

# Performing a Stress Test on Web Application?

Asked 16 years, 4 months ago    Modified 6 years, 3 months ago

Viewed 184k times

---



**256**



In the past, I used Microsoft Web Application Stress Tool and Pylot to stress test web applications. I'd written a simple home page, login script, and site walkthrough (in an ecommerce site adding a few items to a cart and checkout).

Just hitting the homepage hard with a handful of developers would almost always locate a major problem. More scalability problems would surface at the second stage, and even more - after the launch.

The URL of the tools I used were Microsoft Homer (aka [Microsoft Web Application Stress Tool](#)) and [Pylot](#).

The reports generated by these tools never made much sense to me, and I would spend many hours trying to figure out what kind of concurrent load the site would be able to support. It was always worth it because the stupidest bugs and bottlenecks would always come up (for instance, web server misconfigurations).

What have you done, what tools have you used, and what success have you had with your approach? The part that is most interesting to me is coming up with some kind

of a meaningful formula for calculating the number of concurrent users an app can support from the numbers reported by the stress test application.

web-applications

stress-testing

performance

webapplicationstresstool

pylot

Share

Improve this question

Follow

edited Oct 25, 2017 at 14:52



Vadim Kotov

8,274 ● 8 ● 50 ● 63

asked Aug 11, 2008 at 3:00



deadprogrammer

11.3k ● 24 ● 75 ● 85

30 Answers

Sorted by:

Highest score (default)



118



Here's another vote for [JMeter](#).

JMeter is an open-source load testing tool, written in Java. It's capable of testing a number of different server types (for example, web, web services, database, just about anything that uses requests basically).



It does however have a steep learning curve once you start getting to complicated tests, but it's well worth it. You can get up and running very quickly, and depending on what sort of stress-testing you want to do, that might be fine.

## Pros:

- Open-Source/Free tool from the Apache project (helps with buy-in)
- Easy to get started with, and easy to use once you grasp the core concepts. (Ie, how to create a request, how to create an assertion, how to work with variables etc).
- Very scalable. I've run tests with 11 machines generating load on the server to the tune of almost a million hits/hour. It was *much* easier to setup than I was expecting.
- Has an active community and good resources to help you get up and running. Read the tutorials first and play with it for a while.

## Cons:

- The UI is written in Swing. (ugh!)
- JMeter works by parsing the response text returned by the server. So if you're looking to validate any sort of javascript behaviours, you're out of luck.
- Learning curve is steep for non-programmers. If you're familiar with regular expressions, you're already ahead of the game.
- There are large numbers of (*insert expletive*) idiots in the support forum asking stupid questions that could be easily solved if they'd give the documentation even a cursory glance. ('How do I use JMeter to

stress-test my Windows GUI' shows up quite frequently).

- Reporting 'out of the box' leaves much to be desired, particularly for larger tests. In the test I mentioned above, I ended up having to write a quick console app to do some of the 'xml-logfile' to 'html' conversions. That was a few years ago though, so it's probable that this would no longer be required.

Share Improve this answer

edited Oct 9, 2013 at 13:01

Follow

answered Sep 18, 2008 at 13:24



Peter Bernier

8,058 ● 6 ● 40 ● 53

---

please clarify, if JMeter can help you test application installed on a remote VPS ? I'm not sure as it is desktop version

– [Rajat Gupta](#) Aug 6, 2013 at 8:23

- 
- 1 Another JMeter related option to be aware of is JMeter as a service. These types of SaaS provide highly scalable JMeter together with much improved reporting. – [Ophir Prusak](#) Jan 4, 2014 at 14:58

- 
- 6 I disagree that JMeter is very scalable. A million requests per hour is merely 278 requests/second, which - for being run on 11 machines - is extremely low compared to other tools. I would actually put JMeter's scalability on the Cons side. – [heyman](#) May 5, 2014 at 20:11

---

JMeter is not a browser, it works at protocol level. As far as web-services and remote services are concerned, JMeter looks like a browser (or rather, multiple browsers); however

JMeter does not perform all the actions supported by browsers. Web applications should be "executed" to be performed. – [LeonanCarvalho](#) Mar 16, 2018 at 18:06

---



38



I've used [The Grinder](#). It's open source, pretty easy to use, and very configurable. It is Java based and uses Jython for the scripts. We ran it against a .NET web application, so don't think it's a Java only tool (by their nature, any web stress tool should not be tied to the platform it uses).

