

Разработка практико-ориентированного занятия по информационной безопасности на основе уязвимого веб-приложения

Болтовский Гавриил Александрович

Приамурский государственный университет им. Шолом-Алейхема

Студент

Аннотация

Статья описывает конструирование 90-минутного практического занятия «Методы и средства защиты информации» на базе преднамеренно уязвимого веб-приложения. Представлены четыре последовательных задания: эксплуатация SQL-инъекции и дампы БД, переход от открытых паролей к хэшированию, поиск и устранение уязвимости в модуле комментариев, демонстрация port knocking для скрытого SSH-доступа в изолированной среде. Сформулированы требования к наблюдаемости и приёме результатов «до/после», что обеспечивает воспроизводимость учебного сценария в лаборатории.

Ключевые слова: информационная безопасность, учебная песочница, уязвимое веб-приложение, SQL-инъекция

Development of a practice-oriented lesson on information security based on a vulnerable web application

Boltovsky Gavriil Alexandrovich

Sholom-Aleichem Priamursky State University

Student

Abstract

The article describes the design of a 90-minute practical lesson entitled “Methods and Means of Information Protection” based on a deliberately vulnerable web application. Four sequential tasks are presented: exploiting SQL injection and database dumping, transitioning from open passwords to hashing, finding and eliminating vulnerabilities in the comments module, and demonstrating port knocking for hidden SSH access in an isolated environment. Requirements for observability and acceptance of “before/after” results are formulated, ensuring the reproducibility of the training scenario in the lab.

Keywords: information security, training sandbox, vulnerable web application, SQL injection

1 Введение

1.1 Актуальность исследования

В образовательных курсах по ИБ наиболее остро ощущается дефицит простых, быстро разворачиваемых и воспроизводимых практик, которые

связывают теорию угроз с наблюдаемыми эффектами в коде и инфраструктуре. Учебный сценарий, где студент сначала воспроизводит атаку, а затем фиксирует изменение поведения системы после исправления, формирует понимание причинно-следственных связей и минимальный набор инженерных привычек безопасной разработки.

1.2 Обзор исследований

Корпус источников задаёт методические и технологические ориентиры для интеграции учебной песочницы по информационной безопасности в практикум. Б. А. Швырев описывает [1] образовательные потребности в области ИБ, подчёркивая непрерывность подготовки и вариативность методов, что определяет модульную организацию занятий и адаптацию заданий под аудиторию. Н. А. Шутько систематизирует [2] базовые архитектурные принципы веб-приложений и типологию угроз; эти положения задают рамки отбора кейсов и формулировки заданий по анализу и укреплению компонентов приложения. Г. В. Беликов, И. Д. Крылов, В.А.Селищев [3] классифицируют природу SQL-инъекций и подходы к противодействию; соответствующие материалы транслируются в задания по обходу авторизации и последующему устранению уязвимости, включая переход от хранения паролей в открытом виде к хэшированию. И. А. Лубкин [4] вводит метрики оценки средств защиты от атак возвратно-ориентированного программирования; такие показатели могут использоваться для формализации оценивания результатов практикума. Г. А. Воробьёв, И. И. Павленко обосновывают [5] формат ситуационного центра и связку с профильными лабораториями, где программная песочница выступает стендом. И. В. Капгер, Е. Е. Журилова, А. А. Миронова конкретизируют [6] инфраструктурные и регламентные требования, включая пошаговые инструкции и контроль доступа, релевантные развертыванию стенда на Flask в Docker-среде.

1.3 Цель исследования

Описать и структурировать воспроизводимое практическое занятие по выявлению и устранению типовых уязвимостей веб-систем с фиксируемыми результатами «до/после» в контролируемой среде.

1.4 Постановка задачи

В ходе работы были определены учебные цели для четырёх заданий, разработаны пошаговые инструкции и критерии приёмки, установлен порядок логирования и фиксации «следов» атак, заданы временные рамки этапов занятия и регламент обратной связи, сформированы требования к лабораторной среде и правилам сброса состояния между попытками.

2 Методы исследования

В качестве серверной основы выбран Flask: он опирается на широко используемый студентами язык, даёт прозрачную маршрутизацию и позволяет

изолировать «точки атаки» на уровне обработчиков, что удобно для учебных диффов «уязвимо/исправлено». Для изоляции и воспроизводимости использован Docker: фиксированные версии зависимостей, замкнутая сеть и быстрый сброс состояния исключают влияние на внешние системы и сокращают неучебные накладные расходы. Реляционная СУБД применена из-за наглядности SQL-профиля уязвимостей и простоты демонстрации перехода от конкатенации к параметризованным запросам. Для защиты секретов выбран алгоритм классов `bcrypt/argon2`: наличие соли и настраиваемой сложности позволяет объяснить недетерминированность хеша и трудоёмкость перебора. Модуль комментариев спроектирован как отдельный компонент, что облегчает иллюстрацию серверной валидации и автоэкранирования при выводе. Для демонстрации минимизации «поверхности атаки» на уровне сети применён `port knocking (knockd/iptables)`: последовательность «стуков» открывает ограниченное окно доступности SSH-порта, что делает эффект проверки легко наблюдаемым (скан/подключение до/после). Наблюдаемость обеспечена структурированным логированием ключевых событий (аутентификация, SQL-ошибки, операции с комментариями, изменения сетевых правил) и контрольными полезными нагрузками, повторно применяемыми после исправлений. Само учебное приложение доступно в своём GitHub репозитории.

