УДК 004.04

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ИНСТРУМЕНТОВ НАГРУЗОЧНОГО ТЕСТИРОВАНИЯ ДЛЯ СЕТЕВЫХ ПРИЛОЖЕНИЙ

Подсеваткина А.А., Попова Ю.Б.
БНТУ, г.Минск, Беларусь, podsevatkina.anastasia@gmail.com
БНТУ, г. Минск, Беларусь, julia_popova@mail.ru

Нагрузочное тестирование — это один из видов тестирования производительности, проводимый для того, чтобы определить поведение системы под определенной, заранее обозначенной нагрузкой.

При проведении нагрузочного тестирования, как правило, используется следующий набор показателей:

1) Емкость системы, т.е. предельная нагрузка, при которой система работает корректно. Единицей измерения являются запросы в секунду. Таким образом, емкость системы численно равна максимальной пропускной способности.

2) Время отклика, т.е. скорость, с который пользователи системы получают отклики на свои запросы. Единица измерения здесь — это миллисекунды. Времена отклика влияет на емкость системы. Например, если сервер отвечает не так часто, то емкость системы уменьшается.

3) Отказоустойчивость. Этот показатель заключается в том, насколько можно положиться на этот сервис, как он быстро восстановится в случае сбоя, как долго сможет работать до отказа.

4) Доступность — это время, в течение которого сервер находится в безотказном состоянии. Этот показатель вытекает из времени восстановления и времени наработки на отказ. В это понятие вливается инфраструктура, поскольку невозможно быть доступнее, чем текущий провайдер, даже, если сервис способен работать 7 дней в неделю и 24 часа в сутки.

5) Масштабируемость — это способность системы увеличивать свою производительность. Масштабируемость является важным аспектом систем, маршрутизаторов, сетей, если для них требуется возможность работать под большой нагрузкой. Система называется масштабируемой, если она способна увеличивать производительность пропорционально дополнительным ресурсам. Масштабируемость можно оценить через отношение прироста производительности системы к приросту используемых ресурсов. Чаще всего увеличение кластеров, обрабатывающих нагрузку, не приводит к линейному росту [1].

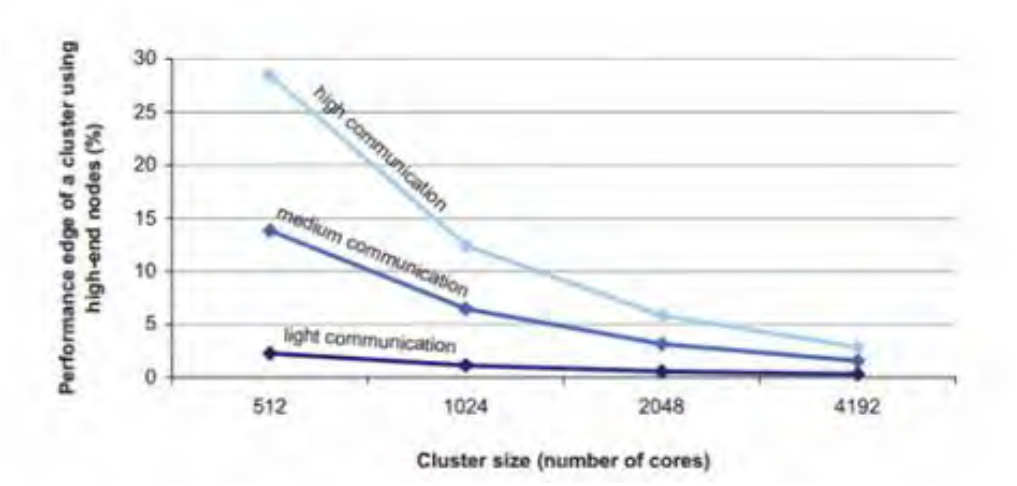

Рисунок 1 — Увеличение кластеров, отображающих нагрузку [1]

Существует довольно большой перечень инструментов для проведения нагрузочного тестирования. Рассмотрим некоторые из них.

