Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Саратовский государственный технический университет

имени Гагарина Ю.А.»

Институт электронной техники и приборостроения

Кафедра Информационная безопасность автоматизированных систем

Специальность 10.05.03 Информационная безопасность автоматизированных систем

**КУРСОВОЙ ПРОЕКТ**

по дисциплине «Разработка и эксплуатация защищенных автоматизированных систем»

по теме **«**Предотвращение уязвимостей при разработке веб-приложений»

|  |  |
| --- | --- |
|  | Выполнили: студент 4 курса  учебной группы с-ИБС42  очной формы обучения  Солодилов Владимир Владимирович \_\_\_\_\_\_\_\_\_\_\_  Руководитель проекта:  доцент. каф. ИБС Ершов А.С.  Комиссия по защите:  доцент. каф. ИБС Ершов А.С.  доцент каф. ИБС Шелудяков Д.А. |

Курсовой проект защищен на оценку \_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата, подпись члена комиссии)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата, подпись члена комиссии)

Саратов 2022

**Введение**

В рамках курсового проекта ставится задача на проведение нагрузочного тестирования с целью выявления предельного уровня нагрузки, при котором приложение остаётся работоспособным, а также тестирование безопасности разработанного веб-приложения.

Для проведения нагрузочного тестирования используется приложение Apache Jmeter, с помощью которого возможно автоматизировать производимые пользователями действия, а также задача их число и время, в течение которого будет осуществляться тестирование. Результаты тесты возможно визуализировать в виде графиков и таблиц, что будет произведено далее.

При анализе безопасности разработанного веб-приложения производится тестирование БД на уязвимости средствами интегрированной среды Microsoft SQL Management Studio, а также самого приложения с помощью приложения OWASP ZAP, которое осуществляет атаку на приложение, после чего выводит результат тестирования и предоставляет информацию по исправлению выявленных уязвимостей.

**Тестирование веб-приложения**

1. Нагрузочное тестирование

Для начала выполним нагрузочное тестирования веб-приложения, используя приложение Apache JMeter. В качестве рабочей задачи были установлены вход пользователя в систему, внесение изменений в БД посредством заполнения соответствующей формы и выход пользователя из системы.

В качестве рабочей станции используется домашний ПК на базе AMD Ryzen 7 1700 и 16Gb DDR4. Полученные результаты выводятся на экран и используются для дальнейшего анализа.

Для выполнения поставленной задачи создадим группу пользователей и определим для неё нагрузочные тесты и графики вывода конечных результатов:

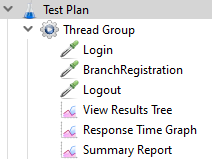


Рисунок 1 – Подключённые модули для тестирования

Нагрузочное тестирование проведём для трёх режимов: обычной нагрузки, повышенной нагрузки и предельной нагрузки.

В качестве обычной нагрузки определим 20 пользователей, которые совершат вышеуказанные действия в течение 60 секунд:

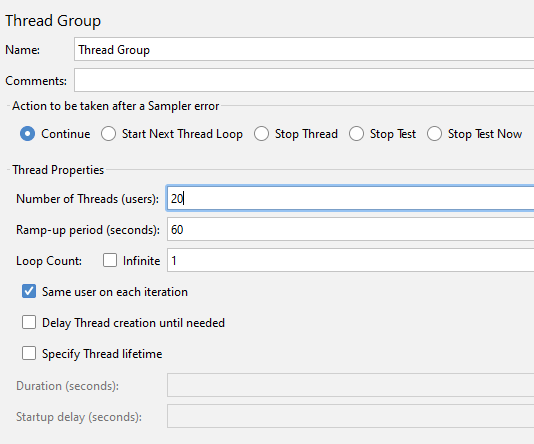


Рисунок 2 – Конфигурация тестирования 1

По окончанию выполнения тестирования получаем, что наибольшее время системы занимает процесс входа пользователя в систему, наименьший – выход из системы. Представим полученные результаты в виде графиков:

