Университет ИТМО

Мегафакультет компьютерных технологий и управления

Факультет программной инженерии и компьютерной техники

Лабораторная работа №4 по Тестированию программного обеспечения

Вариант: 284695

Группа: Р33112

Выполнил: Провоторов Александр Владимирович

Преподаватель: Харитонова Анастасия Евгеньевна

**Текст задания**

С помощью программного пакета Apache JMeter провести нагрузочное и стресс-тестирование веб-приложения в соответствии с вариантом задания.

В ходе нагрузочного тестирования необходимо протестировать 3 конфигурации аппаратного обеспечения и выбрать среди них наиболее дешёвую, удовлетворяющую требованиям по максимальному времени отклика приложения при заданной нагрузке (в соответствии с вариантом).

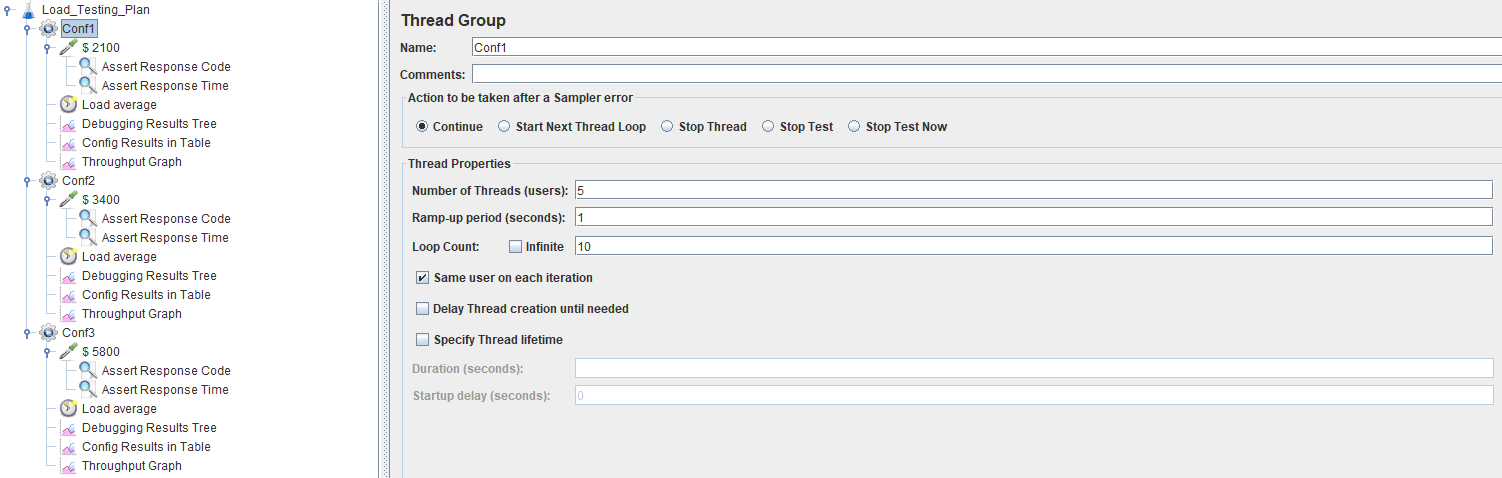
В ходе стресс-тестирования необходимо определить, при какой нагрузке выбранная на предыдущем шаге конфигурация перестаёт удовлетворять требованиями по максимальному времени отклика. Для этого необходимо построить график зависимости времени отклика приложения от нагрузки.

**Параметры тестируемого веб-приложения:**Если запрос содержит некорректные параметры, сервер возвращает HTTP 403.  
Если приложение не справляется с нагрузкой, сервер возвращает HTTP 503.

