совпадает с уже выполненными на предыдущих итерациях.

2. Сгенерированные запросы отправляются сервису на выполнение.

3. Ответ от сервиса проверяется на соответствие JSON-схеме.

4. Если ответ соответствует JSON-схеме, значит ошибка не была выявлена, и необходимо произвести мутацию запроса. Мутация может выполняться как внутри поля, так и над структурой всего запроса целиком. Возврат к пункту 2.

5. Если ответ не соответствует JSON-схеме, считается, что была обнаружена ошибка. Ошибочный ответ запоминается для дальнейшей обработки.

6. Запускается алгоритм поиска гена, который вызывает ошибку. Для этого из запроса по очереди удаляются все гены (поля) и отправляются на выполнение. Если после удаления очередного гена запрос выполнился без ошибки, то удаленный ген либо сам, либо в комплексе с другими генами является причиной ошибки, производится мутация до тех пор, пока не будет выявлен набор генов, являющийся причиной ошибки. Данный набор добавляется в массив для дальнейшей обработки. Возврат к пункту 1.

В качестве критерия останова цикла 1 – 6 могут выступать временные рамки либо достижение определенного количества мутаций запроса, не приведших к ошибке. После цикла 1 – 6 из наборов генов, полученных в 6 пункте, генерируется минимальное число запросов, соответствующих JSON-схеме. Полученный набор тестовых случаев является результатом работы данного алгоритма.

Подобный алгоритм может быть полезен при генерации smoke-тестов, так как при поиске ошибок задействуется максимально возможное число параметров, следовательно, покрытие функционала тестами стремится к 100%. Значительную пользу может принести набор генов, вычисленный в 6 пункте, являющийся причиной ошибки. Программисту будет значительно проще локализовать ошибку в коде, имея запрос состоящий только из тех полей, которые необходимы для воспроизведения ошибки.

Данный алгоритм может быть использован для регрессионного тестирования: тестирование можно считать успешным при безошибочном выполнении изначального набора запросов и последующих, мутировавших определенное число раз.

Алгоритм можно рассматривать в контексте стратегии мутации при генерации более сложных тестовых сценариев (например, CRUD сценарий) с помощью эволюционных алгоритмов.

Список использованных источников:

1. Тестирование программного обеспечения / Сэм Канер, Джек Фолк, Енг Кек Нгуен / ДиаСофт – Москва, 2001 – 544 с.
2. Ю.А.Скобцов, Д.В.Сперанский. Эволюционные вычисления. Москва: ИНТУИТ, 2014.
3. T. Mantere, J.T. Alander Evolutionary software engineering, a review // Applied Soft Computing 5 (2005) pp.315–331.

АНАЛИЗ ЭФФЕКТИВНОСТИ РАБОТЫ СИСТЕМЫ РАСПРЕДЕЛЕННОЙ ПЕРЕДАЧИ ДАННЫХ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Разумов Е.В.

Иванюк А.А. – д-р. техн. наук

Одним из возможных методов, по средствам которого можно достичь большую устойчивость к MITM-атакам, является метод построения системы распределенной передачи данных по средствам peer-to-peer сети. Основная идея, заложенная в данную систему, заключается в разбиении передаваемой информации на блоки, шифровании каждого из этих блоков и последующей передаче их через отдельный промежуточный узел.

Оценивать эффективность работы такой системы целесообразно по нескольким позициям: скорости работы системы в целом, криптостойкости такого способа пересылки данных, зависимости от внешних факторов, преимуществах перед существующими решениями.

С точки зрения скорости, передача данных между адресатами будет значительно медленнее, чем в системах, которые передают данные от отправителя к получателю напрямую. Зависимость времени передачи данных между адресатами от количества пройденных промежуточных узлов не является линейной. Это объясняется в первую очередь тем, что в реальной системе кроме затрат времени, которые необходимы непосредственно для операций шифрования/расшифровывания, необходимо еще время на передачу блоков зашифрованных данный от одного узла к другому. И это время не является постоянной величиной и зависит от качества сети.