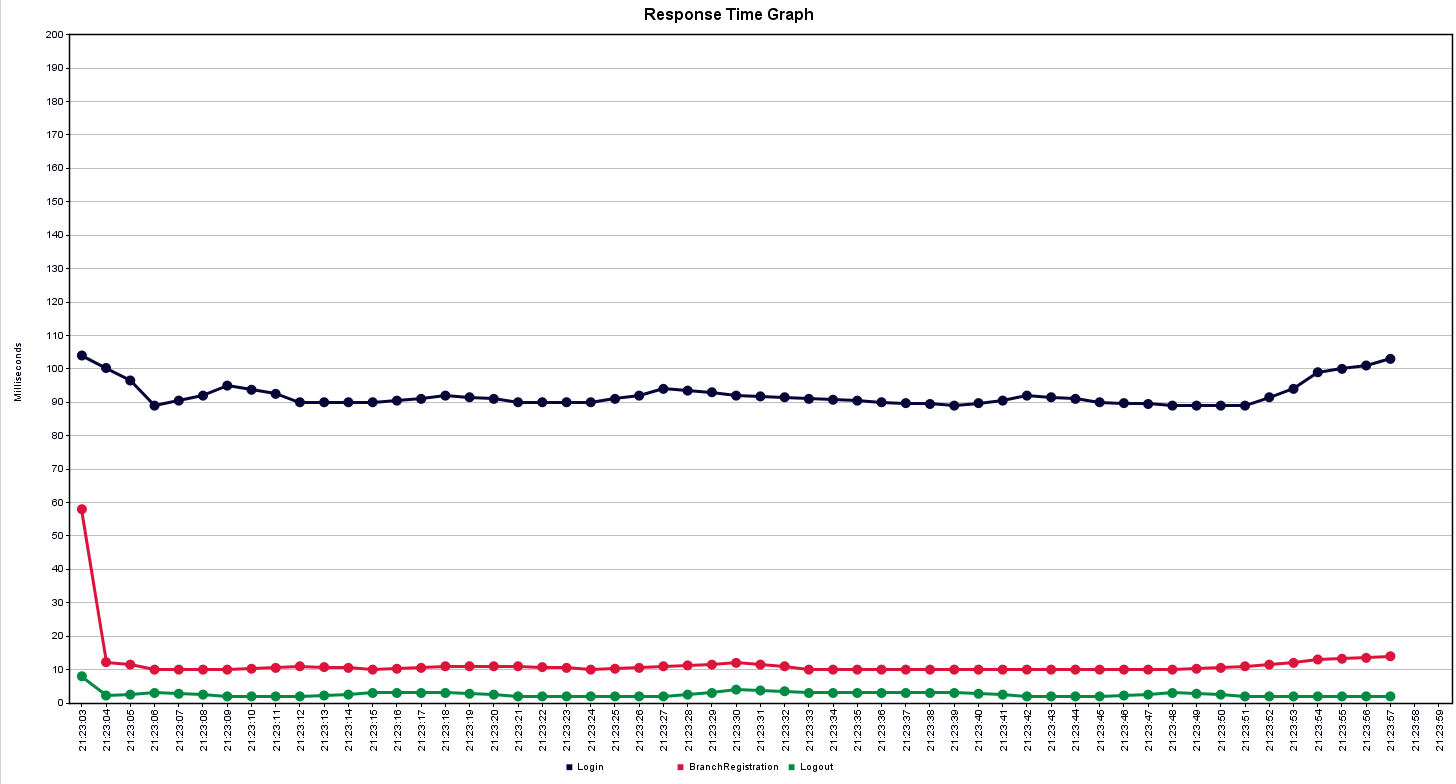


Рисунок 3 – График времени отклика приложения при обычной загрузке

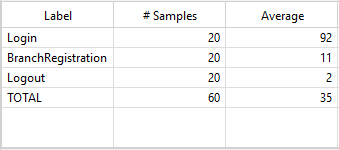


Рисунок 4 – Таблица среднего времени отклика приложения при обычной загрузке

На основе представленных графиков можно сделать вывод, что среднее время отклика приложения составило 35мс, при том, что наиболее сложная для системы задача выполняется в среднем 92мс, а наименее – 2мс.

