

# **Отчёт по лабораторной работе №2**

**Защита научно-технической информации предприятия**

Астраханцева Анастасия  
Ибатулина Дарья  
Ганина Таисия  
Шошина Евгения  
Кадирова Мехрубон  
Хассан Факи Абакар  
(группа НФИбд-01-22)

# **Содержание**

<b>1 Цель работы</b>	<b>4</b>
<b>2 Задание</b>	<b>5</b>
<b>3 Теоретическое введение</b>	<b>6</b>
<b>4 Выполнение лабораторной работы</b>	<b>8</b>
<b>5 Выводы</b>	<b>27</b>
<b>Список литературы</b>	<b>28</b>

# Список иллюстраций

4.1 Установка фильтров . . . . .	9
4.2 Активность трояна LaZagne . . . . .	10
4.3 Попытка выполнения удаленного кода через Microsoft PowerShell на клиентском устройстве . . . . .	10
4.4 Уязвимость XSS . . . . .	11
4.5 Попытка SQL-инъекции с использованием SELECT и SLEEP на веб-сервере redmine.ampire.corp . . . . .	11
4.6 Чужой пользователь Redmine . . . . .	12
4.7 Смена пароля . . . . .	15
4.8 Устранена уязвимость “Слабый пароль пользователя” . . . . .	15
4.9 Вкладка General в планировщике задач . . . . .	17
4.10 Вкладка Actions в планировщике задач - путь к вредоносному файлу	18
4.11 Устранено последствие уязвимости “Слабый пароль пользователя”	18
4.12 Просмотр учетных данных администратора . . . . .	19
4.13 Модификация файла redcloth3.rb для устранения уязвимости . . . . .	20
4.14 Перезапуск сервера . . . . .	20
4.15 Содержимое Wiki-страницы после внесения изменений в код и перезапуска сервера . . . . .	21
4.16 Устранена уязвимость 2 (XSS) . . . . .	21
4.17 Новый пользователь Redmine . . . . .	22
4.18 Удаление нового пользователя Redmine . . . . .	22
4.19 Полностью устранена вторая уязвимость и её последствие . . . . .	23
4.20 Эксплуатация уязвимости Blind SQL-инъекции . . . . .	24
4.21 Внесение изменений в файл query.rb . . . . .	25
4.22 Перезапуск сервера . . . . .	26
4.23 Уязвимость Blind_SQLi была устранена . . . . .	26

# **1 Цель работы**

Целью лабораторной работы является освоение практических навыков выявления, анализа и устранения уязвимостей информационных систем в рамках сценария «Защита научно-технической информации предприятия».

## **2 Задание**

1. Изучить уязвимости: слабый пароль пользователя, Blind SQL, XSS
2. Проанализировать последовательность действий нарушителя на каждом этапе атаки.
3. Освоить методы детектирования атак с использованием средств мониторинга и анализа безопасности.
4. Выполнить мероприятия по устранению последствий атаки [1].

### 3 Теоретическое введение

**VipNet IDS** (Intrusion Detection System) — программно-аппаратный комплекс, предназначенный для обнаружения вторжений в информационные системы. Он осуществляет непрерывный мониторинг сетевого трафика, выявляет признаки известных атак по сигнатурам, а также аномалии в поведении пользователей и систем. При обнаружении угрозы VipNet IDS генерирует оповещения и предлагает рекомендации по реагированию. Система тесно интегрирована с другими компонентами линейки VipNet, что обеспечивает комплексную, многоуровневую защиту корпоративной ИТ-инфраструктуры [2].

**Слабый пароль пользователя** — распространённая конфигурационная ошибка, при которой для учётной записи задаётся предсказуемый или общеизвестный пароль (например, «123456», «password», «admin» и т.п.). Такая практика создаёт условия для успешного подбора или перебора учётных данных, что даёт злоумышленнику несанкционированный доступ к системе.

**XSS (межсайтовый скрипting), CVE-2019-17427** — уязвимость в веб-приложении Redmine, вызванная недостаточной фильтрацией пользовательского ввода в текстовых полях (в частности, при использовании формата textile). Она позволяет внедрить вредоносный JavaScript-код, который будет выполнен в браузере других пользователей при просмотре заражённой страницы. Это может привести к краже сессионных токенов, подмене интерфейса или выполнению действий от имени жертвы.

**Blind SQL Injection, CVE-2019-18890** — уязвимость в Redmine, возникающая из-за отсутствия корректной валидации параметров запроса (например,

`subproject_id`). Злоумышленник может использовать «слепые» SQL-инъекции — метод, при котором данные извлекаются по косвенным признакам (например, по времени ответа сервера). Это позволяет посимвольно читать содержимое базы данных, включая конфиденциальную информацию, а в некоторых случаях — выполнять произвольные команды на сервере.

**Developer backdoor** — скрытый механизм доступа, намеренно или случайно оставленный разработчиком в программном обеспечении. Такой «чёрный ход» позволяет обойти стандартные процедуры аутентификации и получить несанкционированный контроль над системой без видимых следов взлома.

**Redmine User** — компрометация учётной записи в системе Redmine (например, через XSS или утечку данных с рабочей станции) предоставляет злоумышленнику доступ к проектной документации, задачам, исходным кодам и другим элементам научно-технической информации. В контексте сценария №5 лабораторной работы это является ключевым этапом утечки конфиденциальных данных предприятия.

## 4 Выполнение лабораторной работы

Для сценария № 5 «Защита научно-технической информации предприятия» определены три уязвимости и два последствия:

- 1) Уязвимость 1. Слабый пароль пользователя.
- 2) Последствие. Developer backdoor.
- 3) Уязвимость 2. XSS (CVE-2019-17427).
- 4) Последствие. Redmine User.
- 5) Уязвимость 3. Blind SQL (CVE-2019-18890).

### 1. Заполнение карточек инцидентов.

Для обнаружения и анализа атак использовались средства ViPNet IDS NS.

Для обнаружения актуальной подозрительной активности пользуемся фильтрами по дате, времени и важности (рис. 4.1).

События за последние 24 часа

Дата и время событий

За последние 1 День

За период с 04.10.2025 09:20:00 по 04.10.2025 09:40:59

Основные параметры

Уровень важности

Высокий  Средний  Низкий  
 Информационный

Показывать

Агрегированные события  
 Единичные события

Событие

Источник

Получатель

Рис. 4.1: Установка фильтров

Были зафиксированы следующие ключевые инциденты, соответствующие этапам атаки (рис. 4.2, 4.3, 4.4, 4.5, 4.6).

◀ Активность трояна LaZagne

Основная информация Чат

Дата и время события ⓘ  
04.10.2025 09:21

Описание ⓘ  
ET ATTACK\_RESPONSE LaZagne Artifact Outbound in FTP – метод сетевой атаки, при котором злоумышленник использует уязвимости FTP-серверов для отправки исходящего трафика на другое устройство (обычно другой сервер в сети).

