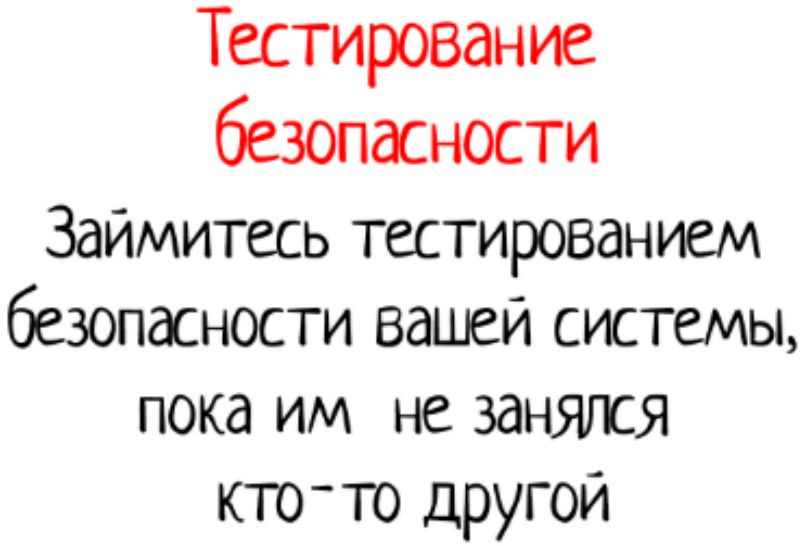
Лекция 13

Тестирование безопасности



Цели и задачи тестирования безопасности веб приложений

> *54% приложений на российском рынке содержат хотя бы одну критичную уязвимость, которая может стать точкой входа для хакеров (данные за 2023г. отсюда).*
>
> *Чтобы избежать таких проблем, важно внедрить цикл безопасной разработки ПО с применением инструментов анализа кода на разных этапах.*

Основной **целью тестирования безопасности веб-приложений является проверка системы на наличие уязвимостей**, которые могут быть использованы злоумышленниками для доступа к конфиденциальной информации или атаки на приложение.

Некоторые из задач тестирования безопасности веб-приложений включают:

- ✓ Идентификация уязвимых мест в приложении
- ✓ Оценка уровня риска для системы при наличии уязвимостей
- ✓ Проверка соответствия веб-приложения стандартам безопасности
- ✓ Проверка правильности работы системы защиты от атак
- ✓ Проверка корректности обработки ошибок в приложении

Основные угрозы безопасности веб-сайтов

SQL-инъекции

SQL-инъекции являются одной из самых распространенных уязвимостей. Злоумышленники могут вставлять вредоносные SQL-запросы в поля ввода, чтобы получить доступ к базе данных. Это может привести к утечке конфиденциальной информации, изменению данных или даже полному разрушению базы данных. Например, злоумышленник может использовать SQL-инъекцию для получения доступа к таблице пользователей, содержащей имена, пароли и другую личную информацию. Это может привести к серьезным последствиям, таким как кража личных данных и финансовых потерь.

Для предотвращения SQL-инъекций рекомендуется использовать подготовленные запросы (prepared statements) и ORM (Object-Relational Mapping) библиотеки. Эти методы позволяют избежать прямого включения пользовательского ввода в SQL-запросы, что значительно снижает риск инъекций.

Основные угрозы безопасности веб-сайтов

XSS (Cross-Site Scripting)

XSS-атаки позволяют злоумышленникам внедрять вредоносный скрипт в веб-страницы, которые затем выполняются на стороне пользователя. Это может привести к краже сессий, данных пользователей или даже к полному контролю над учетной записью. Например, злоумышленник может внедрить скрипт, который перехватывает куки-файлы пользователя и отправляет их на удаленный сервер. Это позволяет злоумышленнику получить доступ к учетной записи пользователя без его ведома.

Для защиты от XSS-атак рекомендуется использовать методы экранирования (escaping) и валидации пользовательского ввода. Также важно использовать Content Security Policy (CSP), которая ограничивает выполнение скриптов на веб-странице.

Основные угрозы безопасности веб-сайтов

CSRF (Cross-Site Request Forgery)

CSRF-атаки заставляют пользователя выполнять нежелательные действия на сайте, на котором он аутентифицирован. Это может привести к изменению данных, отправке нежелательных запросов или даже к выполнению административных действий. Например, злоумышленник может создать ссылку, которая при нажатии изменяет пароль пользователя на сайте, на котором он аутентифицирован.

Для защиты от CSRF-атак рекомендуется использовать токены CSRF. Эти токены генерируются сервером и включаются в каждую форму или запрос, требующий аутентификации. Сервер проверяет наличие и корректность токена перед выполнением запроса.

Основные угрозы безопасности веб-сайтов

Уязвимости в аутентификации и управлении сессиями

Неправильное управление сессиями и аутентификацией может привести к захвату учетных записей. Злоумышленники могут использовать украденные сессии или слабые пароли для получения несанкционированного доступа. Например, если сессии пользователей не защищены должным образом, злоумышленник может перехватить сессионный идентификатор и использовать его для доступа к учетной записи пользователя.