Повысим нагрузку на приложение, увеличив число пользователей до 60 при том же времени в 60 секунд:

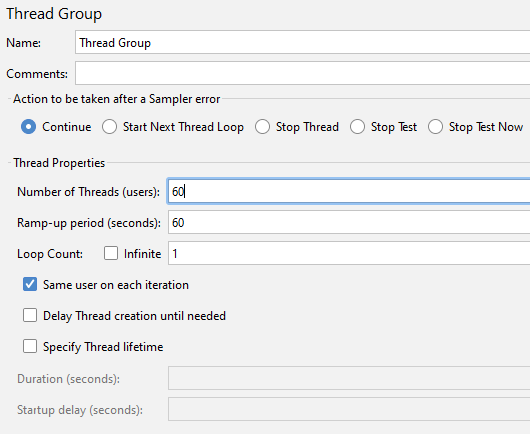


Рисунок 5 – Конфигурация тестирования 2

По окончанию тестирования получаем приблизительно такие же результаты, что и в первом случае, однако на графике времени задержки заметны определённые пики, характеризирующие о появлении очереди при обработке запросов:

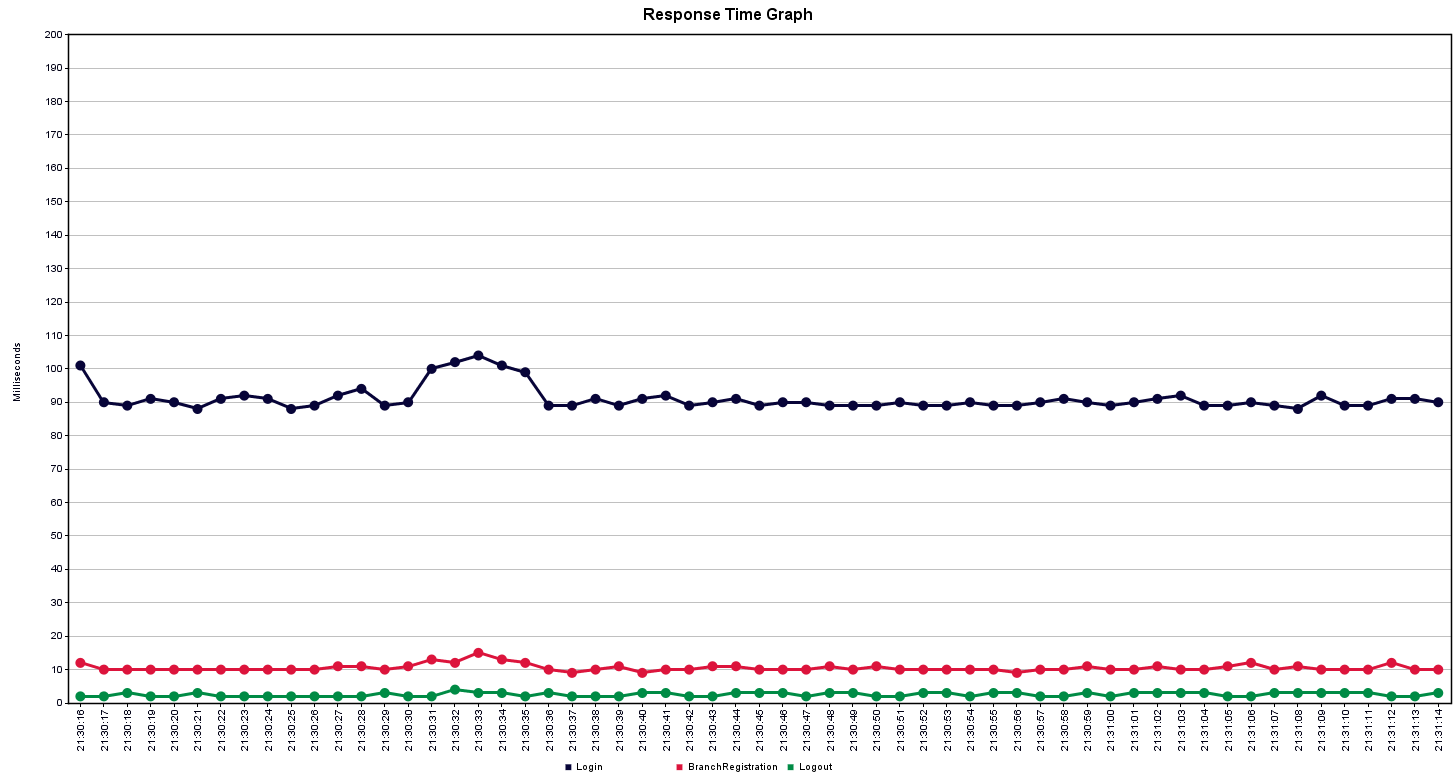


Рисунок 6 – Графики времени задержки при повышенной загрузке приложения

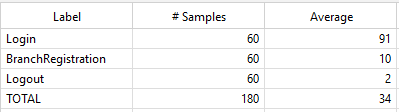


Рисунок 7 – Таблица среднего времени задержки для повышенной нагрузке

Предельной нагрузкой считаем 5000 пользователей, выполняющих поставленные задачи в течение тех же 60 секунд. По результатам выполнения тестирования видны серьёзные изменения по сравнению с предыдущими случаями: увеличилось среднее время обработки запросов до 84мс, а наиболее трудоёмкая задача может занимать до 710мс времени системы. Это выражается в сильном скачке, который на графике представляется в виде пика:

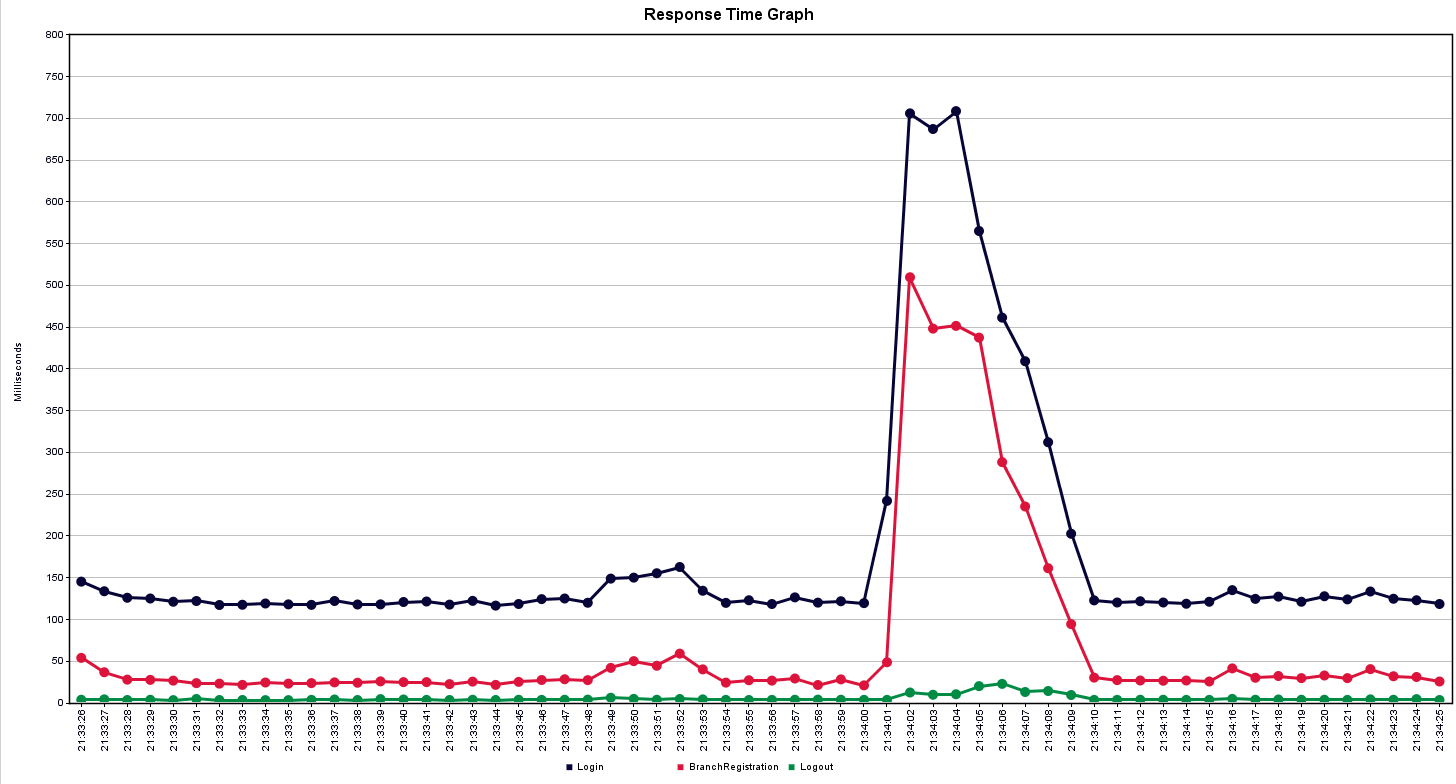


Рисунок 8 – График времени отклика при предельной нагрузке

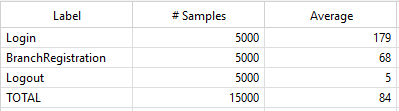


Рисунок 9 – Таблица среднего времени отклика при предельной нагрузке

Обобщая, можно сказать, что система выдержала поставленную на неё нагрузку, оставшись доступной для дальнейшего использования.