1. Apache Jmeter — это самый распространенный инструмент для проведения нагрузочного тестирования, разрабатываемый под эгидой Apache Software Foundation. Изначально Jmeter разрабатывался как средство тестирования веб-приложений, однако в настоящее время он способен проводить нагрузочные тесты для JDBC-соединений, FTP, LDAP, SOAP, JMS, POP3, IMAP, HTTP и TCP [2].

Jmeter имитирует группу пользователей, посылающих запросы на целевой сервер и возвращает статистические данные, которые показывают производительность целевого сервера/приложения с помощью графических диаграмм.

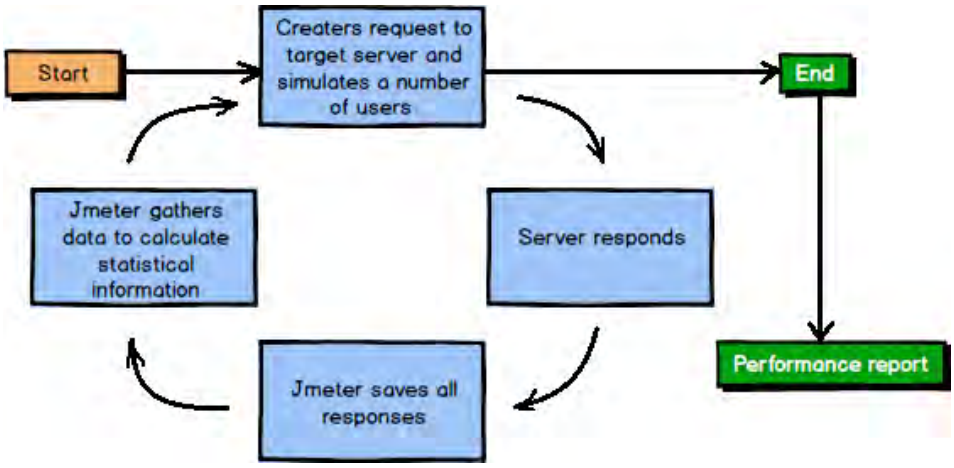

Рисунок 2 – Работа Jmeter [2]

Используя Jmeter, можно проводить автоматизированное функциональное тестирование приложений, тестирование производительности, тестирование сервера, сервера баз данных, веб-сервера и т.д. Также возможно расширение Jmeter за счет плагинов на Java [3]. В таблице ниже представлены преимущества и недостатки инструмента данного инструмента на основе информации из [2-5].

Таблица 1 — Преимущества и недостатки инструмента Jmeter

Преимущества	**Недостатки**
Является бесплатным инструментом и независимой платформой.	Отсутствие полноценного руководства пользователю.
Понятная структура тестов, возможность запускать тесты с любой машины.	Плагин для Jmeter, реализующий авторизацию oAuth, сильно устарел и не работает с oAuth 2.0. Небольшую часть функциональности, связанную с авторизацией через социальные сети, тестировать придется отдельно.
Поддерживается работа с переменными, регулярными выражениями, парсинг JSON, работа с cookie.	Интеграция Selenium с Jmeter представляет сложную задачу, которая требует глубокого понимания внутреннего устройства Jmeter.
Отсутствие необходимости	Сложности освоения для новичков вследствие не совсем

написания новых сценариев за счет модификации существующих.	дружественного интерфейса.
Простота установки.	
Кэширование и автономный анализ/повторное воспроизведение результатов испытаний.	
Визуализация результатов теста.	
Поддержка multi protocol.	

2. Мультипротокольный инструмент Tsung для нагрузочного тестирования написан с использованием платформы Erlang\OTP, которая предоставляет абстракции для простой реализации многопоточного исполнения программы. Tsung конфигурируется одним файлом в формате xml, в котором описываются все настройки и сценарии. Инструмент Tsung может имитировать нагрузку с нескольких клиентских машин, а также на одной клиентской машине несколько IP. Также может создавать дополнительные сессии и встраивать в середину нагрузки тяжеловесные сервисы и проверки. Основные преимущества и недостатки инструмента Tsung представлены в [6].

