Санкт-Петербург
Университет ИТМО

Лабораторная работа №4 по «Тестированию программного обеспечения»

Нагрузочное и стресс-тестирование

Группа P33022
Братчиков Иван

2021

1. Цель работы

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

2. Задание

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:

- URL первой конфигурации ($5600):
  http://aqua:8080?token=466611586&user=1964508982&conf=1;
- URL второй конфигурации ($7100):
  http://aqua:8080?token=466611586&user=1964508982&conf=2;
- URL третьей конфигурации ($10400):
  http://aqua:8080?token=466611586&user=1964508982&conf=3;
- Максимальное количество параллельных пользователей - 11;
- Средняя нагрузка, формируемая одним пользователем 20 запр. в мин.;
- Максимально допустимое время обработки запроса - 620 мс.

3. Описание конфигурации JMeter

Общая конфигурация:

Для формирования кол-ва параллельных пользователей был создан компонент «Thread Group» с параметрами:

- Number of Threads (users): 11
- Ramp-Up Period (in seconds): 0
- Loop Count: Forever=true

Запросы формируются с помощью компонента «HTTP Request»:

- Server Name, Port Number, Protocol Parameters соответствуют таковым в URL
- Для разных конфигураций отличается GET параметр запроса conf

Для ограничения кол-ва запросов в минуту (RPM) используется «Constant Throughput Timer» с установленным:

- Target throughput (in samples per minute): 20.0

Чтобы иметь возможность обнаружить ошибки, связанные с большим временем ответа используется компонент Duration Assertion:

- Duration in miliseconds: 620

Возможность обнаруживать ошибки ответа обеспечивается «Response Assertion» с помощью кодов ответа протокола HTTP:

- Response Field to Test: Response Code
- Pattern Matching Rules: Matches
- Patterns to Test: 200

Графики пропускной способности формируются компонентом «Graph Results».
График изменения времени отклика построен с помощью компонента «Response Time Graph»:

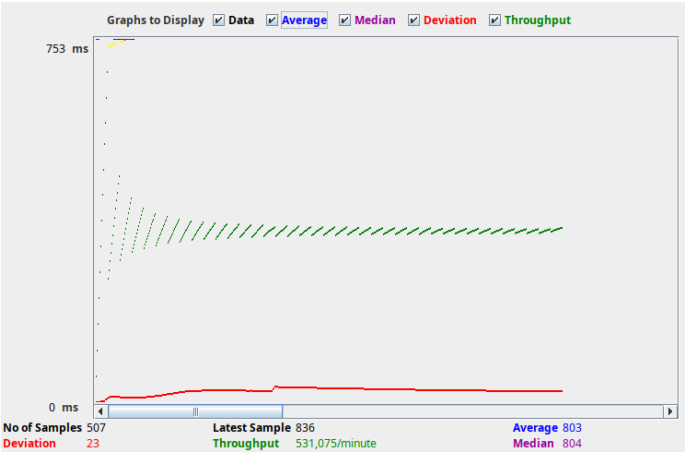
- Interval (ms): 1000

Конфигурация для стресс-тестирования:
Для стресс-тестирования применяется такая же конфигурация, но с изменениями в компоненте «Thread Group»:

- Number of Threads (users): 90
- Ramp-Up Period (in seconds): 90

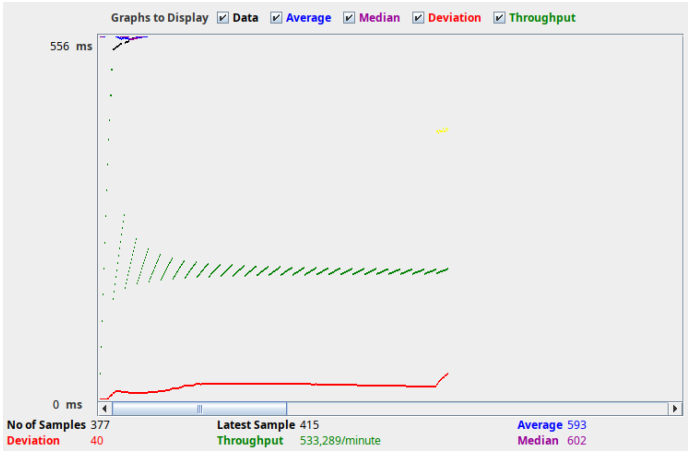
4. Нагрузочное тестирование

Конфигурация 1 ($5600):