1. Тестирование на уязвимости
2. Тестирование БД

Тестирование проводилось встроенными средствами интегрированной среды Microsoft SQL Management Studio, которая после анализа выдала следующие результаты:

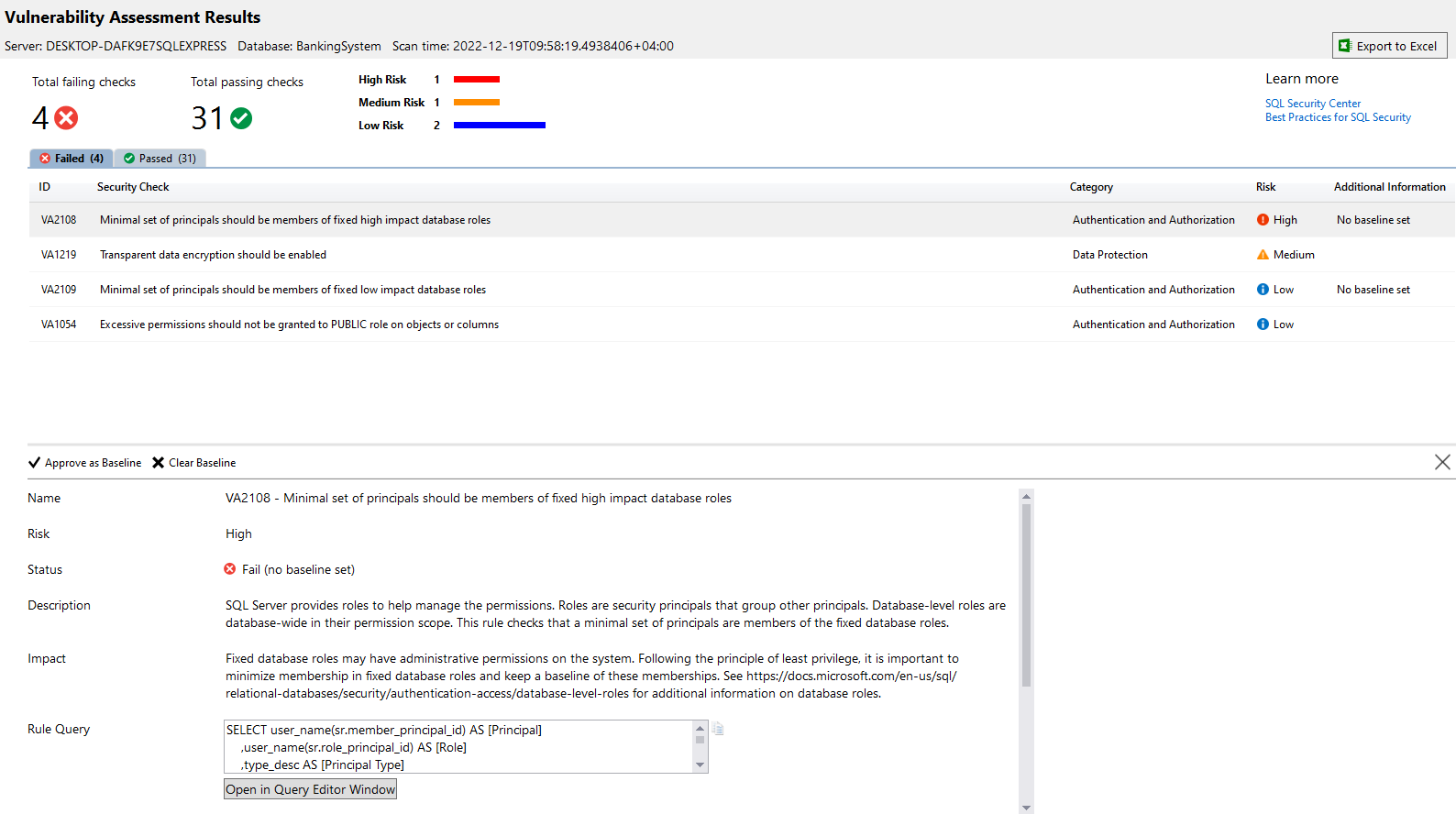
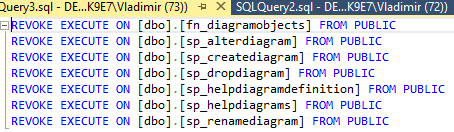