Таблица 2 — Преимущества и недостатки инструмента Tsung

Преимущества	**Недостатки**
Tsung может использоваться для тестирования различных протоколов HTTP, WebDAV, Jabber, LDAP, MySQL и т.д	Ручная установка Tsung из репозитория не является простой и прозрачной. В масштабах нескольких машин она требует определенных профессиональных навыков.
Наличие возможности Tsung Recorder, позволяющей написать сценарий прямо из браузера. Для этого необходимо настроить браузер на прокси Tsung-а (порт 8090) и использовать файл, созданный Recorder-ом для дальнейшей конфигурации. Также существует возможность использовать запись, полученную с использованием инструмента Selenium.	Наличие лишних зависимостей. Например, для того чтобы установить Tsung необходимо использовать make (утилита, предназначенная для автоматизации преобразования файлов из одной формы в другую) + autotools + Perl. С одной стороны, набор такого программного обеспечения можно найти на любой Unix- подобной системе, однако Tsung требует для своей установки некоторые модули Perl, которые могут быть не представлены в дистрибутиве системы, а также требуемые версии autotools могут не подходить для сборки Tsung.

Имитация HTTP, т.е. поддерживаются методы GET/POST/PUT/DELETE/HEAD. Инструмент позволяет автоматически управлять cookie, в том числе и вручную.	Tsung — это огромные XML файлы. Ветвление в логике можно описывать через XML, однако это неудобно и непрактично. Поэтому задача по созданию сценариев с разнородной нагрузкой становится слишком сложной.
Имитация Jabber/XMPP, т.е. поддерживаются сообщения об аутентификации, регистрации и присутствии. Чат сообщения для офлайн и онлайн пользователей. Запросы синхронизации пользователей.	Tsung находится в состоянии застоя. За счет этого страдает поддержка новых протоколов, а также расширение для уже существующих. Одна из главных причин: он может некорректно поддерживать или обеспечивать только частичную поддержку протоколов, тестирование через которые актуально в 2016 году. Tsung не поддерживает веб-сокеты, а также имеет ограничения в работе с базами данных.
Нагрузка может быть распределена на кластере из клиентских машин.	
Свободно распространяемый продукт с открытым кодом.	

3) SOAP UI представляет веб-сервис и, в первую очередь, предназначен для автоматизированного тестирования. Нагрузочное тестирование в нем представлено недоработанным. В качестве протоколов реализации веб-сервисов представлены SOAP (Simple Object Access Protocol), REST (Representational State Transfer) и XML-RPC (XML Remote Procedure Call). Также в SoapUI имеется plugin Jenkins, который используется для непрерывной интеграции. Непрерывная интеграция — это практика разработки программного обеспечения, реализующая слияние рабочих копий в общую основную ветвь разработки несколько раз в день и выполнение частых автоматизированных сборок проекта для скорейшего выявления и решения интеграционных проблем. В SOAP UI используется язык программирования Groovy. В таблице 3 представлены преимущества и недостатки инструмента SOAP UI [7].

Таблица 3 — Преимущества и недостатки инструмента SOAP UI

Преимущества	**Недостатки**
Наличие графического интерфейса пользователя.	Веб-сервис является платным, а бесплатная версия имеет очень ограниченные возможности.
Возможность использования Groovy для разработки тестовых сценариев.	Потребление большого количества памяти, что часто приводит к зависанию приложения.
Возможность добавлять	Избыточность повторяемого кода вследствие того, что

java-библиотеки в тестовые сценарии.	тестовые случаи недоступны между проектами, т.е. в каждом проекте свои Visible Case.
	Нет возможности отладки.
	Минимальная настройка нагрузочных сценариев.
	Отсутствие полноценных отчётов и графиков.
	Неудобная настройка передачи данных для тестов.
	Возможность тестировать только веб-сервисы.