We did some neat stuff with it... we were a web based telecom application, so one cool use I set up was to mimick dialing a number through our web application, then used an auto answer tool we had (which was basically a tutorial app from Microsoft to connect to their RTC LCS server... which is what Microsoft Office Communicator connects to on a local network... then modified to just pick up calls automatically). This then allowed us to use this instead of an expensive telephony tool called The Hammer (or something like that).

Anyways, we also used the tool to see how our application held up under high load, and it was very effective in finding bottlenecks. The tool has built in reporting to show how long requests are taking, but we never used it. The logs can also store all the responses and whatnot, or custom logging.

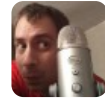
I highly recommend this tool, very useful for the price... but expect to do some custom setup with it (it has a built

in proxy to record a script, but it may need customization for capturing something like sessions... I know I had to customize it to utilize a unique session per thread).

Share Improve this answer

answered Aug 11, 2008 at 3:27

Follow



[Mike Stone](#)

44.6k ● 30 ● 114 ● 140

- 
- 1 +1 for grinder. I particularly liked the proxy scripting option.  
– [davek](#) Dec 8, 2009 at 21:32

---

any chance this can be used to simulate an idle browser. Server requests are made from an idle browser every two seconds for our app. I'd like to know what happens when we have thirty concurrent idle browsers. – [Ramy](#) Aug 13, 2012 at 19:08

- 
- 1 +1 for grinder. paired with EC2, we've successfully used it to spin up 100k concurrent users. – [nategood](#) Dec 12, 2012 at 15:08
- 



25



A little late to this party. I agree that [Pylot](#) is the best up-and-coming open source tool out there. It's simple to use and is actively worked on by a great guy ([Corey Goldberg](#)). As the founder of [OpenQA](#), I'm also happy that Pylot now is listed on our home page and uses some of our infrastructure (namely the forums).

However, I also recently decided that the entire concept of load testing was flawed: emulating HTTP traffic, with applications as complex as they have become, is a pain in the butt. That's why I created the commercial tool

BrowserMob. It's an [external load testing service](#) that uses [Selenium](#) to control real web browsers when playing back load.

The approach obviously requires a **ton** more hardware than normal load testing techniques, but hardware is actually pretty cheap when you are using cloud computing. And a nice side effect of this is that the scripting is **much** easier than normal load testing. You don't have to do any advanced regex matching (like JMeter requires) to extract out cookies, .NET session state, Ajax request parameters, etc. Since you're using real browsers, they just do what they are supposed to do.

Sorry to blatantly pitch a commercial product, but hopefully the concept is interesting to some folks and at least gets them thinking about some new ways to deal with load testing when you have access to a bunch of extra hardware!

Share Improve this answer

Follow

edited Dec 24, 2016 at 13:30



J J B

9,210 ● 1 ● 29 ● 42

answered Feb 17, 2009 at 4:49



Patrick Lightbody

4,534 ● 3 ● 30 ● 40

- 
- 2    Pylot's author has also crated another web testing tool:  
[code.google.com/p/multi-mechanize](https://code.google.com/p/multi-mechanize) – [codeape](#) Sep 13, 2011  
at 7:53
-

- 3 The link to [pylot.org](http://pylot.org) redirects to some suspicious website.  
– [mpiktas](#) Oct 6, 2016 at 14:19
- 



17

I've used [JMeter](#). Besides testing the web server you can also test your database backend, messaging services and email servers.