Индикаторы компрометации ⓘ  
Сигнатура группы IDS ET ATTACK\_RESPONSE LaZagne Artifact Outbound FTP; В сетевом трафике строка "The LaZagne Project"

Рекомендации ⓘ  
1. Отключить порт или VLAN. 2. Проверить наличие файлов LaZagne.exe, The LaZagne Project, credentials.txt. 3. Провести антивирусное сканирование заражённой машины. 4. Сбросить сохранённые учётные данные в браузерах и приложениях. 5. Анализировать сетевые журналы на аналогичные исходящие соединения. 6. Обновить антивирусные базы и сигнатуры IDS/IPS.

Оценка ⚫ ⚫ ⚫ ⚫ ⚫ ⚫

Автор  
Ибатуллина Дарья  
@1132226434@pfur.ru

Ответственный  
Асташкицева Анастасия  
@1132226437@pfur.ru

Источник  
10.10.4.13

Поражённые активы  
10.10.4.11

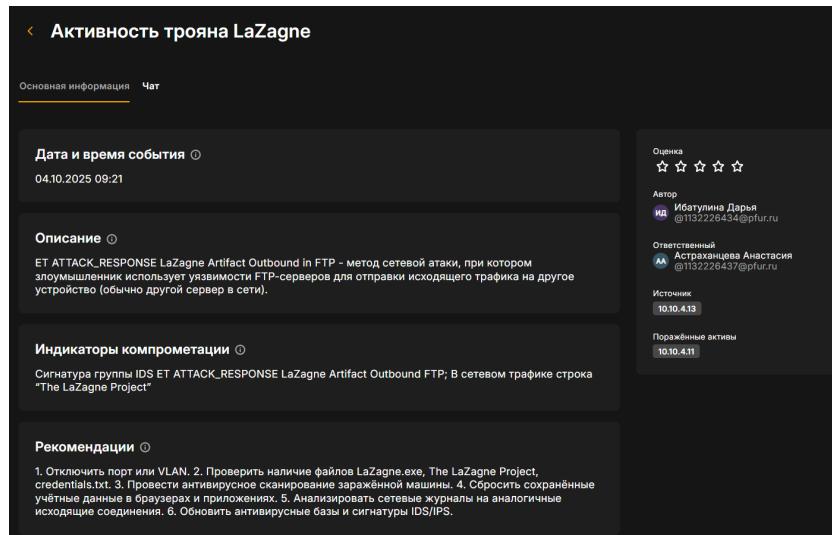


Рис. 4.2: Активность трояна LaZagne

◀ Попытка выполнения удаленного кода через Microsoft PowerShell на клиентском устройстве

Основная информация Чат

Дата и время события ⓘ  
04.10.2025 09:21

Описание ⓘ  
Событие связано с обнаружением ответа атаки, указывающего на попытку выполнения произвольного кода или отправку команд управления на ранее скомпрометированный ресурс. Обнаружен трафик, содержащий сигнатуру Microsoft PowerShell ("Windows PowerShell", "Copyright", "Microsoft Corp"), что указывает на использование PowerShell для эксплуатации уязвимости на клиентском устройстве с OS Windows.

Индикаторы компрометации ⓘ  
successful-admin, attack\_response, Exploitation

Рекомендации ⓘ  
1. Изолировать заражённый компьютер от сети. 2. Проверить запущенные процессы PowerShell и задания в планировщике. 3. Просканировать систему антивирусом и удалить вредоносные файлы. 4. Проверить журналы PowerShell на выполнение подозрительных команд. 5. Сбросить пароли пользователей, особенно администраторов. 6. Проверить сетевые подключения и запретить неизвестные исходящие соединения. 7. Обновить антивирусные базы и сигнатуры IDS.

Оценка ⚫ ⚫ ⚫ ⚫ ⚫ ⚫

Автор  
Ибатуллина Дарья  
@1132226434@pfur.ru

Ответственный  
Не заполнено

Источник  
10.10.4.13

Поражённые активы  
10.10.4.11

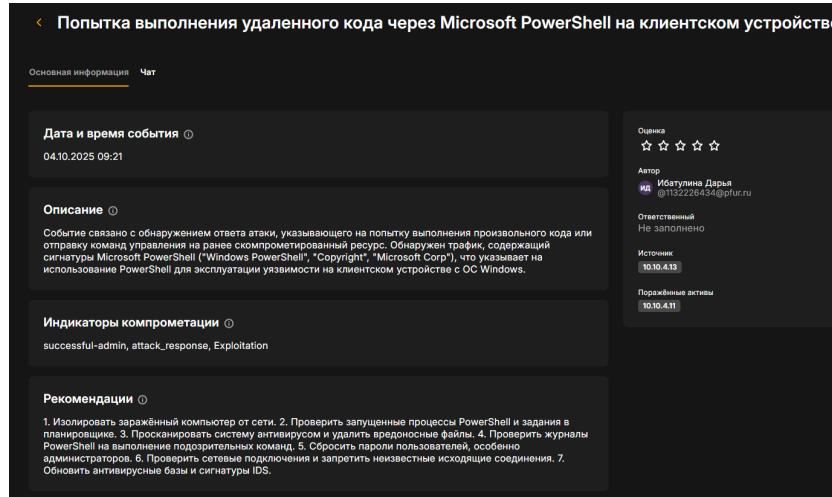


Рис. 4.3: Попытка выполнения удаленного кода через Microsoft PowerShell на клиентском устройстве