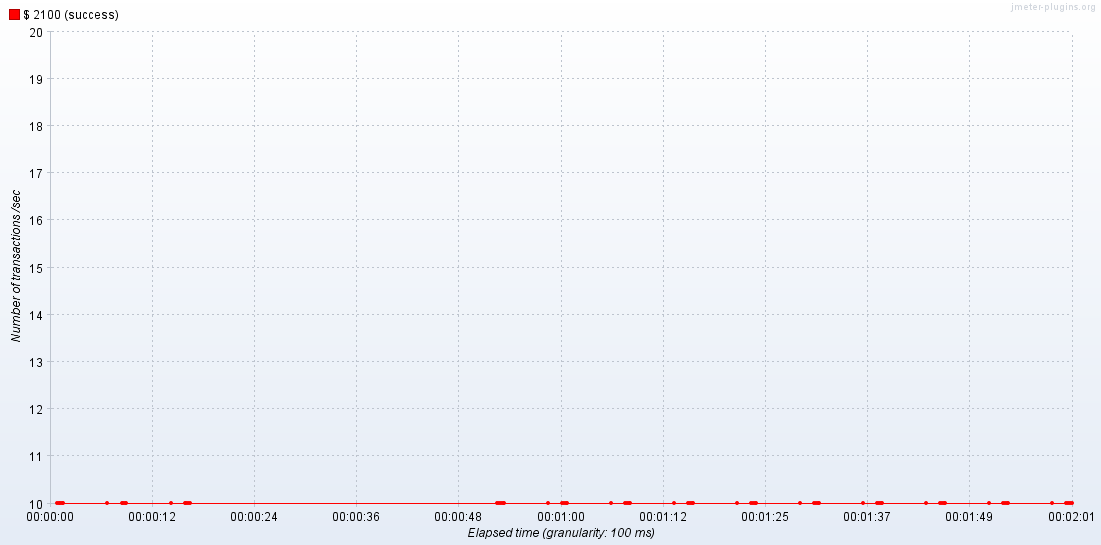
* URL первой конфигурации ($ 2100) – [http://aqua:8080?token=468485783&user=2022903893&conf=1](https://se.ifmo.ru/courses/appUrl);
* URL второй конфигурации ($ 3400) – [http://aqua:8080?token=468485783&user=2022903893&conf=2](https://se.ifmo.ru/courses/appUrl);
* URL третьей конфигурации ($ 5800) – [http://aqua:8080?token=468485783&user=2022903893&conf=3](https://se.ifmo.ru/courses/appUrl);
* Максимальное количество параллельных пользователей – 5;
* Средняя нагрузка, формируемая одним пользователем – 40 запр. В мин.;
* Максимально допустимое время обработки запроса – 770 мс.

**Конфигурация Jmeter для нагрузочного тестирования:**

Конфигурация содержит в себе 3 идентичных теста для 3 разных систем, которые проводят тестирование на 5 пользователях (потоках), таким образом, чтобы число запросов не выходило за 40, а время ответа не превышал 770 мс. GET-запросы с содержаниями, указанными в варианте, отправляются на сервер. С помощью Assertion-ов проверяется, приходит ли код возврата 200 или нет. С помощью Listener-а и плагинов на отрисовку графики отображаем значения в PNG графики и XML-таблицы.