Share Improve this answer  
Follow



edited May 10, 2018 at 6:37



[djthoms](#)

3,076 ● 2 ● 34 ● 57

answered Aug 11, 2008 at 4:26



[grom](#)

16.1k ● 19 ● 65 ● 67



14

[ab](#), [siege](#), [tsung](#), [httpperf](#), [Trample](#), Pylot, [request-log-analyzer](#), [perftools](#)



Share Improve this answer  
Follow



edited Aug 30, 2018 at 15:01



[Bruce Becker](#)

344 ● 1 ● 8 ● 25

answered Apr 19, 2011 at 5:05



[Harry Love](#)

1,910 ● 1 ● 25 ● 25

- 
- 1 Recently I worked with [tsung](#) it's great tool(for really STRESS testing), here how to configure [progrnotes.blogspot.com/2011/11/...](http://progrnotes.blogspot.com/2011/11/...) or oficial docs [tsung.erlang-projects.org/user\\_manual.html](http://tsung.erlang-projects.org/user_manual.html) – [Sergey](#) Dec 6, 2011 at 11:33
-



I found openload useful as well:

[linuxpoison.blogspot.com/2010/12/...](http://linuxpoison.blogspot.com/2010/12/...) – wael34218 Dec 5, 2012 at 7:59

---



For a web based service, check out [loader.io](http://loader.io).

11

Summary:



loader.io is a free load testing service that allows you to stress test your web-apps/apis with thousands of concurrent connections.



They also have an [API](#).

Share Improve this answer

answered Jan 2, 2013 at 22:08

Follow



danriti

907 ● 8 ● 9

---

2 This is a good alternative to testing your own machines with your own machines – [nurettin](#) Dec 18, 2013 at 14:25

---



For simple usage, I prefer ab(apache benchmark) and siege, later one is needed as ab don't support cookie and would create endless sessions from dynamic site.

9



both are simple to start:



```
ab -c n -t 30 url
```



```
siege -b -c n -t 30s url
```

siege can run with more urls.

last siege version turn verbose on in siegerc, which is annoy. you can only disable it by edit that file( `/usr/local/etc/siegerc` ).

Share Improve this answer

edited Nov 24, 2011 at 20:11

Follow



Otiel

18.7k ● 17 ● 81 ● 127

answered Aug 23, 2010 at 9:58



Ben Li

101 ● 1 ● 2



9



As this question is still open, I might as well weigh in.

The good news is that over the past 5 or so years the Open Source tools have really matured and taken off in the space, the bad news is there are so many of them out there.



Here are my thoughts:-



Jmeter vs Grinder

Jmeter is driven from an XML style specification, that is constructed via a GUI.

Grinder uses Jython scripting within a multi-threaded Java framework, so more oriented to programmers.

Both tools will handle HTTP and HTTPS and have a proxy recorder to get you started. Both tools use the Controller model to drive multiple test agents so scalability is not an issue (given access to the Cloud).

Which is better:-

A hard call as the learning curve is steep with both tools as you get into the more complicated scripting requirements for url rewriting, correlation, providing unique data per Virtual User and simulating first time or returning Users (by manipulating the HTTP Headers).

That said I would start with Jmeter as this tool has a huge following and there are many examples and tutorials on the web for using this tool. If and when you come to a 'road block', that is something you can't 'easily' do with Jmeter then have a look at the Grinder. The good news is both these tools have the same Java requirement and a 'mix and match' solution is not out of the question.

Something new to add – Headless browsers running multiple instances of Selenium WebDriver.

This is a relatively new approach because it relies on the availability of resources that can now be provisioned from the Cloud. With this approach a Selenium (WebDriver) script is taken and run within a headless browser (i.e. `WebDriver = New HtmlUnitDriver()`) driver in multiple threads.

From experience around 25 instances of 'headless browsers' can be executed from the Amazon M1 Small Instance.

What this means is that all of the correlation, url rewriting issues disappear as you repurpose your functional testing scripts to become performance testing scripts.

The scalability is compromised as more VMs will be needed to drive the load, as compared with a HTTP driver such as the Grinder or Jmeter. That said, if you are looking to drive 500 Virtual Users then with 20 Amazon Small Instances (6 cents an hour each) at a cost of just \$1.20 per hour gives you load that is very close to the Real User Experience.

Share Improve this answer

answered Mar 28, 2013 at 23:52

Follow



Ian Fleming

176 ● 1 ● 1

---

Grinder can also use Clojure scripting. – [user100464](#) Dec 29, 2013 at 18:51

---



9



Also, there is an awesome open-source pure-python distributed and scaleable [locust](#) framework that uses [greenlets](#). It's great at simulating enormous amount of simultaneous users.