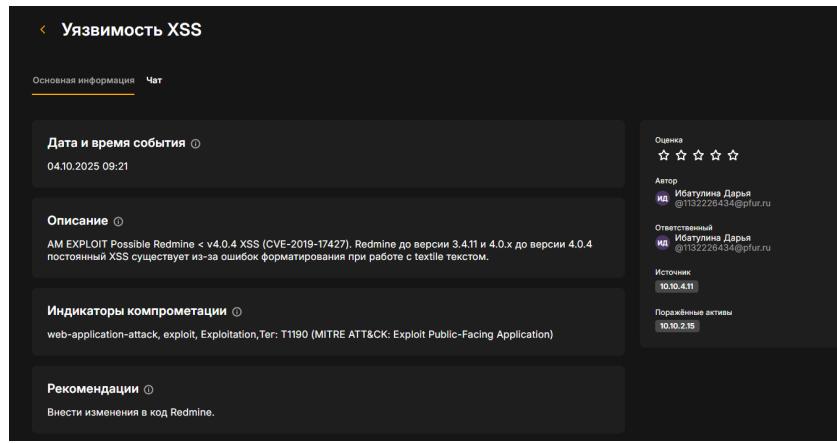


Рис. 4.4: Уязвимость XSS

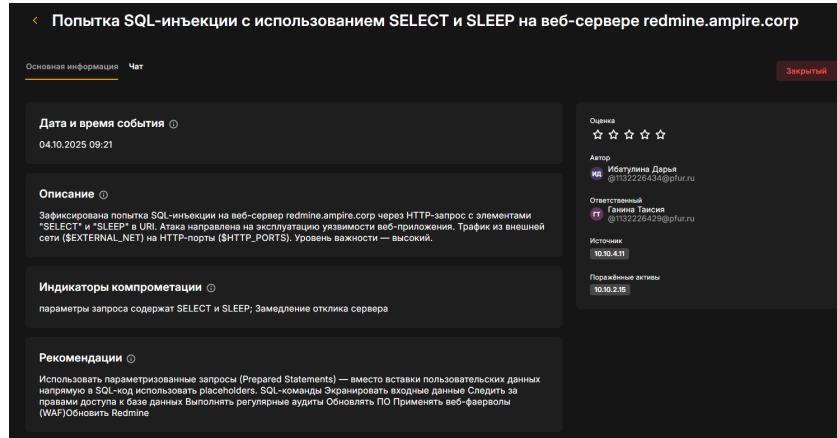


Рис. 4.5: Попытка SQL-инъекции с использованием SELECT и SLEEP на веб-сервере redmine.ampire.corp

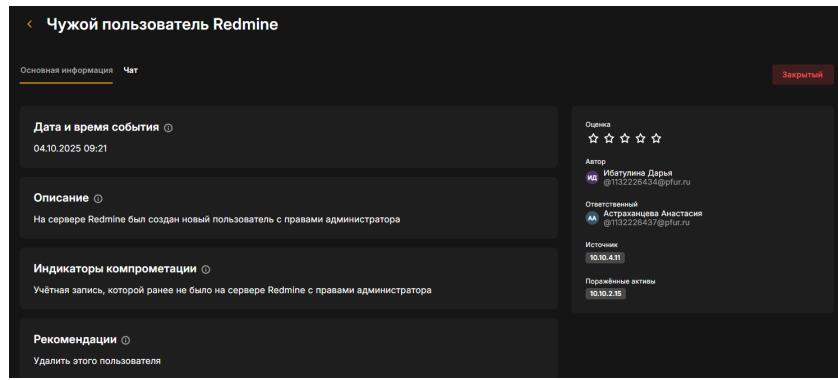


Рис. 4.6: Чужой пользователь Redmine

- **Инцидент 1: Активность трояна LaZagne**

Что произошло: Обнаружена активность сетевой атаки, связанной с трояном LaZagne. Сигнатура IDS указывает на “ET ATTACK\_RESPONSE LaZagne Artifact Outbound in FTP”, а в трафике зафиксирована строка “The LaZagne Project” [3].

Что это означает: Внутренний нарушитель, получив доступ к компьютеру пользователя Dev-1 (после подбора слабого пароля на файловом сервере и запуска backdoor), использует троян LaZagne для кражи учетных данных (логинов и паролей), хранящихся в браузерах и других приложениях. Эти данные необходимы для последующего входа в систему Redmine.

Рекомендации: Необходимо изолировать зараженный хост (10.10.4.13), провести полное антивирусное сканирование, сбросить все пароли, которые могли быть скомпрометированы, и обновить антивирусные базы и сигнатуры IDS/IPS [4].

- **Инцидент 2: Попытка выполнения удаленного кода через Microsoft PowerShell**

Что произошло: Обнаружен трафик, указывающий на попытку выполнения произвольного кода или отправки команд управления на ранее скомпрометированный ресурс. Трафик содержит сигнатуры Microsoft PowerShell (“Windows PowerShell”, “Copyright”, “Microsoft Corp”).

Что это означает: Злоумышленник использует PowerShell для автоматизации дальнейших действий: загрузки дополнительных вредоносных модулей, сбора информации или установки постоянного доступа (backdoor). Это позволяет ему эффективно управлять зараженной машиной.

Рекомендация: Изолировать зараженный клиентский компьютер, проверить и завершить все подозрительные процессы PowerShell, провести антивирусное сканирование, сбросить пароли администраторов и обновить системы защиты [4].

### **Инцидент 3: Уязвимость XSS (Cross-Site Scripting)**

Что произошло: На сервере Redmine обнаружена уязвимость межсайтового скрипtingа (XSS), идентифицированная как CVE-2019-17427. Эта уязвимость существует в версиях Redmine до 3.4.11 и 4.0.x до 4.0.4.

Что это означает: Это ключевой этап атаки. Внутренний нарушитель использует эту уязвимость для внедрения вредоносного JavaScript-кода на Wiki-страницу проекта Dev1. Цель – получить контроль над сессией администратора Redmine, чтобы создать нового пользователя с правами администратора. Это позволяет злоумышленнику закрепиться в системе и получить неограниченный доступ к пользовательской базе данных.

Рекомендация: Необходимо внести изменения в код Redmine, а именно в файл redcloth3.rb, чтобы исправить ошибку форматирования при работе с textile текстом. После внесения изменений требуется перезапустить веб-сервер (`sudo systemctl restart nginx.service`) [4].

- **Инцидент 4: Попытка SQL-инъекции**

Что произошло: Зафиксирована попытка SQL-инъекции на веб-сервере `redmine.ampire.corp`. Злоумышленник отправил HTTP-запрос, содержащий элементы SELECT и SLEEP, что является классическим признаком атаки типа “слепая инъекция”.

Что это означает: Это финальный этап атаки. После того, как нарушитель получил права администратора в Redmine, он использует уязвимость Blind SQL-

инъекции (CVE-2019-18890) для посимвольного перебора и извлечения конфиденциальной информации из базы данных. Использование SLEEP позволяет ему определить, какие символы в запросе являются правильными, основываясь на времени ответа сервера.

Рекомендация: Необходимо внести изменения в код Redmine, в частности, в файл query.rb, добавив фильтрацию входных параметров и закомментировав уязвимый код. После этого также требуется перезапустить веб-сервер [5] [4].

- **Инцидент 5: Чужой пользователь Redmine**

Что произошло: На сервере Redmine был создан новый пользователь с правами администратора [6].

Что это означает: Это прямое следствие успешной эксплуатации уязвимости XSS. Нарушитель, получив контроль над сессией администратора, создал нового пользователя (hacker) с максимальными привилегиями. Это позволяет ему беспрепятственно просматривать, изменять или удалять любую информацию в Redmine, включая конфиденциальные данные проектов.

Рекомендация: Необходимо немедленно удалить этого нового пользователя через веб-интерфейс Redmine (в разделе Administration -> Users). Важно: Это действие будет эффективным только после того, как будет устранена сама уязвимость XSS, иначе нарушитель сможет снова создать нового пользователя [4].