Следующей позицией, по которой можно оценить эффективность разработанной системы, является скорость передачи данных между адресатами в зависимости от числа разбиений исходных данных при условии, что данные передаются через один промежуточный узел. Анализируя результаты, полученные практическим путем, можно сделать вывод, что с увеличением числа разбиений скорость работы системы в целом уменьшается. Причиной тому являются следующие факторы:

- скорость работы сети на каждом из узлов;

– производительность промежуточного узла;
– загруженность промежуточного узла;
– дополнительные затраты времени отправителем на разбиение исходных данных на блоки, шифрование каждого из блоков, установление сессий с каждым из промежуточных узлов;
– затраты времени конечным адресатом на склейку пришедших блоков.

Следовательно в случаях, когда необходимо увеличить криптостойкость системы к MITM-атакам, целесообразно делить исходные данные на большее число пусть и небольших блоков. В противном случае число разбиений исходных данных должно быть в пределе 2-4, что позволит избежать сильного замедления работы системы.

Еще одной немаловажной характеристикой данной системы является производительность каждого из узлов в зависимости от размера передаваемого через него блока. Эта зависимость также не является линейной. Узел способен обработать большее количество байтов информации в единицу времени, если она передается большими блоками. Это связано с тем, что для обработки каждого из блоков информации необходимо устанавливать защищенную сессию с другим узлом сети и инициализировать криптосервисы для обработки полученного блока.

Дополнительным параметром при оценке эффективности разработанной системы является также величина затраченных ресурсов для передачи данных по средствам peer-to-peer сети. Узлы данной сети могут быть использованы не только для ретрансляции зашифрованных блоков данных между адресатами, но и для дополнительных задач, которые решаются при помощи построения peer-to-peer сети. Таким образом нельзя сказать, что зависимость затрат электроэнергии (или интернет-трафика) прямопропорциональна числу одновременно работающих узлов сети. Узел такой сети изредка будет использоваться для ретрансляции (или же передачи/получения) блоков данных, а в оставшееся время будет вести себя как будто это приложение и не было установлено.

На основании проведенных исследований установлено, что разработанную систему целесообразно применять в случаях, когда ключевым параметром является безопасность передачи данных, а скорость передачи при этом не имеет такого весомого значения, так как она всегда будет заметно ниже, чем у классических методов.

Список использованных источников:

1. Trappe, Wade (2005). Introduction to Cryptography with Coding Theory. New York: Pearson. p. 257.
2. Network Forensic Analysis of SSL MITM Attacks – NETRESEC Network Security Blog. Retrieved March 27, 2011.

МОДЕЛИ КАЧЕСТВА КЛИЕНТСКОЙ И СЕРВЕРНОЙ ЧАСТЕЙ ВЕБ-ПРИЛОЖЕНИЙ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Резванов А. В.

Бахтизин В. В. – к-т. техн. наук, профессор

В настоящее время вопрос оценки качества веб-приложений является актуальным, т.к. в современной экономике веб-приложение может выступать в качестве дополнительного канала связи между промышленными предприятиями, организациями и их клиентами (B2C) или для связи между предприятиями (B2B).

При оценке качества веб-приложения его можно разделить на клиентскую часть и серверную часть. Клиентская часть включает файлы гипертекстовой разметки HTML, стили CSS и код JavaScript. Серверная часть включает код на таких языках программирования, как Java, C#, PHP, Python и других.

Оценка качества веб-приложений может быть основана на международных стандартах качества, таких как стандарт ISO/IEC 25010:2011 [1]. В данном стандарте 8 характеристик качества. При этом некоторое подмножество этих характеристик является более важным для клиентской части, в то время как другое подмножество более характерно для серверной части.

Для клиентской части важными характеристиками являются функциональная пригодность, совместимость, удобство использования, надежность и сопровождаемость (рис. 1).

Для серверной части важными являются функциональная пригодность, эффективность функционирования, совместимость, надежность, защищенность, сопровождаемость и мобильность (рис. 2).

Жирным шрифтом выделены наиболее важные характеристики.

Надежность клиентской части включает подхарактеристику завершенности, в то время как для серверной части кроме завершенности имеют значение такие подхарактеристики как готовность, отказоустойчивость и восстанавливаемость.

Характеристика удобства использования клиентской части включает такие подхарактеристики, как