Apache JMeter is another tool that comes as a desktop application, with a user friendly GUI. It works on plugins which have been developed by Apache Software foundation and online contributors.*

Other tools like Grinder, Loadrunner, etc. are used for load testing. All of these tools have their own benefits and disadvantages. This paper provides, in particular, an in-depth comparative analysis of Gatling, Locust, and JMeter commonly used for this purpose. Such a study may help determine an effective tool according to the requirements of an application.

Key Words: **Load Testing**, **Web applications**, **Locust**, **Gatling**, **open-source framework**.

1. INTRODUCTION

In today's internet era, so many websites and applications are being made, and one that rightly serves its customers gets great responses, and traffic to the website could increase exponentially in a day because most of us now have access to the internet due to smartphones. Last year's report stated 4.57 billion to be online in a day, which is about 60% of the world's population. A business wouldn't want to lose accounts because it couldn't handle network traffic properly. Users on the internet lose interest very easily and if they don't get what they want in two clicks or if the application takes longer than 3 seconds to respond, they cannot even return to your website.

Talking about a recent incident, Zerodha's server (an online trading platform) crashed after India's Prime Minister announced a policy. Zerodha faced consumer criticism on all social media sites. Now that Zerodha is well-established, it might not have lost many clients, but for a business not as large as Zerodha, a server crash might cost millions.

Once thousands of users enter an application's endpoints, the real architecture design check begins. If the Load testing has already be done, then there is no point of concern. Otherwise, the system might go into unexpected behaviour when its deployed and fixing these bugs can cost a lot of time and money, a situation, no company wants to run into.

Pu yunming *et. al.*[1] talks about how different types performance parameters needs to be tested before the deployment of an application. Parameters like Response time, concurrent users, throughput in loaded environment. Here they've used Loadrunner and TestDirector to perform this testing, and their results show how load testing can help Find potential bugs, analyze and locate the software defects. Draheim *et. al.*[2] presents an approach for load testing where the web sites are loaded according to the stochastic models of user behaviors. This helps create realistic models. It highlights that the preciseness of the model is important as it paints a better picture of the behavior of the application in adverse conditions. Ali A. *et. al.*[3] proposes an innovative solution to use cloud computing to include testing as a service altogether by leveraging all cloud resources and infrastructure which significantly reduces time, cost, and efforts required to finish testing. This paper proposes an architecture that was used to create a service that does testing in an automatic manner which greatly increases testing efficiency. Verma *et. al.*[4] makes an evaluation of how good Locust is as a tool for load testing. The testing was done in GCC protocol and the efficiency of the tool was found to be high, and it was mentioned as being an easy-to-use and easy-to-configure system. Jovic *et. al.*[5] talks about how to test the GUI applications for by load testing.

In this paper, a comparative review of different Load testing tools has been performed. The analysis is done on the basis of which protocols the tools support, how updated a tool is, and analyze if the tools support parameterization, assertions and give distributed testing capabilities or not paper is organized as follows. The following sub-section talks about the importance of load testing tools. Section II, the concepts and basic working of Gatling, Locust and Apache JMeter have been discussed. This is followed by presentation of a comparison between these tools in Section III. Finally, Section IV discusses some of the conclusions.

1.1 Importance of load testing tools

Other than the points put forth in the previous sections, this subsection will talk about the other advantages a load testing tools have.

- They help analyze the OSI protocol stack, and not just the GUI performance of the application. A load testing tool is capable of sending hypertexts transmitted by web browser when one clicks on a button. Same thing can be expanded for multi-user environment using load testing tools, where hypertexts for all users are sent, each having distinctive login ID and Password.
- They help measure the quality and performance of a web application.
- They help to pin point what is the cause of slow system performance. This is important to troubleshoot the problems and make website more robust. Some of the reasons that can be slowing down an application are data base server, application server, network latency, network congestion, improper load balancing, client side processing.
- They help in determining the number of users application can have with no performance degradation, which consequently helps in improving scalability of the system.
- They help cropping the cost implications that might came later due to server failure.

2. Open source Load testing tools

Multiple Open Source and Enterprise software are available online, helping with the above-mentioned system analysis. These tools help understand system behavior under peak conditions. To perform load testing, multiple users build the planned use model and run into the device with concurrent access. Enterprise solutions aren't always workable, so we're discussing open source tools in detail. The discussion will focus on three tools — Gatling, Locust, and Apache JMeter.

A. Gatling

Gatling is a free open source software that is developed and maintained mainly by Stephane Landelle. Gatling has a basic GUI which is limited to test recorder only. However, Gatling does provide a way to create test cases in readable/writeable format using domain-specific language.