1. Устранение первой уязвимости (Слабый пароль пользователя) и ее последствия:

В начале сценария внутренний нарушитель подбирает слабый пароль для входа на файловый сервер, получает доступ к учетным данным пользователя dev1 и загружает на его компьютер backdoor. Чтобы нейтрализовать эту угрозу и предотвратить дальнейший доступ злоумышленника, необходимо сменить пароль пользователя dev1 (рис. 4.7). Мы установили сложный пароль, состоящий

из латинских букв нижнего и верхнего регистра, цифр и специальных символов, а также не являющийся словом, датой рождения и не содержащий другие персональные данные:

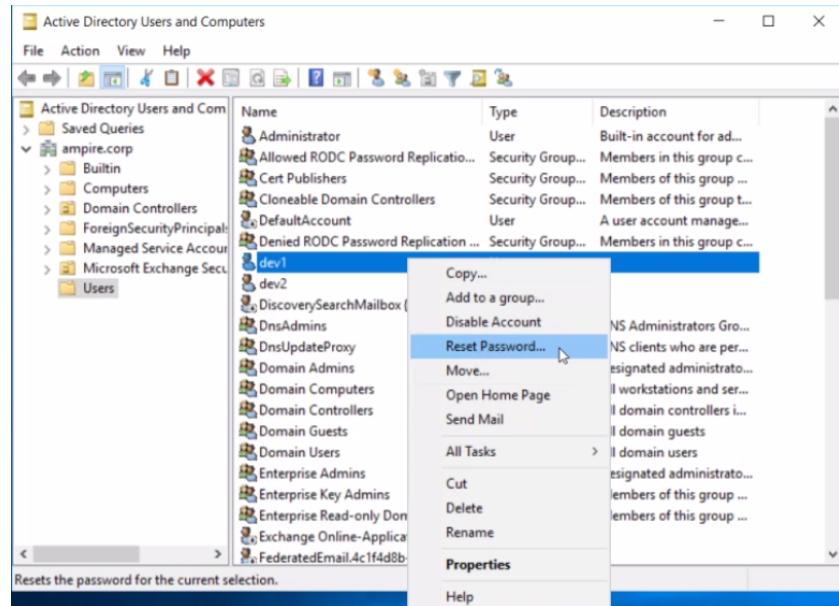


Рис. 4.7: Смена пароля

На скриншоте (рис. 4.8) видно, что у нас устранена первая уязвимость (Слабый пароль пользователя), и можно приниматься за последствие.

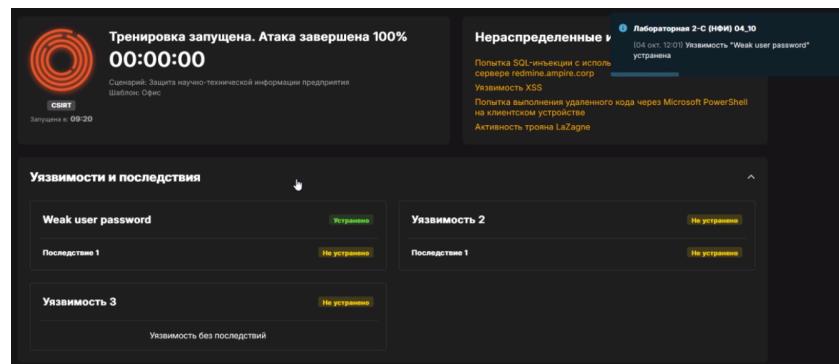


Рис. 4.8: Устранена уязвимость “Слабый пароль пользователя”

Что произошло из-за установки слабого пароля пользователя dev1:

**Начало атаки:** Внутренний нарушитель подобрал слабый пароль на файловом сервере и заменил легитимный файл на вредоносный (backdoor).

**Заражение:** Пользователь *dev1* скачал и запустил этот вредоносный файл.

**Закрепление:** После получения контроля над компьютером *dev1*, нарушитель создал задачу в планировщике, которая будет автоматически запускать вредоносный файл *svchosting.exe* каждый раз при входе пользователя *dev1* в систему. Это позволяет злоумышленнику сохранять доступ к компьютеру даже после перезагрузки.

Открываем планировщик задач и обнаруживаем подозрительную задачу:

Имя задачи: *Evil task*

Автор: *AMPIRE\dev1*. Это означает, что задача была создана от имени легитимного пользователя *dev1*, что помогает ей выглядеть менее подозрительно.

**Настройки безопасности:** Задача будет запускаться только тогда, когда пользователь *dev1* залогинен в системе (Run only when user is logged on). Это типично для атак, где злоумышленник хочет, чтобы вредоносное ПО работало в контексте активной сессии пользователя, чтобы иметь доступ к его данным и ресурсам (рис. 4.9).

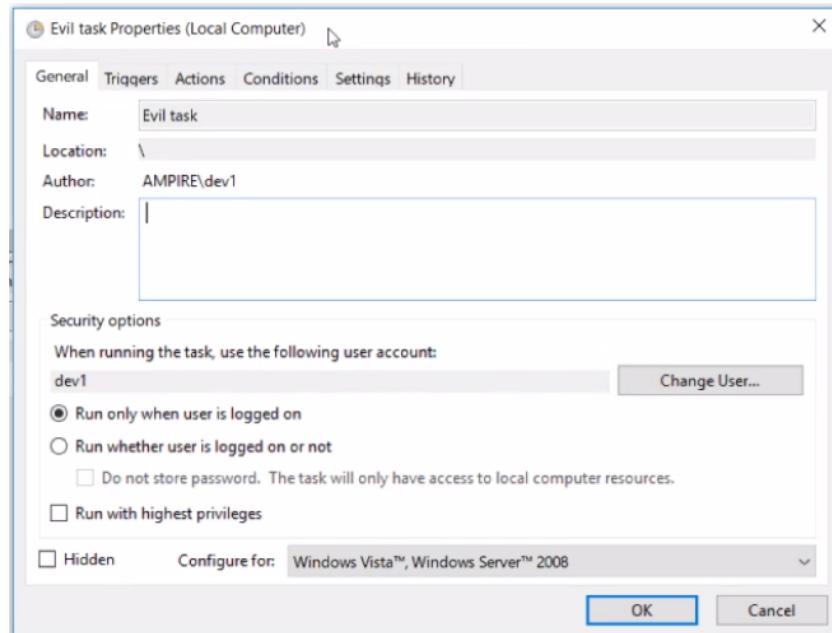


Рис. 4.9: Вкладка General в планировщике задач

Задача настроена на выполнение программы (Start a program). Путь к исполняемому файлу: C:\Users\dev1\Downloads\svchosting.exe. Это указывает на то, что злоумышленник разместил вредоносный файл svchosting.exe в папке загрузок пользователя dev1 и настроил его автоматический запуск через планировщик задач (рис. 4.10).