Share Improve this answer

answered May 7, 2013 at 22:03

Follow



alecxe

473k ● 126 ● 1.1k ● 1.2k



8



We recently started using Gatling for load testing. I would highly recommend to try out this tool for load testing. We had used SOASTA and JMeter in past. Our main reason to consider Gatling is following:

- Recorder to record the scenario
- Using Akka and Netty which gives better performance compare to Jmeter Threading model
- DSL Scala which is very much maintainable compare to Jmeter XML
- Easy to write the tests, don't scare if it's scala.
- Reporting

Let me give you simple example to write the code using Gatling Code:

```
// your code starts here
val scn = scenario("Scenario")
    .exec(http("Page")
        .get("http://example.com"))
// injecting 100 user enter code here's on above
scenario.
setUp(scn.inject(atOnceUsers(100)))
```

However you can make it as complicated as possible. One of the feature which stand out for Gatling is reporting which is very detail.

Here are some links:

[Gatling](#)

[Gatling Tutorial](#)

I recently gave talk on it, you can go through the talk here:

[https://docs.google.com/viewer?](https://docs.google.com/viewer?url=http%3A%2F%2Ffiles.meetup.com%2F3872152%2FExploring-Load-Testing-with-Gatling.pdf)

[url=http%3A%2F%2Ffiles.meetup.com%2F3872152%2FExploring-Load-Testing-with-Gatling.pdf](https://docs.google.com/viewer?url=http%3A%2F%2Ffiles.meetup.com%2F3872152%2FExploring-Load-Testing-with-Gatling.pdf)

Share Improve this answer

edited Nov 23, 2015 at 16:50

Follow

answered Nov 23, 2015 at 16:44



Sunil Kapil

1,040 ● 13 ● 12



6

This is an old question, but I think newer solutions are worthy of a mention. Checkout LoadImpact:

<http://www.loadimpact.com>.



Share Improve this answer

answered Jan 5, 2012 at 21:33

Follow




Chris F

2,946 ● 2 ● 29 ● 33



Yeah. I've just taken a look at this. Found it on Google before finding this Q/A. I think a web based application is a good approach, but I couldn't be sure if it's really pushing my server. It was worthwhile to try out though, no doubt. Tbh, I'm

really tempted to sign up for a full account. – [Charlie](#) May 18, 2014 at 1:04 

---



4




I tried [WebLoad](#) it's a pretty neat tool. It comes with and test script IDE which allows you to record user action on a website. It also draws a graph as it perform stress test on your web server. Try it out, I highly recommend it.

Share Improve this answer

[edited Jan 14, 2014 at 11:48](#)

Follow

 [Yasei No Umi](#)  
1,574 ● 9 ● 23

answered Sep 30, 2008 at 1:07



[Alvin](#)  
10.4k ● 9 ● 39 ● 50

---

1 I recommend WebLoad too. It's a great tool, easy to use, and the reports are pretty helpful. I'm assuming you're on a Windows platform, so these results combined with perfmon will let you know just about everything you need to know.  
– [Babak Naffas](#) Jun 22, 2009 at 2:37

---

2 Note that WebLoad is purely commercial now. They sent out emails saying, quote: ----- -WebLOAD Open Source has been declared End of life (EOL) -If you still have a version of the product we remind you that under the EULA, any distribution of the product or using it to service third parties is strictly forbidden. ----- Distributing the Open Source version is prohibited? Even using it a way they don't like is banned? Not the sort of behavior I want anything to do with. – [Joshdan](#) Jul 15, 2009 at 21:24

---

1 The linked to domain is now just advertising - the original domain has expired. – [dodgy\\_coder](#) Jun 17, 2012 at 13:22

---

@Joshdan this is why GPL is important.

– [Thorbjørn Ravn Andersen](#) Aug 16, 2012 at 13:04

---



3



Trying all mentioned here, I found [curl-loader](#) as best for my purposes. very easy interface, real-time monitoring, useful statistics, from which I build graphs of performance. All features of libcurl are included.