Some Key features of Gatling includes:

- It's based on Scala
- It has a self-explanatory DSL which makes it easy to use and understand
- Comes with an HTTP recorder
- Powerful validation and assertion system
- Higher test loads are generated using asynchronous non-blocking approach
- Informative and clean reports

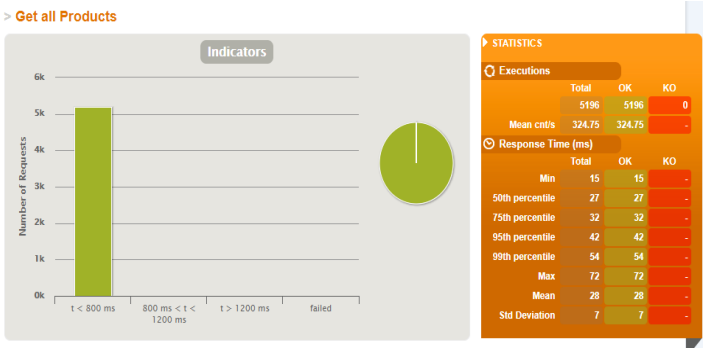

Fig. 1: Sample gatling report screenshot

B. Locust

Locust is an open source framework based purely on python. It was developed by developers, for the use by developers which means it doesn't have a GUI, and everything done in Locust is through Python scripts. The main target of Locust are web applications and web-based services. However, if one is comfortable with python, almost anything can be tested. Locust uses a different way to stimulate users, an approach based on events and gevent_coroutine as the backbone for this process. This make it easy to create users on a system with low hardware capabilities very easily, even when complex scenarios are involved.

Some key features of Locust includes:

- Cross-platform independence
- High scalability due to event based stimulation
- Assertion ability, which can be limited by knowledge of python
- Impressive web-based load monitoring
- Version control is easily done
- Ability to test almost anything given good language knowledge

C. Apache JMeter

JMeter is one of the very few desktop application based load testing tool. It has a very friendly GUI[6], making the testing and debugging much easier. JMeter has a modular structure and its core is extended from plugins. This implies that all the protocols and features of JMeter comes from plugins which have been developed by Apache Software

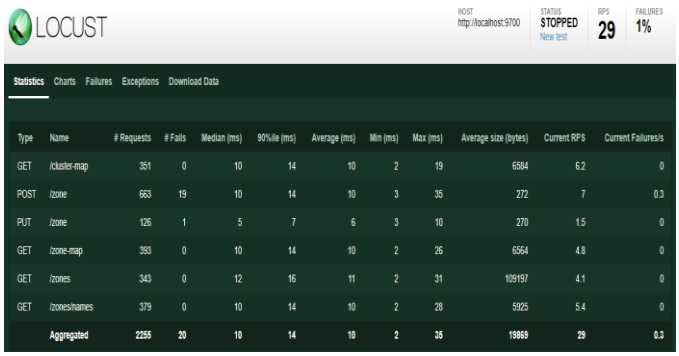

Fig. 2: Sample Locust report screenshot

foundation or community developers.

Some key features of JMeter includes:

- Cross-platform independence, it just needs Java in the system
- Highly Scalable
- Multiple protocol support- HTTP, SMTP, JDBC, FTP, SOAP, TCP, POP3, etc.
- Assertion support available
- Multiple pre-processor and post-processor implementations are done around the sampler. This provides for advanced setup, parameterization and correlation capabilities.
- Built-in external listeners to visualize and analyze performance results
- Can be integrated with major CI/CD systems which make load testing a part of continuous development cycle

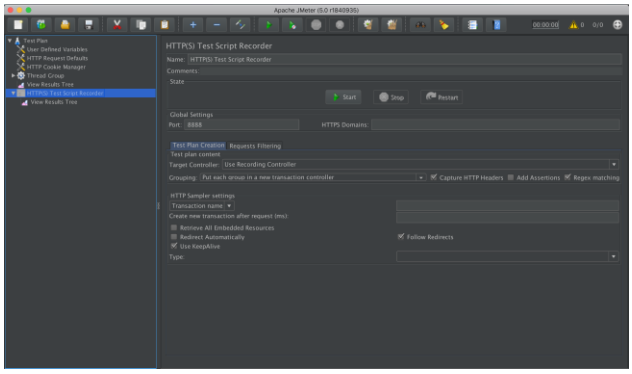

Fig. 3: Sample JMeter GUI screenshot

3.0 Comparison of Gatling, Locust and JMeter

For making comparison between the 3 tools, we are going to hit same server HTTP get request end point from 20 threads over 100000 iterations. Each tool will send out the request as fast as they can and the data for Average response time, throughput and average execution time will be noted, and consequently used for making basic comparisons. Following are few of the comparison graphs.