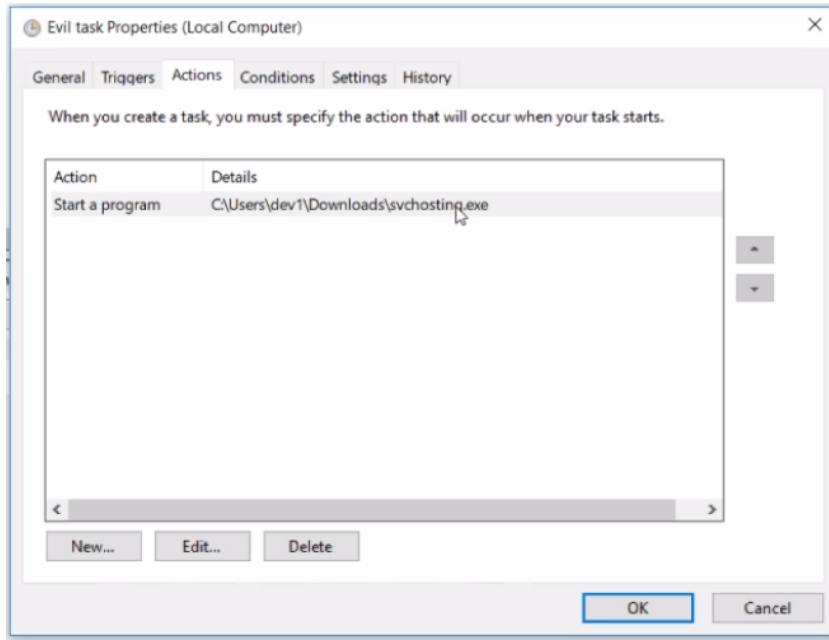


Рис. 4.10: Вкладка Actions в планировщике задач - путь к вредоносному файлу

Для устранения последствия мы удалили задачу и вредоносный exe-файл в директории C:\Users\dev1\Downloads.

Переходим на сервер и видим, что устанено последствие уязвимости “Слабый пароль пользователя” (рис. 4.11).

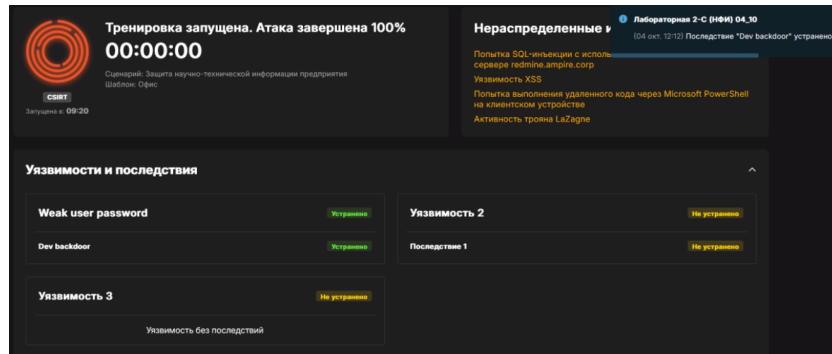


Рис. 4.11: Устранено последствие уязвимости “Слабый пароль пользователя”

## 2. Устранение второй уязвимости и её последствия:

Последовательность действий, показанная на скриншотах (рис. 4.12-4.16), пред-

ставляет собой процесс устранения уязвимости XSS (CVE-2019-17427) в системе Redmine, которая была эксплуатирована злоумышленником для создания нового пользователя с правами администратора.

На скриншоте (рис. 4.12) видим, что для доступа к серверу Redmine (10.10.2.15) были использованы учетные данные администратора (admin). Это необходимо для получения прав на редактирование кода сервера.

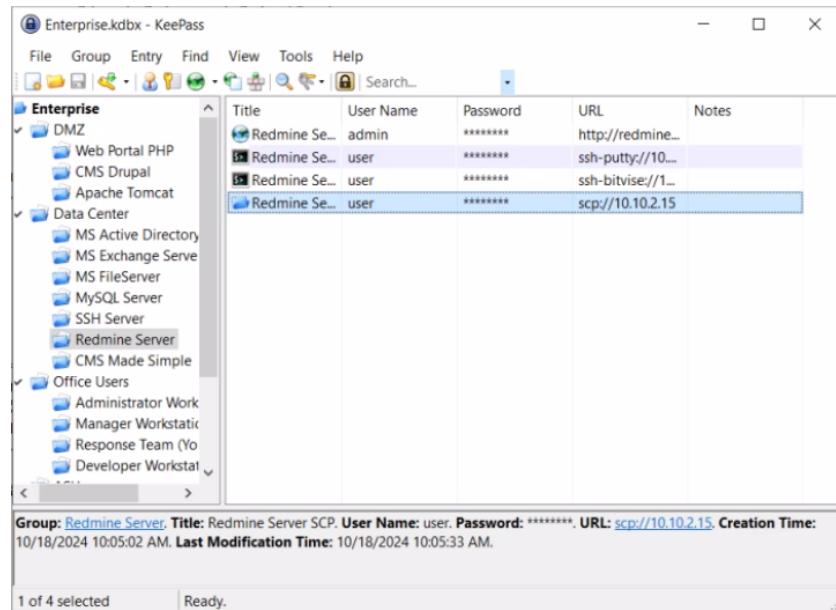


Рис. 4.12: Просмотр учетных данных администратора

Далее, открываем файл `/var/www/redmine/lib/redcloth3.rb`. Это библиотека, отвечающая за преобразование текста в формате textile в HTML. В этом файле было найдено место, где обрабатываются HTML-теги, и внесены изменения для исправления уязвимости.

Видим, что злоумышленником была добавлена константа `ALLOWED_TAGS`, которая определяет список тегов, которые не будут экранироваться. Тег `,`, который использовался злоумышленником для внедрения вредоносного JavaScript-кода, мы исключили из этого списка. Теперь, при обнаружении любого тега, не входящего в `ALLOWED_TAGS`, он теперь будет автоматически экранирован (заменен на `<` и `>`), что делает его безопасным для отображения (рис. 4.13).

```

#!/var/www/redmine/lib/redcloth3.rb - user@10.10.2.15 - Editor - WinSCP
Encoding: 1252 (ANSI - Lc)
if tags.has_key? tag
  pcs = [tag]
  tags[tag].each do |prop|
    ['"', "'", '&'].each do |q|
      q2 = ( q =~ '' ? q : '\s' )
      if raw[3] =~ /\#(prop)\$\s*=\$\s*(#(q)([^#(q2)]+)(#(q)/i
        attrv = $1
        next if prop == 'src' and attrv =~ /\#(prop)=\${$1.gsub('','\\'))\w+:/i
        pcs << "#{$prop}=\${$1.gsub('','\\'))}\w+"
        break
      end
    end
  end if tags[tag]
  "<#{$raw[1]}#{pcs.join " "}>"  

else
  ""
end
end

ALLOWED_TAGS = %w(redpre pre code kbd notextile)
def escape_html_tags(text)
  text.gsub!(/\<(.|?)([!w]+)(~>\w*)(>?)\>/) do |m|
    if ALLOWED_TAGS.include?(m[2]) && m[3].present?
      "<#{m[1]}#{m[3]}>"
    else
      "&lt;#{m[1]}#{'&gt;' unless m[3].blank?}"  

    end
  end
end
end