Share Improve this answer

answered Sep 3, 2010 at 9:24



Follow



[DominiCane](#)

1,293 ● 3 ● 16 ● 30



3



Blaze meter has a chrome extension for recording sessions and exporting them to JMeter (currently requires login). You also have the option of paying them money to run it on their cluster of JMeter servers (their pricing seems much better than LoadImpact which I've just stopped using):



- [BlazeMeter Chrome Extension](#)
- [Blog entry about it](#)



I don't have any association with them, I just like the look of their service, although I haven't used the paid version yet.

Share Improve this answer

edited Aug 4, 2014 at 12:53

Follow



answered Nov 1, 2013 at 3:03



Gerry

11.1k ● 4 ● 45 ● 50



2



You asked this question almost a year ago and I don't know if you still are looking for another way of benchmarking your website. However since this question is still not marked as solved I would like to suggest the free webservice [LoadImpact](#) (btw. not affiliated). Just got this link via twitter and would like to share this find. They create a reasonable good overview and for a few bucks more you get the "full impact mode". This probably sounds strange, but good luck pushing and braking your service :)

Share Improve this answer

Follow

answered Jun 22, 2009 at 1:38



merkuro

6,169 ● 2 ● 28 ● 29



1



I found [IBM Page Detailer](#) also an interesting tool to work with.

Share Improve this answer

Follow

answered Aug 28, 2008 at 19:38



Michael Stum

181k ● 119 ● 407 ● 541

I've used [openSTA](#).



This allows a session with a web site to be recorded and then played back via a relatively simple script language.

1



You can easily test web services and write your own scripts.



It allows you to put scripts together in a test in any way you want and configure the number of iterations, the number of users in each iteration, the ramp up time to introduce each new user and the delay between each iteration. Tests can also be scheduled in the future.

It's open source and free.

It produces a number of reports which can be saved to a spreadsheet. We then use a pivot table to easily analyse and graph the results.

Share Improve this answer

answered Sep 15, 2008 at 20:39

Follow



[rbrayb](#)

46.7k ● 34 ● 118 ● 179



We use the Microsoft tool mentioned - Microsoft Web Application Stress Tool. It is the easiest tool I have used.

1



It is limited in many ways, including only being able to hit port 80 on manually created tests. But, its ease of use means it actually gets used.



We supplement the load from this tool with other tools including OpenSTA and link check spiders.

JMeter looks good from my initial evaluation, I hope to include it in our continuous integration going forward. But, JMeter is complex and non trivial to roll out.

I'd suggest opening another question regarding interpreting the MS stress tool results.

Share Improve this answer

answered Sep 28, 2008 at 15:37

Follow



Jerry B

193 ● 1 ● 1 ● 7



1

Visual Studio Test Edition 2010 (2008 good too). This is a really easy and powerful tool to create web/load tests with.



The bonus with this tool when using against Windows servers is that you get integrated access to all the perfmon server stats in your report. Really useful.



The other bonus is that with Visual Studio project you can integrate a "Performance Session" that will profile the code execution of your website.

If you are serving webpages from a windows server, this is the best tool out there.

There is a separate and expensive licence required to use several machines to load test the application however.

Share Improve this answer

edited Dec 16, 2010 at 21:25

Follow

answered Dec 8, 2009 at 21:28



Nat

14.3k ● 5 ● 43 ● 64



1



We have developed a process that treats load and performance measurement as a first-class concern - as you say, leaving it to the end of the project tends to lead to disappointment...

So, during development, we include very basic multi-user testing (using selenium), which checks for basic craziness like broken session management, obvious concurrency issues, and obvious resource contention problems. Non-trivial projects include this in the continuous integration process, so we get very regular feedback.

For projects that don't have extreme performance requirements, we include basic performance testing in our testing; usually, we script out the tests using BadBoy, and import them into JMeter, replacing the login details and other thread-specific things. We then ramp these up to the level that the server is dealing with 100 requests per second; if the response time is less than 1 second, that's usually sufficient. We launch and move on with our lives.

For projects with extreme performance requirements, we still use BadBoy and JMeter, but put a lot of energy into

understanding the bottlenecks on the servers on our test rig(web and database servers, usually). There's a good [tool for analyzing Microsoft event logs](#) which helps a lot with this. We typically find unexpected bottlenecks, which we optimize if possible; that gives us an application that is as fast as it can be on "1 web server, 1 database server". We then usually deploy to our target infrastructure, and use one of the "Jmeter in the cloud" services to re-run the tests at scale.