3 Результаты и обсуждение

Собранный сценарий обеспечивает линейную прогрессию трудности и измеримость результатов: уязвимая аутентификация воспроизводимо обходится SQL-инъекцией, после исправления попытка даёт отказ и чистые журналы; переход к хэшированию устраняет риск чтения паролей из дампа, что подтверждается недетерминированными хешами и изменением логики проверки; модуль комментариев перестаёт позволять подмену авторства и внедрение опасных последовательностей, контент корректно нормализуется и экранируется; `port knocking` делает служебный порт невидимым без правильной последовательности и доступным в коротком окне после «стуков», что фиксируется как на стороне клиента, так и в логах системы фильтрации. Все шаги выполняются в изолированной среде, а критерии приёма заданы как наблюдаемые технические эффекты, что облегчает контроль и повторение результатов в аудитории.

В предлагаемой структуре занятие разворачивается в изолированной среде с предсказуемыми условиями запуска и сброса состояния; сценарий намеренно удерживается простым, чтобы внимание фокусировалось на механике атак и исправлений. На первом этапе предполагается демонстрация типового поведения уязвимой аутентификации: подстановка логического предиката в параметры формы входа теоретически делает возможным вход в административный раздел; после перехода к параметризованным обращениям и нормализации ввода те же действия должны приводить к отказу в доступе и отсутствию характерных следов некорректного SQL в журналах. Далее рассматривается обращение с секретами: исходная модель хранит пароли в

открытом виде, тогда как переход к хэшированию с солью и проверке на этапе входа иллюстрирует недетерминированность хеша и практическую неосуществимость восстановления исходного значения. Третий блок касается компонента комментариев, где уязвимая реализация допускает подмену авторства и вмешательство в структуру запросов; перенос проверки на серверную сторону, нормализация ввода и отказ от конкатенации предполагают нейтрализацию этих эффектов и отображение потенциально опасных последовательностей как обычного текста. Завершающий эпизод посвящён снижению сетевой видимости: метод port knocking открывает кратковременное окно служебного доступа после корректной последовательности обращений и закрывает его по обратной, что удобно фиксировать средствами клиентской диагностики и системных журналов фильтрации. Во всех эпизодах ключевая идея состоит в сопоставлении одного и того же действия до и после изменения кода или конфигурации, благодаря чему причинно-следственные связи остаются наблюдаемыми и проверяемыми без выхода за пределы тестовой среды; ограничения сознательно охватывают узкий класс рисков и не претендуют на полноту производственного профиля.

4 Выводы

Предложена воспроизводимая структура 90-минутного практического занятия по ИБ на базе уязвимого веб-приложения; определены критерии наблюдаемости «до/после» и требования к изолированной среде, что обеспечивает проверяемость результатов без риска для внешней инфраструктуры.

Библиографический список

1. Швырев Б. А. Образовательные потребности в информационной безопасности // Международный журнал гуманитарных и естественных наук. 2019. №. 5-1. С. 31-33.
2. Шутько Н. А. Теоретические понятия защиты WEB-приложений от уязвимостей // Вестник науки. 2022. Т. 4. №. 11 (56). С. 253-269.
3. Беликов Г. В., Крылов И. Д., Селищев В. А. SQL-инъекция как способ обхода авторизации // Известия Тульского государственного университета. Технические науки. 2021. №. 12. С. 217-221.
4. Лубкин И. А. Доклады Томского государственного университета систем управления и радиоэлектроники // Доклады Томского Государственного университета систем управления и радиоэлектроники Учредители: Томский государственный университет систем управления и радиоэлектроники. 2021. Т. 24. №. 4. С. 46-51.
5. Воробьев Г. А., Павленко И. И. Ситуационный центр в обучении техническим и гуманитарным аспектам информационной безопасности // Информационное противодействие угрозам терроризма. 2015. Т. 1. №. 25. С. 117-122.
6. Капгер И. В., Журилова Е. Е., Миронова А. А. О разработке учебно-

лабораторного стенда для изучения аппаратного модуля доверенной загрузки «Аккорд» // Вестник Пермского национального исследовательского политехнического университета. Электротехника, информационные технологии, системы управления. 2016. №. 17. С. 131-142.

7. GitHub course_materials URL:
https://github.com/Gavriilbolt/course_materials/tree/main/01_basic_info_security/Prac%231software (дата обращения: 03.09.2025).