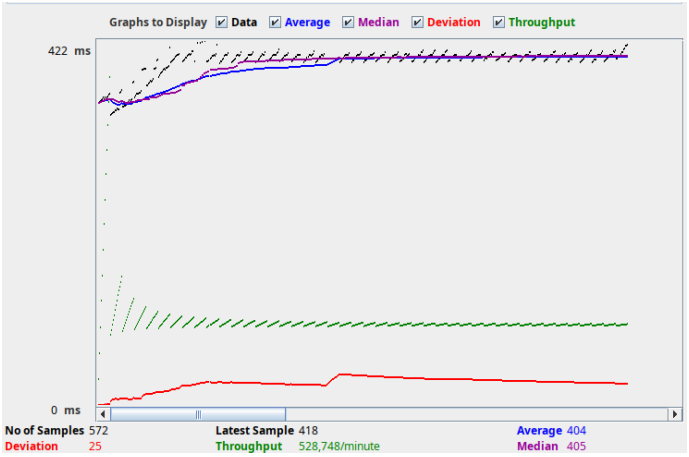
Среднее время ответа: ~800мс

Конфигурация 2 ($7100):



Среднее время ответа: ~590мс

Конфигурация 3 ($10400):



Среднее время ответа: ~400мс

На графиках присутствуют небольшие значения Deviation, которые в основном связаны с ошибками передачи данных ввиду опосредованного подключения к серверу с помощью SSH-туннелей.

Конфигурация 1 совсем не соответствует требованиям к нагрузке (570мс).

Время ответа конфигурации 2 удовлетворяет полностью и даже с большим запасом по времени ответа.

Конфигурация 3 тоже соответствует предъявляемым требованиям с запасом, но стоит больше.

Стресс тестирование будет проводиться по 2 конфигурации, так как она удовлетворяет требованиям и дешевле третьей.

4. Стресс-тестирование

График времени изменения времени отклика (до 600мс):