Again, PAL reports help to analyze what happened during the tests - you often see very different bottlenecks on production environments.

The key is to make sure you don't just run your stress tests, but also that you collect the information you need to understand the performance of your application.

Share Improve this answer

answered Jul 15, 2012 at 14:39

Follow



Neville Kuyt

29.6k ● 2 ● 39 ● 52



1



There are a lot of good tools mentioned here. I wonder if tools are an answer the question: "How do you stress test a web application?" The tools don't really provide a method to stress a Web app. Here's what I know:



Stress testing shows how a Web app fails while serving responses to an increasing population of users. Stress testing shows how the Web app functions while it fails. Most Web apps today - especially the Social/Mobile Web

apps - are integrations of services. For example, when Facebook had its outage in May 2011 you could not log onto Pepsi.com's Web app. The app didn't fail entirely, just a big portion of its normal function become unavailable to users.

Performance testing shows a Web app's ability to maintain response times independent of how many users are concurrently using the app. For example, an app that handles 10 transactions per second with 10 concurrent users should handle 20 transactions per second at 20 users. If the app handles less than 20 transactions per second the response times are growing longer and the app is not able to achieve linear scalability.

Also, in the above example the transaction-per-second count should be of only successful operations of a test use case/workflow. Failures typically happen in shorter timespans and will make the TPS measurement overly optimistic. Failures are important to a stress and performance test since they generate load on the app too.

I wrote up the PushToTest methodology in the TestMaker User Guide at <http://www.pushtotest.com/pushtotest-testmaker-6-methodology>. TestMaker comes in two flavors: Open Source (GPL) Community version and TestMaker Enterprise (commercial with great professional support.)

-Frank

Share Improve this answer

answered Aug 17, 2012 at 21:01

Follow



user1227037

31 ● 4

---

1 this doesn't answer the OP's question whatsoever

– Corey Goldberg Jun 1, 2018 at 17:24

---



1



Take a look at

LoadBooster(<https://www.loadbooster.com>). It utilizes headless scriptable browser PhantomJS/CasperJs to test web sites. Phantomjs will parse and render every page, execute the client-side script. The headless browser approach is easier to write test scenarios to support complex AJAX heavy Web 2.0 app, browser navigation, mouse click and keystrokes into the browser or wait until an element exists in DOM. LoadBooster support selenium HTML script too.

Disclaimer: I work for LoadBooster.

Share Improve this answer

answered Dec 29, 2016 at 5:43

Follow



QingHai

138 ● 3

---

Note for future readers : LoadBooster no longer exists

– OxTheOldOne Aug 30, 2022 at 9:12 ✎

---



Try [ZebraTester](#) which is much easier to use than jMeter. I have used jMeter for a long time but the total setup time

1



for a load test was always an issue. Although ZebraTester isn't open source, the time that I have saved in the last six months makes up for it. They also have a SaaS portal which can be used for quickly running tests using their load generators.

Share Improve this answer

edited Feb 23, 2017 at 23:23

Follow

answered Sep 30, 2016 at 21:21



Vinit

1,825 ● 17 ● 38



0

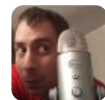


One more note, for our web application, I found that we had huge performance issues due to contention between threads over locks... so the moral was to think over the locking scheme very carefully. We ended up having worker threads to throttle too many requests using an asynchronous http handler, otherwise the application would just get overwhelmed and crash and burn. It meant a huge backlog could pile up, but at least the site would stay up.

Share Improve this answer

answered Aug 11, 2008 at 3:31

Follow



Mike Stone

44.6k ● 30 ● 114 ● 140

this doesn't answer the OP's question whatsoever

– Corey Goldberg Jun 1, 2018 at 17:25





Take a look at [TestComplete](#).