Fig 4: Comparison of average response time and total execution time of the 3 tools

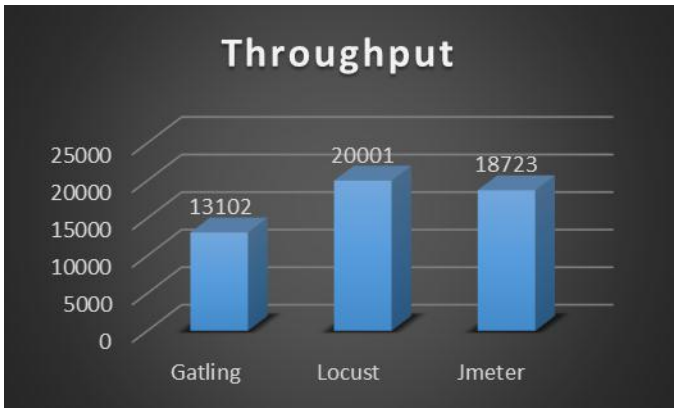

Fig 5: Comparison throughput of the 3 tools

The charts clealy show that Locust has highest throughput, and lowest response and execution time in this particular scenario. Locust is followed by JMeter, and then comes Gatling.

Locust lies ahead when it comes to the performance of the tool itself. Now let's delve into feature-based comparison to understand which tool can be a better choice in which situation. Locust does provide the best performance but it comes with a pre-requisite of having good python knowledge for you to be able to do everything with this tool, and attain maximum efficiency.

Gatling is a good choice given its comprehensive visualizations. If you prefer Scala, Gatling is a fine way to go, since it also supports JDBC and JMS protocols. Also, Gatling's Recorder Interface makes it a little easier to understand. JMeter is the way to go if other protocols are required for testing purposes, as it supports around 10 protocols like SMTP, FTP, IMAP, SOAP, POP3. It also reports in any format the tester wants, unlike Locust and Gatling, where the final reports come in HTML format only. Reports can come in CSV, XML, Graphs, Embedded table formats. JMeter is much more user-friendly than Locust and Gatling, since it comes as a Desktop application[7].

So if you have python knowledge, and would prefer higher tool performance[8], Locust should be the choice. If you are not as comfortable with Python or if protocols like JDBC,JMS are to be used[9], Gatling is a great option. If the tester doesn't come with a development background, and/or the testing requires use of the protocols not directly suppported by Locust and Gatling, JMeter is the clear way to go.

4. CONCLUSION

This paper presents a general analysis of open-source load testing tools. Based on Locust, Gatling, JMeter comparisons, it can be concluded that each of the tools does the job perfectly. Locust gets its efficiency edge, Gatling gets an edge with visualizations it provides, while JMeter gets its edge from the variety of protocols it supports and its ease of use. According to specifications, an educated decision on which resources to select from three.

REFERENCES

[1] Pu, Y., & Xu, M. (2009). Load Testing for Web Applications. 2009 First International Conference on Information Science and Engineering. doi:10.1109/icise.2009.720M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[2] Draheim, D., Grundy, I., Hosking, I., Lutteroth, C., & Weber, G. (2006). Realistic load testing of Web applications. Conference on Software Maintenance and Reengineering (CSMR'06). doi:10.1109/csmr.2006.43

[3] Ali, A., & Badr, N. (2015). Performance testing as a service for web applications. 2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS). doi:10.1109/intelcis.2015.7397245

[4] Verma, M., & Chiueh, T.-C. (n.d.). Implementation and performance evaluation of Locust. Proceedings. 1998 International Conference on Parallel Processing (Cat. No.98EX205). doi:10.1109/icpp.1998.708468

[5] Jovic, M., & Hauswirth, M. (2010). Performance Testing of GUI Applications. 2010 Third International Conference on Software Testing, Verification, and Validation Workshops. doi:10.1109/icstw.2010.27

[6] Kao, C. H., Lin, C. C., & Chen, J.-N. (2013). Performance Testing Framework for REST-Based Web Applications. 2013 13th International Conference on Quality Software. doi:10.1109/qsic.2013.32

[7] Using JMeter to Performance Test Web Services by Dmitri Nevedrov

[8] Locust official website. URL https://locust.io/

[9] Gatling official website. URL https://gatling.io/