Для защиты от уязвимостей в аутентификации и управлении сессиями рекомендуется:

- ✓ использовать безопасные методы хранения паролей, такие как хеширование с солью (salted hashing)
- ✓ использовать безопасные сессионные идентификаторы и ограничивать время жизни сессий.

Основные угрозы безопасности веб-сайтов

Уязвимости в конфигурации

Неправильная конфигурация серверов и приложений может предоставить злоумышленникам доступ к конфиденциальной информации или позволить им выполнять нежелательные действия. **Это может включать в себя неправильные настройки прав доступа, открытые порты и уязвимые версии программного обеспечения.** Например, если сервер настроен таким образом, что позволяет доступ к административным панелям без аутентификации, злоумышленник может получить полный контроль над сайтом.

Для предотвращения уязвимостей в конфигурации рекомендуется регулярно проверять и обновлять настройки серверов и приложений. Также важно использовать автоматизированные инструменты для сканирования конфигураций на наличие уязвимостей.

Методы защиты от угроз на этапе кодирования

- Валидация и фильтрация полей ввода
- Использование токенов для защиты от CSRF
- Использование WAF (Web Application Firewall)
- Шифрование данных

Методы тестирования безопасности веб приложений

Статическое тестирование кода SAST (Static Application Security Testing)

Статическое тестирование кода включает в себя анализ исходного кода приложения без его выполнения.

Инструменты для статического анализа могут автоматически сканировать код и находить потенциальные проблемы. Например, статический анализатор может обнаружить использование небезопасных функций или неправильное управление памятью.

Динамическое тестирование DAST (Dynamic Application Security Testing)

Динамическое тестирование проводится на работающем приложении. Позволяет выявить уязвимости, которые могут возникнуть в результате взаимодействия различных компонентов системы.

Динамическое тестирование может включать в себя как ручное, так и автоматизированное тестирование. Например, тестировщик может использовать автоматизированный сканер уязвимостей для проверки веб-приложения на наличие известных уязвимостей.

Методы тестирования безопасности веб приложений

IAST **(Interactive Application Security Testing)** — интерактивное тестирование безопасности приложений. IAST разработан для устранения недостатков SAST и DAST путем объединения элементов обоих подходов. IAST выполняет весь анализ в режиме реального времени.

IAST работает внутри приложения и может анализировать:

- код приложения
- потоки данных
- конфигурации
- HTTP-запросы и ответы
- библиотеки, фреймворки и другие компоненты
- информацию о внутреннем подключении

Методы тестирования безопасности веб приложений

Тестирование на проникновение (Penetration testing)

Это метод, при котором тестировщики пытаются взломать систему, используя те же методы, что и злоумышленники.

Цель — выявить уязвимости, которые могут быть использованы для реальных атак.

Пентестинг может быть черным, серым или белым, в зависимости от уровня доступа тестировщиков к системе. Например, при черном пентестинге тестировщики не имеют никакой информации о системе и действуют как реальные злоумышленники.

Наиболее популярные инструменты тестирования безопасности веб-приложений

Сканирование веб-приложений, перехват и изменение трафика, управление сессиями.

Сканирование веб-приложений. Один из самых популярных платных сканеров.

Сканирование веб-серверов на наличие уязвимостей

Позволяет сканировать сеть на наличие открытых портов и получить информацию о запущенных службах

Реальный пример из жизни:

Стек тестирования безопасности для сайта «5-й элемент»

Для **Sast** используются

- Snyk
- Semgrep
- SonarQube
- PT Application Inspector (платное ПО)

Для Snyk и Semgrep можно писать свои правила

Для **Dast** решений используются

- Burp Suite
- Owasp zap
- Sqlmap

+ свой код для проверки уязвимостей

Базы данных угроз

В 90-х появились первые публичные справочники угроз.

Самая популярная сегодня база **CVE (Common Vulnerabilities and Exposures)**. Она является публичной и финансируется подразделением US-CERT (Национального управления кибербезопасности Министерства безопасности США). Актуальная база представляет собой словарь идентификаторов уязвимостей в компьютерной безопасности. CVE является общепринятым международным стандартом, используемым для однозначной идентификации и обмена информацией о различных уязвимостях.

Каждая запись в базе представляет собой уникальный идентификатор в форме «CVE-год-номер», где «год» это год создания идентификатора, а «номер» - уникальный номер уязвимости.

Каждая запись связана с **NVD (National Vulnerability Database)**.

Открытый стандарт CVSS (Common Vulnerability Scoring System) предоставляет собой числовую оценку, которая позволяет оценить влияние уязвимости.

Кто занимается актуализацией баз данных угроз

Благодаря открытому подходу, редактором базы могут быть разные пользователи. Сравнить такой подход можно с созданием и редактированием статей Википедии.

Помимо базы NVD существуют и другие базы.

MITRE ATT&CK – содержит процедуры, техники и тактики, используемые злоумышленниками

Exploit Database – содержит описания и коды эксплойтов и предоставляет информацию, которая может быть использована для тестирования и обучения.

Vulners – использует уникальные технологии для сбора данных о уязвимостях, эксплойтах, патчах и других аспектах кибербезопасности.