0

Share Improve this answer

answered Aug 21, 2008 at 10:34

Follow



[Erick Sasse](#)

2,819 ● 3 ● 25 ● 30



---

1 Test Complete is a commercial tool. – [Ripon Al Wasim](#) Feb 28, 2013 at 4:21

---



0

I second the opensta suggestion. I would just add that it allows you to do things to monitor the server you're testing using SMTP. We keep track of processor load, memory used, bytes sent, etc. The only downside is that if you find something broken and want to do a fix it relies on several open-source libraries that are no longer kept up, so getting a compiling version of the source is more tricky than with most OSS.



Share Improve this answer

answered Sep 20, 2008 at 22:09

Follow



[tloach](#)

8,040 ● 1 ● 35 ● 44



0

I played with JMeter. One think it could not not test was ASP.NET Webforms. The viewstate broke my tests. I am not shure why, but there are a couple of tools out there



that dont handle viewstate right. My current project is ASP.NET MVC and JMeter works well with it.



Share Improve this answer

answered Feb 3, 2009 at 10:55

Follow



**Mathias F**

15.9k ● 25 ● 93 ● 165



0

I had good results with [FunkLoad](#) :

- easy to script user interaction
- reports are clear
- can monitor server load



Share Improve this answer

answered Nov 13, 2012 at 10:32

Follow



**yanjost**

5,441 ● 2 ● 27 ● 28



0

At the risk of being accused of shameless self promotion I would like to point out that in my quest for a free load testing tool I went to this article:

<http://www.devcurry.com/2010/07/10-free-tools-to-loadstress-test-your.html>



Either I couldn't get the throughput I wanted, or I couldn't get the flexibility I wanted. AND I wanted to easily aggregate the results of multiple load test generation hosts in post test analysis.

I tried out every tool on the list and to my frustration found that none of them quite did what I wanted to be able to

do. So I built one and am sharing it.

Here it is: <http://sourceforge.net/projects/loadmonger>

PS: No snide comments on the name from folks who are familiar with urban slang. I wasn't but am slightly more worldly now.

Share Improve this answer

answered Dec 11, 2012 at 1:22

Follow



Steve Owens

77 ● 1 ● 3



0



I vote for **jMeter** too and I want add some quotes to @PeterBernier answer.

The main question that load testing answers is how many concurrent users can my web application support? In order to get a proper answer, **load testing should represent real application usage, as close as possible.**

Keep above in mind, **jMeter** has many building blocks **Logical Controllers, Config Elements, Pre Processors, Listeners** ,... which can help you in this.

You can mimic real world situation with jMeter, for example you can:

1. Configure jMeter to act as real Browser by configuring ( concurrent resource download , browser cache , http headers , setting request

time out , cookie management , https support ,  
encoding , ajax support ,... )

2. Configure jMeter to generate user requests (by defining number of users per second , ramp-up time , scheduling ,...)
3. Configure lots of client with jMeter on them, to do a distributed load test.
4. Process response to find if the server is responding correctly during test. ( For example assert response to find a text in it)

Please consider:

- It is easy to start a real web application test with jMeter in minutes. The jMeter has a very easy tool which record your test scenario ( know as HTTP(S) Test Script Recorder ).
- jMeter has lots of plugins at <http://jmeter-plugins.org>.
- The jMeter UI is swing based and has made good changes in jMeter 3.2. On the other hand please consider that JMeter GUI should only be used for test and debugging. It is not good practice to use it in GUI mode for actual test.  
<https://www.blazemeter.com/blog/5-ways-launch-jmeter-test-without-using-jmeter-gui>. Configure and test your scenario and run it on non-gui mode.
- There are lots of reporting showing tools in jMeter (Known as listeners ) but there are not meant to be on during test. You must run your test and generate

reports ( `.jtl` files). Then you must use these tools to analyze result. Please have a look at <https://www.blazemeter.com/blog/jmeter-listeners-part-1-basic-display-formats> or [https://www.tutorialspoint.com/jmeter/jmeter\\_listener\\_s.htm](https://www.tutorialspoint.com/jmeter/jmeter_listener_s.htm).

The <https://www.blazemeter.com/jmeter> has very good and practical information to help you configure your test environment.

Share Improve this answer

answered Jun 28, 2017 at 13:37

Follow



**Alireza Fattahi**

45.3k ● 16 ● 131 ● 184



**Highly active question.** Earn 10 reputation (not counting the [association bonus](#)) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.