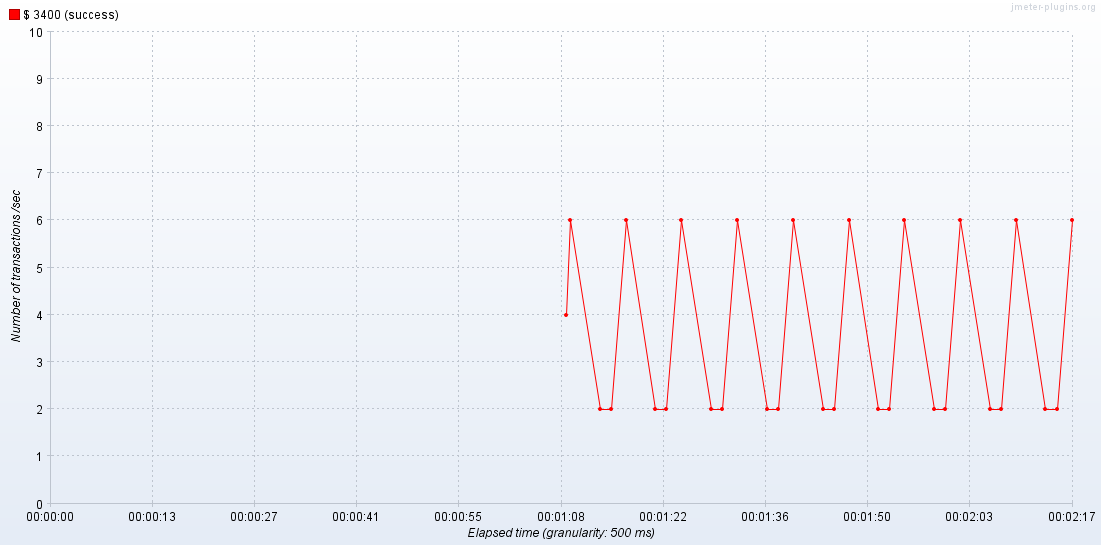


**Тест системы 1**

****

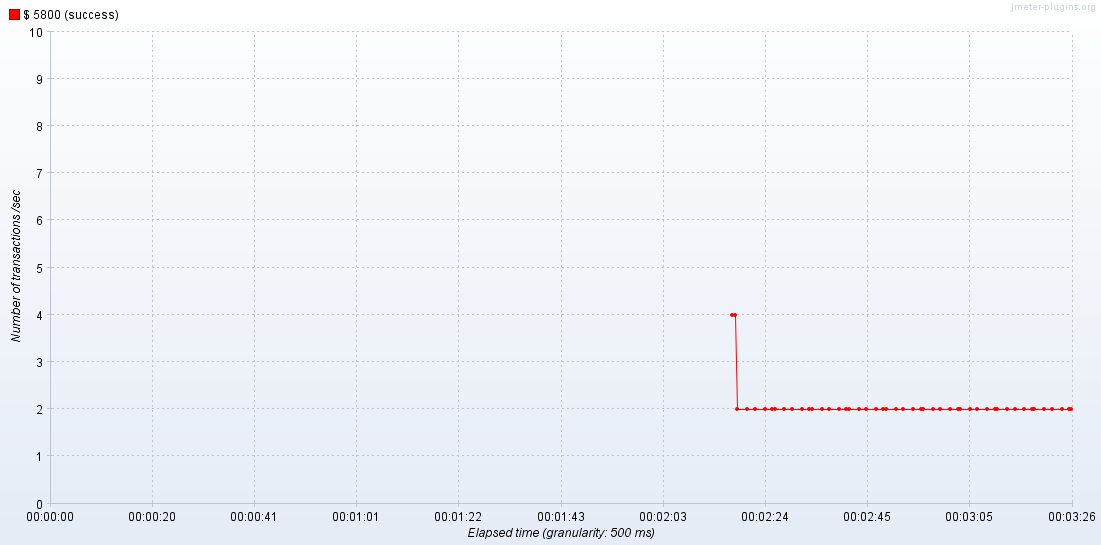
В данном случае система прошла все тесты, а время ответа каждого теста в среднем составляло 614 мс.

**Тест системы 2**

****

В данном случае система прошла все тесты, а время ответа каждого теста в среднем составляло 414 мс.

**Тест системы 3**

****

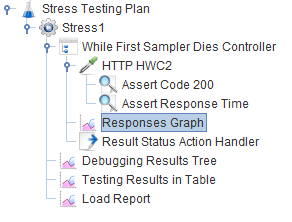
В данном случае система прошла все тесты, а время ответа каждого теста в среднем составляло 313 мс.

**Вывод по первой части:**

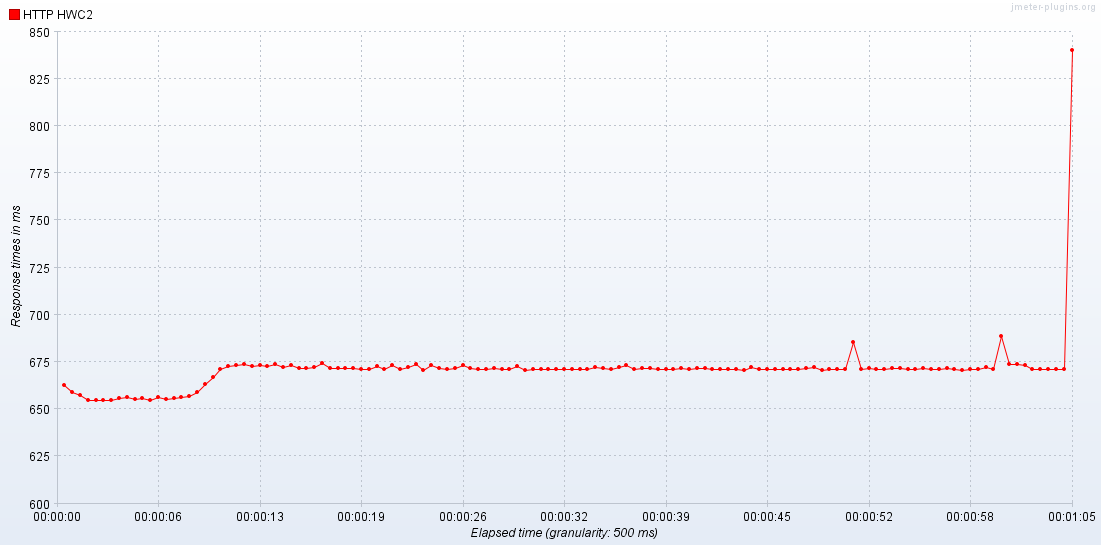
В данном случае каждая система прошла полный набор тестов в заданном промежутке времени, что означает выбор аппаратного обеспечения будет зависеть только от его стоимости. Поэтому для стресс тестирования мы **выберем 1-ю конфигурацию.**

**Конфигурация Jmeter для стресс тестирования:**

Теперь в плане мы имеем только 1-ю конфигурацию, которую мы и будем использовать в стресс-тесте. В данном случае мы имеем модуль While, который контролирует условие: пока один из потоков не присылает какое-либо сообщение кроме 200 либо time-limit, он продолжает работу и наращивает нагрузку на сервер.



**График времени отклика от нагрузки**

****

**Вывод:**

В ходе данной лабораторной работы, я научился использовать JMeter для нагрузочного и стресс-тестирования, а также понял концепцию этих тестирования в рамки проверок больших систем, ведь данный инструмент хорошо помогает понять, насколько данная нам система отказоустойчива в нужных нам рамках. По итогу получилось, что самая дешевая конфигурация осталась достаточной, для работы нашего сервера.