Рисунок 10 – Тестирование БД на уязвимости до их исправления

Проанализировав данные уязвимости, можно сказать, что они связаны с неправильным назначением прав доступа для БД и отдельных таблиц. Исправим данные уязвимости, выполнив запросы из полученной справки:







После выполнения запросов произведём повторное тестирование БД на наличие уязвимостей, в результате которого видно 1 сообщение об отсутствии шифрования БД. Данную уязвимость исправить в редакции Express Microsoft SQL Server не представляется возможным, так как она не поддерживает данный функционал:

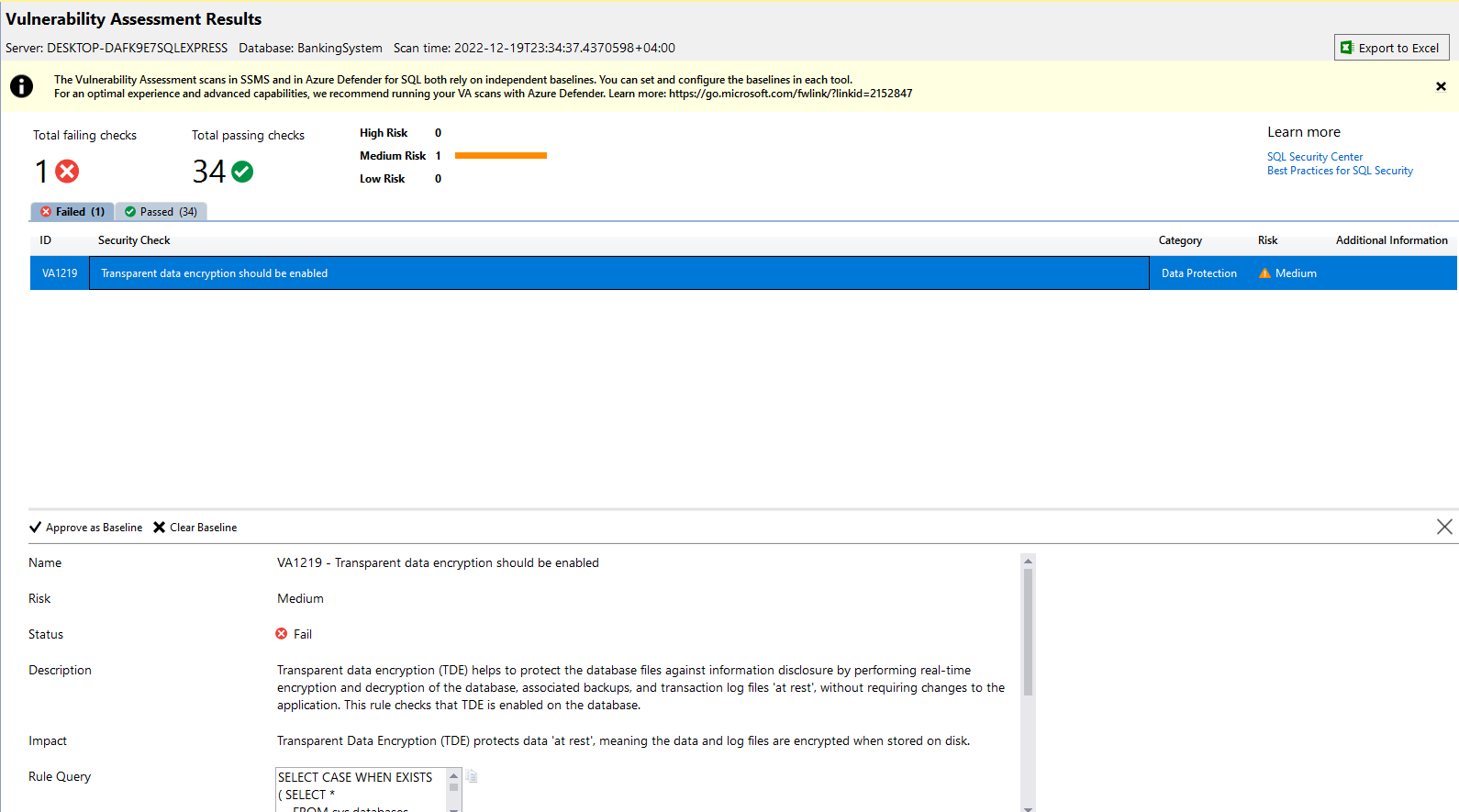
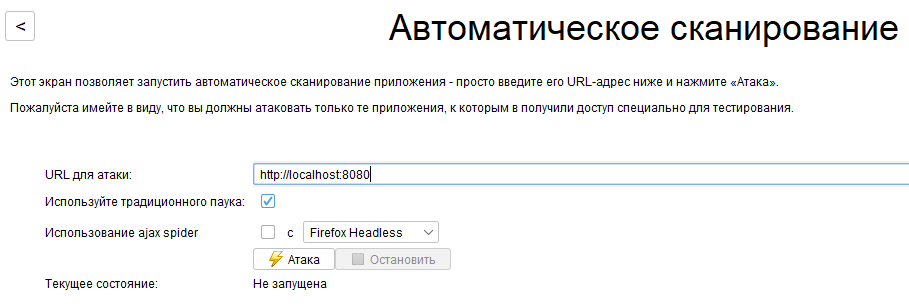