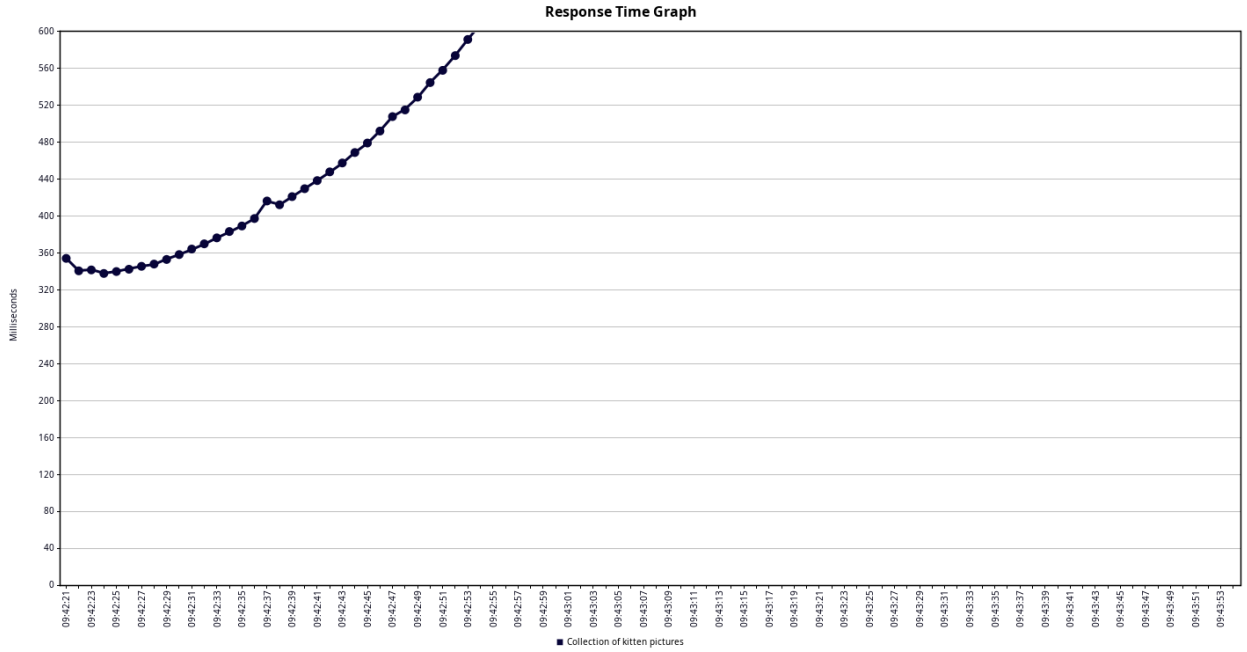
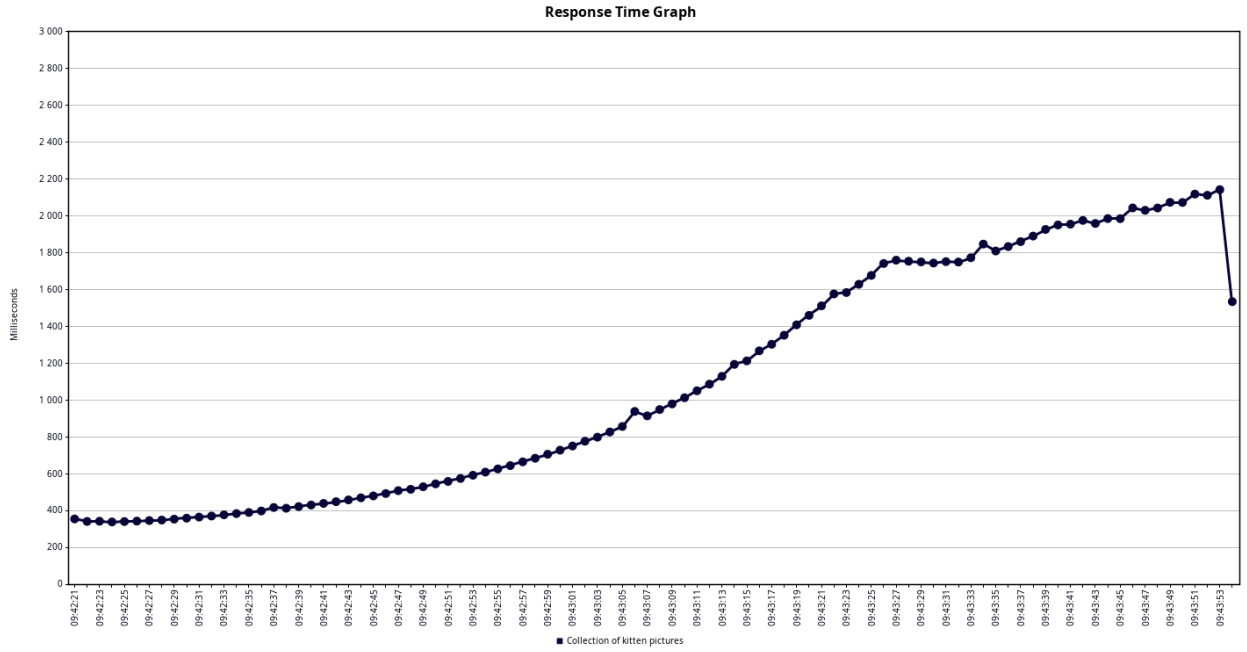


График времени изменения времени отклика (полный):



Так как графики строились с интервалом 1сек и «Ramp-Up Period» равен «Number of Threads», то каждая точка на графике показывает время отклика для разного кол-ва пользователей. Так можно установить, что конфигурация перестаёт соответствовать требованиям по времени отклика при 34-х пользователях со средним кол-вом запросов в минуту — 40.

5. Вывод

При выполнении лабораторной работы проводилось нагрузочное и стресс-тестирование программного обеспечения.

Нагрузочное тестирование позволяет проверить соответствие программного обеспечения, предъявляемым требованиям производительности (например, при

приёмочном тестировании), а также проверить поведение приложения при заданных условиях.

Стресс-тестирование позволяет определить максимальную производительность приложения в рамках данного аппаратного обеспечения. Также оно может быть полезно для определения системных требований программного обеспечения под нагрузкой.

В рамках лабораторной работы тестирование было синтетическим, но JMeter позволяет также производить тестирование с моделированием запросов, похожих на генерируемые реальными пользователями.