Проведя анализ указанных выше инструментов, было принято решение реализовать свой инструмент для проведения нагрузочного тестирования, включающий преимущества рассмотренных инструментов и исключая их недостатки. Разрабатываемый инструмент получил название **HisWrath** и представляет собой программное решение, которое, будучи основанным на базе последней версии Erlang\OTP, является золотой серединой между утилитами, созданными для тестировщиков и программистов. Для тестировщиков предоставлен базовый проблемно-ориентированный язык, который позволяет расширять стандартные сценарии HisWrath и добавлять условия в логику исполнения тестовых сценариев. Для программистов предоставлена возможность расширять встроенный проблемно-ориентированный язык, используя возможности метапрограммирования языка, программирования на Elixir. Также HisWrath не предоставляет полноценного графического интерфейса, что делает его независимым от графических библиотек, но при этом он предоставляет возможность генерации отчетов в формате JSON для последующего преобразования в любой удобный для клиента вид. Параметры отчета генерируются на базе общих показателей, которые измеряются в ходе нагрузочного тестирования, и которые были приведены в начале работы. Таким образом, преимущества и недостатки HisWrath могут быть сведены в таблицу ниже.

Таблица 4 — Преимущества и недостатки инструмента HisWrath

Преимущества	**Недостатки**
Бесплатный инструмент и независимая платформа.	Отсутствие графического интерфейса пользователя.
Понятная структура тестов, возможность запускать тесты с любой машины.	
Работа с переменными, регулярными выражениями, парсинг JSON, работа с cookie.	
Отсутствие необходимости разрабатывать новые тестовые сценарии вследствие возможности модифицировать существующие.	
Простота установки.	
Повторное воспроизведение результатов испытаний.	

Визуализация результатов теста.	
Возможность работать практически с любым протоколом, у которого имеется открытая спецификация.	

Заключение

Рассмотренные выше инструменты являются лишь небольшой частью арсенала тестировщика. Однако уже на приведенных выше примерах можно увидеть, что несмотря на длительное развитие и широкую функциональность, ни один из инструментов не предоставляет простой возможности для расширения, программирования поведения, доставки данных о проведенных тестах в различных форматах, полноценной возможности тестирования сетевых инфраструктур, свободно выходя за рамки сетевых протоколов 7 и 6 уровней модели OSI, в отличие от инструмента HisWrath, который планируется реализовать.

Список использованных источников

1. Distributed systems for fun & profit [Электронный ресурс] / M. Takada. – San Francisco, California, 2013. – Режим доступа: http://book.mixu.net/distsys/intro.html. – Дата доступа: 25.10.2016.
2. Halili, E. Apache Jmeter / E. Halili. – Packt Publishing Ltd, June 2008. – 82 p.
3. Erinle, B. Performance Testing with Jmeter – Second Edition / B. Erinle., April 2015. – 145 p.
4. Apache Jmeter [Электронный ресурс] / Apache Software Foundation. – North America, 1999-2016. – Режим доступа: http://jmeter.apache.org/. – Дата доступа: 25.10.2016.
5. Тематические истории, основанные на опыте компании JETRUBY [Электронный ресурс] / Отдел обеспечения качества. – Краснодар, Россия, 2016. – Режим доступа: http://jetruby.com/ru/blog/%D0%BF%D0%BE%D1%87%D0%B5%D0%BC%D1%83-%D0%BC%D1%8B-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D1%83%D0%B5%D0%BC-jmeter/. – Дата доступа: 25.10.2016.
6. Tsung's documentation [Электронный ресурс] / N. Niclausse. Niagara, 2013. – Режим доступа: http://tsung.erlang-projects.org/user_manual/index.html. – Дата доступа: 25.10.2016.
7. SoapUI by SMARTBEAR [Электронный ресурс] / Boston, 2016. – Режим доступа: https://www.soapui.org/load-testing/concept.html. – Дата доступа: 25.10.2016.