CWE (Common Weakness Enumeration) – описывает типы слабостей в ПО, включая ошибки кодирования, архитектурные проблемы и другие.

CIRCL (Computer Incident Response Center Luxembourg) описывает инструменты для анализа потенциальных утечек информации.

https://www.securityvision.ru/blog/obzor-baz-dannykh-ugroz/

В России актуализацией баз данных угроз занимается **ФСТЭК (Федеральная Служба по Техническому и Экспортному Контролю)**.

В Беларуси Оперативно-аналитического центра при Президенте Республики Беларусь.

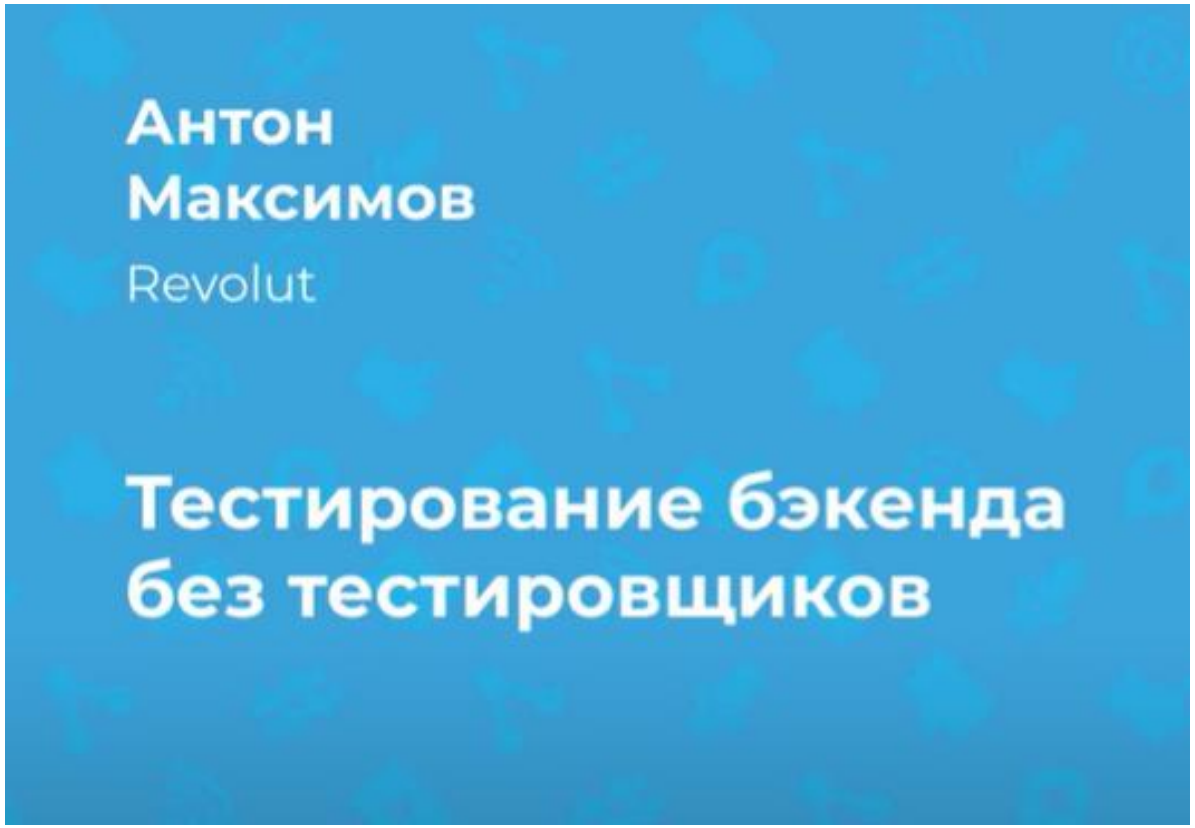
Уязвимость в GPT-4 открыла бесплатный доступ к нейросети

Пример эксплойта

Разработчик под ником xtekky представил проект GPT4Free — набор инструментов, предоставляющих бесплатный и почти неограниченный доступ к чат-боту на базе GPT-4 и его предшественнику GPT-3.5. Эксплойт был разработан методом обратного проектирования, которое помогло обнаружить уязвимость в API нейросети.

Эксплойт GPT4Free не обходит механизмы платного доступа к платформе OpenAI, а «обманывает» API, заставляя платформу считать, что запросы поступают от ресурсов, к которым привязаны платные учётные записи.

В свою защиту разработчик заявляет, что GPT4Free предназначен для работы только в *«образовательных целях»*. Он также заявил, что будет пытаться продолжить работу над проектом, даже если его попробуют засудить. По всей вероятности, оказавшиеся жертвами GPT4Free ресурсы в обозримом будущем ликвидируют бреши безопасности и закроют доступ для инструмента — исходный код проекта может удалить и GitHub, но это только подстегнёт его последователей.



Доклад о том, почему разработчику важно знать тестирование

https://www.youtube.com/watch?v=BEPGrEWZVUE&list=PL8761XQAJnradvEvL8QLs4VloEqtOWDZj