```

Line: 1223/1225 Column: 8 Encoding: 1252 (ANSI - Lc)

Рис. 4.13: Модификация файла redcloth3.rb для устранения уязвимости

После внесения изменений в код в терминале мы выполняем команду `sudo systemctl restart nginx.service`. Это необходимо для того, чтобы веб-сервер загрузил обновленный код и изменения вступили в силу (рис. 4.14).

```

user@redmine: ~
Using username "user".
Last login: Wed Mar 12 13:14:21 2025
user@redmine:~$ sudo systemctl restart nginx.service
[sudo] password for user:
user@redmine:~$ sudo systemctl restart nginx.service
user@redmine:~$ sudo systemctl restart nginx.service
user@redmine:~$ 

```

Рис. 4.14: Перезапуск сервера

На скриншоте (рис. 4.15) показано содержимое Wiki-страницы проекта Dev1 до и после перезапуска сервера. До перезапуска вредоносный код отображался

“как есть”, запускался при переходе на веб-страницу, а после — был экранирован и стал просто текстом.

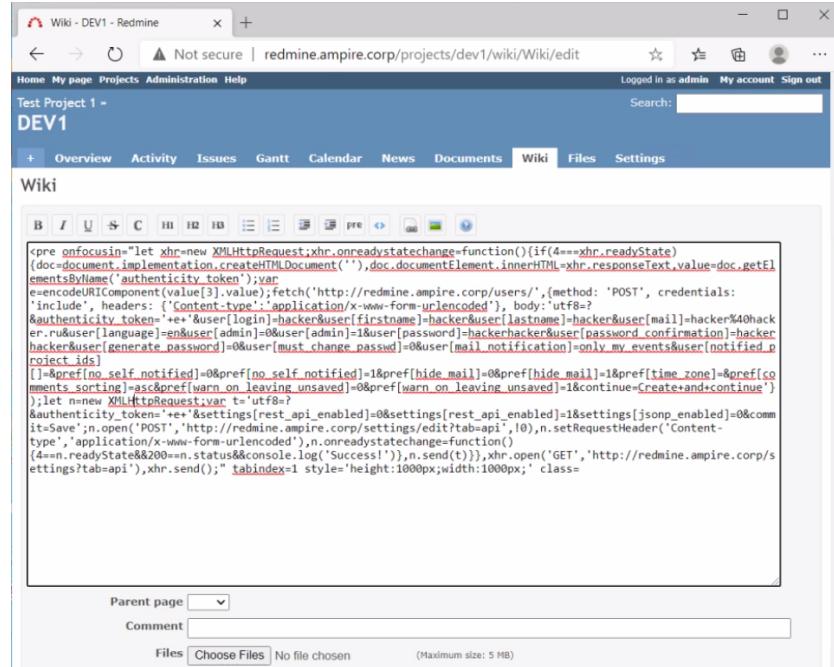


Рис. 4.15: Содержимое Wiki-страницы после внесения изменений в код и перезапуска сервера

Видим, что уязвимость XSS успешно устранена (рис. 4.16).

A screenshot of a web application interface titled "Лабораторная 2-С (НФИ) 04\_10". The main header includes "Группа: НФИбд-01-22 (C) - вторник" and a "Добавить инцидент" button. The top navigation bar has tabs: "Основная информация", "Инциденты", "Цепочки кибератак", "Beta", "Схема шаблона", and "Материалы". On the left, there's a circular progress indicator with the text "Тренировка запущена. Атака завершена 100% 00:00:00" and "Сценарий: Защита научно-технической информации предприятия". Below it, "Запущена в 09:20". On the right, a section titled "Нераспределенные инциденты" lists two items: "Попытка SQL-инъекции с использованием SELECT и SLEEP на веб-сервере redmine.ampire.corp" (status: Извлечено) and "Попытка выполнения удаленного кода через Microsoft PowerShell на клиентском устройстве" (status: Извлечено). A sidebar on the right shows a summary of vulnerabilities: "Уязвимости и последствия" (Weak user password, Dev backdoor) and "Извлечение XSS" (XSS, Последствия 1). A note at the bottom says "Уязвимость без последствий".

Рис. 4.16: Устранена уязвимость 2 (XSS)

Далее работаем с последствием этой уязвимости.

В ходе сценария внутренний нарушитель успешно эксплуатировал уязвимость XSS (CVE-2019-17427) для внедрения вредоносного JavaScript-кода на Wiki-страницу проекта Dev1. Этот код был направлен на создание нового пользователя с правами администратора, что позволило злоумышленнику получить неограниченный доступ к системе Redmine и ее конфиденциальной информации.

На данных скриншотах (рис. 4.17-4.18) показан процесс удаления созданного злоумышленником пользователя через веб-интерфейс администратора Redmine.

The screenshot shows the Redmine 'Users' administration page. A new user named 'hacker' has been added, appearing in the list with the following details:

Login	First name	Last name	Email	Administrator	Created	Last connection
admin	Redmine	Admin	admin@example.net	✓	02/13/2020 02:10 PM	10/04/2025 12:56 PM
DEV1	John	Doe	dev1@ampire.corp	✓	02/17/2020 08:18 AM	10/04/2025 12:56 PM
DEV2	Jane	Dow	janedow@ampire.corp	✓	02/19/2020 12:31 PM	10/04/2025 12:56 PM
hacker	hacker	hacker	hacker@hacker.ru	✓	10/04/2025 09:21 AM	

A confirmation dialog box is overlaid on the page, asking 'Are you sure?' with 'OK' and 'Cancel' buttons.

Рис. 4.17: Новый пользователь Redmine

The screenshot shows the Redmine 'Users' administration page again. The user 'hacker' is selected, and a confirmation dialog box is overlaid, asking 'Are you sure?' with 'OK' and 'Cancel' buttons.

Рис. 4.18: Удаление нового пользователя Redmine

Последствие Redmine User теперь также имеет статус «Устраниено».