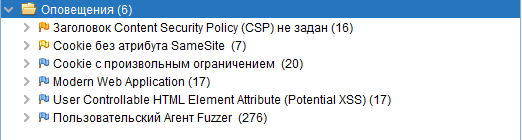
Рисунок 11 – Тестирование БД на уязвимости после их исправления

1. Тестирование веб-приложения

Для тестирования веб-приложения использовалось приложение OWASP ZAP, которое сконфигурировано для анализа уязвимостей, после чего запущено:

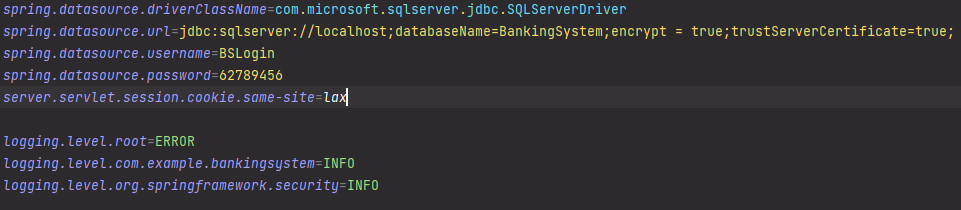


В результате тестирования были выявлены некритические уязвимости, представленные ниже:



Для исправления данных уязвимостей произведена дополнительная конфигурация файла с Spring Security, в который добавлена поддержка CSP, а также в файл application.properties установлен слабый режим слежения за куками:





При повторном сканировании данных проблем нет:

