Performance Testing

Type of testing to ensure software applications will perform well under the particular workload.

Load testing is the simplest form of performance testing

Stress testing, volume testing, load testing is some of the other performance testing

Popular tools – Jmeter, Loadrunner, Neoload, Load UI

Why Jmeter?

- Open Source
- Cross platform support – since its built using JAVA
- Scripting isn't essential to learn JMeter
- JMeter GUI is more user-friendly.

Thread Group

A thread group is a set of threads/users executing the same scenario.

Listeners

Listeners will track and listen on how jmeter is executing, and then it collects parameter results, graphs and it shows the output

- Understand the Jmeter Load parameters to analyze results
  Samples : No of users hit that specific request

Average: It is the average time taken by all the samples to execute specific label.

Min: The shortest time taken by a sample for specific label.

Max: The longest time taken by a sample for specific label

Std. Dev.: This shows the set of exceptional cases which were deviating from the average value of sample response time. The lesser this value more consistent the data. Standard deviation should be less than or equal to half of the average time for a label.

Error%: Percentage of Failed requests per Label.

Throughput: Throughput is the number of request that are processed per time unit(seconds, minutes, hours) by the server. This time is calculated from the start of first sample to the end of the last sample. Larger throughput is better.

Median: It is the time in the middle of a set of samples result. It indicates that 50% of the samples took no more than this time i.e the remainder took at least as long.

90% Line: 90% of the samples took no more than this time. The remaining samples took at least as long as this. (90th percentile)

- ## Additional Plugins for Simulating real time Load

  Download the jar from
  https://jmeter-plugins.org/wiki/PluginsManager/
  Concurrency Thread Group
  Ultimate Thread Group

Controllers are more like a folder where we can control the action of the complete folder.

Simple controller

The simple controller provides no functionality beyond that of grouping, Primarily to organize samplers and other logic controllers.

Transactional controllers

It gives overall or sum off response time of the transaction in the transactional grouping

The transactional controller is considered successful only if all the samplers under it are successful

The transaction controller can function in parent mode.

In the parent mode, the individual samples can still appear in the view results tree, but no longer appear as separate entries in view results table or in other listeners

By Default, TC does not include time taken by timers and preprocessors. However, it can be configured to include the timers by enabling a checkbox

Loop controller

The loop controller can be configure to loop indefinitely by enabling the Forever checkbox

If you run the test with the forever checkbox enabled, you need to click on the shutdown button and terminate the test gracefully.

Clicking on the stop button terminates the threads abruptly, and you may see some errors in the request

Runtime controller

The runtime controller controls the duration for which its child elements run

A RC executes its child elements in its hierarchy the duration. at the end of the nested elements it'll loop through again

Using the same logic, if the specified time runs out, the Runtime controller stops the execution even if it has executed only a part of the nested elements. The currently executing element is allowed to complete, but the newer elements are not executed once the specified time is over.

Throughput controller

The Throughput controller controls the number of executions of its child elements. This is a misnomer, as it does not control the throughput

You can configure this in two modes:

1: Number of executions: the child elements are executed until the specified count is reached and the subsequent executions are skipped. TO configure this, choose Total executions and specify a value for throughput

2: Percentage of executions: This concept is same except the number fo executions of the child elements is restricted by the percentage configured.

Once only controller

Once only controller executes its child elements only once per thread/VUser

This is typically used to perform logins or another use-case that's needed only once for a user session

The once only controller should be a child element of the thread group or loop controller. otherwise, the behavior is not defined.

Interleave controller

The interleave controller executes only one of its child elements per loop iteration. Each time it iterates, it picks the next child element in sequence

How the values will be picked for multiple threads?

What if there are child controllers

Random controller

The random controller is similar to interleave controller except that the order of interleaving is random instead of sequential. The config is same as interleave controller

Random order controller

The random order controller executes all its child elements but in random order.

Random controller executes only one of the child elements.

Switch controller

The switch controller is analogous to the switch/case programming construct

The switch Controller executes only one of its child elements after match the elements name with the configured switch value.