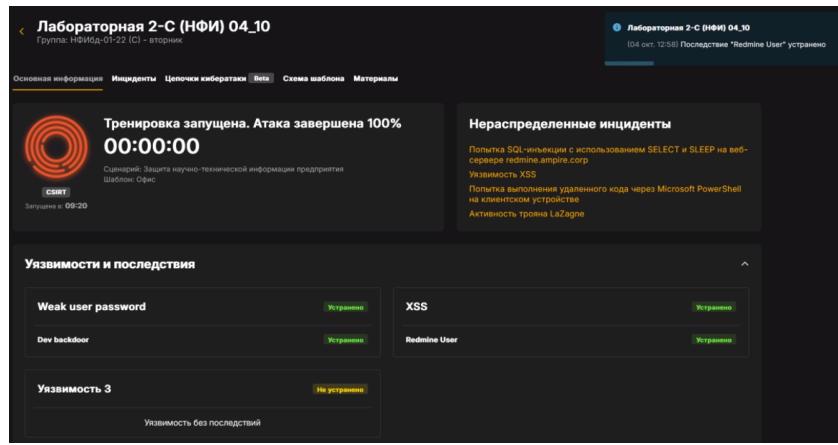


Рис. 4.19: Полностью устранена вторая уязвимость и её последствие

### 3. Устранение третьей уязвимости (последствия у нее нет):

Следующие скриншоты (рис. 4.20-4.21) демонстрируют процесс устранения уязвимости Blind SQL-инъекции (CVE-2019-18890) в системе Redmine, которая была эксплуатирована злоумышленником для похищения конфиденциальной информации [5].

На рис. 4.20 показан процесс эксплуатации уязвимости до ее устранения. Он демонстрирует, как злоумышленник проводит атаку Blind SQL-инъекции.

Что видно:

В адресной строке браузера введен URL: `redmine.ampire.corp/issues.xml?project_id=1&subproject_id=2,3-SLEEP(2)`.

Вкладка “Network” (Сеть) в инструментах разработчика показывает список HTTP-запросов.

Запрос `issues.xml?project_id=1&subproject_id=2,3-SLEEP(2)` имеет время выполнения 8.13 секунд (Time).

В левой части окна отображается XML-ответ сервера, который содержит данные о задачах проекта.

Это классический пример Blind SQL-инъекции. Злоумышленник использует функцию `SLEEP(2)` для создания искусственной задержки ответа сервера.

Если параметр `subproject_id` не был бы обработан корректно, сервер выполнил бы SQL-запрос, содержащий `SLEEP(2)`, и ответил бы с задержкой в 2 секунды (плюс время выполнения самого запроса). Длинное время ответа (8.13 секунд) — это сигнал для атакующего, что его запрос был успешно выполнен.

Используя эту технику, злоумышленник может посимвольно перебирать данные из базы данных (например, пароли, названия проектов, содержимое задач), просто измеряя время ответа сервера. Если время ответа увеличивается, значит, символ подобран верно.

Этот скриншот служит доказательством того, что уязвимость существовала и была активно эксплуатирована. После внесения исправления в `query.rb` и перезапуска сервера, такой запрос больше не вызовет задержки, так как параметр `subproject_id` будет автоматически приведен к целому числу, и `SLEEP(2)` будет проигнорирован.

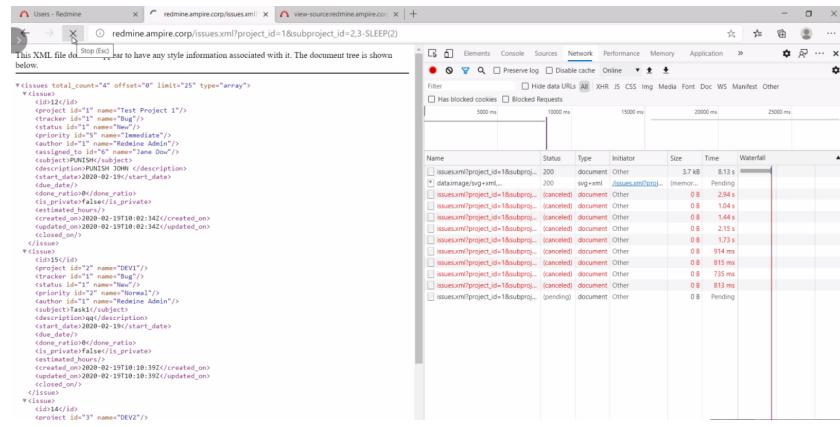


Рис. 4.20: Эксплуатация уязвимости Blind SQL-инъекции

Внесем изменения в файл `query.rb` (рис. 4.21). Этот файл является частью модели данных Redmine и отвечает за формирование SQL-запросов к базе данных. В нем находится код, который обрабатывает параметр `subproject_id`.

Что было сделано: Была внесена критическая правка. Оригинальная строка кода:

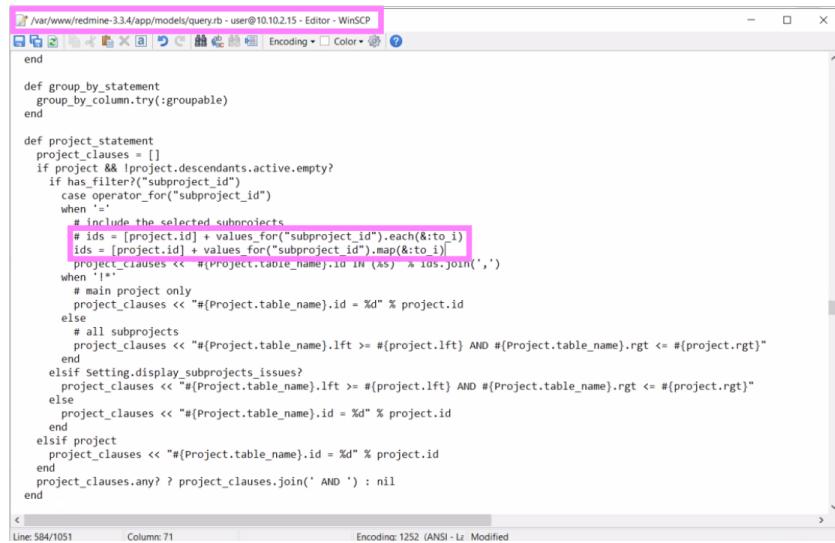
```
ids = [project.id] + values_for("subproject_id").each(&:to_i)
```

была закомментирована. Вместо нее была добавлена новая строка:

```
ids = [project.id] + values_for("subproject_id").map(&:to_i)
```

Исходный код использовал метод `.each`, который не выполнял должной фильтрации или преобразования входных данных. Это позволяло злоумышленнику передавать в параметр `subproject_id` произвольные SQL-команды, которые затем вставлялись в запрос без проверки.

Решение: Метод `.map` применяет функцию `&:to_i` ко всем элементам массива. Функция `to_i` (convert to integer) преобразует любое значение в целое число. Если входное значение не является числом, оно будет приведено к 0. Таким образом, любой попытке передать SQL-код (например, `or 1=1 --`) будет противопоставлено число 0, что делает атаку невозможной. Это простая, но эффективная форма фильтрации входных данных.