If the switch value is an integer, it executes the child element based on the sequence number

IF Controller

More like the if controller used in java.

2 modes

without interpretation as variable expression where LHS == RHS is verified

With interpretation as variable expression -> groovy/jexl3 ->where LHS == RHS is verified and returns True/False. Based on true/False the script executes.

HTTP Cookie manager

Browsers will store cookies certain unique session cookies which helps the webpages to identify that the user is the one who stored the cookie.

Cookie will help you identify yourself uniquely to the application

Jmeter is not a browser to store cookies. That's why we are giving cookie manager.

User Defined Variables

FOr using the variable use the template ${VARIBLE_NAME}

Random and Random string functions-

Timers

Constant Timer

The constant Timer introduces a specified delay before the samplers in its scope are executed. The only configuration is the delay that is needed.

Constant Timer delays all the child elements in scope.

What if you wanted to delay only a specific sampler by 5 seconds?

You can achieve this by making the constant timer a child element of the sampler to be delayed.

Constant timer is mostly used for pacing

Pacing is time between two iterations

Uniform Random Timer

The delay introduced by the uniform random timer has two parts:

Constant Delay: Fixed and equal to the configured value.

Random Delay: Varies between Zero and the configured value

The actual delay will range between the constant delay and the constant delay plus the random delay.

As the term "Uniform " indicated, the delay varies within its range with equal probability

If the constant delay was configured to 10 seconds and the Random delay was configured to 5 seconds the actual delay will vary between 10 and 15 seconds.

Use Loop controller to demonstrate the Gaussian delays

Gaussian Random Timer

The gaussian random timer introduces a delay according to the gaussian distribution. The delay varies around a central mean.

Lets assume that the mean value is 10 seconds and the variance is 2 seconds. The delay introduced would vary as follows.

1. 68% of the time, the delay would vary between mean -variance and mean + variance (Between 8 and 12 seconds)
2. 95% of the time, the delay would vary between mean -2*variance and mean + 2*variance (Between 6 and 14 seconds)
3. 99% of the time, the delay would vary between mean -3*variance and mean + 3*variance (Between 4 and 16 seconds)

Constant Throughput Timer

The constant throughput timer calculates and introduces delays between samplers so as to keep the throughput at the configured value.

Target throughput (in samples per min) = 5

Configure number of threads (Users) as 1 and Loop count as 20

In these results, each of the requests to /jmeter/alpha has been delayed by a varying amount so as to keep the overall throughput at configured level.

In the aggregate results, observe the throughput is approx 5 requests per minute

Synchronizing TImer

The synchronizing timer blocks VUsers and releases them all at once creating a large load at the same instant. This is very helpful to test how the application handles simultaneous requests

Number of threads(users) as 4, Loop count as 12.

Assertions

Response Assertion

Main Samples – Samplers

Sub samples – embedded resources

Add Assertion Listener - this listener will show only error. Success will not be shown

JSON Assertion

When we are getting a JSON response like in testing a webservice request, this assertion is used.

Size Assertion

This assertion is used to check the size of the transactions.

Duration Assertion

This assertion is used to check the sample and sub samples response time.

How to download files from Jmeter.

Using HTTP request call the transaction with. it'll download the file.

For saving - use a listener for that transaction called "Save response to a file"

If we don't enable the add the number to prefix checkbox and iterate for 100 times there will be 100 files downloaded in that path

Use size assertion and validate the size of the file HTML Assertion and XML Assertions

HTML Assertion is used to check the HTML response for the number of warnings and errors in the HTML response.

XML response does the same for the XMLs

JSON Assertion

it helps in checking the webservice requests in the JSON format

Use the JSON assertion

Find the JSON path using JSON path tester in the view result in tree listener.

Syntax of Path tester in View results in tree is **&..name**

Correlation

1. Correlation is done for handling dynamic values

  Values which change from iteration to iteration /user to user

2. These dynamic values not handled - script will fail

Dynamic value is something that servers sends in response for one of the samplers before it is sent in a sampler as a request. we will be recording a script which has requests only. In that we can see the dynamic value is passed.

Where we can find the response from the servers while recording?

View result tree in HTTPS script recorder. While recoridng the script, we will have a log. In that check fromthe responses of samplers before the sampler request found

===============

1.Identify the dynamic values -

Record the script twice and then compare both the recording - whatever the value that is changing - typically those values might be dynamic values

Dynamic values are mostly at samplers or at header manager

Dynamic value is something that servers sends in response for one of the samplers before it is sent in a sampler as a request. we will be recording a script which has requests only. In that we can see the dynamic value is passed.

Where we can find the response from the servers while recording?

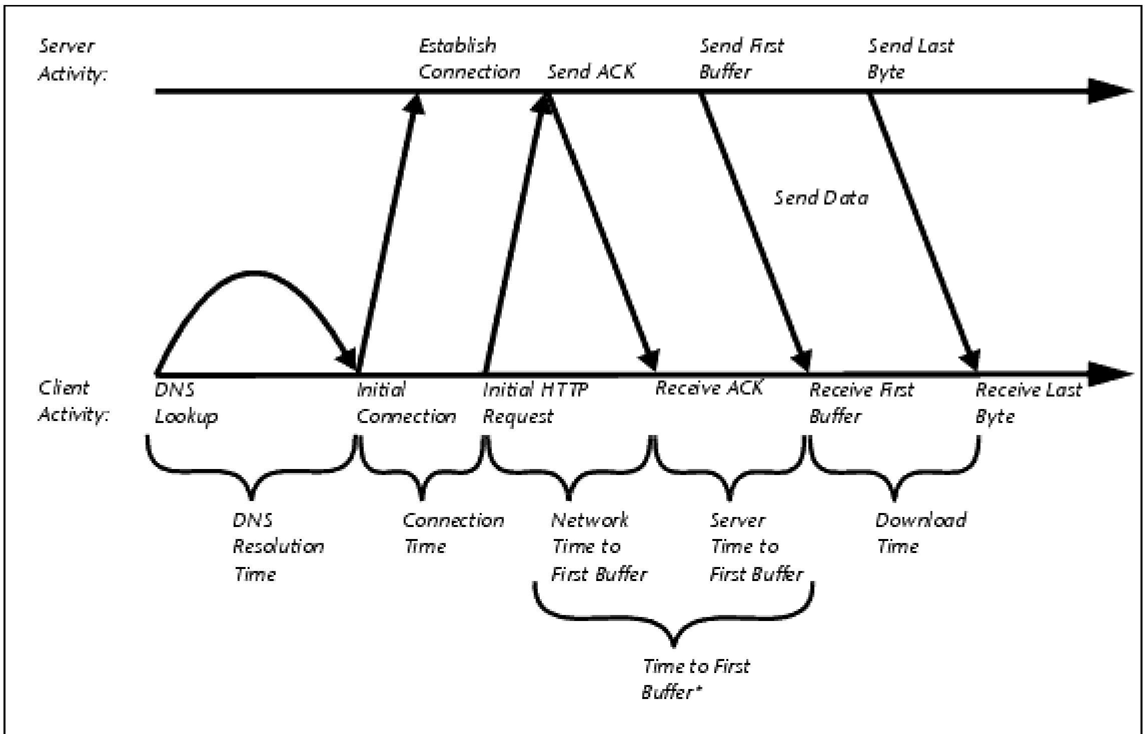
View result tree in HTTPS script recorder. While recoridng the script, we will have a log. In that check from the responses of samplers before the sampler request found

2.Correlate the dynamic values.

Steps:

1. Identify the Dynamic value in the response
2. Build the regular expressions of the Dynamic value - why? - Because its a dynamic value.
3. Copy the regular expression and modify the Jmeter script to capture the dynamic value
4. Add the post processor - Regular expression extractor on the sampler request where we found the Dynamic value as response
5. Add the regex value and other values
6. Add debug sampler with view results tree. It will capture the variable response.

Response time breakup



*Business Service Management combines: Network Time to First Buffer and Server Time to First Buffer