The screenshot shows a WinSCP file editor window with the title '/var/www/redmine-3.3.4/app/models/query.rb - user@10.10.2.15 - Editor - WinSCP'. The code in the editor is as follows:

```
end

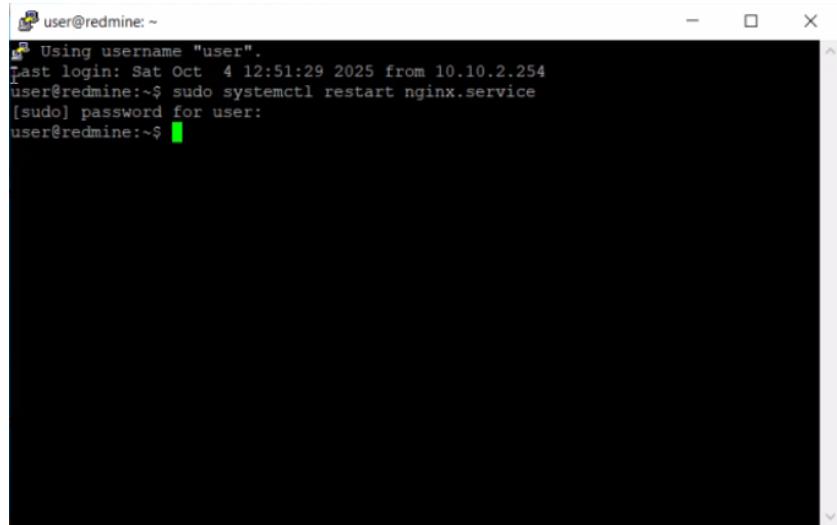
def group_by_statement
  group_by_column.try(:groupable)
end

def project_statement
  project_clauses = []
  if project && !project.descendants.active.empty?
    if has_filter?("subproject_id")
      case operator_for("subproject_id")
      when '='
        # include the selected subprojects
        # ids = [project.id] + values_for("subproject_id").each(&:to_i)
        ids = [project.id] + values_for("subproject_id").map(&:to_i)
        project_clauses << "#{$(Project.table_name)}.id IN (#{ids})"
      when '!='
        # main project only
        project_clauses << "#{Project.table_name}.id = #{project.id}"
      else
        # all subprojects
        project_clauses << "#{Project.table_name}.lft >= #{project.lft} AND #{Project.table_name}.rgt <= #{project.rgt}"
      end
    elsif Setting.display_subprojects_issues?
      project_clauses << "#{Project.table_name}.lft >= #{project.lft} AND #{Project.table_name}.rgt <= #{project.rgt}"
    else
      project_clauses << "#{Project.table_name}.id = #{project.id}"
    end
  elsif project
    project_clauses << "#{Project.table_name}.id = #{project.id}"
  end
  project_clauses.any? ? project_clauses.join(' AND ') : nil
end
```

Line: 504/1051 Column: 71 Encoding: 1252 (ANSI - Lz Modified)

Рис. 4.21: Внесение изменений в файл query.rb

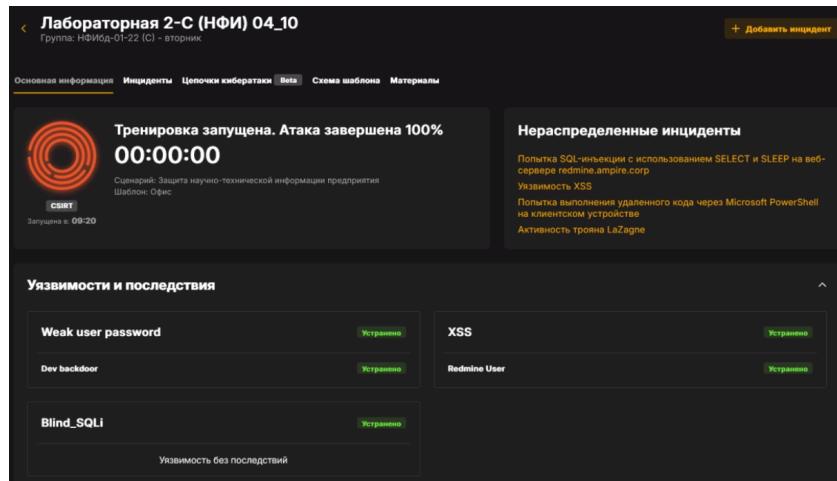
После внесения изменений в файл `query.rb` в терминале мы выполняем команду `sudo systemctl restart nginx.service`. Это необходимо для того, чтобы изменения вступили в силу (рис. 4.22).



```
user@redmine: ~
└─ Using username "user".
└─ Last login: Sat Oct  4 12:51:29 2025 from 10.10.2.254
user@redmine:~$ sudo systemctl restart nginx.service
[sudo] password for user:
user@redmine:~$
```

Рис. 4.22: Перезапуск сервера

На скриншоте (рис. 4.23) видно, что после настройки брандмауэра уязвимость IGSS32 была устранена, можно приниматься за последствия



Лабораторная 2-С (НФИ) 04\_10  
Группа: НФИод-01-22 (С) - вторник

+ Добавить инцидент

Основная информация Инциденты Цепочки кибератаки Beta Схема шаблона Материалы

Тренировка запущена. Атака завершена 100%  
00:00:00

Сценарий: Защита научно-технической информации предприятия  
Шаблон: Офис  
Запущено в 08:20

Нераспределенные инциденты

Попытка SQL-инъекции с использованием SELECT и SLEEP на веб-сервере redmine.ampire.corp  
уязвимость XSS  
Попытка выполнения удаленного кода через Microsoft PowerShell  
на клиентском устройстве  
Активность троины LaZagne

Уязвимости и последствия

Weak user password	Устранино
Dev backdoor	Устранино
Blind_SQLi	Устранино
XSS	Устранино
Redmine User	Устранино

Рис. 4.23: Уязвимость Blind\_SQLi была устранена

## **5 Выводы**

В ходе лабораторной работы был успешно реализован сценарий защиты научно-технической информации предприятия: обнаружены и устраниены уязвимости (слабый пароль, XSS, Blind SQL-инъекция), нейтрализованы последствия атаки (удалён backdoor и несанкционированный пользователь Redmine).

# **Список литературы**

1. Программный комплекс обучения методам обнаружения, анализа и устранения последствий компьютерных атака "Ampire". Сценарий №5 «ЗАЩИТА НАУЧНО-ТЕХНИЧЕСКОЙ ИНФОРМАЦИИ ПРЕДПРИЯТИЯ» [Электронный ресурс]. O'Reilly Media. 24 с.
2. Сетевой сенсор системы обнаружения атак программно-аппаратный комплекс ViPNet IDS NS 3. infotechs. 321 с.
3. The LaZagne Project [Электронный ресурс]. 2015.
4. AMTIP [Электронный ресурс].
5. CVE-2019-18890 POC (Proof of Concept) [Электронный ресурс].
6. Redmine [Электронный ресурс].